aws

模式

AWS Prescriptive Guidance



Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Prescriptive Guidance: 模式

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon 的商标和商业外观不得用于任何非 Amazon 的商品或服务，也不得以任何可能引起客户混淆、贬低或诋毁 Amazon 的方式使用。所有非 Amazon 拥有的其他商标均为各自所有者的财产，这些所有者可能附属于 Amazon、与 Amazon 有关联或由 Amazon 赞助，也可能不是如此。

Table of Contents

AWS 规范性指导模式 1
云基础 2
使用着陆区域加速器自动创建账户 AWS 3
摘要 3
先决条件和限制 3
架构 3
工具 5
最佳实践 7
操作说明 7
相关资源 30
其他信息 30
自动清点 AWS 资源 32
摘要 32
先决条件和限制 32
架构 33
工具 34
最佳实践 35
操作说明 35
故障排除 40
相关资源 41
跨账户配置 VPC 流日志 42
摘要 42
先决条件和限制 42
架构 43
工具 43
最佳实践 44
操作说明 46
相关资源 47
其他信息 47
自动标记中转网关连接 49
摘要 49
先决条件和限制 49
架构 49
工具 51

操作说明 52
相关资源 56
更多模式 57
人工智能和机器学习 58
聚合 DynamoDB 数据以在 Athena 中进行 ML 预测 59
摘要 59
先决条件和限制 59
架构 59
工具 60
操作说明 61
相关资源 70
跨账户将 AWS CodeCommit 存储库与 Amazon A SageMaker I Studio 经典版关联 71
摘要 71
先决条件和限制 71
架构 71
工具 72
操作说明 73
其他信息 78
从 PDF 文件中自动提取内容 81
摘要 81
先决条件和限制 81
架构 82
工具 83
操作说明 83
相关资源 87
附件 88
使用 SageMaker AI Deepar 构建冷启动预测模型 89
摘要 89
先决条件和限制 89
架构 90
工具 90
最佳实践 91
操作说明 91
相关资源 93
使用 SageMaker AI 和 Azure 构建 MLOps 工作流程 DevOps 94
摘要 94

先决条件和限制 94
架构 95
工具 96
最佳实践 97
操作说明 97
故障排除 104
相关资源 104
使用在 Amazon Bedrock 中配置日志记录 CloudFormation 106
摘要 106
先决条件和限制 106
架构 106
工具 107
操作说明 108
相关资源 110
在 Step Functions 中创建 Docker 容器以 SageMaker 进行模型训练 111
摘要 111
先决条件和限制 111
架构 112
工具 112
操作说明 113
相关资源 124
使用亚马逊 Bedrock 代理在 Amazon EKS 中创建访问控制 125
摘要 125
先决条件和限制 125
架构 126
工具 126
最佳实践 127
操作说明 127
故障排除 143
相关资源 143
在上部署 RAG 用例 AWS 145
摘要 145
先决条件和限制 145
架构 146
工具 147
最佳实践 148

操作说明 149
相关资源 151
其他信息 152
在单个 SageMaker 端点中部署多个管道模型对象 154
摘要 154
先决条件和限制 154
架构 154
工具 155
操作说明 156
相关资源 165
使用 RAG 和提示开发基于 AI 聊天的助手 ReAct 166
摘要 166
先决条件和限制 166
架构 167
工具 169
最佳实践 170
操作说明 170
故障排除 176
相关资源 176
其他信息 176
使用 Amazon Bedrock 开发一款基于聊天的助手 178
摘要 178
先决条件和限制 178
架构 179
工具 180
最佳实践 181
操作说明 182
相关资源 185
其他信息 186
通过语音输入记录机构知识 188
摘要 188
先决条件和限制 188
架构 189
工具 190
最佳实践 191
操作说明 191

相关资源 196
使用 Amazon Personalize 生成个性化推荐 198
摘要 198
先决条件和限制 198
架构 198
工具 199
操作说明 200
相关资源 203
其他信息 203
使用 SageMaker AI 和 Hydra 简化机器学习工作流程 207
摘要 207
先决条件和限制 207
架构 208
工具 208
最佳实践 209
操作说明 210
故障排除 215
相关资源 216
其他信息 216
训练和部署支持 GPU 的自定义机器学习模型 217
摘要 217
先决条件和限制 217
架构 217
工具 218
操作说明 218
相关资源 233
其他信息 233
..... 236
摘要 236
先决条件和限制 236
架构 239
工具 240
最佳实践 241
操作说明 241
相关资源 247
其他信息 247

使用 Amazon Q 开发者作为编码助手 252
摘要 252
先决条件和限制 252
工具 253
最佳实践 253
操作说明 253
故障排除 260
相关资源 261
使用 Processing 对 TB 级机器学习 SageMaker 数据集进行分布式特征工程 262
摘要 262
先决条件和限制 262
架构 263
工具 265
操作说明 265
相关资源 275
附件 275
使用 Flask 和 Elastic Beanstalk 查看人工智能/机器学习 (AI/ML) 模型结果 276
摘要 276
先决条件和限制 276
架构 277
工具 278
操作说明 279
相关资源 287
其他信息 287
更多模式 291
Analytics 292
在 Microsoft SQL Server Analysis Services 中分析 Amazon Redshift 数据 294
摘要 294
先决条件和限制 294
架构 294
工具 295
操作说明 295
相关资源 297
..... 298
摘要 298
先决条件和限制 298

架构 298
工具 299
操作说明 300
相关资源 304
自动将数据从 Amazon S3 AWS Data Exchange 中提取数据 305
摘要 305
先决条件和限制 305
架构 305
工具 306
操作说明 307
相关资源 308
附件 308
在 AWS Glue 中自动加密 309
摘要 309
先决条件和限制 309
架构 309
工具 310
最佳实践 311
操作说明 311
相关资源 313
使用 AWS DataOps 开发套件构建数据管道来处理 Google Analytics 数据 314
摘要 314
先决条件和限制 314
架构 314
工具 315
操作说明 316
故障排除 318
相关资源 318
其他信息 318
构建视频处理管道 321
摘要 321
先决条件和限制 321
架构 322
工具 322
操作说明 323
相关资源 329

其他信息 329
附件 330
使用 AWS Glue 构建从 Amazon S3 到 Amazon Redshift 的 ETL 管道 331
摘要 331
先决条件和限制 331
架构 332
工具 332
操作说明 333
相关资源 338
其他信息 338
使用 Amazon 构建企业数据网格 DataZone 339
摘要 339
先决条件和限制 339
架构 340
工具 341
操作说明 342
相关资源 352
其他信息 353
通过 Amazon Web Services 计算风险价值 (VaR) 355
摘要 355
先决条件和限制 355
架构 356
工具 357
最佳实践 357
操作说明 358
相关资源 360
使用 Athena 来配置对共享 AWS Glue Data Catalog 的跨账户存取 361
摘要 361
先决条件和限制 361
架构 361
工具 362
操作说明 363
相关资源 374
其他信息 374
将 NORMALIZE 转换为 Amazon Redshift SQL 375
摘要 375

先决条件和限制 375
架构 375
工具 376
操作说明 380
相关资源 381
将 RESET WHEN 转换为 Amazon Redshift SQL 382
摘要 382
先决条件和限制 382
架构 382
工具 383
操作说明 386
相关资源 386
在 AWS 上部署和管理无服务器数据湖 388
摘要 388
先决条件和限制 388
架构 388
工具 390
操作说明 391
相关资源 392
..... 394
摘要 394
先决条件和限制 394
架构 395
工具 395
操作说明 396
相关资源 399
附件 399
确保 Amazon EMR 日志记录到 Amazon S3 400
摘要 400
先决条件和限制 400
架构 401
工具 401
操作说明 402
相关资源 404
附件 404
使用 AWS Glue 生成测试数据 405

摘要 405
先决条件和限制 405
架构 405
工具 406
最佳实践 406
操作说明 407
相关资源 415
其他信息 415
将物联网数据直接摄取至 Amazon S3 420
摘要 420
先决条件和限制 420
架构 421
工具 421
最佳实践 422
操作说明 422
故障排除 428
相关资源 429
其他信息 429
使用 Lambda 函数在 Amazon EMR 中启动 Spark 作业 433
摘要 433
先决条件和限制 433
架构 433
工具 434
操作说明 435
相关资源 437
其他信息 438
附件 440
将 Apache Cassandra 工作负载迁移到 Amazon Keyspaces 441
摘要 441
先决条件和限制 441
架构 441
工具 442
最佳实践 443
操作说明 443
故障排除 455
相关资源 455

其他信息 455
使用迁移器将 Hadoop 数据迁移到 Amazon S3 WANdisco LiveData 457
摘要 457
先决条件和限制 457
架构 457
操作说明 458
相关资源 462
其他信息 463
将 Oracle 商业智能 12c 迁移到 AWS Cloud 464
摘要 464
先决条件和限制 464
架构 465
工具 466
操作说明 467
相关资源 476
其他信息 476
使用将 Kafka 集群迁移到 Amazon MSK MirrorMaker 480
摘要 480
先决条件和限制 480
架构 481
工具 481
最佳实践 482
操作说明 482
相关资源 485
其他信息 485
将 ELK 堆栈迁移至 AWS Cloud 486
摘要 486
先决条件和限制 486
架构 487
工具 489
操作说明 490
相关资源 496
其他信息 497
AWS 使用 Starburst 将数据迁移到 498
摘要 498
先决条件和限制 498

架构 498
工具 500
操作说明 500
相关资源 502
优化输入文件大小的 ETL 摄取 504
摘要 504
先决条件和限制 504
架构 504
工具 505
操作说明 505
相关资源 507
其他信息 508
使用 AWS Step Functions 编排 ETL 管道 509
摘要 509
先决条件和限制 509
架构 510
工具 510
操作说明 512
故障排除 517
相关资源 517
其他信息 517
使用 Amazon Redshift ML 执行 ML 分析 518
摘要 518
先决条件和限制 518
架构 519
工具 519
操作说明 520
相关资源 523
使用亚马逊 Athena 查询亚马逊 DynamoDB 525
摘要 525
先决条件和限制 525
架构 525
工具 526
操作说明 526
故障排除 529
相关资源 529

使用 Athena 查询 DynamoDB 表 530
摘要 530
先决条件和限制 530
架构 530
工具 531
操作说明 531
相关资源 539
其他信息 539
设置最小可行数据空间 541
摘要 541
先决条件和限制 542
架构 543
工具 543
最佳实践 544
操作说明 545
故障排除 588
相关资源 588
其他信息 588
为 Amazon Redshift 查询结果设置特定语言的排序 592
摘要 592
先决条件和限制 592
架构 592
工具 593
操作说明 593
相关资源 597
其他信息 598
使用 Lambda 函数订阅来自跨区域 S3 存储桶的事件通知 602
摘要 602
先决条件和限制 602
架构 602
工具 603
操作说明 604
相关资源 607
三种用于转换数据的 AWS Glue 作业类型 608
摘要 608
先决条件和限制 608

架构 608
工具 609
操作说明 610
相关资源 612
其他信息 612
附件 618
使用 Athena 可视化 Amazon Redshift 审计日志 QuickSight 619
摘要 619
先决条件和限制 619
架构 619
工具 620
操作说明 620
相关资源 623
附件 623
使用 Amazon 可视化 IAM 凭证报告 QuickSight 624
摘要 624
先决条件和限制 624
架构 625
工具 625
操作说明 626
其他信息 631
更多模式 632
计算 634
容器和微服务 635
从亚马逊 EKS 访问亚马逊 Neptune 637
在 Amazon ECS 上访问容器应用程序 648
使用 AWS Fargate 启动类型访问 Amazon ECS 上的容器应用程序 661
在 Amazon EKS 上私下访问容器应用程序 672
在 Amazon EKS 上的 App Mesh 中激活 mTLS 678
Amazon RDS for PostgreSQL 数据库实例的自动备份 685
自动部署 Node Termination Handler 696
自动构建 Java 应用程序并将其部署到 Amazon EKS 709
使用 Amazon EFS 在 EC2 实例上创建 Amazon ECS 任务定义 729
使用容器映像部署 Lambda 函数 734
使用 AWS Fargate 在 Amazon ECS 上部署 Java 微服务 741
使用 Amazon EKS 和 Helm 部署 Kubernetes 软件包 746

在 Amazon EKS 上部署 Java 微服务并使用应用程序负载均衡器将其公开 755
使用 AWS Copilot 将集群应用程序部署至 Amazon ECS 766
在 Amazon EKS 上部署基于 gRPC 的应用程序 775
部署和调试 Amazon EKS 集群 786
使用 Elastic Beanstalk 部署容器 813
使用 Lambda 和 Amazon VPC 生成静态出站 IP 地址 818
自动识别重复的容器镜像 829
在 Amazon EKS Worker 节点上安装 SSM Agent 855
使用在 Amazon EKS 工作节点上安装 SSM CloudWatch 代理和代理 preBootstrapCommands 860
迁移适用于 EKS 自动模式的 NGINX 入口控制器 866
将容器工作负载从 ARO 迁移到 ROSA 884
优化生成的 Docker 映像 896
将 Kubernetes 容器组（pod）放置在 Amazon EKS 中的兼容节点上 904
跨账户或区域复制已筛选的 Amazon ECR 容器映像 919
在不重启容器的情况下轮换凭证 934
在亚马逊上运行 Amazon ECS 任务 WorkSpaces 939
在 AWS 上运行 ASP.NET Web API Docker 容器 949
使用 AWS Fargate 运行消息驱动型工作负载 960
使用持久数据存储运行有状态工作负载 967
使用 Amazon EKS Pod Identity 和 KEDA 设置自动缩放 988
..... 1008
在 Amazon ECS 中使用双向 TLS 和 ALB 1027
更多模式 1052
无服务器 1054
使用 AWS Amplify 构建 React Native 应用程序 1055
跨多个 SaaS 产品集中管理租户 1072
创建跨账户 Amazon 连接 EventBridge 1082
使用 Kinesis Data Streams 和 Firehose 将 DynamoDB 记录传送到亚马逊 S3 1094
使用 API Gateway 实现基于路径的 API 版本控制 1100
将 psycopg2 库导入 AWS Lambda 1110
将 API Gateway 与亚马逊 SQS 集成 1119
使用 AWS L APIs ambda 进行异步处理 1132
使用 Amazon DynamoDB 直播进行 APIs 异步处理 1140
使用 Amazon S APIs QS 进行异步处理 1149
从 Step Functions 同步运行 Systems Manager 自动化任务 1157

使用并行读取 S3 对象 AWS Lambda 1169
将遥测数据从 Lambda 发送到 OpenSearch 1180
设置无服务器蜂窝路由器 1192
设置对 Amazon S3 存储桶的私有访问权限 1208
解决 Step Functions 状态的问题 1214
使用无服务器方法将 Amazon Web Services 串在一起 1223
更多模式 1228
网络连接 1230
AWS Transit Gateway 自动对等互连 1231
使用 AWS Transit Gateway 集中网络连接 1236
使用应用程序负载均衡器为 Oracle JD Edward EnterpriseOne s 配置 HTTPS 加密 1241
通过私有网络连接到 Application Migration Service 数据和控制面板 1251
使用 AWS CloudFormation 自定义资源创建 Infoblox 对象 1262
为 Network Firewall 自定义 CloudWatch 警报 1274
使用 Terraform 在 Wavelength 区域部署资源 1290
将 DNS 记录批量迁移至 Route 53 私有托管区 1297
在 AWS 上从 F5 迁移到应用程序负载均衡器时修改 HTTP 标头 1306
从多个私密访问 AWS 服务终端节点 VPCs 1311
多份报告网络访问分析器的发现结果 AWS 账户 1319
在多账户 AWS 环境中为混合网络设置 DNS 解析 1341
..... 1353
使用 Splunk 查看 AWS Network Firewall 日志和指标 1358
更多模式 1367
内容分发 1368
使用 Firehose 向 Splunk 发送 AWS WAF 日志 1369
使用以下方法通过 VPC 在 S3 存储桶中提供静态内容 CloudFront 1376
更多模式 1383
数据库和存储 1384
数据库 1385
使用链接服务器访问本地 SQL Server 数据 1387
在 AWS PeopleSoft 上为 Oracle 添加 HA 1392
评测将 SQL Server 数据库迁移至 MongoDB Atlas on AWS 的查询性能 1414
使用 iaC 自动部署 Aurora 全球数据库的蓝/绿 1422
使用 AWS Lambda 和任务计划程序自动执行数据库任务 1448
使用 DR Orchestrator 框架自动执行故障转移和故障恢复 1457
在跨区域自动复制 Amazon RDS 实例 AWS 账户 1482

自动备份 SAP HANA 数据库 1495
为 DynamoDB 自动生成 PynamoDB 模型和 CRUD 函数 1502
阻止对 Amazon RDS 的公有访问 1509
配置跨账户的 Amazon DynamoDB 访问 1515
在“始终打开”可用性组中配置只读路由 1527
在 pgAdmin 中使用 SSH 隧道进行连接 1534
将 JSON Oracle 查询转换至 PostgreSQL 数据库 SQL 1538
跨账户复制 Amazon DynamoDB 表 1567
跨账户复制 Amazon DynamoDB 表 1572
为 Amazon RDS 和 Amazon Aurora 创建成本和使用情况报告 1582
使用 Aurora PostgreSQL 模拟 Oracle RAC 工作负载 1588
为 PostgreSQL 数据库实例启用加密连接 1593
加密现有 Amazon RDS for PostgreSQL 数据库实例 1600
在启动时强制对 Amazon RDS 数据库执行自动标记 1606
估算 DynamoDB 成本 1610
估算 Amazon DynamoDB 表的存储成本 1619
使用 AWR 报告估计 Oracle 数据库的 Amazon RDS 引擎大小 1623
将 Amazon RDS for SQL Server 表导出至 S3 存储桶 1653
处理动态 SQL 语句中的匿名块 1662
在 Aurora PostgreSQL 兼容中处理重载的 Oracle 函数 1669
帮助强制执行 DynamoDB 标签 1675
实施跨区域灾转移 1680
将含有 100 多个参数的 Oracle 函数迁移到 PostgreSQL 1693
将 Amazon RDS for Oracle 数据库实例迁移到 AMS 账户 1698
将 Oracle OUT 绑定变量迁移到 PostgreSQL 1706
使用 HSR 将 SAP HANA 迁移至 AWS 1714
使用分布式可用性组将 SQL Server 迁移至 AWS 1724
使用和 AWS DMS 从 Oracle 8i 或 9i 迁移到适用于 Oracle 的 Amazon RD SharePlex S 1733
监控 Amazon Aurora 的加密 1740
使用 Amazon 监控 GoldenGate 日志 CloudWatch 1745
将 Oracle 数据库 EE 平台改为适用于 Oracle 的 Amazon RDS SE2 1757
使用 Precission Connect 将大型机数据库复制到 AWS 1766
计划适用于 Amazon RDS 和 Aurora PostgreSQL 的任务 1779
使用本地 SMTP 服务器发送 RDS for SQL Server 通知 1784
为 IBM Db2 on AWS 上的 SAP 设置灾难恢复 1795
使用 Terraform 为数据库迁移设置 CI/CD 管道 1811

为 Amazon RDS Custom 上的 Oracle 电子商务套件设置 HA/DR 架构 1818
在适用于 MySQL 的 RDS 和亚马逊上的 MySQL 之间设置数据复制 EC2 1826
Oracle PeopleSoft 应用程序的过渡角色 1832
跨账户从亚马逊 Redshift 卸载数据 1863
按照工作负载分类的数据库迁移模式 1887
更多模式 1898
存储和备份 1903
允许 EC2 实例在 AMS 中对 S3 存储桶进行写入访问 1904
自动将数据流摄入至 Snowflake 数据库 1909
自动加密 EBS 卷 1919
在 AWS 上的 Charon-SSP 仿真器中备份 Sun SPARC 服务器 1928
使用 Veeam 备份数据并将其存档至 Amazon S3 1946
在 AWS 上 NetBackup 为 VMware Cloud 进行配置 1965
使用 AWS CLI 在账户和区域之间复制 S3 对象 1971
使用 S3 批量复制在账户和区域之间复制 S3 对象 1984
使用 PrivateLink 适用于亚马逊 S3 的 DistCp AWS 将 Hadoop 数据迁移到亚马逊 S3 1995
更多模式 2008
开发人员工具 2010
DevOps 2011
实现 AWS 基础设施运营自动化 2014
自动 CloudFront 更新负载平衡器端点的更改 2022
自动 CodeGuru 审查 AWS CDK Python 应用程序 2029
自动执行 AWS 资源评测 2039
自动安装 SAP 系统 2051
使用 AWS CDK 自动部署 Service Catalog 产品组合与产品 2061
自动从 AWS 备份 CodeCommit 到 Amazon S3 2076
自动删除堆 CloudFormation 栈 2083
自动进行动态管道管理以部署修补程序解决方案 2093
自动获取 Amazon MWAA 自定义指标 2117
使用和自动管理 Amazon Lex 资源 AWS CDK CloudFormation 2132
使用 AWS CodePipeline 和 AWS 自动部署堆栈集 CodeBuild 2145
自动将 Systems Manager 的托管策略附加到 EC2 实例配置文件 2170
自动为微服务构建 CI/CD 管道与 Amazon ECS 集群 2184
使用微服务构建松耦合架构 2194
构建 Docker 镜像并将其推送到亚马逊 ECR 2204
使用 Amazon Web Services 构建与测试 iOS 应用程序 2210

使用规则包查看 AWS CDK 应用程序或 CloudFormation 模板以了解最佳实践 2215
为在 Amazon EKS 上的应用程序配置双向 TLS 2218
使用 AWS 创建 AppStream 2.0 资源 CloudFormation 2228
使用 Firelens 为 Amazon ECS 创建自定义日志解析器 2233
使用和 P HashiCorp acker 创建管道 CodePipeline 和 AMI 2240
使用创建管道并将更新部署到本地 EC2 实例 CodePipeline 2246
为 Java 和 Python 项目创建动态 CI 管道 2253
部署 Synt CloudWatch hetics 加那利群岛 2266
在 Amazon ECS 上部署 Java 微服务 CI/CD 管道 2272
部署 ChatOps 解决方案来管理 SAST 扫描结果 2280
使用 AWS Network Firewall 和 AWS Transit Gateway 部署防火墙 2294
..... 2304
使用实例配置文件从 AWS Cloud9 部署 Amazon EKS EC2 集群 2308
在多个 AWS 区域部署代码 2318
执行 Amazon Redshift SQL 查询 2328
将 AWS Backup 报告导出为 CSV 格式文件 2339
将亚马逊 EC2 实例标签导出到 CSV 文件 2346
生成包含 AWS Config 托管规则的 AWS CloudFormation 模板 2351
为 SageMaker 笔记本实例提供跨账户访问存储库的 CodeCommit 权限 2357
实施 GitHub Flow 分支策略 2366
实施 Gitflow 分支策略 2373
实施中继分支策略 2383
实现集中式自定义 Checkov 扫描 2388
在检测到 monorepo 中的变化后，启动不同的 CI/CD 管道 2397
将 Bitbucket 存储库与 AWS Amplify 集成 2411
使用 Lambda 在 AWS 账户上启动 CodeBuild 项目 2418
使用应用程序恢复控制器管理 EMR 集群的多可用区故障转移 2430
管理微服务到多个账户和区域的蓝/绿部署 2442
监控 Amazon ECR 存储库的通配符权限 2474
优化多账户无服务器部署 2478
从 AWS CodeCommit 事件中执行自定义操作 2495
使用 GitHub 操作配置 AWS Service Catalog 产品 2499
配置权限最低的 IAM 角色 2508
将亚马逊 CloudWatch 指标发布到 CSV 文件 2518
移除正面 AWS 账户 的 Amazon EC2 条目 AWS Managed Microsoft AD 2523
从中移除同一个 AWS 账户 亚马逊 EC2 条目 AWS Managed Microsoft AD 2537

在中对 Python ETL 作业运行单元测试 AWS Glue 2545
在 Amazon S3 中设置 Helm v3 图表 2553
使用设置 CI/CD 管道 CodePipeline 2561
在 Amazon EKS 上为应用程序设置 end-to-end加密 2573
简化 Amazon EKS 多租户应用程序部署 2585
为多个电子邮件端点订阅 SNS 主题 2601
使用 AWS Fargate WaitCondition 钩子结构 2606
在 AWS 中使用第三方 Git 存储库 CodePipeline 2616
使用 AWS 验证 Terraform 配置 CodePipeline 2623
更多模式 2636
基础设施 2639
使用会话管理器和 Amazon Instance Connect 访问堡垒主 EC2 机 2640
使用集中化 DNS 解析 AWS Managed Microsoft AD 2654
使用 Observability Access Manager 集中监控 2663
在启动时检查 EC2 实例是否有必需的标签 2674
使用会话管理器连接到 EC2 实例 2679
在不支持 AWS 的 AWS 区域创建管道 CodePipeline 2685
使用 AWS CDK 外观和逃生舱口自定义默认角色名称 2691
在亚马逊 EC2 上使用私有静态部署一个 Cassandra 集群 IPs 2698
使用 Transit Gateway Connect 扩展 VRFs 到 AWS 2703
获取有关 AWS KMS 密钥状态变更的 Amazon SNS 通知 2716
在非工作负载子网的多账户 VPC 设计中保留可路由的 IP 空间 2722
通过代码存储库在 Service Catalog 中配置 Terraform 产品 2726
使用单个电子邮件地址注册多个 Amazon Web Services account 2742
在单账户 AWS 环境中为混合网络设置 DNS 解析 2754
在亚马逊上自动设置 UiPath RPA 机器人 EC2 2758
在 AWS 上设置高度可用的 PeopleSoft 架构 2771
为 Oracle JD Edwards 设置灾难恢复 EnterpriseOne 2793
设置漂移检测和报告 2814
成功将 S3 存储桶导入为 CloudFormation 堆栈 2819
同步不同区域的 Amazon EFS 文件系统 2829
使用测试 AWS 基础架构 LocalStack 和 Terraform 测试 2836
将 SAP Pacemaker 集群从升级到 ENSA1 ENSA2 2843
在不同账户中 VPCs 使用一致的可用区 2864
IDs 在 IAM 策略中使用用户 2869
在本地验证 Account Factory for Terraform 代码 2878

更多模式 2892
网络和移动应用程序 2895
对 React Web 应用程序用户执行身份验证 2896
持续部署 Amplify web 应用程序 2906
使用 AWS Amplify 和 Amazon Cognito 创建 React 应用程序 2913
为微前端创建门户 2927
将基于 React 的 SPA 部署到 Amazon S3 然后 CloudFront 2954
使用私有端点和应用程序负载均衡器部署 Amazon API Gateway API 2960
在本地 Angular 应用程序中嵌入亚马逊 QuickSight 控制面板 2966
使用 Green Boost 探索 Web 应用程序开发 2982
使用 AWS 运行单元测试 CodeBuild 3004
以六边形架构构建 Python 项目 3011
更多模式 3034
IoT 3036
在 IoT 环境中配置安全事件的日志记录和监控 3037
摘要 3037
先决条件和限制 3037
架构 3038
工具 3039
操作说明 3040
相关资源 3043
提取和查询 AWS IoT SiteWise 元数据属性 3044
摘要 3044
先决条件和限制 3044
架构 3044
工具 3045
操作说明 3046
相关资源 3048
其他信息 3048
..... 3051
摘要 3051
先决条件和限制 3051
架构 3052
工具 3053
最佳实践 3053
操作说明 3053

故障排除 3066
相关资源 3068
其他信息 3068
更多模式 3070
迁移和现代化 3071
迁移 3072
为 AWS DMS 创建 AWS CloudFormation 模板 3073
开始使用自动发现产品组合 3077
将 Cloudera 本地工作负载迁移到 AWS 3082
解决 SQL Server 迁移至 AWS 后的连接错误 3094
无需禁用即可自动重启 AWS 复制代理 SELinux 3097
重构 3103
更换主机 3437
重新定位 3653
更换平台 3722
按照工作负载分类的迁移模式 4233
更多模式 4241
现代化 4243
在 CAST Imaging 中分析和可视化软件架构 4244
使用 CAST Highlight 评测迁移至 AWS 之前的应用程序就绪情况 4253
自动将过期的 DynamoDB 数据归档到 Amazon S3 4270
在 Amazon 服务中构建多租户无服务器架构 OpenSearch 4285
部署多堆栈应用程序 4328
使用 AWS SAM 部署嵌套应用程序 4336
使用 AWS Lambda TVM 为 Amazon S3 实施 SaaS 租户隔离 4343
使用 AWS Step Functions 实施无服务器 saga 模式 4366
使用 Amazon ECS Anywhere 管理本地容器应用程序 4378
在 AWS 上实现 ASP.NET Web 表单应用程序的现代化 4386
使用 AWS Fargate 运行事件驱动型工作负载 4398
SaaS 架构中的租户登录 4406
使用 CQRS 和事件溯源 4428
更多模式 4448
主机 4450
通过安装，AWS 服务 从 IBM z/OS 进行访问 AWS CLI 4451
备份大型机数据并将其存档至 Amazon S3 4469
使用和构建 COBOL Db2 程序 AWS Mainframe Modernization AWS CodeBuild 4490

构建 Micro Focus Enterprise Server PAC 4510
在 Amazon Web Services Cloud 中构建大型机文件查看器 4527
对现代化改造的 Blu Age 应用程序容器化 4539
在 AWS 上将 EBCDIC 数据转换为 ASCII。 4547
使用 AWS Lambda 将大型机 EBCDIC 文件转换为 ASCII 文件 4564
转换具有复杂记录布局的大型机数据文件 4581
为容器化应用程序部署环境 4594
生成 Db2 z/OS 数据见解 4603
使用 AWS 大型机现代化和 Amazon Q 生成见解 QuickSight 4643
将 Stonebranch Universal Controller 与 AWS 集成 4656
使用 Precisely 将 VSAM 文件迁移和复制到 Amazon Web Services Cloud 4684
对大型机输出管理进行现代化改造 AWS 4699
使用 Amazon Q 实现 CardDemo 大型机应用程序的现代化 4737
对大型机批量打印工作负载进行现代化改造 AWS 4750
使用 Rocket 软件企业套件实现大型机环境的现代化 4771
在 AWS 上实现大型机在线打印工作负载的现代化 4786
使用 Transfer Family 将大型机文件移动到 Amazon S3 4812
在 Db2 联合身份验证数据库中保护用户访问 4823
将 Db2 z/OS 数据传输到 AWS 4830
更多模式 4854
管理 4855
成本管理 4856
为 AWS Glue 任务创建详细的成本和使用情况报告 4857
为 Amazon EMR 集群创建详细的成本和使用情况报告 4862
更多模式 4866
高性能计算 4867
使用 Terraform 和 DRA 部署 Lustre 文件系统 4868
为 AWS 设置一个 Grafana 监控控制面板 ParallelCluster 4875
使用 NICE DCV 设置自动扩缩 VDI 4886
混合云 4897
在 AWS 上配置数据中心扩展到 VMware Cloud 4898
将 vRealize Automation 配置为在 A VMs W VMware S 上的云端上 4902
在 AWS 上使用 C VMware loud 部署 SDDC 4910
在 AWS 上将 VMware vRealize 网络洞察与 VMware 云集成 4918
使用 HCX OSAM 迁移 VMs 到 AWS 上的 VMware 云端 4922
将日志从 VMware 云端发送 AWS 到 Splunk 4927

在 Amazon ECS Anywhere 上为混合工作负载设置 CI/CD 管线 4933
更多模式 4950
管理与治理 4951
当 Amazon Data Firehose 资源未加密时发出警报 4952
自动添加或更新 Windows 注册表项 4956
使用 Python 自动创建 RFC 4960
自动停止和启用 Amazon RDS 数据库实例 4966
使用 Terraform 在 AWS Organizations 中集中分发软件包 4978
在日志中为.NET 应用程序配置 CloudWatch 日志记录 4988
在 Amazon Web Services account 和区域之间复制 AWS Service Catalog 产品 4995
为云运营创建 RACI 矩阵 5003
使用为自定义指标创建警报 CloudWatch 5007
使用默认加密的 EBS 卷创建 AWS Cloud9 IDE 5012
自动创建基于标签的 CloudWatch 仪表板 5017
记录你的着陆区设计 5024
使用 AWS CDK 在整个组织中启用 Amazon DevOps Guru 5027
使用引导流水线实现 AFT 5048
管理多个 Amazon Web Services account 和区域中的 AWS Service Catalog 产品 5068
将 Amazon Web Services account 从 AWS Organizations 迁移至 AWS Control Tower 5075
监控 SAP RHEL Pacemaker 集群 5086
监控 AMI 的使用情况 AWS 账户 5101
在 AWS Organizations 中设置程序账户关闭警报 5114
查看您的 Amazon Web Services account 或组织 EBS 快照详情 5121
更多模式 5127
消息和通信 5129
在 Amazon MQ 中自动化 RabbitMQ 配置 5130
提高 Amazon Connect 联系中心的座席工作站的通话质量 5136
更多模式 5147
安全、身份和合规性 5148
使用 Amazon Cognito AWS 服务 从 ASP.NET 进行访问 5151
摘要 5151
先决条件和限制 5151
架构 5152
工具 5152
操作说明 5153
故障排除 5156

相关资源 5156
附件 5157
使用 AWS Directory Service 对 SQL Server 进行身份验证 5158
摘要 5158
先决条件和限制 5158
架构 5158
工具 5159
操作说明 5159
相关的资源 5162
自动执行事件响应和取证 5163
摘要 5163
先决条件和限制 5163
架构 5164
工具 5166
操作说明 5167
相关资源 5169
其他信息 5170
附件 5170
自动修复 Security Hub 标准调查发现 5171
摘要 5171
先决条件和限制 5171
架构 5172
工具 5172
最佳实践 5173
操作说明 5173
相关资源 5175
附件 5176
使用 Amazon Inspector 自动执行跨账户工作负载的安全扫描 5177
摘要 5177
先决条件和限制 5177
架构 5178
工具 5178
操作说明 5179
相关资源 5182
附件 5182
自动审核来自公有 IP 地址的访问权限 5183

摘要 5183
先决条件和限制 5183
架构 5184
工具 5185
最佳实践 5186
操作说明 5186
故障排除 5190
相关资源 5190
CloudTrail 使用安全最佳实践自动重新启用 AWS 5191
摘要 5191
先决条件和限制 5191
架构 5192
工具 5192
操作说明 5193
相关资源 5197
附件 5198
自动修复未加密的 Amazon RDS 数据库实例和集群 5199
摘要 5199
先决条件和限制 5199
架构 5200
工具 5200
最佳实践 5201
操作说明 5202
相关资源 5206
其他信息 5207
自动轮换 IAM 用户访问密钥 5208
摘要 5208
先决条件和限制 5209
架构 5209
工具 5211
操作说明 5212
相关资源 5221
在 Amazon Web Services account 中自动验证和部署 IAM policy 与角色 5222
摘要 5222
先决条件和限制 5222
架构 5223

工具 5224
操作说明 5224
相关资源 5228
双向集成 Security Hub 与 Jira 5229
摘要 5229
先决条件和限制 5229
架构 5230
工具 5231
操作说明 5232
相关资源 5239
其他信息 5240
为经过强化的容器映像构建管线 5242
摘要 5242
先决条件和限制 5242
架构 5243
工具 5245
操作说明 5246
故障排除 5252
相关资源 5253
使用 Terraform 在 AWS Organizations 中集中管理 IAM 访问密钥 5254
摘要 5254
先决条件和限制 5254
架构 5255
工具 5256
最佳实践 5257
操作说明 5257
故障排除 5264
相关资源 5265
查看亚马逊 CloudFront 分配的访问日志、HTTPS 和 TLS 版本 5266
摘要 5266
先决条件和限制 5266
架构 5267
工具 5267
操作说明 5268
相关资源 5270
附件 5270

检查和的安全组入口规则中是否有单主机网络条目 IPv4 IPv6 5271
摘要 5271
先决条件和限制 5271
架构 5271
工具 5272
操作说明 5272
相关资源 5275
附件 5275
选择 Amazon Cognito 身份验证流程 5276
摘要 5276
先决条件和限制 5276
架构 5276
工具 5280
操作说明 5281
相关资源 5283
其他信息 5284
使用 Guard 创建 AWS Config 自定义规则 5285
摘要 5285
先决条件和限制 5285
架构 5286
工具 5290
操作说明 5291
故障排除 5292
相关资源 5293
创建一份关于多个 Prowler 发现的报告 AWS 账户 5294
摘要 5294
先决条件和限制 5294
架构 5295
工具 5296
操作说明 5297
故障排除 5314
相关资源 5314
其他信息 5314
使用删除未使用的 Amazon EBS 卷 AWS Config 5317
摘要 5317
先决条件和限制 5317

架构 5317
工具 5318
操作说明 5318
故障排除 5321
相关资源 5321
使用部署 AWS Control Tower 控件 AWS CDK 5322
摘要 5322
先决条件和限制 5322
架构 5324
工具 5324
最佳实践 5325
操作说明 5325
相关资源 5331
其他信息 5332
使用 Terraform 部署 AWS Control Tower 控件 5335
摘要 5335
先决条件和限制 5335
架构 5336
工具 5337
最佳实践 5338
操作说明 5338
故障排除 5342
相关资源 5343
其他信息 5344
部署可检测代码中安全问题的管道 5346
摘要 5346
先决条件和限制 5346
架构 5346
工具 5347
操作说明 5348
故障排除 5350
相关资源 5350
其他信息 5350
为公共子网部署侦探控制 5353
摘要 5353
先决条件和限制 5353

架构 5354
工具 5355
最佳实践 5355
操作说明 5355
相关资源 5363
其他信息 5363
为公有子网部署预防性控制 5366
摘要 5366
先决条件和限制 5366
架构 5367
工具 5368
操作说明 5368
相关资源 5372
其他信息 5373
使用 Terraform 为 AWS WAF 解决方案部署安全自动化 5375
摘要 5375
先决条件和限制 5375
架构 5375
工具 5376
最佳实践 5376
操作说明 5377
故障排除 5379
相关资源 5380
检测 CA 证书即将过期的 Amazon RDS 实例 5381
摘要 5381
先决条件和限制 5381
架构 5382
工具 5383
最佳实践 5384
操作说明 5384
故障排除 5387
相关资源 5388
通过 IAM Access Analyzer 动态生成 IAM policy 5389
摘要 5389
先决条件和限制 5389
架构 5390

工具 5391
操作说明 5392
相关资源 5397
启用 GuardDuty 使用 CloudFormation 模板 5398
摘要 5398
先决条件和限制 5398
架构 5398
工具 5399
操作说明 5399
相关资源 5401
其他信息 5401
在 Amazon RDS for SQL Server 中启用透明数据加密 5405
摘要 5405
先决条件和限制 5405
架构 5405
工具 5406
操作说明 5406
相关资源 5408
确保 AWS 负载均衡器使用安全侦听器协议 5409
摘要 5409
先决条件和限制 5409
架构 5410
工具 5410
最佳实践 5410
操作说明 5411
故障排除 5413
相关资源 5413
附件 5414
确保对 Amazon EMR 静态数据加密 5415
摘要 5415
先决条件和限制 5415
架构 5416
工具 5416
操作说明 5417
相关资源 5419
附件 5419

确保 IAM 配置文件与 EC2 实例关联 5420
摘要 5420
先决条件和限制 5420
架构 5421
工具 5421
操作说明 5422
相关资源 5423
附件 5424
确保新 Amazon Redshift 集群已加密 5425
摘要 5425
先决条件和限制 5425
架构 5425
工具 5426
操作说明 5426
相关资源 5428
附件 5429
导出 IAM 身份中心身份及其分配报告 5430
摘要 5430
先决条件和限制 5430
架构 5431
工具 5432
操作说明 5432
故障排除 5433
相关资源 5434
其他信息 5435
帮助防止计划的 KMS 密钥删除 5437
摘要 5437
先决条件和限制 5437
架构 5438
工具 5438
操作说明 5439
相关资源 5442
其他信息 5442
附件 5443
识别公共的 Amazon S3 存储桶 AWS Organizations 5444
摘要 5444

先决条件和限制 5444
架构 5444
工具 5445
操作说明 5446
故障排除 5449
相关资源 5449
其他信息 5449
将 AWS 安全日志提取到微软 Sentinel 5451
摘要 5451
先决条件和限制 5451
架构 5452
工具 5453
最佳实践 5454
操作说明 5454
相关资源 5462
使用管理 IAM 身份中心权限集 CodePipeline 5463
摘要 5463
先决条件和限制 5463
架构 5464
工具 5465
最佳实践 5466
操作说明 5466
故障排除 5474
相关资源 5475
使用 AWS Secrets Manager 管理凭证 5476
摘要 5476
先决条件和限制 5476
架构 5476
工具 5477
操作说明 5477
相关资源 5478
其他信息 5478
..... 5481
摘要 5481
先决条件和限制 5481
架构 5482

工具 5482
操作说明 5483
相关资源 5485
附件 5485
在启动时监控 Amazon EMR 集群的传输中加密 5486
摘要 5486
先决条件和限制 5486
架构 5487
工具 5487
操作说明 5488
相关资源 5490
附件 5490
监控 Amazon ElastiCache 集群的静态加密 5491
摘要 5491
先决条件和限制 5491
架构 5492
工具 5492
操作说明 5493
相关资源 5495
附件 5495
监控 EC2 实例密钥对 5496
摘要 5496
先决条件和限制 5496
架构 5496
工具 5497
操作说明 5498
相关资源 5500
附件 5501
监控 IAM 根用户活动 5502
摘要 5502
先决条件和限制 5502
架构 5503
工具 5503
操作说明 5504
相关资源 5508
其他信息 5508

创建 IAM 用户时发出通知 5509
摘要 5509
先决条件和限制 5509
架构 5509
工具 5510
操作说明 5511
相关资源 5512
附件 5513
使用 SCP 阻止互联网访问 5514
摘要 5514
先决条件和限制 5514
工具 5515
最佳实践 5515
操作说明 5515
相关资源 5517
根据 IP 地址或地理位置限制访问 5518
摘要 5518
先决条件和限制 5518
工具 5519
操作说明 5520
相关资源 5524
扫描 Git 存储库中的获取敏感信息 5525
摘要 5525
先决条件和限制 5525
架构 5525
工具 5525
最佳实践 5525
操作说明 5526
相关资源 5530
将警报从 AWS Network Firewall 发送到 Slack 通道 5531
摘要 5531
先决条件和限制 5531
架构 5532
工具 5532
操作说明 5533
相关资源 5539

其他信息 5539
使用 AWS Private CA 和 AWS RAM 简化私有证书管理 5543
摘要 5543
先决条件和限制 5543
架构 5544
工具 5544
操作说明 5545
相关资源 5551
其他信息 5551
关闭所有 Security Hub 成员账户的安全标准控制 5552
摘要 5552
先决条件和限制 5552
架构 5552
工具 5553
操作说明 5554
相关资源 5558
使用从 IAM 身份中心更新 AWS CLI 证书 PowerShell 5559
摘要 5559
先决条件和限制 5559
架构 5560
工具 5560
最佳实践 5560
操作说明 5561
故障排除 5563
相关资源 5563
其他信息 5563
使用 AWS Config 监控 Amazon Redshift 5566
摘要 5566
先决条件和限制 5566
架构 5566
工具 5567
操作说明 5568
相关资源 5570
其他信息 5571
使用网络防火墙从出站网络流量中捕获 DNS 域名 5572
摘要 5572

先决条件和限制 5572
架构 5572
工具 5573
操作说明 5574
使用 Terraform 自动启用 GuardDuty 5587
摘要 5587
先决条件和限制 5587
架构 5589
工具 5590
操作说明 5591
相关资源 5598
其他信息 5598
验证 PCI DSS 4.0 的最佳实践 5600
摘要 5600
先决条件和限制 5600
工具 5601
操作说明 5602
相关资源 5603
其他信息 5604
附件 5604
..... 5605
摘要 5605
先决条件和限制 5605
架构 5605
工具 5606
操作说明 5607
相关资源 5609
附件 5609
..... 5610
摘要 5610
先决条件和限制 5610
架构 5610
工具 5611
操作说明 5612
相关资源 5614
附件 5614

更多模式 5615
..... 5617

AWS 规范性指导模式

Amazon Web Services (AWS) 规范性指导模式为实施特定的云迁移、现代化和部署场景提供了 step-by-step说明、架构、工具和代码。这些模式由主题专家审查 AWS，适用于计划或正在迁移到 AWS的建筑商和动手用户。它们还为已经上线 AWS 并正在寻找优化或现代化云运营的方法的用户提供支持。

无论您处于项目的概念验证、规划还是实施阶段，您都可以使用这些模式将不同复杂性的本地或云工作负载迁移到云端并加快云的采用、优化和现代化工作。 AWS 例如，对于云迁移项目：

- 在规划阶段，您可评估可供迁移至 AWS的不同选项。您可以选择适合自己需求的正确模式，具体取决于您的目的是重新定位、更换主机、更换平台还是重新架构。您还可了解可用于迁移的不同工具，并开始计划购买许可证或开始与供应商进行初步对话。
- 在概念验证和实施阶段，您可以按照模式中提供的 step-by-step说明将工作负载迁移到 AWS。每种模式都包括先决条件、目标参考架构、工具、 step-by-step任务、最佳实践、故障排除和代码等详细信息。
- 如果您已经在使用 AWS Cloud，则可以找到有助于实现云资源使用现代化、优化、扩展和保护其使用的模式。

若要按技术领域查看模式列表，请使用以下链接或AWS 规范指南主页 上的筛选和搜索选项。

- 云基础
- 人工智能和机器学习
- 分析
- 计算
- 数据库和存储
- 开发者工具
- IoT
- 迁移和现代化
- 管理
- 安全、身份和合规性

要查看所有出版物，包括指南、策略和模式，请参阅AWS 规范性指导主页。

云基础

主题

- 使用着陆区域加速器自动创建账户 AWS
- 自动清点多个账户和地区的 AWS 资源
- 配置 VPC 流日志以实现对 Amazon Web Services account 的集中管理
- 使用 AWS Organizations 自动标记中转网关连接
- 更多模式

使用着陆区域加速器自动创建账户 AWS

由贾斯汀·库斯科夫斯基（AWS）、乔·贝伦斯（AWS）和内森·斯科特（AWS）创作

摘要

此模式说明了如何在授权用户提交请求 AWS 账户 时使用 AWS解决方案上的着陆区加速器自动部署新的解决方案。它 AWS Step Functions 用来编排许多 AWS Lambda 函数。Lambda 函数将账户信息添加到 Git 存储库，启动 AWS CodePipeline 管道，并验证是否已配置必要的 AWS 资源。该过程完成后，用户会收到一条通知，告知账户已创建。

或者，你可以在帐户创建过程中集成 Microsoft Entra ID 组并分配 AWS IAM Identity Center 权限集。如果您的组织使用 Microsoft Entra ID 作为身份源，则此可选功能可帮助您自动管理和配置对新帐户的访问权限。

先决条件和限制

先决条件

- 访问中的管理账户 AWS Organizations
- AWS Cloud Development Kit (AWS CDK) 版本 2.118.0 或更高版本，已安装并配置
- Python 版本 3.9 或更高版本，已安装
- AWS Command Line Interface (AWS CLI) 版本 2.13.19 或更高版本，已安装
- Docker 版本 24.0.6 或更高版本，已安装
- AWS 解决方案上的着陆区域加速器，部署在管理账户中
- （可选）微软 Entra ID 和 IAM 身份中心，已集成

限制

账户创建工作流程支持按顺序执行以部署单个 AWS 账户。此限制可确保账户创建工作流程已成功完成，而无需在并行运行期间争夺资源。

架构

目标架构

下图显示了使用着陆区加速器自动创建新 AWS 账户 版本的高级架构。 AWS AWS Step Functions 协调自动化。Step Functions 工作流程中的每项任务都由一个或多个 AWS Lambda 函数执行。

图表显示了以下工作流：

1. 用户通过运行 Python 脚本或使用 Amazon API Gateway 来申请账户。
2. 账户创建 Orchestrator 工作流程始于。 AWS Step Functions
3. 该工作流程更新源代码存储库中的`account-config.yaml`文件。它还会在 AWS 管道上启动着陆区加速器并检查管道的状态。此管道创建并设置了新账户。有关其工作原理的更多信息，请参阅上的 “着陆区加速器” 架构概述 AWS。
4. （可选）管道完成后，工作流将检查 Microsoft Entra ID 中是否存在该群组。如果 Microsoft Entra ID 中不存在该群组，则该工作流程会将该群组添加到 Microsoft Entra ID 中。
5. 该工作流程会执行 AWS 解决方案上的 “着陆区域加速器” 无法执行的其他步骤。默认步骤包括：
   - 在 AWS Identity and Access Management (IAM) 中创建账户别名
   - 在中为账户添加标签 AWS Organizations
   - 根据分配给账户的标签在 P AWS Systems Manager arameter Store 中创建参数
6. （可选）该工作流程会将一个或多个权限集分配给你之前指定的 Microsoft Entra ID 组。权限集允许群组中的用户访问新帐户并允许他们执行您配置的操作。
7. AWS Lambda 函数运行 QA 和验证测试。它会验证资源创建，检查标签是否已创建，并验证是否已部署安全资源。
8. 该工作流程释放账户并使用亚马逊简单电子邮件服务 (Amazon SES) 通知用户该流程已成功完成。

有关 Step Functions 工作流程的更多信息，请参阅此模式的 “其他信息” 部分中的 Step Functions 工作流程图。

微软 Entra ID 应用程序

如果你选择与 Microsoft Entra ID 集成，则在部署此模式时将创建以下两个应用程序：

- 链接到 IAM 身份中心并确保 Microsoft Entra ID 组在 IAM 身份中心中可用的应用程序。在本示例中，这个 Microsoft Entra ID 应用程序命名为`LZA2`。
- 允许 Lambda 函数与微软 Entra ID 通信并调用微软 Graph 的应用程序。 APIs在这种模式中，此应用程序被命名为`create_aws_account`。

这些应用程序收集的数据用于同步 Microsoft Entra ID 组和分配权限集。

工具

AWS 服务

- Amazon API Gateway 可帮助您创建、发布、维护、监控和保护任何规模的 RES WebSocket APIs T、HTTP。在此模式中，您可以使用 API Gateway 来检查 AWS 账户 名称的可用性、启动 AWS Step Functions 工作流程以及检查 Step Functions 的执行状态。
- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，可帮助您在代码中定义和配置 AWS Cloud 基础架构。
- AWS Control Tower按照规范性最佳实践，帮助您设置和管理 AWS 多账户环境。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如， AWS Lambda 函数、使用 API 目的地的 HTTP 调用端点或其他 AWS 账户目的地的事件总线。此解决方案使用一条EventBridge 规则，即在 Step Functions 工作流程状态更改为`FailedTimed-out`、或时启动 Lambda 函数。`Aborted`
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS IAM Identity Center帮助您集中管理对所有应用程序 AWS 账户 和云应用程序的单点登录 (SSO) 访问权限。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥以帮助保护您的数据。在这种模式中， AWS KMS 密钥用于加密数据，例如存储在亚马逊简单存储服务 (Amazon S3) Simple S3 中的数据、Lambda 环境变量和 Step Functions 中的数据。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Organizations是一项账户管理服务，可帮助您将多个账户整合 AWS 账户 到一个由您创建和集中管理的组织中。
- Amazon Simple Email Service (Amazon SES) 帮助您通过使用您自己的电子邮件地址和域发送和接收电子邮件。成功创建新账户后，您将通过 Amazon SES 收到通知。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。如果账户创建过程中出现错误，Amazon SNS 会向您配置的电子邮件地址发送通知。
- AWS Step Functions是一项无服务器编排服务，可帮助您组合 AWS Lambda 功能和其他功能 AWS 服务 来构建关键业务应用程序。
- AWS Systems Manager Parameter Store 为配置数据管理和密钥管理提供安全的分层存储。

其他工具

- awscurl 可自动执行 AWS API 请求的签名，并帮助您以标准 curl 命令的形式发出请求。
- 微软 Entra ID，前身为 Azure Active Directory，是一项基于云的身份和访问管理服务。
- Microsoft Graph 可 APIs帮助你访问微软云服务（例如微软 Entra 和微软 365）中的数据和情报。

代码存储库

此模式的代码可在 GitHub lza-account-creation-workflow存储库中找到。

lambda_layer 目录包含以下层，这些层在多个 Lambda 函数中被引用：

- account_creation_helper — 此层包括用于担任角色和检查进度的模块。 AWS Service Catalog
- boto3 — 此层包含适用于 Python (Boto3) 的 AWS SDK模块，以确保该模块 AWS Lambda 具有最新版本。
- identity_center_helper — 此层支持对 IAM 身份中心的调用。

lambda_src 目录包含以下 Lambda 函数：

- AccountTagToSsmParameter— 此函数使用附加到账户的标签 AWS Organizations 在 Parameter Store 中创建参数。每个参数都以`/account/tags/`前缀开头。
- AttachPermissionSet— 此函数向 IAM 身份中心群组添加权限集。
- Azure ADGroup 同步 — 此功能将目标 Microsoft Entra ID 组同步到 IAM 身份中心。
- CheckForRunningProcesses— 此函数检查`AWSAccelerator-Pipeline`管道当前是否正在运行。如果管道正在运行，则该函数会延迟 AWS Step Functions 工作流程。
- CreateAccount— 此函数使用 AWS Service Catalog 和 AWS Control Tower 来创建新的 AWS 账户。
- CreateAdditionalResources— 此功能可创建不由着陆区加速器管理的 AWS 资源 AWS CloudFormation，例如账户别名和 AWS Service Catalog 标签。
- GetAccountStatus— 此功能在中扫描已配置的产品 AWS Service Catalog ，以确定账户创建过程是否已完成。
- GetExecutionStatus— 此函数检索正在运行或已完成的 AWS Step Functions 执行的状态。
- NameAvailability— 此函数检查中是否已存在 AWS 账户 名称 AWS Organizations。
- ReturnResponse— 如果账户创建成功，则此函数返回新账户的 ID。如果账户创建失败，则会返回一条错误消息。

- RunStepFunction— 此函数运行创建账户 AWS Step Functions 的工作流程。
- SendEmailWithSES — 此功能向正在等待账户创建完成的用户发送电子邮件。
- 验证 ADGroup SyncTo SSO — 此功能检查指定的 Microsoft Entra ID 组是否与 IAM 身份中心同步。
- ValidateResources— 此函数验证所有 AWS Control Tower 自定义设置是否已成功运行。

最佳实践

我们建议使用以下命名惯例 AWS CDK：

- 所有参数都以p前缀开头。
- 所有条件都以c前缀开头。
- 使用r前缀启动所有资源。
- 以o前缀开始所有输出。

操作说明

部署 IAM 角色以进行验证和标记

Task	描述	所需技能
准备开启着陆区域加速器 AWS 以进行自定义。	1. 在 AWS 代码存储库的着陆区域加速器中，创建一个名为的文件`customizations-config.yaml`。您可以使用此文件来定义核心解决方案的自定义设置。有关更多信息，请参阅自定义解决方案。 2. 在`customizations-config.yaml`文件中，创建一个名为的部分`cloudFormationStacks`。	AWS DevOps

Task	描述	所需技能
准备部署`lza-account-creation-validation` 角色。	现在，您可以自定义解决方案，以便在除管理账户之外的所有账户中部署 `lza-account-creation-validation` IAM 角色。此角色为 `ValidateResources` Lambda 函数提供了对新账户的只读访问权限。 1. 从中下载 account-creation-validation-role.yaml 文件。GitHub 2. 将文件保存到文件模板部分所`customizations-config.yaml` 示的位置。 3. 打开 `customizations-config.yaml` 文件。 4. 在该`cloudFormationStacks` 部分中，添加以下代码。根据需要更新您的 AWS 区域 landing zone 的目标： `- deploymentTargets: organizationalUnits: - Root excludedAccounts: - Management description: IAM Role to allow Account Validation`	AWS DevOps

Task	描述	所需技能

	<pre>name: lza-account-creation-validation
	regions:
	- us-east-1
	template: cloudformation/account-creation-validation-role.yaml
	runOrder: 1
	terminationProtection: true
	parameters:
	- name: pManagementAccountId
	value: "{{ account Management }}"</pre>
	5. 保存并关闭 customizations-config.yaml 文件。	

Task	描述	所需技能
准备部署account-tagging-to-ssm-parameter-role 角色。	现在，您可以自定义解决方案，以便在除管理账户之外的所有账户中部署 account-tagging-to-ssm-parameter-role IAM 角色。此角色用于在参数存储中创建 AWS Systems Manager 参数。 1. 从中下载-p account-tagging-to-ssmarameter-role.yaml 文件。 GitHub 2. 将文件保存到文件模板部分所customizations-config.yaml 示的位置。 3. 打开 customizations-config.yaml 文件。 4. 在该cloudFormationStacks 部分中，添加以下代码。根据需要更新您的 AWS 区域 landing zone 的目标： `- deploymentTargets: organizationalUnits: - Root excludedAccounts: - Management description: IAM Role to create SSM Parameter`	AWS DevOps

Task	描述	所需技能
	s based on Account Tagging name: lza-account-tagging-to-ssm-parameter regions: - us-east-1 template: cloudformation/account-tagging-to-ssm-parameter-role.yaml runOrder: 1 terminationProtection: true parameters: - name: pManagementAccountId value: "{{ account Management }}" 5. 保存并关闭 customizations-config.yaml 文件。	

Task	描述	所需技能
准备部署config-log-validation-role 角色。	现在，您可以自定义解决方案以在日志存档账户中部署 config-log-validation-role IAM 角色。此角色允许 ValidateResources Lambda 函数访问 Amazon S3 存储桶以获取日志记录和访问 AWS Config 规则。 1. 从中下载 config-log-validation-role.yaml 文件。 GitHub 2. 将文件保存到文件模板部分所customizations-config.yaml 示的位置。 3. 打开 customizations-config.yaml 文件。 4. 在该cloudFormationStacks 部分中，添加以下代码。根据需要更新您的 AWS 区域 landing zone 的目标： `- deploymentTargets: accounts: - LogArchive description: IAM Role to validate Config and Logs name: lza-config-log-validation-role regions:`	AWS DevOps

Task	描述	所需技能

	<pre>- us-
	east-1
	template:
	cloudformation/con
	fig-log-validation-
	role.yaml
	runOrder: 1
	terminati
	onProtection: true
	
	parameters:
	- name:
	pManagementAccount
	Id
	value:
	"{{ account Managemen
	t }}"</pre> 5. 保存、关闭并提交对 `customizations-config.yaml` 文件所做的更改。	

（可选）从微软 Entra ID 获取数据

Task	描述	所需技能
创建允许 Lambda 函数与 Microsoft Entra ID 通信的应用程序。	1. 在 Microsoft Entra ID 管理中心注册`create_aws_account` 应用程序。有关说明，请参阅 Microsoft 文档中的注册应用程序。 2. 按照 Microsoft 文档中更新应用程序的请求权限中的说明为`create_aws_account` 应用程序配	Microsoft Entra ID

Task	描述	所需技能

	置以下 Microsoft Graph 权限：
	• 应用程序. Read.All
	  • 类型：应用程序
	  • 需要管理员同意：是
	• AppRoleAssignment。 ReadWrite.All
	  • 类型：应用程序
	  • 需要管理员同意：是
	• 组。 ReadWrite.All
	  • 类型：应用程序
	  • 需要管理员同意：是
	• ServicePrincipalEndpoint.Read.All
	  • 类型：应用程序
	  • 需要管理员同意：是
	• 同步。 ReadWrite.All
	  • 类型：应用程序
	  • 需要管理员同意：是
	• 用户. 读取
	  • 类型：已委托
	  • 需要管理员同意：否	

Task	描述	所需技能
检索`create_aws_account` 应用程序的值。	现在，您可以检索`create_aws_account` 应用程序所需的值。 1. 在 Microsoft Entra ID 管理中心中，导航到应用程序注册，然后选择。`create_aws_account` 2. 在左侧窗格中，选择概览。 3. 在概述页面上，记下以下值： • 应用程序（客户端）ID • 目录（租户）ID 4. 在左侧窗格的 “管理” 下，选择 “证书和机密”。 5. 在 “证书和密钥” 页面上，选择 “客户机密” 选项卡，然后记下以下值： • 客户机密值 • 客户端密钥 ID	Microsoft Entra ID
创建将 Microsoft Entra ID 与 IAM 身份中心集成的应用程序。	在 Microsoft Entra ID 管理中心注册LZA2应用程序。有关说明，请参阅 Microsoft 文档中的注册应用程序。	Microsoft Entra ID

Task	描述	所需技能
检索LZA2应用程序的值。	现在，您可以检索LZA2应用程序所需的值。 1. 在 Microsoft Entra ID 管理中心中，导航到 “企业应用程序”，然后选择LZA2。 2. 在左侧窗格中，选择概览。 3. 在概述页面上，记下以下值： • 名称 • 对象 ID 4. 在左侧窗格的 “管理” 下，选择 “清单”。 5. 在 JSON 文件中的`appRoles`部分中，找到名为的应用程序角色`User`。 6. 记下此应用程序角色的`id`值。	Microsoft Entra ID

Task	描述	所需技能

创建密钥。	1. 在中 AWS CLI，输入以下命令来创建变量。使用您为`create_aws_account` 和LZA2应用程序检索到的值：
	
	`# Variables for create_aws_account app`
	`TENANT_ID='<Directory ID>'`
	`CLIENT_ID='<Application ID>'`
	`SECRET_ID='<Client secret ID>'`
	`SECRET_VALUE='<Client secret value>'`
	
	`# Variables for LZA2 app`
	`OBJECT_ID='<Object ID>'`
	`APP_ROLE_ID='<App role ID>'`
	`ENTERPRISE_APP_NAME='<Name>'`
	
	2. 输入以下命令以创建名为`GraphApiSecret` 的密钥 AWS Secrets Manager：
	
	`aws secretsmanager create-secret \`
	`--name GraphApiSecret \`
	`--secret-string "{\"client_id\": \"${CLIENT_ID}\", \"tenant_id\": \"${TENANT_ID}\",`
		AWS DevOps

Task	描述	所需技能
	\"object_id\": \"${OBJECT_ID}\", \"app_role_id\": \"${APP_ROLE_ID}\", \"secret_value\": \"${SECRET_VALUE}\", \"secret_id\": \"${SECRET_ID}\"}" 如果将来需要更新密钥，则可以更新变量并运行以下命令： aws secretsmanager update-secret \ --secret-id GraphApiSecret \ --secret-string "{\"client_id\": \"${CLIENT_ID}\", \"tenant_id\": \"${TENANT_ID}\", \"object_id\": \"${OBJECT_ID}\", \"app_role_id\": \"${APP_ROLE_ID}\", \"secret_value\": \"${SECRET_VALUE}\", \"secret_id\": \"${SECRET_ID}\"}"	

部署解决方案

Task	描述	所需技能
克隆源代码。	1. 输入以下命令来克隆lza-account-creation-workflow存储库：	DevOps 工程师

Task	描述	所需技能
	``` git clone https://github.com/aws-samples/lza-account-creation-workflow ``` 2. 输入以下命令以导航到克隆存储库的config目录： ``` cd lza-account-creation-workflow/config ```	

Task	描述	所需技能
更新 deploy-config.yaml 文件。	1. 打开 deploy-config.yaml 文件。 2. 查看模板并根据需要更新值，以便在您的 AWS 环境中进行部署。例如，更新以下各项的值： • accountCreationFailure • accountCompletionFromEmail • ssoLoginUrl • rootEmailPrefix • rootEmailDomain 3. 如果你要与 Microsoft Entra ID 集成，请执行以下操作： • 将 enableAzureADIntegration 设置为 true。 • 对于graphApiSecretName 值，请输入您之前创建的密钥（GraphApiSecret ）。 4. 保存并关闭 deploy-config.yaml 文件。	AWS DevOps

Task	描述	所需技能

在您的 AWS 环境中部署解决方案。	1. 输入以下命令：
	`cdk bootstrap <account-number>/<Region>`
	有关更多信息，请参阅文档中的引导。 AWS CDK
	2. 输入以下命令合成 CloudFormation 模板：
	`cdk synth`
	有关更多信息，请参阅 AWS CDK 文档中的配置和执行 AWS CDK 堆栈合成。
	3. 输入以下命令部署解决方案。
	`cdk deploy`
	有关更多信息，请参阅 AWS CDK 文档中的部署 AWS CDK 应用程序。
	Note
	此解决方案使用 Amazon S3 存储桶来存储此解决方案的源代码。您可以使用 upload_to_source_bucket.py 脚本创建源代
		AWS DevOps

Task	描述	所需技能
	码存档并上传更新的版本。	

选项 1 — 使用 Python 创建账户

Task	描述	所需技能
确定要使用哪些参数。	选择在运行启动 Step Functions 工作流程的 Python 脚本时要使用的参数。有关参数的完整列表，请参阅此模式的 “其他信息” 部分。	AWS DevOps，Python

启动 Python 脚本。	1. 输入以下命令以导航到克隆的存储库：
	`cd lza-account-creation-workflow`
	2. 运行启动 Step Functions 工作流程的 Python 脚本。以下是包含示例参数和值的命令示例：
	`python ./run-stepfunction.py \ --account-name "lza-test-01" \ --support-dl "johnsmith@example.com" \ --managed-org-unit "Workloads/Workload-1" \ --purpose "Testing new micro service" \`
		DevOps 工程师，Python

Task	描述	所需技能
	--force-update true \ --ad-integration "{\"CustomerAccountAdmin\": \"platform-admin\", \"CustomerAccountDev\": \"workload1-app1\"}" \ --bypass-creation true \ --tags APPLICATION=TestingMicroService 3. 请等待，直到收到账户已成功创建的通知。 **Note** 如果账户创建过程失败，Amazon SNS 会向您在文件accountCreationFailure 中定义的电子邮件地址发送通知。deploy-config.yaml 账户申请者未收到通知。	

选项 2 — 使用 API Gateway 和 awscurl 创建账户

Task	描述	所需技能
为 awscurl 设置变量。	1. 输入以下命令以导航到源代码目录：	AWS DevOps

Task	描述	所需技能

	`cd lza-account-creation-workflow`
	2. 输入以下命令来设置 AWS 访问密钥变量。您可以从AWS 访问门户中复制变量，然后将其粘贴到 shell 中。以下是示例：
	`export AWS_ACCESS_KEY_ID="<id>"`
	`export AWS_SECRET_ACCESS_KEY="<key>"`
	`export AWS_SESSION_TOKEN="<token>"`
	3. 输入以下命令为 API 调用设置： AWS 区域
	`export AWS_REGION=$(aws configure get region)`
	4. 输入以下命令从`lza-account-creation-workflow-application` CloudFormation 输出中检索 API Gateway 端点：
	`export API_GATEWAY_ENDPOINT=$(aws cloudformation describe-stacks --stack-name "lza-account-creation-workflow-application" --query 'Stacks[*`	

Task	描述	所需技能
	```].Outputs[?OutputK ey==`oApiGatewayCr eateAccountEndpoin t`].OutputValue' -- output text) ```	

查看姓名可用性。	输入以下命令以验证该名称是否可用于 AWS 账户。<AWS_ACCOUNT_NAME> 替换为目标账户的名称：
	``
	awscurl --service
	 execute-api \
	    --region ${AWS_REG
	ION} \
	    --access_key
	 ${AWS_ACCESS_KEY_ID}
	 \
	    --secret_key
	 ${AWS_SECRET_ACCES
	S_KEY} \
	    --security_token
	 ${AWS_SESSION_TOKEN}
	 \
	    -X POST ${API_GAT
	EWAY_ENDPOINT}chec
	k_name?account_nam
	e=<AWS_ACCOUNT_NAME>
	``
		AWS DevOps

Task	描述	所需技能
运行账户创建工作流程。	1. 在克隆存储库的根文件夹中，打开该`api_example.json` 文件。 2. 使用您的配置值更新参数： `{ "account_name": "lza-test-01", "account_email": "johnsmith+lzatest01@example.com", "support_dl": "johnsmith@example.com", "managed_org_unit": "Workloads/Workload-1", "ad_integration": [{ "PermissionSetName": "CustomerAccountAdmin", "ActiveDirectoryGroupName": "platform-admin" }, { "PermissionSetName": "CustomerAccountDev", "ActiveDirectoryGroupName": "workload1-app1" }], "force_update": "true",`	AWS DevOps

Task	描述	所需技能
	<pre> "bypass_c reation": "false", "account_tags": [{ "Key": "Environment", "Value": "Dev" }, { "Key": "DeploymentMethod", "Value": "ApiGateway" }] }</pre> 3. 保存并关闭 `api_examp le.json` 文件。 4. 输入以下命令以启动 Step Functions 工作流程： <pre>awscurl --service execute-api \ --data @api-exam ple.json \ --region ${AWS_REG ION} \ --access_key ${AWS_ACCESS_KEY_I D} \ --secret_key ${AWS_SECRET_ACCES S_KEY} \ --security_token ${AWS_SESSION_TOKE N} \ -X POST ${API_GAT EWAY_ENDPOINT}exec ute</pre>	

Task	描述	所需技能
	5. 在上一个命令的输出中，记下 Step Functions 的执行亚马逊资源名称 (ARN)。 6. 如果要检查 Step Functions 工作流程的状态，请输入以下命令。<STEP_FUNCTION_EXECUTION_NAME> 替换为 Step Functions 执行 ARN 或名称： `awscurl --service execute-api \ --region ${AWS_REGION} \ --access_key ${AWS_ACCESS_KEY_ID} \ --secret_key ${AWS_SECRET_ACCESS_KEY} \ --security_token ${AWS_SESSION_TOKEN} \ -X GET ${API_GATEWAY_ENDPOINT}get_execution_status?execution=<STEP_FUNCTION_EXECUTION_NAME>` 7. 请等待，直到收到账户已成功创建的通知。 **Note** 如果账户创建过程失败，Amazon SNS 会向您在文	

Task	描述	所需技能
	件accountCr eationFai lure 中定义的电子邮件地址发送通知。deploy-co nfig.yaml 账户申请者未收到通知。	

（可选）清理解决方案

Task	描述	所需技能
从 Amazon S3 存储桶中移除对象。	移除以下 Amazon S3 存储桶中的所有对象： • lza-account-creation-work-<CDK_UNIQUE_ID> • lza-account-creation-workflow-src-<AWS_REGION>-<AWS_ACCOUNT> • lza-account-creation-workflow-<AWS_REGION>-<AWS_ACCOUNT>	AWS DevOps
删除 CloudFormation 堆栈。	输入以下命令以删除 CloudFormation 堆栈： `aws cloudformation delete-stack \`	AWS DevOps

Task	描述	所需技能

	`--stack-name lza-account-creation-workflow-application`
	`aws cloudformation wait stack-delete-complete \`
	`--stack-name lza-account-creation-workflow-application`	

删除管道。	输入以下命令以删除`lza-account-creation-workflow-pipeline` 管道： `cdk destroy lza-account-creation-workflow-pipeline --force`	AWS DevOps

相关资源

- 着陆区域加速器已开启 AWS（AWS 解决方案库）
- 常见 AWS CDK 问题疑难解答（AWS CDK 文档）

其他信息

Step Functions 工作流程图

下图显示了 Step Functions 工作流程中的状态。

Arguments (参数)

以下是运行启动 Step Functions 工作流程的 Python 脚本时可以使用的参数。

以下参数为必需参数：

- `account-name (-a)`（字符串）-新建的名称 AWS 账户。
- `support-dl (-s)`（字符串）-账户创建过程完成后接收通知的电子邮件地址。
- `managed-org-unit (-m)`（字符串）-将包含新帐户的托管组织单位 (OU)。

以下参数是可选的：

- `ad-integration (-ad)`（字符串字典）— Microsoft Entra ID 组和分配的权限集。以下是如何使用此参数的示例：

```
--ad-integration "{\"<PermissionSetName>\": \"<EntraIdGroupName>\"}"
```

- `account-email (-e)`（字符串）-新用户的 root 用户的电子邮件地址 AWS 账户。

> **Note**
>
> 如果不使用此参数，则将使用`configs/deploy-config.yaml`文件中的值`rootEmailPrefix`生成电子邮件地址。`rootEmailDomain`如果未提供电子邮件地址，则使用以下格式生成电子邮件地址：`rootEmailPrefix+accountName@rootEmailDomain`。

- `region (-r)`（字符串）— Step Functions 工作流程的部署 AWS 区域 位置。默认值为 `us-east-1`。
- `force-update (-f)`（字符串布尔值）-输入`true`强制 AWS Service Catalog 更新预配置的产品。
- `bypass-creation (-b)`（字符串布尔值）-输入`true`以绕过向`accounts-config.yaml`文件添加帐户并绕过运行`AWSAccelerator-Pipeline`管道。此参数通常用于测试帐户创建工作流程或在`Landing Zone Accelerator`管道中出现错误时运行 Step Functions 的其余步骤。
- `tags (-t)`（字符串）-要添加到的其他标签 AWS 账户。默认情况下，会添加以下标签：`account-name`、`support-dl`和`purpose`。以下是如何使用此参数的示例：

```
--tags TEST1=VALUE1 TEST2=VALUE2
```

自动清点多个账户和地区的 AWS 资源

由 Matej Macek (AWS) 创建

摘要

这种模式概述了一种自动化方法，用于维护多个帐户的全面 AWS 资源清单，以及 AWS 区域. 它旨在帮助基础设施和安全工程师改善其资源管理实践。它用于跟踪资源变化，Amazon Athena 用于查询，Amazon 用于 AWS Config 交互式控制面板 QuickSight 。您可以通过部署 AWS CloudFormation 堆栈来实现此解决方案。

此解决方案类似于使用 Amazon Athena QuickSight 和AWS Amazon 可视化 AWS Config 数据（博客文章）中介绍的解决方案。此模式扩展了该解决方案，以满足以下常见要求并提供以下主要好处：

- 以@@ 合规为中心 — 这种方法可以帮助您满足监管要求，例如 PCI DSS、NIST SP 800-53、ISO/IEC 27001、HIPAA、GDP R以及其他要求准确资产库存的监管要求。
- 自定义框架 — 它为为各种 AWS 资源创建 QuickSight 仪表板提供了基础，因此您可以根据自己的特定要求自定义解决方案。
- 以用户为导向的增强功能 — 这种方法结合了来自实际用例的反馈，并满足了对更全面解决方案的要求。

在动态、多账户或多区域环境中，基础设施、安全和财务团队经常面临可见性和协作方面的挑战。该解决方案旨在解决这些难题，并显著减少创建和维护资源清单所需的时间和精力。结果是资源集中视图，可帮助您改进资源分配决策，识别和降低风险，优化成本，并提高整体可见性和协作。这种方法弥合了概念性解决方案与实际实施需求之间的差距，以实现安全、合规和运营目的。

先决条件和限制

先决条件

- 以下处于活动状态 AWS 账户：
  - 管理账户-用于在整个组织内计费、创建账户和控制访问权限的集中账户
  - 审计账户 — 用于安全监控、合规性检查和偏差通知的集中中心
  - 日志存档账户 — 用于存储和分析所收集数据的集中账户
- 在审计账户中，一种 AWS Config 聚合器，用于收集和汇总来自目标账户和地区的配置数据
- 在日志存档帐户中，设置以下内容：
  - 亚马逊简单存储服务 (Amazon S3) S ervice 存储桶，用于存储来自聚合器的数据 AWS Config

  - 亚马逊 QuickSight 订阅
  - 与亚马逊 Athena QuickSight 之间的授权连接
  - 通过 Athena 查询访问 Amazon S3 存储桶的@@ 权限
- AWS Command Line Interface (AWS CLI)，已安装并配置
- 部署预置以下资源的 CloudFormation 堆栈的权限：
  - 一个 AWS Lambda 函数
  - 亚马逊 S3 通知配置
  - Athena 数据库、表和视图
  - QuickSight 数据集和数据源
- 在中运行自动化的权限 AWS Systems Manager
- 访问权限 QuickSight

限制

- 解决方案依赖于 AWS Config。 AWS Config 通常会在检测到更改后立即或按您指定的频率记录对资源的配置更改。但是，这是尽力而为，有时可能需要更长的时间。
- 此解决方案仅跟踪AWS Config 支持的资源类型。
- 该解决方案不跟踪其他云提供商或本地环境中的资源库存。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅 AWS 文档中的服务终端节点和配额页面，然后选择该服务的链接。

架构

下图显示了收集、组织、分析和可视化组织中多个账户的配置和合规性数据的简化流程。 AWS

图表显示了以下工作流：

1. AWS Config 聚合器定期收集有关目标账户和区域中资源的配置和合规性数据，然后将这些数据传输到日志存档账户中的 Amazon S3 存储桶。
2. 向 Amazon S3 存储桶添加新 AWS Config 数据会调用一个 AWS Lambda 函数。
3. Lambda 函数通过为密钥配置与每个快照文件的区域和日期相对应的值来对数据进行分区。这有助于 AWS Glue 有效地查询和处理配置和合规性数据。

4. Amazon Athena 使用架构 AWS Glue 对存储在 Amazon S3 存储桶中的数据运行 SQL 查询。它利用中的架构元数据 AWS Glue 来理解数据的结构。
5. Athena 中的@@ 视图定义和提取目标数据集。
6. Amazon 中的@@ 控制面板可 QuickSight 帮助您可视化和分析数据集。

工具

AWS 服务

- Amazon Athena 是一种交互式查询服务，它可帮助您通过使用标准 SQL 直接在 Amazon S3 中分析数据。
- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。
- AWS Config提供了您的资源 AWS 账户 及其配置方式的详细视图。它可以帮助您确定资源之间的关联方式，以及它们的配置如何随时间变化。 AWS Config 聚合器从多个 AWS 账户 区域收集 AWS Config 配置和合规性数据。
- AWS Glue是一项完全托管的提取、转换和加载 (ETL) 服务。它可以帮助您在数据存储和数据流之间对数据进行可靠地分类、清理、扩充和移动。此模式使用 AWS Glue 数据目录和架构注册表。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Organizations是一项账户管理服务，可帮助您将多个账户整合 AWS 账户 到一个由您创建和集中管理的组织中。
- Amazon QuickSight 是一项云规模的商业智能 (BI) 服务，可帮助您在单个控制面板中可视化、分析和报告数据。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Systems Manager 可帮助您管理在 AWS Cloud中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。AWS Systems Manager 自动化简化了许多人的常见维护、部署和补救任务 AWS 服务。

代码存储库

此模式的 AWS CloudFormation 模板可在可AWS Config 视化 GitHub 存储库中找到。此 CloudFormation 模板部署了 AWS Systems Manager 自动运行手册，该操作手册设置 AWS Config

为与 Amazon Athena 配合使用。此自动化 AWS Glue 可以准备连接指定的 Amazon S3 存储桶，在 Amazon Athena 中创建视图，并配置亚马逊 QuickSight 以实现控制面板可视化。

最佳实践

- 我们建议您遵循设置和管理安全的多账户 AWS 环境中的最佳实践，而不是 AWS 规范 AWS Control Tower性指导。
- 我们建议您创建一个 AWS Config 聚合器来收集整个 AWS 组织的配置和合规性数据。有关更多信息，请参阅文档中的多账户多区域数据聚合。 AWS Config
- 在部署此解决方案之前，我们建议您查看 A mazon S3、AWS Config、Athena 和的当前定价信息。QuickSight

操作说明

部署堆 CloudFormation 栈

Task	描述	所需技能
下载 CloudFormation 模板。	下载 Config-Visualizat ion QuickSight-SSM-Automation.y aml 模板 CloudFormation 。	AWS 管理员、云管理员、DevOps 工程师
修改 CloudFormation 模板。	只有当你正在使用AWS Control Tower并 AWS Config 由其管理时，才能完成此步骤 AWS Control Tower。您需要修改 CloudFormation 模板。 1. 登录 管理账户。 2. 打开 AWS Organizations 管理控制台。 3. 导航到 Settings (设置) 页面。此页面显示有关该组织的详细信息，包括组织 ID。 4. 复制组织 ID。	DevOps 工程师，AWS 管理员

Task	描述	所需技能

	5. 在你首选的文本编辑器中，打开 Config-VisualizQuickSightation-SSM-Automation.yaml 文件。
	6. 找到以下行：
	`return re.match( '^AWSLogs/(\d+)/Config/([\w-]+)/(\d+)/(\d+)/(\d+)/(\d+)/ConfigSnapshot/[^\\]+$', object_key)`
	7. 将此行替换为以下内容，其中<ORGANIZATION_ID> 是您之前复制的 ID：
	`return re.match( '^<ORGANIZATION_ID>/AWSLogs/(\d+)/Config/([\w-]+)/(\d+)/(\d+)/(\d+)/(\d+)/ConfigSnapshot/[^\\]+$', object_key)`
	8. 保存并关闭 Config-VisualizQuickSightation-SSM-Automation.yaml 文件。	

Task	描述	所需技能
创建堆 CloudFormation 栈。	按照从 CloudFormation 控制台创建堆栈中的说明进行操作。请注意以下几点： 1. 选择 “上传模板文件”，然后选择您下载的 YAML 文件。 2. 对于堆栈名称，输入 `Config-QuickSight-Visualization-SSM-Automation` 。 3. 选择提交。	AWS 管理员、云管理员、DevOps 工程师

在 Systems Manager 中运行自动化

Task	描述	所需技能
找到你的 QuickSight 用户名。	1. 打开 QuickSight 管理控制台。 2. 打开个人资料菜单。 3. 记下用户名。您稍后会需要此值。	AWS 管理员、云管理员、DevOps 工程师
找到传送渠道名称和 Amazon S3 存储桶名称。	1. 在中 AWS CLI，输入以下命令： `aws configservice describe-delivery-channels` 2. 记下 Amazon S3 存储桶的名称和您的AWS Config 交付渠道的名称。你稍后需要这些值。	AWS 管理员、云管理员、DevOps 工程师

Task	描述	所需技能
在 Systems Manager 中运行自动化。	1. 打开 AWS Systems Manager 管理控制台。 2. 在导航窗格中，选择文档。 3. 选择我拥有的。 4. 选择 Config-QuickSight-可视化。 5. 选择执行自动化。 6. 在输入参数部分，输入以下参数的值： • `ConfigDeliveryChannelName` — 输入您的 AWS Config 配送渠道的名称。此参数为必需参数。 • `ConfigS3BucketLocation` — 输入用于存储 AWS Config 配置数据的 Amazon S3 存储桶的名称。此参数为必需参数。 • `QuickSightUserName` — 输入具有管理权限的用户名 QuickSight。此参数为必需参数。 • `AutomationAssumeRole` — 允许 Systems Manager Automation 代表你执行操作的 AWS Identity and Access Management (IAM) 角色的亚马逊资源名称 (ARN)。此参数为可选的。将此参数留空。	AWS 管理员、云管理员、DevOps 工程师

Task	描述	所需技能
	• `DeleteConfigVisualization` — 选择`false`。 7. 选择执行。	

在 Amazon 中可视化数据 QuickSight

Task	描述	所需技能
刷新数据。	要根据您的特定要求安排数据集刷新，请按照刷新 SPICE 数据中的说明进行操作。	AWS 管理员、 DevOps 工程师、云管理员
创建 分析。	要在中创建可帮助您 QuickSight 可视化资源的控制面板，请按照在 Amazon 中开始分析中的说明进行操作 QuickSight。	QuickSight 管理员
创建控制面板。	1. 修改完 QuickSight 分析后，按照发布仪表板中的说明创建仪表板。仪表板是您可以与其他 QuickSight 用户共享的分析。 2. 按照授予控制面板访问权限中的说明与目标 QuickSight 用户共享控制面板。	QuickSight 管理员

(可选)清除

Task	描述	所需技能
删除 Systems Manager 自动化创建的资源。	1. 打开 AWS Systems Manager 管理控制台。 2. 在导航窗格中，选择文档。 3. 选择我拥有的。 4. 选择 Config-QuickSight-可视化。 5. 选择执行自动化。 6. 在输入参数部分中，为`DeleteConfigVisualization` 参数输入`true`。 7. 选择执行。	AWS 管理员、云管理员、DevOps 工程师
删除 CloudFormation 堆栈。	要删除`Config-QuickSight-Visualization-SSM-Automation` 堆栈中的资源，请按照从 CloudFormation 控制台删除堆栈中的说明进行操作。	AWS 管理员、云管理员、DevOps 工程师

故障排除

事务	解决方案
Amazon QuickSight 正在尝试连接到 `us-east-1` AWS 区域，但不允许在该地区创建资源。	一项服务控制政策限制您在该 QuickSight 地区订阅 Amazon。在服务控制策略中，手动指定目标 AWS 区域。`<REGION_ID>` 替换为相应的区域标识符： `https://<REGION_ID>.quicksight.aws.amazon.com/sn/start/dashboards`

事务	解决方案
	以下是示例： `https://eu-central-1.quicksight.aws.amazon.com/sn/start/dashboards`
在亚马逊 Athena 中，您会看到以下消息： `Before you run your first query, you need to set up a query result location in Amazon S3.`	确保您已准备好一个 Amazon S3 存储桶，用于存储来自亚马逊 Athena 的查询结果。然后，按照使用 Amazon Athena 控制台指定查询结果位置中的说明进行操作。

相关资源

AWS 文档

- AWS Config 文档
- 亚马逊 QuickSight 文档

AWS 博客文章

- 使用自动实现 AWS Config 数据可视化 AWS Systems Manager
- 如何使用定期记录资源配置的变化 AWS Config

其他资源

- 亚马逊 QuickSight 社区学习中心
- 亚马逊 QuickSight 社区画廊

配置 VPC 流日志以实现对 Amazon Web Services account 的集中管理

创建者：Benjamin Morris (AWS) 和 Aman Kaur Gandhi (AWS)

摘要

在 Amazon Web Services（AWS）虚拟私有云（VPC）中，VPC 流日志功能可以为运营和安全故障排除提供有用的数据。但是，在多账户环境中使用 VPC 流日志存在限制。具体而言，不支持 Amazon Logs 中的跨账户流 CloudWatch 日志。相反，您可以通过使用相应存储桶策略配置 Amazon Simple Storage Service（Amazon S3）存储桶来集中管理日志。

> **Note**
>
> 此模式讨论了将流日志发送到集中位置的要求。但是，如果您还希望在成员账户中本地提供日志，则可以为每个 VPC 创建多个流日志。无法访问日志存档账户的用户可以查看流量日志以进行故障排除。或者，您可以为向日志发送日志的每个 VPC 配置一个流 CloudWatch 日志。然后，您可以使用 Amazon Data Firehose 订阅筛选器将日志转发到 S3 存储桶。有关更多信息，请参阅相关资源部分。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- AWS Organizations 组织，其账户用于集中管理日志（例如，日志存档）

限制

如果您使用 AWS Key Management Service（AWS KMS）托管密钥 `aws/s3` 来加密您的中央存储桶，它将不会收到来自其他账户的日志。相反，你会看到一个`Unsuccessful`错误代码 400，其中包含一条消息，例如你`"LogDestination: <bucketName> is undeliverable"`给定的消息`ResourceId`。

这是因为账户的 AWS 托管式密钥无法在账户之间共享。

解决方案是使用 Amazon S3 托管加密（SSE-S3）或 AWS KMS 客户托管密钥，您可以与成员账户共享该密钥。

架构

目标技术堆栈

在下图中，为每个 VPC 部署了两个流日志。一个向本地日志组发送 CloudWatch 日志。另一个将日志发送到集中式日志账户中的 S3 存储桶。存储桶策略允许日志传输服务将日志写入存储桶。

> **Note**
>
> 自 2023 年 11 月起，AWS 现在支持 a ws: SourceOrg ID 条件密钥。此条件允许您拒绝向 AWS Organizations 组织以外的账户写入中央存储桶。

目标架构

自动化和扩缩

每个 VPC 都配置为将日志发送到中央日志账户中的 S3 存储桶。使用以下自动化解决方案之一来帮助确保正确配置流日志：

- AWS CloudFormation StackSets
- AWS Control Tower Account Factory for Terraform（AFT）
- 带有补救功能的 AWS Config 规则

工具

工具

- Amazon CloudWatch Lo gs 可帮助您集中管理来自所有系统、应用程序和 AWS 服务的日志，以便您可以监控它们并安全地将其存档。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。此模式使用 VPC 流日志功能来捕获有关在您的 VPC 中传入和传出各个网络接口的 IP 流量的信息。

最佳实践

使用基础设施即代码（IaC）可以极大地简化 VPC 流日志的部署过程。将您的 VPC 部署定义抽象化为包含流日志资源构造，将自动 VPCs 使用流日志进行部署。这将在下一节中演示。

集中式流日志

在 HashiCorp Terraform 中向 VPC 模块添加集中式流日志的语法示例

此代码创建流日志，用于将日志从 VPC 发送到集中式 S3 存储桶。请注意，此模式不包括 S3 存储桶的创建。

有关推荐的存储桶策略语句，请参阅其他信息部分。

```
variable "vpc_id" { type = string }
locals { custom_log_format_v5 = "$${version} $${account-id} $${interface-id} $
${srcaddr} $${dstaddr} $${srcport} $${dstport} $${protocol} $${packets} $${bytes}
 $${start} $${end} $${action} $${log-status} $${vpc-id} $${subnet-id} $${instance-
id} $${tcp-flags} $${type} $${pkt-srcaddr} $${pkt-dstaddr} $${region} $${az-id} $
${sublocation-type} $${sublocation-id} $${pkt-src-aws-service} $${pkt-dst-aws-service}
 $${flow-direction} $${traffic-path}" }
resource "aws_flow_log" "centralized_flow_log" {
  log_destination      = "arn:aws:s3:::centralized-vpc-flow-logs-
<log_archive_account_id>" # Optionally, a prefix can be added after the ARN.
  log_destination_type = "s3"
  traffic_type         = "ALL"
  vpc_id               = var.vpc_id
  log_format           = local.custom_log_format_v5 # If you want fields from VPC Flow
 Logs v3+, you will need to create a custom log format.
}
```

有关自定义日志格式的更多信息，请参阅 AWS 文档。

本地流日志

使用所需权限将本地流日志添加到 Terraform 中的 VPC 模块的语法示例

此代码创建流日志，用于将日志从 VPC 发送到本地 CloudWatch 日志组。

```
data "aws_region" "current" {}
variable "vpc_id" { type = string }
resource "aws_iam_role" "local_flow_log_role" {
  name = "flow-logs-policy-${var.vpc_id}"
  assume_role_policy = <<EOF
{
  "Version": "2012-10-17",
  "Statement": [{
      "Effect": "Allow",
      "Principal": {"Service": "vpc-flow-logs.amazonaws.com"},
      "Action": "sts:AssumeRole"
  }]
}
EOF
}
resource "aws_iam_role_policy" "logs_permissions" {
  name = "flow-logs-policy-${var.vpc_id}"
  role = aws_iam_role.local_flow_log_role.id
  policy = <<EOF
{
  "Version": "2012-10-17",
  "Statement": [{
      "Action": ["logs:CreateLog*", "logs:PutLogEvents", "logs:DescribeLog*",
 "logs:DeleteLogDelivery"],
      "Effect": "Allow",
      "Resource": "arn:aws:logs:${data.aws_region.current.name}:*:log-group:vpc-flow-
logs*"
  }]
}
EOF
}
resource "aws_cloudwatch_log_group" "local_flow_logs" {
  name              = "vpc-flow-logs/${var.vpc_id}"
  retention_in_days = 30
}
resource "aws_flow_log" "local_flow_log" {
  iam_role_arn    = aws_iam_role.local_flow_log_role.arn
  log_destination = aws_cloudwatch_log_group.local_flow_logs.arn
  traffic_type    = "ALL"
  vpc_id          = var.vpc_id
}
```

操作说明

部署 VPC 流日志基础设施

Task	描述	所需技能
确定加密策略并为中央 S3 存储桶创建策略。	中央存储桶不支持 aws/s3 AWS KMS 密钥，因此您必须使用 SSE-S3 或 AWS KMS 客户托管密钥。如果您使用 AWS KMS 密钥，密钥政策必须允许成员账户使用该密钥。	合规
创建中央流日志存储桶。	创建将向其发送流日志的中央存储桶，然后应用您在上一步中选择的加密策略。这应该在日志存档或类似用途的账户中。 从其他信息部分获取存储桶策略，并在使用您的环境特定值更新占位符后，将其应用于您的中央存储桶。	常规 AWS
配置 VPC 流日志以将日志发送到中央流日志存储桶。	向要从中收集数据的每个 VPC 添加流日志。实现这一目标的最具可扩展性的方法是使用 IaC 工具，例如 AFT 或 AWS Cloud Development Kit（AWS CDK）。例如，您可以创建一个在流日志旁边部署 VPC 的 Terraform 模块。如有必要，您可以手动添加流日志。	网络管理员
将 VPC 流日志配置为发送到本地 CloudWatch 日志。	（可选）如果您希望流日志在生成日志的账户中可见，请创建另一个流日志，将数据发送到本地账户中的 CloudWatch	常规 AWS

Task	描述	所需技能
	日志。或者，您可以将数据发送到本地账户中特定于账户的 S3 存储桶。	

相关资源

- 如何使用集中式流日志数据促进数据分析并满足安全要求（博客文章）
- 如何使用 AWS Config 规则自动启用 VPC 流日志（博客文章）

其他信息

桶策略

在为占位符名称添加值后，可以将此存储桶策略示例应用于流日志的中央 S3 存储桶。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSLogDeliveryWrite",
            "Effect": "Allow",
            "Principal": {
                "Service": "delivery.logs.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::<BUCKET_NAME>/*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control",
                    "aws:SourceOrgID": "<ORG_ID>"
                }
            }
        },
        {
            "Sid": "AWSLogDeliveryCheck",
            "Effect": "Allow",
            "Principal": {
                "Service": "delivery.logs.amazonaws.com"
```

```
            },
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::<BUCKET_NAME>",
            "Condition": {
                "StringEquals": {
                    "aws:SourceOrgID": "<ORG_ID>"
                }
            }
        },
        {
            "Sid": "DenyUnencryptedTraffic",
            "Effect": "Deny",
            "Principal": {
                "AWS": "*"
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::<BUCKET_NAME>/*",
                "arn:aws:s3:::<BUCKET_NAME>"
            ],
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": "false"
                }
            }
        }
    ]
}
```

使用 AWS Organizations 自动标记中转网关连接

由 Richard Milner-Watts (AWS)、Haris Bin Ayub (AWS) 和 John Capps (AWS) 编写

摘要

在 Amazon Web Services (AWS) 上 AWS Resource Access Manager，您可以使用AWS Transit Gateway跨 AWS 账户 界共享。但是，当您跨账户边界创建中转网关连接时，创建的连接没有名称标签。这可能会使识别这些连接变得耗时。

此解决方案提供了一种自动机制，用于收集由管理的组织内账户的每个 Transit Gateway 附件的相关信息AWS Organizations。该过程包括从中转网关路由表中查找无类别域间路由 (CIDR) 范围。然后，该解决方案会将格式为 `<CIDR-range>-<AccountName>` 的名称标签应用于持有中转网关的账户内的连接。

该解决方案可以与解决方案库中的无服务器公交网络 Orchestrator 等解决方案一起使用。 AWS 无服务器公交网络 Orchestrator 支持大规模自动创建 Transit Gateway Gateway 附件。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 包含所有相关账户的 AWS Organizations 组织
- 访问组织根目录下的组织管理账户以创建必需 AWS Identity and Access Management (IAM) 角色
- 包含一个或多个与组织共享并带有连接的中转网关的共享网络成员账户

架构

以下屏幕截图 AWS Management Console 显示了此解决方案生成的没有关联名称标签的 Transit Gateway 附件和两个带有名称标签的 Transit Gateway 附件的示例。生成的名称标签的结构为 `<CIDR-range>-<AccountName>`。

此解决方案用于AWS CloudFormation部署一个AWS Step Functions工作流，用于管理所有配置的 Transit Gateway Name 标签的创建 AWS 区域。工作流调用执行底层任务的AWS Lambda函数。

解决方案从中获取账户名后，Step Function AWS Organizations s 状态机将获取所有 Transit Gateway 附件 IDs。它们按地区并行处理。此处理包含查找每个连接的 CIDR 范围。CIDR 范围是通过在区域内的中转网关路由表中搜索匹配的中转网关连接 ID 来获取的。如果所有必需的信息均可用，该解决方案会将名称标签应用于附件。该解决方案不会覆盖任何现有名称标签。

该解决方案按照 Amazon EventBridge 事件控制的时间表运行。该事件每天早上 6:00 UTC 启动解决方案。

目标技术堆栈

- 亚马逊 EventBridge
- AWS Lambda
- AWS Organizations
- AWS Transit Gateway
- Amazon Virtual Private Cloud（Amazon VPC）
- AWS X-Ray

目标架构

下图演示了参考架构和工作流。

1. 计划的事件启动规则。
2. 该 EventBridge 规则启动 Step Functions 状态机。
3. 状态机调用 Lambda 函数 `tgw-tagger-organizations-account-query`。
4. `tgw-tagger-organizations-account-query` Lambda 函数在组织管理账户中代入该角色。
5. `tgw-tagger-organizations-account-query`Lambda 函数调用 Organizations API 来返回元数据 AWS 账户 。
6. 状态机调用 Lambda 函数 `tgw-tagger-attachment-query`。
7. 对于每个区域，状态机并行调用 `tgw-tagger-rtb-query` Lambda 函数来读取每个连接的 CIDR 范围。
8. 对于每个区域，状态机并行调用 `tgw-tagger-attachment-tagger` Lambda 函数。
9. 名称标签是在共享网络账户中为中转网关连接创建的。

自动化和扩缩

该解决方案并行处理每个区域以减少运行的总持续时间。

工具

Amazon Web Services

- AWS CloudFormation通过将基础架构视为代码，提供了一种对相关资源 AWS 和第三方资源集合进行建模、快速一致地配置这些资源并在其整个生命周期中对其进行管理的方法。
- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和您运行 AWS 的应用程序的指标。
- Amazon EventBridge 是一项无服务器事件总线服务，可用于将应用程序与来自各种来源的数据连接起来。 EventBridge 接收事件（环境变化的指示器），并应用规则将事件路由到目标。规则根据事件的结构（称为事件模式）或计划将事件与目标匹配。
- AWS Lambda是一项计算服务，支持在不预置或管理服务器的情况下运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。您只需按使用的计算时间付费。代码不运行时不会产生任何费用。
- AWS Organizations随着 AWS 资源的增长和扩展，可以帮助你集中管理和治理环境。使用 Organizations，您可以通过编程方式创建新资源 AWS 账户 和分配资源，对账户进行分组以组织工作流程，将策略应用于账户或群组进行治理，并通过对所有账户使用单一付款方式来简化计费。
- AWS Step Functions是一种低代码的可视化工作流服务，用于编排 AWS 服务、自动化业务流程和构建无服务器应用程序。工作流可以管理故障、重试、并行化、服务集成和可观测性，因此您可专注于更高价值的业务逻辑。
- AWS Transit Gateway通过中央集线器连接 VPCs 和本地网络。这简化了您的网络，结束了复杂的对等关系。它充当云路由器，因此每个新连接仅需建立一次。
- Amazon Virtual Private Cloud（亚马逊 VPC）是一项用于在您定义的逻辑隔离的虚拟网络中启动 AWS 资源的服务。
- AWS X-Ray收集有关您的应用程序所服务的请求的数据，并提供可用于查看、筛选和深入了解这些数据的工具，以识别问题和优化机会。

代码

该解决方案的源代码可在 Tr ansit Gateway Attachment Tagger GitHub 存储库中找到。存储库包含以下文件：

- `tgw-attachment-tagger-main-stack.yaml` 在共享网络账户中创建支持此解决方案的所有资源。
- `tgw-attachment-tagger-organizations-stack.yaml` 在组织的管理账户中创建角色。

操作说明

部署主解决方案堆栈

Task	描述	所需技能
收集必要的先决条件信息。	要配置从 Lambda 函数对 API AWS Organizations 的跨账户访问权限，您需要组织管理账户的账户 ID。 **Note** 这两个 AWS CloudFormation 堆栈的创建顺序很重要。您必须首先将资源部署到共享网络账户中。在将资源部署到组织的管理账户之前，共享网络账户中的角色必须已存在。有关更多信息，请参阅 AWS 文档。	DevOps 工程师
启动主解决方案堆栈的 AWS CloudFormation 模板。	主解决方案堆栈的模板将部署 IAM 角色、Step Functions 工作流程、Lambda 函数和亚马逊 CloudWatch 事件。 打开共享网络账户 AWS Management Console 的，然后打开:&CFN 控制台。 使用 `tgw-attachment-tagger-main-stack.yaml` 模板并指定以下值来创建堆栈：	DevOps 工程师

Task	描述	所需技能
	• 堆栈名称 — tgw-attachment-tagger-main堆栈 • awsOrganizationsRootAccountId— 组织管理账户的账户 ID • TGWRegions参数 — AWS 区域 对于解决方案，以逗号分隔的字符串输入 • TGWListparameter — IDs 要从解决方案中排除的公交网关，以逗号分隔的字符串输入 有关启动 AWS CloudFormation 堆栈的更多信息，请参阅AWS 文档。	
验证解决方案是否已成功启动。	等待 CloudFormation 堆栈达到 CRE ATE_ COMPLETE 状态。这应该需要不到 1 分钟的时间。 打开 Step Functions 控制台，验证是否创建了一台名为 tgw-attachment-tagger-state-machine 的新状态机。	DevOps 工程师

部署 AWS Organizations 堆栈

Task	描述	所需技能
收集必要的先决条件信息。	要配置从 Lambda 函数到 AWS Organizations API 的跨	DevOps 工程师

Task	描述	所需技能
	账户存取，您需要共享联网账户的账户 ID。	
启动 Organizations 堆栈的 CloudFormation 模板	AWS Organizations 堆栈模板将在组织管理账户中部署 IAM 角色。 访问组织管理账户的 AWS 控制台。然后打开 CloudFormation 控制台。 使用 `tgw-attachment-tagger-organizations-stack.yaml` 模板并指定以下值来创建堆栈： • 堆栈名称 — tgw-attachment-tagger-organizations 堆栈 • NetworkingAccountId参数- 共享网络帐户的帐户 ID 对于其他堆栈创建选项，请用默认值。	DevOps 工程师
验证解决方案是否已成功启动。	等待 AWS CloudFormation 堆栈达到 CREATE_COMPLETE 状态。这应该需要不到 1 分钟的时间。 打开 AWS Identity and Access Management (IAM) 控制台，确认已创建名为 tgw-attachment-tagger-organization-query- role 的新角色。	DevOps 工程师

验证解决方案

Task	描述	所需技能
运行状态机。	打开共享网络账户的 Step Functions 控制台，然后在导航窗格选择状态机。 选择状态机 tgw-attachment-tagger-state-机器，然后选择开始执行。 由于解决方案不使用此状态机输入，因此您可使用默认值。 `{ "Comment": "Insert your JSON here" }` 选择启动执行。	DevOps 工程师
观察状态机直至完成。	在打开的新页面，您可观看状态机的运行。持续时间将取决于待处理中转网关连接的数量。 在此页面，您可检查状态机的每个步骤。您可以查看状态机中的各种任务，并点击 Lambda 函数的 CloudWatch 日志链接。对于在地图中并行运行的任务，您可使用索引下拉列表来查看每个区域的具体实现。	DevOps 工程师
验证中转网关连接标签。	打开共享网络账户的 VPC 控制台，然后选择中转网关连接。 在控制台上，为满足标准	DevOps 工程师

Task	描述	所需技能
	的连接提供名称标签（连接将传播到中转网关路由表，并且资源所有者是组织的成员）。	
验证 CloudWatch 事件的启动。	等待 CloudWatch 事件启动。预定时间为 UTC 时间 06:00。 然后打开共享网络账户的 Step Functions 控制台，然后在导航窗格选择状态机。 选择状态机 tgw-attachment-tagger-state-机器。验证解决方案是否在 06:00 UTC 运行。	DevOps 工程师

相关资源

- AWS Organizations
- AWS Resource Access Manager
- 无服务器中转网络编排工具
- 创建 IAM 角色
- 在 AWS CloudFormation 控制台上创建堆栈

更多模式

- 使用 GitHub 操作基于 AWS CloudFormation 模板配置 AWS Service Catalog 产品
- 通过部署角色自动售货机解决方案来配置权限最低的 IAM 角色

人工智能和机器学习

主题

- 在 Amazon DynamoDB 中聚合数据，以便在 Athena 中进行 ML 预测
- 将一个 AWS CodeCommit 存储库 AWS 账户 与另一个账户中的 Amazon A SageMaker I Studio Classic 关联起来
- 使用 Amazon Textract 从 PDF 文件中自动提取内容
- 在 Amazon A SageMaker I Studio Lab 中使用 Deepar 计算时间序列构建冷启动预测模型
- 使用亚马逊 A SageMaker I 和 Azure 构建 MLOps 工作流程 DevOps
- 使用在 Amazon Bedrock 中配置模型调用日志 AWS CloudFormation
- 为 AWS Step Functions SageMaker 创建自定义 Docker 容器镜像并将其用于模型训练
- 使用 Amazon Bedrock 代理通过基于文本的提示在 Amazon EKS 中自动创建访问入口控制
- 使用 Terraform 和 Amazon AWS Bedrock 部署 RAG 用例
- 使用 Amazon 中的推理管道将预处理逻辑部署到单个终端节点的 ML 模型中 SageMaker
- 使用 RAG 和提示开发基于 AI 聊天的高级生成式 AI 助手 ReAct
- 使用 Amazon Bedrock 代理和知识库开发基于聊天的全自动助手
- 使用 Amazon Bedrock 和 Amazon Transcribe 从语音输入中记录机构知识
- 使用 Amazon Personalize 生成个性化和重新排名的推荐
- 使用 SageMaker AI 和 Hydra 简化从本地开发到可扩展实验的机器学习工作流程
- 在 Amazon 上训练和部署支持 GPU 的自定义机器学习模型 SageMaker
- 将自然语言翻译成查询 DSL OpenSearch 和 Elasticsearch 查询
- 使用 Amazon Q Developer 作为编码助手来提高工作效率
- 使用 Processing 对 TB 级机器学习 SageMaker 数据集进行分布式特征工程
- 使用 Flask 和 AWS Elastic Beanstalk 查看人工智能/机器学习 (AI/ML) 模型结果
- 更多模式

在 Amazon DynamoDB 中聚合数据，以便在 Athena 中进行 ML 预测

由 Sachin Doshi (AWS) 和 Peter Molnar (AWS) 创建

摘要

此模式向您展示如何使用 Amazon Athena 在 Amazon DynamoDB 表中构建物联网（IoT）数据的复杂聚合。您还将学习如何使用 Amazon A SageMaker I 通过机器学习 (ML) 推理来丰富数据，以及如何使用 Athena 查询地理空间数据。您可以使用此模式作为创建满足组织要求的 ML 预测解决方案的基础。

出于演示目的，此模式使用一个示例方案，该企业正在运营踏板车拼车，并希望预测必须为不同城市社区的客户部署的最佳踏板车数量。该企业使用预先训练的 ML 模型，该模型根据过去四个小时预测下一小时的客户需求。该场景使用路易斯维尔地铁政府公民创新与技术办公室的公共数据集。此场景的资源可在 GitHub 存储库中找到。

先决条件和限制

- 活跃的 AWS 账户
- 创建包含 AWS Identity and Access Management (IAM) 角色的 AWS CloudFormation 堆栈的权限，用于以下内容：
  - Amazon Simple Storage Service (Amazon S3)桶
  - Athena
  - DynamoDB
  - SageMaker AI
  - AWS Lambda

架构

技术堆栈

- Amazon QuickSight
- Amazon S3
- Athena
- DynamoDB

- Lambda
- SageMaker AI

目标架构

下图显示了使用 Athena 的查询功能、Lambda 函数、Amazon S3 存储、AI 终端节点和控制面板在 DynamoDB 中构建复杂数据聚合的架构。 SageMaker QuickSight

图表显示了以下工作流：

1. DynamoDB 表提取从踏板车实例集传输的 IoT 数据。
2. Lambda 函数使用提取的数据加载 DynamoDB 表。
3. Athena 查询为表示城市社区的地理空间数据创建一个新的 DynamoDB 表。
4. 查询位置保存在 S3 存储桶中。
5. Athena 函数从 SageMaker 托管预训练机器学习模型的 AI 端点查询机器学习推理。
6. Athena 直接从 DynamoDB 表中查询数据，并聚合数据进行分析。
7. 用户在 QuickSight 仪表板中查看分析数据的输出。

工具

AWS 服务

- Amazon Athena 是一种交互式查询服务，它可帮助您通过使用标准 SQL 直接在 Amazon S3 中分析数据。
- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- Amazon SageMaker AI 是一项托管机器学习服务，可帮助您构建和训练机器学习模型，然后将其部署到可用于生产的托管环境中。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon QuickSight 是一项云规模的商业智能 (BI) 服务，可帮助您在单个控制面板中可视化、分析和报告数据。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。

代码存储库

此模式的代码可在 Amazon Athena 机器学习 GitHub 存储库中通过亚马逊 DynamoDB 数据使用机器学习预测中找到。您可以使用存储库中的 CloudFormation 模板来创建示例场景中使用的以下资源：

- DynamoDB 表
- 一个 Lambda 函数，用于加载包含相关数据的表
- 用于推理请求的 A SageMaker I 终端节点，其预训练 XGBoost 模型存储在 Amazon S3 中
- 名为 `V2EngineWorkGroup` 的 Athena 工作组
- 命名 Athena 查询以查找地理空间 shapefile 并预测踏板车需求
- 预先构建的 A mazon Athena DynamoDB 连接器，它允许 Athena 与 DynamoDB 通信并使用 () 参照 DynamoDB 连接器构建应用程序AWS Serverless Application ModelAWS SAM

操作说明

获取示例数据集

Task	描述	所需技能
下载数据集和资源。	1. 下载无桩车辆租赁的公共数据集。出于演示目的，这些数据作为用例的一部分在 DynamoDB 中预先填充，但在生产环境中，您可以通过各种机制（例如物联网设备或 Amazon Kinesis 使用者）将这些数据发送到 DynamoDB。这些机制使用 Lambda 将数据插入 DynamoDB。 2. 下载表示肯塔基州路易斯维尔市内历史和文化街区边界的 GIS shapefiles。公共数据集由肯塔基州路易斯维尔和杰斐逊县信息联盟提供。最初的 shapefile 已	应用程序开发人员、数据科学家

Task	描述	所需技能
	经转换为文本文件，你可以用 Athena 进行查询，但是你可以在 Jupyter 笔记本中找到用于转换 shapefile 的 Python 代码，在使用亚马逊 Athena 的 GIS shapefile 的地理空间处理中。 GitHub 3. 下载预训练的 Python 代码，该代码使用 SageMaker AI 和 Athena 训练机器学习模型进行每小时预测。 4. 在 Athena 中获取 SQL 查询，该查询将所有内容汇集在一起，以便根据 DynamoDB 中存储的数据进行实时预测。 5. （可选）QuickSight 用于在肯塔基州路易斯维尔地图上可视化地理空间数据。	

使用 CloudFormation 模板部署所需的资源

Task	描述	所需技能
创建堆 CloudFormation 栈。	1. 从 GitHub 存储库下载 CloudFormation 模板。 2. **Note** 登录 AWS Management Console，然后选择`us-east-1` 。：机器学习模型存储	AWS DevOps

Task	描述	所需技能
	在 Amazon Elastic Container Registry (Amazon ECR) `us-east-1` AWS 区域中，但该模式与区域无关。您可以在任何支持此模式中 AWS 服务 使用的区域中复制该模式。 3. 打开CloudFormation 控制台，然后在导航窗格上选择 Stacks。 4. 选择创建堆栈，然后选择使用现有的资源（导入资源）。 5. 在标识资源页面上，选择下一步。 6. 在指定模板部分中，对于模板源，选择上传模板文件。 7. 选择 “文件”，然后选择您之前下载的 CloudFormation 模板。 8. 选择下一步，接受默认参数值，然后选择下一步以逐步完成设置向导的其余部分。 9. 选中 “我确认 AWS CloudFormation 可能会使用自定义名称创建 IAM 资源” 复选框。 10选择创建堆栈。	

Task	描述	所需技能
	Note CloudFormation 堆栈可能需要 15—20 分钟才能创建这些资源。	
验证部 CloudFormation 署。	要验证 CloudFormation 模板中的示例数据是否已加载到 DynamoDB 中，请执行以下操作： 1. 打开 DynamoDB 控制台，然后在导航窗格中选择实例。 2. 在表部分中，检查 `DynamoDBTableDocklessVehicles` 表。 3. 资源创建完成后，打开 Athena 控制台，然后从导航窗格中选择工作组。 4. 选择 `V2EngineWorkGroup` 工作组，然后选择切换工作组。 5. 如果您收到保存查询结果位置的提示，请选择您具有写入权限的 Amazon S3 位置。 6. 选择保存。 7. 在导航窗格中，选择查询编辑器，然后选择 `athena-ml-db-<your-AWS-account-number>` 数据库。	应用程序开发人员

将地理位置文件加载到 Athena 中

Task	描述	所需技能
使用地理空间数据创建 Athena 表。	要将地理位置文件加载到 Athena 中，请执行以下操作： 1. 打开 Athena 控制台，然后从导航窗格中选择查询编辑器。 2. 选择已保存的查询选项卡。 3. 搜索并选择Q1：Neighborhoods。 4. 要返回到查询编辑器，请选择编辑器选项卡。 5. 选择运行。这将在数据库中创建一个名为 `louisville_ky_neighborhoods` 的表。确保该表已在 `athena-ml-db-<your-AWS-account-number>` 数据库中创建。 该查询将为表示城市社区的地理空间数据创建一个新表。数据表是根据 GIS shapefile 创建的。`CREATE EXTERNAL TABLE` 语句定义了表的架构以及基础数据文件的位置和格式。 有关处理 shapefile 和生成此表的 Python 代码，请参阅示例中的使用 Amazon Athena 对 GIS shapefile 进行地理空间处理。 AWS 有关详细的 SQL	数据工程师

Task	描述	所需技能
	代码，请参阅上的 create_neighborhood_table.sql GitHub。	

根据汇总的 DynamoDB 数据按社区预测踏板车需求

Task	描述	所需技能
在 Athena 中声明一个用于查询 AI 的函数。 SageMaker	1. 打开 Athena 控制台，从导航窗格中选择查询编辑器，然后选择编辑器选项卡。 2. 将以下 SQL 语句复制并粘贴到查询编辑器中。 `USING EXTERNAL FUNCTION predict_demand ( location_id BIGINT, hr BIGINT , dow BIGINT, n_pickup_1 BIGINT, n_pickup_2 BIGINT, n_pickup_3 BIGINT, n_pickup_4 BIGINT, n_dropoff_1 BIGINT, n_dropoff_2 BIGINT, n_dropoff_3 BIGINT, n_dropoff_4 BIGINT ) RETURNS DOUBLE SAGEMAKER '<Your SageMaker endpoint>'` SQL 语句的第一部分声明外部函数，用于从托管预训练模型的 SageMaker AI 端点查询机器学习推论。	数据科学家、数据工程师

Task	描述	所需技能
	3. 定义输入参数的顺序和类型以及返回值的类型。 4. 选择运行。	

Task	描述	所需技能
根据汇总的 DynamoDB 数据按社区预测踏板车需求。	现在，您可以使用 Athena 直接从 DynamoDB 查询事务数据，然后聚合数据以进行分析和预测。通过直接查询 DynamoDB NoSQL 数据库不容易实现这一点。 1. 打开 Athena 控制台，然后从导航窗格中选择查询编辑器。 2. 选择示例查询选项卡。 3. 搜索并选择 Q2：DynamoDBAthena MLScooter Predict。 4. 要返回到查询编辑器，请选择编辑器选项卡。 5. 选择运行。 SQL 语句执行以下操作： • 使用 Athena 联合查询来查询包含原始行程数据的 DynamoDB 表 • 使用 Athena 的地理空间函数将地理坐标放置在邻域中 • 使用 AI 通过 ML 推理丰富数据 SageMaker 有关使用 SQL 在 Athena 中聚合 DynamoDB 数据 SageMaker 和 AI 推理数据的信息，请参阅中的 athena_long.sql。 GitHub	应用程序开发人员、数据科学家

Task	描述	所需技能
验证输出。	输出表包括邻域质心的邻域、经度和纬度。它还包括下一小时预测的车辆数量。 该查询生成所选时间点的预测。您可以通过更改语句中所有位置的表达式 `TIMESTAMP '2019-09-07 15:00'` 来预测任何其他时间。 如果您的 DynamoDB 表中有实时数据源，请将时间戳更改为 `NOW()`。	应用程序开发人员、数据科学家

清理环境

Task	描述	所需技能
删除资源。	1. 打开 Athena 控制台，清空您在堆栈中创建的存储桶。CloudFormation 2. 打开CloudFormation 控制台，然后删除名为的堆栈`bdb-1462-athena-dynamodb-ml-stack` 。 3. 打开 Amazon CloudWatch 控制台，然后删除名为的日志组`/aws/sagemaker/Endpoints/Sg-athena-ml-dynamodb-model-endpoint` 。	AWS 应用程序开发人员 DevOps

相关资源

- 亚马逊 Athena 查询联合体 SDK () GitHub
- 查询地理空间数据（AWS 文档）
- 在 Amazon Athena ML（大数据博客）中使用机器学习预测而不是亚马逊 DynamoDB 数据AWS
- 亚马逊 ElastiCache（Redis OSS）（AWS 文档）
- 亚马逊 Neptune（文档）AWS

将一个 AWS CodeCommit 存储库 AWS 账户 与另一个账户中的 Amazon A SageMaker I Studio Classic 关联起来

由 Laurens van der Maas (AWS) 和 Aubrey Oosthuizen (AWS) 创建

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

此模式提供了有关如何将一个账户（账户 A）中的 AWS CodeCommit 存储库与另一个账户 AWS 账户 AWS 账户 （账户 B）中的 Amazon A SageMaker I Studio Classic 关联的说明和代码。要设置关联，您必须在账户 A 中创建 AWS Identity and Access Management (IAM) 策略和角色，在账户 B 中创建 IAM 内联策略。然后，使用外壳脚本将 CodeCommit 存储库从账户 A 克隆到账户 B 中的 Amazon A SageMaker I Classic

先决条件和限制

先决条件

- 第二 AWS 账户，一个包含 CodeCommit 存储库，另一个包含带有用户的 SageMaker AI 域
- 已配置的 SageMaker AI 域和用户，可以访问互联网或通过虚拟专用网络 AWS Security Token Service (VPC AWS STS) 端点访问 CodeCommit 和 ()
- 对 IAM 有基本的了解
- 对 SageMaker AI Studio 经典版的基本理解
- 对 Git 的基本了解和 CodeCommit

限制

此模式仅适用于 SageMaker AI Studio Classic，不适用于亚马 RStudio 逊 SageMaker AI。

架构

技术堆栈

- 亚马逊 SageMaker AI

- 亚马逊 A SageMaker I Studio 经典版
- AWS CodeCommit
- AWS Identity and Access Management (IAM)
- Git

目标架构

下图显示了一种架构，该架构将账户 A CodeCommit 中的存储库与账户 B 中的 SageMaker AI Studio Classic 关联起来

图表显示了以下工作流：

1. 用户通过该`MyCrossAccountRepositoryContributorRole`角色在账户 A 中扮演`sts:AssumeRole`角色，而在账户 B 中使用 SageMaker AI Studio Classic 中的 SageMaker AI 执行角色。代入的角色包括克隆指定存储库并与之交互的 CodeCommit 权限。
2. 在 SageMaker AI Studio Classic 中，用户从系统终端执行 Git 命令。

自动化和扩缩

此模式由手动步骤组成，可以使用AWS Cloud Development Kit (AWS CDK)AWS CloudFormation、或 Terra form 自动执行这些步骤。

工具

AWS 工具

- Amazon SageMaker AI 是一项托管机器学习 (ML) 服务，可帮助您构建和训练机器学习模型，然后将其部署到生产就绪的托管环境中。
- Amazon SageMaker AI Studio Classic 是一个基于 Web 的机器学习集成开发环境 (IDE)，允许您构建、训练、调试、部署和监控您的机器学习模型。
- AWS CodeCommit是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。

  注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。

其他工具

- Git 是分布式版本控制系统，用于追踪软件开发期间源代码的更改。

操作说明

在账户 A 中创建 IAM policy 和 IAM 角色

Task	描述	所需技能
在账户 A 中创建用于存储库访问的 IAM policy。	1. 登录 AWS Management Console 并打开 IAM 控制台。 2. 在导航窗格中选择策略，然后选择创建策略。 3. 选择 JSON 选项卡。 4. 从此模式的其他信息 部分中的示例 IAM policy 中复制策略语句，然后将该语句粘贴到 JSON 编辑器中。请确保替换策略中的所有占位符值。 5. 选择 Next: Tags，然后选择 Next:Review。 6. 在名称中，为策略输入名称。注意：在此模式中，IAM policy 称为 `CrossAccountAccessForMySharedDemoRepo` ，但您可以选择您喜欢的任何策略名称。 7. 选择创建策略。	AWS DevOps

Task	描述	所需技能
	Tip 最佳做法是将 IAM 策略的范围限制为使用案例所需的最低权限。	

Task	描述	所需技能
在账户 A 中创建用于存储库访问的 IAM 角色。	1. 在 IAM 控制台的导航窗格中，选择角色，然后选择创建角色。 2. 对于可信实体类型，请选择 Amazon Web Services account。 3. 在 Amazon Web Services account 部分，选择另一个 Amazon Web Services account。 4. 对于账户 ID，输入账户 B 的账户 ID。 5. 在添加权限页面上，搜索并选择您之前创建的 `CrossAccountAccessForMySharedDemoRepo` 策略。 6. 选择下一步。 7. 对于角色名称，输入一个名称。注意：在此模式中，IAM 角色称为 `MyCrossAccountRepositoryContributorRole` ，但您可以选择您喜欢的任何角色名称。 8. 选择创建角色，然后复制新角色的 Amazon 资源名称（ARN）。	AWS DevOps

在账户 B 中创建 IAM 内联策略

Task	描述	所需技能
将内联策略附加到账户 B 中您的 SageMaker 域用户的执行角色。	1. 在 IAM 控制台的导航窗格中，选择角色。 2. 在账户 B 中搜索并选择与您的 SageMaker AI 域用户关联的执行角色。 3. 选择添加权限，然后选择创建内联策略。 4. 选择 JSON 选项卡。 5. 复制以下策略语句，然后将其粘贴到 JSON 编辑器中。 `{ "Version": "2012-10-17", "Statement": [ { "Sid": "VisualEditor0", "Effect": "Allow", "Action": "sts:AssumeRole", "Resource": "arn:aws:iam::<Account_A_ID>:role/<Account_A_Role_Name>" } ] }` 6. 将 `<Account_A_ID>` 替换为账户 A 的账户 ID。将 `<Account_A_Role_Na`	AWS DevOps

Task	描述	所需技能
	me> 替换为您之前创建的 IAM 角色的名称。 7. 选择查看策略。 8. 对于名称，输入内联策略的名称。 9. 选择创建策略。	

在 SageMaker AI Studio 经典版中为账户 B 克隆存储库

Task	描述	所需技能
在 SageMaker AI Studio Classic 中使用账户 B 创建外壳脚本	1. 在SageMaker 控制台的导航窗格中，选择 Studio。 2. 选择您的用户配置文件，然后选择打开 Studio。 3. 在主页部分中，选择打开启动程序。 4. 在实用程序和文件部分中，选择文本文件。 5. 从此模式的 “其他信息” 部分的 “示例 SageMaker shell 脚本” 中复制脚本，然后将该语句粘贴到新文件中。请确保替换脚本中的所有占位符值。 6. 右键单击新文件的 untitled.txt 选项卡，然后选择重命名文本。对于新名称，输入 cross_account_git_clone.sh，然后选择重命名。	AWS DevOps

Task	描述	所需技能
从系统终端调用 Shell 脚本。	1. 在SageMaker 主机的 “主页” 部分，选择 “打开启动器”。 2. 在实用程序和文件部分中，选择系统终端。 3. 在终端中，运行以下命令： `chmod u+x ./cross_account_git_clone.sh && ./cross_account_git_clone.sh` 您已在 SageMaker AI Studio 跨账户中克隆 CodeCommit 仓库。您现在可以从系统终端执行所有 Git 命令。	AWS DevOps

其他信息

示例 IAM policy

如果您使用此示例策略，请执行以下操作：

- `<CodeCommit_Repository_Region>`替换 AWS 区域 为存储库的。
- 将 `<Account_A_ID>` 替换为账户 A 的账户 ID。
- `<CodeCommit_Repository_Name>`替换为账户 A 中存储 CodeCommit 库的名称。

```
{
"Version": "2012-10-17",
"Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "codecommit:BatchGet*",
            "codecommit:Create*",
```

```
            "codecommit:DeleteBranch",
            "codecommit:Get*",
            "codecommit:List*",
            "codecommit:Describe*",
            "codecommit:Put*",
            "codecommit:Post*",
            "codecommit:Merge*",
            "codecommit:Test*",
            "codecommit:Update*",
            "codecommit:GitPull",
            "codecommit:GitPush"
        ],
        "Resource": [

 "arn:aws:codecommit:<CodeCommit_Repository_Region>:<Account_A_ID>:<CodeCommit_Repository_Name>
        ]
    }
]
}
```

SageMaker AI 外壳脚本示例

如果您使用此示例脚本，请执行以下操作：

- 将 <Account_A_ID>替换为账户 A 的账户 ID。
- 将 <Account_A_Role_Name> 替换为您之前创建的 IAM 角色的名称。
- <CodeCommit_Repository_Region>替换 AWS 区域 为存储库的。
- <CodeCommit_Repository_Name>替换为账户 A 中存储 CodeCommit 库的名称。

```
#!/usr/bin/env bash
#Launch from system terminal
pip install --quiet git-remote-codecommit

mkdir -p ~/.aws
touch ~/.aws/config

echo "[profile CrossAccountAccessProfile]
region = <CodeCommit_Repository_Region>
credential_source=EcsContainer
role_arn = arn:aws:iam::<Account_A_ID>:role/<Account_A_Role_Name>
output = json" > ~/.aws/config
```

```
echo '[credential "https://git-
codecommit.<CodeCommit_Repository_Region>.amazonaws.com"]
        helper = !aws codecommit credential-helper $@ --profile
 CrossAccountAccessProfile
        UseHttpPath = true' > ~/.gitconfig

git clone codecommit::<CodeCommit_Repository_Region>://
CrossAccountAccessProfile@<CodeCommit_Repository_Name>
```

使用 Amazon Textract 从 PDF 文件中自动提取内容

由 Tianxia Jia (AWS) 创建

摘要

许多组织需要从上传至其业务应用程序的 PDF 文件中提取信息。例如，组织可能需要准确地从税务或医疗 PDF 文件中提取用于税务分析或医疗索赔处理的信息。

在 Amazon Web Services (AWS) Cloud 上，Amazon Textract 会自动从 PDF 文件中提取信息（例如已打印文本、表单和表格），并生成包含原始 PDF 文件信息的 JSON 格式文件。您可以在 AWS 管理控制台 中使用 Amazon Textract，也可以通过 API 调用使用。建议使用编程 API 调用来扩展和自动处理大量 PDF 文件。

当 Amazon Textract 处理文件时，它会创建以下 `Block` 对象列表：页面、文本行和单词、表单（键值对）、表格和单元格以及选择元素。还包括其他对象信息，例如边界框、置信区间和关系。IDsAmazon Textract 将提取字符串形式的内容信息。需要正确识别和转换数据值，以便下游应用程序更轻松的使用。

此模式描述了使用 Amazon Textract 自动从 PDF 文件中提取内容并将其处理成干净输出 step-by-step 的工作流程。此模式使用模板匹配技术正确识别必填字段、密钥名称和表，然后对每种数据类型进行后期处理更正。您可以使用此模式处理不同类型的 PDF 文件，然后可以扩展和自动化此工作流程，以处理相同格式的 PDF 文件。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 用于存储待转换为 JPEG 格式以供 Amazon Textract 处理的 PDF 文件的现有 Amazon Simple Storage Service(Amazon S3) 存储桶。有关 S3 存储桶的更多信息，请参阅 Amazon S3 文档中的存储桶概述。
- `Textract_PostProcessing.ipynb` Jupyter 笔记本（附件），已安装并配置。有关 Jupyter 笔记本的更多信息，请参阅亚马逊文档中的创建 Jupyter 笔记本。 SageMaker
- 格式相同的现有 PDF 文件。
- 了解 Python。

限制

- 您的 PDF 文件必须质量良好、且清晰可读。建议使用原生 PDF 文件，但如果所有单词均清晰可见，则可以使用转换至 PDF 格式的扫描文档。有关这方面的更多信息，请参阅 AWS 机器学习博客上的使用 Amazon Textract 预处理 PDF 文档：视觉效果检测和删除。
- 对于多页文件，您可以使用异步操作，或将 PDF 文件拆分为单个页面并使用同步操作。有关这两个选项的更多信息，请参阅 Amazon Textract 文档中的检测和分析多页文档的文本和检测和分析单页文档的文本。

架构

此模式的工作流程是首先在示例 PDF 文件上运行 Amazon Textract（首次运行），然后在与首个 PDF 文件相同格式的 PDF 文件上运行（重复运行）。下图显示了首次运行和重复运行的组合工作流程，该工作流程自动重复地从相同格式的 PDF 文件中提取内容。

图表显示了此模式的以下工作流程：

1. 将 PDF 文件转换为 JPEG 格式，并将其存储在 S3 存储桶中。
2. 调用 Amazon Textract API 并解析 Amazon Textract 响应 JSON 文件。
3. 通过为每个必填字段添加正确的 `KeyName:DataType` 对编辑 JSON 文件。为重复运行阶段创建 `TemplateJSON` 文件。
4. 为每种数据类型（例如浮点数、整数和日期）定义后处理校正函数。
5. 准备与首个 PDF 文件格式相同的 PDF 文件。
6. 调用 Amazon Textract API 并解析 Amazon Textract 响应 JSON。
7. 将已解析 JSON 文件与 `TemplateJSON` 文件相匹配。
8. 实施后期处理校正。

最终的 JSON 输出文件包含了每个必需字段的正确 `KeyName` 和 `Value`。

目标技术堆栈

- Amazon SageMaker
- Amazon S3
- Amazon Textract

自动化和扩缩

您可以使用 AWS Lambda 函数自动执行重复运行工作流程，该函数会在新的 PDF 文件添加至 Amazon S3 中时启动 Amazon Textract。然后，Amazon Textract 会运行处理脚本，并将最终输出保存至存储位置。有关这方面的更多信息，请参阅 Lambda 文档中的使用 Amazon S3 触发器调用 Lambda 函数。

工具

- Amazon SageMaker 是一项完全托管的机器学习服务，可帮助您快速轻松地构建和训练机器学习模型，然后将其直接部署到可用于生产的托管环境中。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon Textract 可以轻松地将文档文本检测和分析添加至您的应用程序。

操作说明

首次运行

Task	描述	所需技能
转换 PDF 文件。	为首次运行准备 PDF 文件，方式为将 PDF 文件拆分为单页，并将其转换为适合 Amazon Textract 同步操作(Syn API) 的 JPEG 格式。 **Note** 您也可以对多页 PDF 文件使用 Amazon Textract 异步操作 (Asyn API)。	数据科学家，开发人员
解析 Amazon Textract 响应 JSON。	打开 `Textract_PostProcessing.ipynb` Jupyter 笔记本（附件），然后使用以	数据科学家，开发人员

Task	描述	所需技能

	下代码调用 Amazon Textract API：
	
	`response = textract.analyze_document( Document={ 'S3Object': { 'Bucket': BUCKET, 'Name': '{}'.format(filename) } }, FeatureTypes=["TABLES", "FORMS"])`
	
	使用以下代码将响应 JSON 解析为表单与表格：
	
	`parseformKV=form_kv_from_JSON(response) parseformTables=get_tables_fromJSON(response)`	

编辑 TemplateJSON 文件。	对每个 KeyName 和对应的 DataType 编辑已解析的 JSON（例如：字符串、浮点、整数或日期）以及表格标题（例如 ColumnNames 和 RowNames）。 此模板用于每种单独的 PDF 文件类型，这意味着该模板可重复用于相同格式的 PDF 文件。	数据科学家，开发人员

Task	描述	所需技能
定义后处理校正函数。	Amazon Textract 对 TemplateJSON 文件的响应中的值为字符串。日期、浮点数、整数或货币无区别。必须将这些值转换至适合下游用例的正确数据类型。 使用以下代码根据 TemplateJSON 文件更正每种数据类型： `finalJSON=postprocessingCorrection(parsedJSON,templateJSON)`	数据科学家，开发人员

重复运行

Task	描述	所需技能
准备 PDF 文件。	准备 PDF 文件，方式为将 PDF 文件拆分为单页，并将其转换为适合 Amazon Textract 同步操作 (Syn API) 的 JPEG 格式。 **Note** 您也可以对多页 PDF 文件使用 Amazon Textract 异步操作 (Asyn API)。	数据科学家，开发人员
调用 Amazon Textract API。	使用以下代码调用 Amazon Textract API：	数据科学家，开发人员

Task	描述	所需技能
	`response = textract.analyze_document( Document={ 'S3Object': { 'Bucket': BUCKET, 'Name': '{}'.format(filename) } }, FeatureTypes=["TABLES", "FORMS"])`	
解析 Amazon Textract 响应 JSON。	使用以下代码将响应 JSON 解析为表单与表格： `parseformKV=form_kv_from_JSON(response) parseformTables=get_tables_fromJSON(response)`	数据科学家，开发人员

Task	描述	所需技能
加载 templateJSON 文件并使其与已解析 JSON 匹配。	通过以下命令使用 TemplateJSON 文件提取正确的键值对和表： `form_kv_corrected= form_kv_correction (parseformKV,templ ateJSON) form_table_correct ed=form_Table_corr ection(parseformTa bles, templateJSON) form_kv_table_correcte d_final={**form_kv _corrected , **form_ta ble_corrected}`	数据科学家，开发人员
后期处理校正。	在 TemplateJSON 文件和后处理功能中使用 DataType，以使用以下代码校正数据： `finalJSON=postproc essingCorrection(f orm_kv_table_corre cted_final,templat eJSON)`	数据科学家，开发人员

相关资源

- 使用 Amazon Textract 从文档中自动提取文本和结构化数据
- 使用 Amazon Textract 提取文本和结构化数据
- Amazon Textract 资源

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

在 Amazon A SageMaker I Studio Lab 中使用 Deepar 计算时间序列构建冷启动预测模型

由 Ivan Cui (AWS) 和 Eyal Shacham (AWS) 创作

摘要

无论您是更有效地为网络流量分配资源，预测患者对人员配备需求的需求，还是预测公司产品的销售，预测都是必不可少的工具。冷启动预测可以对几乎没有历史数据的时间序列进行预测，例如刚刚进入零售市场的新产品。这种模式使用 Amazon SageMaker AI Deepar 预测算法来训练冷启动预测模型，并演示如何对冷启动项目进行预测。

Deepar 是一种监督学习算法，用于使用循环神经网络 (RNN) 预测标量（一维）时间序列。Deepar采用的方法是在相关产品的所有时间序列中联合训练单个模型。

传统的时间序列预测方法，例如自回归积分移动平均线 (ARIMA) 或指数平滑 (ETS)，严重依赖于每种产品的历史时间序列。因此，这些方法对冷启动预测无效。当您的数据集包含数百个相关的时间序列时，DeepAR 要优于标准 ARIMA 和 ETS 方法。您还可以使用经过训练的模型为新的时间序列生成预测，这些预测与其训练的时间序列相似。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- 亚马逊 A SageMaker I 域名。
- 亚马逊 A SageMaker I Studio 实验室或 Jupiter 实验室应用程序。
- 具有读写权限的亚马逊简单存储服务 (Amazon S3) Service 存储桶。
- 具有 Python 编程知识。
- 使用 Jupyter 笔记本的知识。

限制

- 在没有任何历史数据点的情况下调用预测模型将返回错误。使用最少的历史数据点调用模型将返回不准确且高度可信的预测。这种模式为解决冷启动预测的这些已知局限性提供了一种方法。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅按地区划分的 AWS 服务。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

产品版本

- Python 版本 3.10 或更高版本。
- 该模式的笔记本在 Amazon A SageMaker I Studio 中使用带有 Python 3（数据科学）内核的 ml.t3.medium 实例进行了测试。

架构

下图显示了此模式的工作流和体系结构组件。

该工作流执行以下任务：

1. 合成训练和测试数据的输入文件，然后将其上传到 Amazon S3 存储桶。这些数据包括具有分类和动态特征的多个时间序列以及目标值（待预测）。Jupyter 笔记本可视化数据，以更好地了解训练数据的要求和预期的预测值。
2. 创建了超参数调谐器作业，用于训练模型并根据预定义的指标找到最佳模型。
3. 输入文件将从 Amazon S3 存储桶下载到超参数调整任务的每个实例。
4. 在调谐器作业根据调谐器的预定义阈值选择最佳模型后，该模型将部署为 A SageMaker I 端点。
5. 然后，可以调用已部署的模型，并根据测试数据验证其预测。

本笔记本演示了当有足够数量的历史数据点可用时，模型预测目标值的效果如何。但是，当我们使用较少的历史数据点（代表冷产品）调用模型时，即使在模型的置信度范围内，模型的预测也与原始测试数据不匹配。在该模式中，为冷产品构建了一个新模型，其中其初始上下文长度（预测点）定义为可用历史点的数量，并在获取新数据点时迭代训练新模型。笔记本显示，只要历史数据点的数量接近其上下文长度，模型就会有准确的预测。

工具

AWS 服务

- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- Amazon SageMaker AI 是一项托管机器学习 (ML) 服务，可帮助您构建和训练机器学习模型，然后将其部署到生产就绪的托管环境中。

- Amazon SageMaker AI Studio 是一个基于 Web 的机器学习集成开发环境 (IDE)，允许您构建、训练、调试、部署和监控您的机器学习模型。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

其他工具

- Python 是通用的计算机编程语言。

代码存储库

此模式的代码可在 GitHub Deepar ColdProduct- Pattern 存储库中找到。

最佳实践

- 在虚拟环境中训练模型，并始终使用版本控制来实现最高的可重复性。
- 尽可能多地包含高质量的分类特征，以获得最高的预测模型。
- 确保元数据包含相似的分类项目，以便模型能够充分推断出冷启动产物的预测。
- 运行超参数调整作业以获得最高的预测模型。
- 在这种模式下，您构建的模型的上下文长度为 24 小时，这意味着它将预测接下来的 24 小时。如果您尝试预测未来 24 小时的历史数据少于 24 小时，则模型的预测精度会根据历史数据点的数量线性降低。要缓解此问题，请为每组历史数据点创建一个新模型，直到该数字达到所需的预测（上下文）长度。例如，从 2 小时的上下文长度模型开始，然后将模型逐渐增加到 4 小时、8 小时、16 小时和 24 小时。

操作说明

启动你的 SageMaker AI Studio 经典版应用程序

Task	描述	所需技能
启动您的笔记本环境。	1. 登录并打开 SageMaker AI Studio 主页。 AWS Management Console然后选择 “开放工作室”。	数据科学家

Task	描述	所需技能
	2. 在左侧导航窗格中，选择应用程序中的 Studio Classic 图标。然后，选择 “应用程序” 列表上的 “打开” 按钮。 有关更多信息，请参阅 AI 文档中的启动 Amazon SageMaker A SageMaker I Studio。	

创建并激活笔记本

Task	描述	所需技能
为模型训练设置虚拟环境。	要为模型训练设置虚拟环境，请执行以下操作： 1. 将deepar_synthetic.ipynb 笔记本从此模式的 GitHub 存储库下载到本地计算机。 2. 在 Amazon SageMaker AI Studio 经典版中，从 Studio Classic 菜单栏中选择 “上传文件” 图标，然后选择已下载的笔记本电脑。 3. 在左侧导航窗格的文件浏览器中选择笔记本。按照提示设置笔记本环境。选择数据科学 3.0 镜像和 Python 3 内核。 有关更多信息，请参阅 AI 文档中的将文件上传	数据科学家

Task	描述	所需技能
	到 SageMaker AI Studio Classic。 SageMaker	
创建和验证预测模型。	• 按照笔记本中的说明创建训练和测试数据，训练模型，然后调用模型。 • 观察为模型提供足够的历史数据点后预测的准确性。	数据科学家

相关资源

- Deepar 超参数
- 使用 AWS 机器学习服务预测对推出新产品的需求
- 推出亚马逊 A SageMaker I Studio 经典版
- 使用 SageMaker AI Deepar 预测算法

使用亚马逊 A SageMaker I 和 Azure 构建 MLOps 工作流程 DevOps

由 Deepika Kumar (AWS)、Philips Kokoh Prasetyo (AWS) 和 Sara van de Moosdijk (AWS) 创作

摘要

机器学习操作 (MLOps) 是一组自动化和简化机器学习 (ML) 工作流程和部署的实践。 MLOps 侧重于机器学习生命周期的自动化。它有助于确保不仅开发模型，还能系统地、反复地部署、监控和再训练。它为机器学习带来了 DevOps 原理。 MLOps 可以更快地部署机器学习模型，随着时间的推移提高准确性，并更有力地保证机器学习模型提供真正的商业价值。

在开始 MLOps 旅程之前，Organizations 通常会拥有现有的 DevOps 工具和数据存储解决方案。这种模式展示了如何利用微软 Azure 和 AWS. 它可以帮助你将 Azure DevOps 与 Amazon SageMaker AI 集成，从而创建 MLOps 工作流程。

该解决方案简化了 Azure 和之间的工作 AWS。你可以使用 Azure 进行开发和 AWS 机器学习。它促进了从头到尾制作机器学习模型的有效流程，包括数据处理、训练和部署 AWS。为了提高效率，你可以通过 Azure DevOps 管道管理这些流程。该解决方案适用于生成式 AI 中的基础模型操作 (FMOps) 和大型语言模型操作 (LLMOps)，其中包括微调、向量数据库和提示管理。

先决条件和限制

先决条件

- Azure 订阅 — 访问 Azure 服务，例如 Azure DevOps，用于设置持续集成和持续部署 (CI/CD) 管道。
- 活跃 AWS 账户 — 使用此模式中 AWS 服务 使用的账户的权限。
- 数据-访问用于训练机器学习模型的历史数据。
- 熟悉机器学习概念 — 了解 Python、Jupyter 笔记本和机器学习模型开发。
- 安全配置 — 在 Azure 和 AWS 上正确配置角色、策略和权限，以确保数据传输和访问的安全性。
- （可选）矢量数据库-如果您对矢量数据库使用检索增强生成 (RAG) 方法和第三方服务，则需要访问外部矢量数据库。

限制

- 本指南不讨论安全的跨云数据传输。有关跨云数据传输的更多信息，请参阅混合云和多云AWS 解决方案。
- 多云解决方案可能会增加实时数据处理和模型推断的延迟。

- 本指南提供了多账户 MLOps 架构的一个示例。需要根据您的机器学习和 AWS 策略进行调整。
- 除了 Ama SageMaker zon AI 之外，本指南未描述人工智能/机器学习服务的使用情况。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额页面，然后选择服务的链接。

架构

目标架构

目标架构将 Azure DevOps 与 Amazon A SageMaker I 集成在一起，创建了跨云端机器学习工作流程。它将 Azure 用于模型构建和部署、数据准备、基础架构管理的CI/CD processes and SageMaker AI for ML model training and deployment. It outlines the process of obtaining data (from sources such as Amazon S3, Snowflake, and Azure Data Lake) through model building and deployment. Key components include CI/CD管道，并使用 SageMaker Amazon AI 来训练和微调、评估和部署机器学习模型。该架构旨在跨云平台提供高效、自动化和可扩展的机器学习工作流程。

该架构由以下组件组成：

1. 数据科学家在开发账户中执行机器学习实验，通过使用各种数据源探索机器学习用例的不同方法。数据科学家进行单元测试和试验，为了跟踪他们的实验，他们可以将 Amazon AI 与 Amazon A SageMaker I 配合使用 MLflow。在生成式 AI 模型开发中，数据科学家对 Amazon A SageMaker I 模型中心的基础 JumpStart 模型进行微调。模型评估后，数据科学家将代码推送并合并到托管在 Azure 上的模型生成存储库 DevOps。此存储库包含用于多步骤模型构建管道的代码。
2. 在 Azure DevOps 上，提供持续集成 (CI) 的模型生成管道可以在代码合并到主分支时自动或手动激活。在 Automation 账户中，这会激活 SageMaker AI 管道，用于数据预处理、模型训练和微调、模型评估以及基于准确性的条件模型注册。
3. 自动化账户是跨机器学习平台的中央账户，用于托管机器学习环境 (Amazon ECR)、模型 (Amazon S3)、模型元数据（SageMaker AI 模型注册表）、功能（SageMaker AI 功能存储）、自动管道（SageMaker AI 管道）和机器学习日志见解（CloudWatch）。对于生成式 AI 工作负载，您可能需要对下游应用程序中的提示进行额外评估。即时管理应用程序有助于简化和自动化流程。此账户允许机器学习资产的可重复使用性，并强制实施最佳实践，加快机器学习用例的交付。
4. 最新模型版本已添加到 SageMaker AI 模型注册表中以供审核。它跟踪模型版本和相应的工件（血统和元数据）。它还管理模型的状态（批准、拒绝或待定），并管理下游部署的版本。
5. 在模型注册表中训练过的模型通过工作室界面或 API 调用获得批准后，可以向 Amazon 发送事件 EventBridge。 EventBridge 在 Azure 上启动模型部署管道 DevOps。

6. 提供持续部署 (CD) 的 Model Deploy 管道从 Model Deploy 存储库中检出源代码。源代码包含代码、模型部署配置和质量基准测试脚本。模型部署管道可以根据您的推理类型进行定制。
7. 质量控制检查后，模型部署管道会将模型部署到暂存账户。 Staging 账户是生产账户的副本，用于集成测试和评估。对于批量转换，Model Deploy 管道可以自动更新批量推理流程，以使用最新批准的模型版本。对于实时、无服务器或异步推理，它会设置或更新相应的模型端点。
8. 在暂存账户中成功测试后，可以通过模型部署管道手动批准将模型部署到生产账户。该管道在 “部署到生产” 步骤中配置了生产端点，包括模型监控和数据反馈机制。
9. 模型投入生产后，使用 SageMaker AI Model Monitor 和 A SageMaker I Clarify 等工具来识别偏差、检测漂移并持续监控模型的性能。

自动化和扩缩

使用基础设施即代码 (IaC) 自动部署到多个账户和环境。通过实现 MLOps 工作流程设置过程的自动化，可以将从事不同项目的机器学习团队所使用的环境分开。 AWS CloudFormation通过将基础设施视为代码，帮助您建模、配置和管理 AWS 资源。

工具

AWS 服务

- Amazon SageMaker AI 是一项托管机器学习服务，可帮助您构建和训练机器学习模型，然后将其部署到可用于生产的托管环境中。
- AWS Glue是一项完全托管的提取、转换和加载 (ETL) 服务。它可以帮助您在数据存储和数据流之间对数据进行可靠地分类、清理、扩充和移动。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。在这种模式中，Amazon S3 用于数据存储，并与 SageMaker AI 集成用于模型训练和模型对象。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。在这种模式中，Lambda 用于数据预处理和后处理任务。
- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。在这种模式下，它存储 SageMaker AI 用作训练和部署环境的 Docker 容器。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。在这种模式下， EventBridge 协调事件驱动或基于时间的工作流程，以启动自动模型再训练或部署。

- Amazon API Gateway 可帮助您创建、发布、维护、监控和保护任何规模的 RES WebSocket APIs T、HTTP。 在这种模式中，它用于为 SageMaker AI 端点创建面向外部的单一入口点。
- 对于 RAG 应用程序，您可以使用 AWS 服务诸如 Amazon S OpenSearch ervice 和 Amazon RDS for PostgreSQL 存储向量嵌入内容，这些嵌入为 LLM 提供您的内部数据。

其他工具

- Azure DevOps 可帮助你管理 CI/CD 管道并简化代码构建、测试和部署。
- Azure 数据湖存储或 Snowfla ke 可能是机器学习模型训练数据的第三方来源。
- Pinecone、Milvus 或 ChromaDB 可能是用于存储矢量嵌入的第三方矢量数据库。

最佳实践

在实施此多云 MLOps 工作流程的任何组件之前，请完成以下活动：

- 定义和了解机器学习工作流程以及支持该工作流程所需的工具。不同的用例需要不同的工作流程和组件。例如，在个性化用例中，功能重用和低延迟推断可能需要功能存储，但其他用例可能不需要 feature store。要成功定制架构，需要了解数据科学团队的目标工作流程、用例要求和首选的协作方法。
- 为架构的每个组件制定明确的责任分工。在 Azure 数据湖存储、Snowflake 和 Amazon S3 上分布数据存储可能会增加复杂性和成本。如果可能，请选择一致的存储机制。同样，请避免使用 Azure 和 AWS DevOps 服务的组合，或者同时使用 Azure 和 AWS ML 服务。
- 选择一个或多个现有模型和数据集对 MLOps 工作流程进行 end-to-end测试。测试工件应反映平台投入生产时数据科学团队开发的真实用例。

操作说明

设计您的 MLOps 架构

Task	描述	所需技能
识别数据源。	根据当前和未来的用例、可用数据源和数据类型（例如机密数据），记录需要与 MLOps 平台集成的数据源。数据可以	数据工程师、数据科学家、云架构师

Task	描述	所需技能
	存储在 Amazon S3、Azure 数据湖存储、Snowflake 或其他来源中。对于生成式 AI 工作负载，数据还可能包括为生成的响应奠定基础的知识库。这些数据以矢量嵌入的形式存储在矢量数据库中。制定计划，将这些资源与您的平台集成，并确保对正确资源的访问。	
选择适用的服务。	根据数据科学团队所需的工作流程、适用的数据源和现有云架构，通过添加或删除服务来自定义架构。例如，数据工程师和数据科学家可以在 SageMaker AI 或 Amazon EMR 中执行数据预处理和特征工程。 AWS Glue不太可能全部需要这三种服务。	AWS 管理员、数据工程师、数据科学家、机器学习工程师

Task	描述	所需技能
分析安全需求。	收集并记录安全要求。这包括确定： • 哪些团队或工程师可以访问特定的数据源 • 哪些团队或工程师可以访问预先训练好的基础模型 • 是否允许团队访问其他团队的代码和模型 • 对于非开发账户，团队成员应拥有哪些权限（如果有） • 跨云数据传输需要实施哪些安全措施 有关保护生成式 AI 工作负载的更多信息，请参阅保护生成式 AI：生成式 AI 安全范围矩阵简介（AWS 博客文章）。	AWS 管理员、云架构师

设置 AWS Organizations

Task	描述	所需技能
设置 AWS Organizations。	在根 AWS Organizations 上设置 AWS 账户。这可以帮助您管理作为多账户 MLOps 策略的一部分创建的后续账户。有关更多信息，请参阅 AWS Organizations 文档。	AWS 管理员

设置开发环境和版本控制

Task	描述	所需技能
创建 AWS 开发账户。	创建数据工程师和数据科学家有权实验和创建机器学习模型 AWS 账户 的地方。有关说明，请参阅 AWS Organizations 文档中的在组织中创建成员帐户。	AWS 管理员
创建 “模型构建” 存储库。	在 Azure 中创建 Git 存储库，数据科学家可以在实验阶段完成后推送模型构建和部署代码。有关说明，请参阅 Azure DevOps 文档中的设置 Git 存储库。	DevOps 工程师、机器学习工程师
创建模型部署存储库。	在 Azure 中创建用于存储标准部署代码和模板的 Git 存储库。它应包括组织使用的每个部署选项的代码，如设计阶段所确定的那样。例如，它应包括实时端点、异步端点、无服务器推理或批量转换。有关说明，请参阅 Azure DevOps 文档中的设置 Git 存储库。	DevOps 工程师、机器学习工程师
创建 Amazon ECR 存储库。	设置一个 Amazon ECR 存储库，将经批准的机器学习环境存储为 Docker 映像。允许数据科学家和机器学习工程师定义新环境。有关说明，请参阅 Amazon ECR 文档中的创建私有存储库。	机器学习工程师
设置 SageMaker AI 工作室。	根据先前定义的安全要求、首选的数据科学工具（例如）	数据科学家、机器学习工程师、提示工程师

Task	描述	所需技能
	和首选的集成开发环境 (IDE MLflow)，在开发账户上设置 SageMaker AI Studio。使用生命周期配置自动安装关键功能，并为数据科学家创建统一的开发环境。有关更多信息，请参阅 A SageMaker I 文档中的 Amazon AI Studio 和MLflow 跟踪服务器。SageMaker	

集成 CI/CD 管道

Task	描述	所需技能
创建自动化账户。	创建自动管道和作业的运行 AWS 账户 位置。您可以授予数据科学团队对此帐户的读取权限。有关说明，请参阅 AWS Organizations 文档中的在组织中创建成员帐户。	AWS 管理员
设置模型注册表。	在自动化账户中设置 SageMaker AI 模型注册表。该注册表存储机器学习模型的元数据，并帮助某些数据科学家或团队负责人批准或拒绝模型。有关更多信息，请参阅 SageMaker AI 文档中的使用模型注册表注册和部署模型。	机器学习工程师
创建模型构建管道。	在 Azure 中创建 CI/CD 管道，当代码推送到模型生成存储库时，该管道会手动启动或自动启动。管道应查看源代码，并	DevOps 工程师、机器学习工程师

Task	描述	所需技能
	在 Automation 账户中创建或更新 SageMaker AI 管道。管道应向模型注册表中添加一个新模型。有关创建管道的更多信息，请参阅 Azure 管道文档。	

构建部署堆栈

Task	描述	所需技能
创建 AWS 暂存和部署帐户。	AWS 账户 为机器学习模型的暂存和部署而创建。这些帐户应相同，以便在投入生产之前对试运行中的模型进行准确的测试。您可以授予数据科学团队对暂存账户的读取权限。有关说明，请参阅 AWS Organizations 文档中的在组织中创建成员帐户。	AWS 管理员
设置 S3 存储桶以进行模型监控。	如果您要为由 Model Deploy 管道创建的已部署模型启用模型监控，请完成此步骤。创建用于存储输入和输出数据的 Amazon S3 存储桶。有关创建 S3 存储桶的更多信息，请参阅 Amazon S3 文档中的创建存储桶。设置跨账户权限，以便自动模型监控作业在 Automation 账户中运行。有关更多信息，请参阅 SageMaker AI 文档中的监控数据和模型质量。	机器学习工程师
创建模型部署管道。	在 Azure 中创建一个 CI/CD 管道，该管道在模型注册表中	DevOps 工程师、机器学习工程师

Task	描述	所需技能
	获得批准后启动。管道应检查源代码和模型工件，构建用于在暂存账户和生产账户中部署模型的基础架构模板，在暂存账户中部署模型，运行自动测试，等待手动批准，然后将批准的模型部署到生产账户。有关创建管道的更多信息，请参阅 Azure 管道文档。	

（可选）自动化 ML 环境基础架构

Task	描述	所需技能
构建 AWS CDK 或 CloudFormation 模板。	为需要自动部署的所有环境定义 AWS Cloud Development Kit (AWS CDK) 或 AWS CloudFormation 模板。这可能包括开发环境、自动化环境以及暂存和部署环境。有关更多信息，请参阅AWS CDK和CloudFormation文档。	AWS DevOps
创建基础设施管道。	在 Azure 中创建用于基础架构部署的 CI/CD 管道。管理员可以启动此管道来创建机器学习团队所需的新环境 AWS 账户和设置环境。	DevOps 工程师

故障排除

事务	解决方案
监控和偏差检测不足 — 监控不足可能导致错过模型性能问题或数据漂移的检测。	使用 Amazon CloudWatch、A SageMaker I 模型监控器和 SageMaker AI Clarify 等工具加强监控框架。配置警报，以便对已发现的问题立即采取行动。
CI 管道触发错误 — 由于配置错误， DevOps 可能不会在代码合并时触发 Azure 中的 CI 管道。	检查 Azure DevOps 项目设置，确保 webhook 已正确设置并指向正确的 SageMaker AI 端点。
治理 — 中央自动化账户可能无法在机器学习平台上强制执行最佳实践，从而导致工作流程不一致。	审计 Automation 账户设置，确保所有机器学习环境和模型都符合预定义的最佳实践和策略。
模型注册机构批准延迟 — 当模型的检查和批准延迟时，就会发生这种情况，要么是因为人们需要时间进行审查，要么是因为技术问题。	实施通知系统，提醒利益相关者注意待批准的模型，并简化审核流程。
模型部署事件失败 — 为启动模型部署管道而调度的事件可能会失败，从而导致部署延迟。	确认 Amazon EventBridge 拥有成功调用 Azure DevOps 管道的正确权限和事件模式。
生产部署瓶颈 — 手动批准流程可能会造成瓶颈，从而延迟模型的生产部署。	优化模型部署管道内的审批工作流程，促进及时审核和畅通沟通渠道。

相关资源

AWS 文档

- 亚马逊 SageMaker AI 文档
- M@@ achine Learning Lens（架构AWS 良好的框架）
- 成功规划 MLOps（AWS 规范性指导）

其他 AWS 资源

- MLOps 采@@ 用 Amazon A SageMaker I 的企业的基础路线图（AWS 博客文章）

- AWS 2022 年澳新银行峰会- End-to-end MLOps 面向建筑师（YouTube 视频）
- FMOps/LLMOps: 将生成式 AI 和差异化付诸实践 MLOps（AWS 博客文章）
- 使用 Amazon A SageMaker I Clarify 和 MLOps 服务大规模实施法学硕士评估（AWS 博客文章）
- 矢量数据库在生成式人工智能应用中的作用（AWS 博客文章）

Azure 文档

- Azure DevOps 文档
- Azure 管道文档

使用在 Amazon Bedrock 中配置模型调用日志 AWS CloudFormation

由 Vikramaditya Bhatnagar (AWS) 创作

摘要

您可以将 Amazon Bedrock 配置为收集您的中所有模型调用的调用日志、模型输入数据和模型输出数据。 AWS 账户这是使用 Amazon Bedrock 构建强大的生成式 AI 应用程序的最佳实践。您可以将模型调用日志存储在 Amazon Logs CloudWatch 日志组、亚马逊简单存储服务 (Amazon S3) Simple Service 存储桶中，或者同时存储在两者中。在日志中 CloudWatch 保存日志数据可以帮助您创建自定义指标筛选器、警报和仪表板。根据贵组织的政策，Amazon S3 非常适合跨部门复制数据 AWS 区域或用于长期存储。

此模式提供了一个示例 AWS CloudFormation 模板，该模板使用基础设施即代码 (IaC) 方法为 Amazon Bedrock 配置模型调用日志。该模板在日志和 Amazon S3 中配置 CloudWatch 日志存储。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 具备以下权限：
  - 创建 CloudFormation 堆栈的@@ 权限
  - 访问 Amazon Bedrock 的@@ 权限
  - 创建和访问 Amazon S3 存储桶的@@ 权限
  - 创建和访问 CloudWatch 日志组的@@ 权限
  - 创建和访问 AWS Lambda 函数的@@ 权限
  - 创建和访问 AWS Key Management Service (AWS KMS) 密钥的@@ 权限

限制

此模式将模型调用记录到 CloudWatch 日志和 Amazon S3。它不支持只选择这两个服务中的一个。

架构

目标架构

该 CloudFormation 模板在您的目标中预置了以下资源 AWS 账户：

- 用于存储模型调用 CloudWatch 日志的日志日志组
- 用于存储模型调用日志的 Amazon S3 存储桶和相应的存储桶策略
- 用于存储服务器端访问日志的 Amazon S3 存储桶和相应的存储桶策略
- 在 Amazon Bedrock 中配置日志设置的 AWS Lambda 功能
- AWS KMS key 和相应的密钥别名
- 亚马逊 Bedrock 的 AWS Identity and Access Management (IAM) 服务角色

下图显示了部署与此模式关联的 CloudFormation 堆栈后如何存储调用日志。当基础模型提供文本、图像、视频或嵌入数据时，Amazon Bedrock 会发布日志数据。如图所示，Amazon S3 存储桶和 CloudWatch 日志日志组使用加 AWS KMS key密。

图表显示了以下工作流：

1. 用户向 Amazon Bedrock 中的基础模型提交查询。
2. Amazon Bedrock 担任 IAM 服务角色。
3. Amazon Bedrock 生成日志数据并将其存储在 CloudWatch 日志日志组和 Amazon S3 存储桶中。
4. 如果用户读取、上传或删除包含模型调用日志的 Amazon S3 存储桶中的任何文件，则这些活动将记录在另一个 Amazon S3 存储桶中以获取服务器端访问日志。

自动化和扩缩

要扩展此解决方案，您可以将 CloudFormation 模板部署为多个 AWS 区域 和 AWS 账户。有关更多信息，请参阅 CloudFormation 文档中的使用管理跨账户和区域 StackSets的堆栈。

工具

AWS 服务

- Amazon Bedrock 是一项完全托管的服务，它通过统一的 API 提供来自领先人工智能公司和亚马逊的高性能基础模型 (FMs) 供您使用。
- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。

- Amazon CloudWatch L ogs 可帮助您集中所有系统和应用程序的日志， AWS 服务 这样您就可以监控它们并安全地将其存档。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Key Management Service (AWS KMS) 帮助您创建和控制加密密钥以帮助保护您的数据。
- Amazon Simple Storage Service (Amazon S3) 是一项对象存储服务，可提供业界领先的可扩展性、数据可用性、安全性和性能。

其他工具

- Git 是一个开源的分布式版本控制系统。

代码存储库

此模式的代码可在 GitHub enable-bedrock-logging-using-cloudformation 存储库中找到。

操作说明

创建堆 CloudFormation 栈

Task	描述	所需技能
下载 CloudFormation 模板。	从 GitHub 存储库下载CloudFormation 模板。	云架构师
部署 模板。	在您的目标账户和区域中创建堆栈。在参数部分中，为模板中定义的参数指定值。有关说明，请参阅 CloudFormation 文档中的创建堆栈。	云架构师

测试解决方案

Task	描述	所需技能
启用模型访问权限。	在 Amazon Bedrock 中，添加对基础模型的访问权限。有关说明，请参阅亚马逊 Bedrock 文档中的添加或删除对亚马逊 Bedrock 基础模型的访问权限。	云架构师
运行示例提示。	在 Amazon Bedrock 游乐场中，运行示例提示。有关说明，请参阅 Amazon Bedrock 文档中的使用游戏场在控制台中生成响应。	云架构师
查看日志配置。	1. 登录 Amazon Bedrock 控制台。 2. 在导航栏中，选择 CloudFormation 堆栈的部署 AWS 区域 位置。 3. 在左侧导航窗格的 B edrock 配置下，选择设置。 4. 请确认以下内容： • 模型调用日志已启用。 • 所有数据类型均已选中。 • 对于日志目的地，同时选择 S3 和 CloudWatch 日志。	云架构师
查看亚马逊 S3 存储桶。	1. 在 S3 配置部分，选择浏览 S3。这将在 Amazon S3 控制台中打开目标存储桶。 2. 确认您之前运行的示例提示中存在日志数据。	云架构师

Task	描述	所需技能
查看日志组。	1. 返回亚马逊 Bedrock 控制台中的 “设置” 页面。 2. 在CloudWatch 日志配置部分，查看 CloudWatch 日志日志组的设置。记下日志组的名称。 3. 打开 CloudWatch 管理控制台。 4. 在导航窗格中，在日志下选择日志组。 5. 选择 Amazon Bedrock 发布日志数据的日志组的名称。 6. 确认您之前运行的示例提示中存在日志数据。	云架构师

相关资源

AWS 文档

- 访问亚马逊 S3 存储桶（亚马逊 S3 文档）
- 创建和管理堆栈（CloudFormation 文档）
- 监控模型调用（Amazon Bedrock 文档）
- 使用日志组和日志流（CloudWatch 日志文档）

AWS 博客文章

- 使用 Amazon Bedrock 和 Amazon 集成监控生成式 A CloudWatch I 应用程序
- 使用 Amazon Bedrock Agents 构建强大的生成式 AI 应用程序的最佳实践 — 第 1 部分
- 使用 Amazon Bedrock Agents 构建强大的生成式 AI 应用程序的最佳实践 — 第 2 部分

为 AWS Step Functions SageMaker 创建自定义 Docker 容器镜像并将其用于模型训练

由 Julia Bluszcz (AWS)、Neha Sharma (AWS)、Aubrey Oosthuizen (AWS)、Mohan Gowda Purushothama (AWS) 和 Mateusz Zaremba (AWS) 创作

摘要

此模式展示了如何在 AWS Step Functions 中为亚马逊创建 Docker 容器镜像 SageMaker并将其用于 AWS Step Functions 中的训练模型。通过将自定义算法打包到容器中，您几乎可以在 SageMaker 环境中运行任何代码，无论编程语言、框架或依赖关系如何。

在提供的示例SageMaker 笔记本中，自定义 Docker 容器镜像存储在亚马逊弹性容器注册表 (Amazon ECR) Container Registry 中。然后，Step Functions 使用存储在 Amazon ECR 中的容器为其运行 Python 处理脚本。 SageMaker然后，容器将模型导出到亚马逊简单存储服务 (Amazon S3) Semple S3。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- SageMaker具有 Amazon S3 权限的 AWS 身份和访问管理 (IAM) 角色
- Step Functions 的 IAM 角色
- 基本熟悉 Python
- 熟悉亚马逊 SageMaker Python 开发工具包
- 熟悉 AWS 命令行界面（AWS CLI）
- 熟悉适用于 Python 的 Amazon SDK (Boto3)
- 熟悉 Amazon ECR
- 熟悉 Docker

产品版本

- AWS Step Functions 数据科学软件开发工具包版本 2.3.0
- 亚马逊 SageMaker Python SDK 版本 2.78.0

架构

下图显示了在 Step Functions 中为其创建 Docker 容器镜像 SageMaker，然后将其用于训练模型的示例工作流程：

图表显示了以下工作流：

1. 数据科学家或 DevOps 工程师使用 Amazon SageMaker 笔记本创建自定义 Docker 容器镜像。
2. 数据科学家或 DevOps 工程师将 Docker 容器映像存储在私有注册表中的 Amazon ECR 私有存储库中。
3. 数据科学家或 DevOps 工程师使用 Docker 容器在 Step Functions 工作流程中运行 Python SageMaker 处理作业。

自动化和扩缩

此模式中的示例 SageMaker 笔记本使用`ml.m5.xlarge`笔记本实例类型。您可以更改实例类型，以适合您的用例。有关 SageMaker 笔记本实例类型的更多信息，请参阅 Amazon SageMaker 定价。

工具

- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- Amazon SageMaker 是一项托管机器学习 (ML) 服务，可帮助您构建和训练机器学习模型，然后将其部署到生产就绪的托管环境中。
- Amaz SageMaker on Python 软件开发工具包是一个开源库，用于在上 SageMaker训练和部署机器学习模型。
- AWS Step Functions 是一项无服务器编排服务，可让您搭配使用 AWS Lambda 函数和其他 Amazon Web Services 来构建业务关键型应用程序。
- AWS Step Functions 数据科学 Python SDK 是一个开源库，可帮助您创建处理和发布机器学习模型的 Step Functions 工作流程。

操作说明

创建自定义 Docker 容器映像，并将其存储在 Amazon ECR 中

Task	描述	所需技能
设置 Amazon ECR 并新建私有注册表。	如果您尚未设置 Amazon ECR，请按照 Amazon ECR 用户指南中的设置 Amazon ECR进行操作。每个 Amazon Web Services account 都提供有原定设置的私有 Amazon ECR 注册表。	DevOps 工程师
创建 Amazon ECR 私有存储库。	请按照 Amazon ECR 用户指南中的创建私有存储库进行操作。 **Note** 您创建的存储库是存储自定义 Docker 容器镜像的地方。	DevOps 工程师
创建一个 Dockerfile，其中包含运行 SageMaker 处理作业所需的规范。	通过配置 Dockerfile 来创建包含运行 SageMaker 处理作业所需的规格的 Dockerfile。有关说明，请参阅《Amazon SageMaker 开发者指南》中的调整自己的训练容器。 有关 Dockerfiles 的更多信息，请参阅 Doc ker 文档中的 DockerFile 参考。 用于创建 DockerFile 的 Jupyter 笔记本代码单元格示例	DevOps 工程师

Task	描述	所需技能

	单元格 1
	
	`# Make docker folder`
	`!mkdir -p docker`
	
	单元格 2
	
	`%%writefile docker/Dockerfile`
	
	`FROM python:3.7-slim-buster`
	
	`RUN pip3 install pandas==0.25.3 scikit-learn==0.21.3`
	`ENV PYTHONUNBUFFERED=TRUE`
	
	`ENTRYPOINT ["python3"]`	

Task	描述	所需技能

构建您的 Docker 容器映像并将其推送至 Amazon ECR。	1. 通过在 AWS CLI 中运行 `docker build` 命令创建 Dockerfile，然后使用此 Dockerfile 构建容器映像。
	2. 通过运行 `docker push` 命令将容器映像推送至 Amazon ECR。
	
	有关更多信息，请参阅 “在上构建自己的算法容器” 中的 “构建和注册容器” GitHub。
	
	用于构建和注册 Docker 映像的 Jupyter 笔记本代码单元格示例
	
	Important
	在运行以下单元格之前，请确保已创建一个 Dockerfile 并将其存储在名为的目录中。`docker`此外，请确保您已创建一个 Amazon ECR 存储库，并将第一个单元格中的 `ecr_repository` 值替换为存储库名称。
	
	单元格 1
	
	`import boto3`
	`tag = ':latest'`
		DevOps 工程师

Task	描述	所需技能
	account_id = boto3.cli ent('sts').get_cal ler_identity().get ('Account') region = boto3.Ses sion().region_name ecr_repository = 'byoc' image_uri = '{}.dkr.e cr.{}.amazonaws.com/ {}'.format(account_id, region, ecr_repository + tag) 单元格 2 # Build docker image !docker build -t $image_uri docker 单元格 3 # Authenticate to ECR !aws ecr get-login -password --region {region} | docker login --username AWS --password-stdin {account_id}.dkr.ecr. {region}.amazonaws.com 单元格 4 # Push docker image !docker push $image_ur i	

Task	描述	所需技能
	Note 您必须向私有注册表对 Docker 客户端进行身份验证，这样才能使用`docker push`和`docker pull`命令。这些命令将图像推送和拉出注册表中存储库。	

创建采用您自定义 Docker 容器映像的 Step Functions 工作流

Task	描述	所需技能

创建包含自定义处理和模型训练逻辑的 Python 脚本。	编写将在数据处理脚本中运行的自定义处理逻辑。然后，将其另存为名为 `training.py` 的 Python 脚本。
	
	有关更多信息，请参阅在开启 SageMaker 脚本模式的情况下自带模型 GitHub。
	
	含自定义处理和模型训练逻辑的 Python 脚本示例
	
	`%%writefile training.py`
	`from numpy import empty`
	`import pandas as pd`
	`import os`
	`from sklearn import datasets, svm`
	`from joblib import dump, load`
		数据科学家

Task	描述	所需技能

	<pre>if __name__ == '__main__':
	    digits = datasets.load_digits()
	    #create classifier object
	    clf = svm.SVC(gamma=0.001, C=100.)
	
	    #fit the model
	    clf.fit(digits.data[:-1], digits.target[:-1])
	
	    #model output in binary format
	    output_path = os.path.join('/opt/ml/processing/model', "model.joblib")
	    dump(clf, output_path)</pre>	

Task	描述	所需技能

创建一个 Step Functions 工作流程，其中包含您的 SageMaker 处理作业。	安装并导入 AWS Step Functions Data Science SDK，然后将 training.py 文件上传至 Amazon S3。然后，使用 Amaz SageMaker on Python 软件开发工具包在 Step Functions 中定义处理步骤。
	
	Important
	确保您已在您的 AWS 账户中为 Step Functions 创建了 IAM 执行角色。
	
	要上传至 Amazon S3 的环境设置和自定义训练脚本示例
	
	`!pip install stepfunct ions`
	
	`import boto3`
	`import stepfunctions`
	`import sagemaker`
	`import datetime`
	
	`from stepfunctions import steps`
	`from stepfunct ions.inputs import ExecutionInput`
	`from stepfunct ions.steps import ( Chain )`
		数据科学家

Task	描述	所需技能
	from stepfunct ions.workflow import Workflow from sagemaker .processing import ScriptProcessor, ProcessingInput, ProcessingOutput sagemaker_session = sagemaker.Session() bucket = sagemaker _session.default_b ucket() role = sagemaker .get_execution_role() prefix = 'byoc-tra ining-model' # See prerequisites section to create this role workflow_execution_rol e = f"arn:aws:iam:: {account_id}:role/Ama zonSageMaker-StepF unctionsWorkflowEx ecutionRole" execution_input = ExecutionInput(schema={ "Preproce ssingJobName": str}) input_code = sagemaker _session.upload_data("training.py", bucket=bucket, key_prefix="prepro cessing.py",	

Task	描述	所需技能
	) 使用自定义 Amazon ECR 图像和 Python 脚本的 SageMaker 处理步骤定义示例 **Note** 确保使用`execution_input` 参数来指定作业名称。每次运行作业时，参数值必须是唯一的。此外，training.py 文件的代码作为 `input` 参数传递至 `ProcessingStep` ，这意味着它将被复制到容器中。`ProcessingInput` 代码的目标与 `container_entrypoint` 内部的第二个参数相同。 `script_processor = ScriptProcessor(command=['python3'], image_uri=image_uri, role=role, instance_count=1, instance_type='ml.m5.xlarge')`	

Task	描述	所需技能
	<pre>processing_step = steps.ProcessingStep("training-step", processor=script_processor, job_name=execution_input["PreprocessingJobName"], inputs=[ProcessingInput(source=input_code, destination="/opt/ml/processing/input/code", input_name="code",),], outputs=[ProcessingOutput(source='/opt/ml/processing/model', destination="s3://{}/{}".format(bucket, prefix), output_name='byoc-example')], container_entrypoint=["python3", "/opt/ml/processing/input/code/training.py"],)</pre>	

Task	描述	所需技能

	运行处理作业的 Step Functions 工作 SageMaker 流程示例
	
	Note
	此示例工作流仅包括 SageMaker 处理作业步骤，不包括完整的 Step Functions 工作流程。有关完整的工作流程示例，请参阅 AWS Step Functions 数据科学软件开发工具包文档 SageMaker中的示例笔记本。
	
	`workflow_graph = Chain([processing_step])`
	
	`workflow = Workflow( name="ProcessingWorkflow", definition=workflow_graph, role=workflow_execution_role )`
	
	`workflow.create()`
	`# Execute workflow`
	`execution = workflow.execute( inputs={ "PreprocessingJobName":`	

Task	描述	所需技能
	str(datetime.datet ime.now().strftime ("%Y%m%d%H%M-%SS")), # Each pre processin g job (SageMaker processing job) requires a unique name, }) execution_output = execution.get_outp ut(wait=True)	

相关资源

- 处理数据（Amazon SageMaker 开发者指南）
- 调整自己的训练容器（Amazon SageMaker 开发者指南）

使用 Amazon Bedrock 代理通过基于文本的提示在 Amazon EKS 中自动创建访问入口控制

由 Keshav Ganesh (AWS) 和 Sudhanshu Saurav (AWS) 创作

摘要

当多个团队需要使用共享的 Amazon Elastic Kubernetes Service (Amazon EKS) 集群时，组织在管理访问控制和资源配置方面面临挑战。诸如 Amazon EKS 之类的托管 Kubernetes 服务简化了集群操作。但是，管理团队访问权限和资源权限的管理开销仍然复杂且耗时。

此模式显示了 Amazon Bedrock 代理如何帮助您自动执行 Amazon EKS 集群访问管理。这种自动化使开发团队能够专注于其核心应用程序开发，而不必处理访问控制的设置和管理。您可以自定义 Amazon Bedrock 代理，使其通过简单的自然语言提示对各种任务执行操作。

通过使用 AWS Lambda 功能作为操作组，Amazon Bedrock 代理可以处理诸如创建用户访问条目和管理访问策略之类的任务。此外，Amazon Bedrock 代理可以配置容器身份关联，从而允许集群中运行的 pod 访问 AWS Identity and Access Management (IAM) 资源。使用此解决方案，组织可以通过基于文本的简单提示来简化 Amazon EKS 集群管理，减少手动开销，并提高整体开发效率。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- 已为部署过程设定了 IAM 角色和权限。这包括访问 Amazon Bedrock 基础模型 (FM)、创建 Lambda 函数以及跨目标的任何其他所需资源的权限。 AWS 账户
- 在活跃状态 AWS 账户 下@@ 可以访问这些 Amazon Bedrock FMs：Amazon Titan Text Embeddings V2 和 Anthropic Claude 3 Haiku。
- AWS Command Line Interface (AWS CLI) 版本 2.9.11 或更高版本，已安装并配置。
- eksctl 0.194.0 或更高版本，已安装。

限制

- 可能需要培训和文档，以帮助确保顺利采用和有效使用这些技术。使用亚马逊 Bedrock，亚马逊 EKS、Lambda、A OpenSearch mazon Service 和 OpenAPI 会给开发人员和团队带来重要的学习曲线。 DevOps

- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅按地区划分的 AWS 服务。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

架构

下图显示了此模式的工作流和体系结构组件。

此解决方案执行以下步骤：

1. 用户通过提交提示或查询来与 Amazon Bedrock 代理进行互动，这些提示或查询用作代理处理和采取行动的输入。
2. 根据提示，Amazon Bedrock 代理会检查 OpenAPI 架构，以确定要定位的正确 API。如果 Amazon Bedrock 代理找到了正确的 API 调用，则请求将发送到与实现这些操作的 Lambda 函数关联的操作组。
3. 如果找不到相关的 API，Amazon Bedrock 代理会查询该 OpenSearch 馆藏。该 OpenSearch 馆藏使用索引知识库内容，这些内容来自包含 Amazon EKS 用户指南的 Amazon S3 存储桶。
4. 该 OpenSearch 集合将相关的上下文信息返回给 Amazon Bedrock 代理。
5. 对于可操作的请求（与 API 操作相匹配的请求），Amazon Bedrock 代理在虚拟私有云 (VPC) 中执行并触发 Lambda 函数。
6. Lambda 函数根据用户在 Amazon EKS 集群中的输入执行操作。
7. 用于 Lambda 代码的 Amazon S3 存储桶存储包含为 Lambda 函数编写的代码和逻辑的项目。

工具

AWS 服务

- Amazon Bedrock 是一项完全托管的服务，它通过统一的 API 提供来自领先的人工智能初创公司和亚马逊的高性能基础模型 (FMs) 供您使用。
- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。
- 亚马逊 Elastic Kubernetes Service（亚马逊 EKS）可帮助你在上面运行 AWS Kubernetes，而无需安装或维护自己的 Kubernetes 控制平面或节点。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。

- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon OpenSearch Service 是一项托管服务，可帮助您在中部署、操作和扩展 OpenSearch 集群 AWS Cloud。它的集合功能可帮助您整理数据并建立全面的知识库，供诸如Amazon Bedrock代理之类的人工智能助手使用。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

其他工具

- eksctl 是一种用于在 Amazon EKS 上创建和管理 Kubernetes 集群的命令行实用程序。

代码存储库

此模式的代码可在 GitHub eks-access-controls-bedrock-agent 存储库中找到。

最佳实践

- 在实施此模式时，请保持尽可能高的安全性。确保 Amazon EKS 集群是私有的，访问权限有限，并且所有资源都在虚拟私有云 (VPC) 内。有关更多信息，请参阅 Amazon EKS 文档中的安全最佳实践。
- 尽可能使用 AWS KMS 客户托管的密钥，并向其授予有限的访问权限。
- 遵循最低权限原则，授予执行任务所需的最低权限。有关更多信息，请参阅 IAM 文档中的授予最低权限和安全最佳实践。

操作说明

设置环境

Task	描述	所需技能
克隆存储库。	要克隆此模式的存储库，请在本地工作站中运行以下命令： `git clone https://github.com/aws-samp`	AWS DevOps

Task	描述	所需技能
	`les/eks-access-con trols-bedrock-agen t.git`	
获取 AWS 账户 身份证。	要获取 AWS 账户 ID，请按以下步骤操作： 1. 在克隆存储库的根文件夹中打开一个 shell，。`eks-access-controls-bedrock-agent` 2. 要获取您的 AWS 账户 ID，请导航到克隆的目录并运行以下命令： `AWS_ACCOUNT=$(aws sts get-caller-identity --query "Account" --output text)` 此命令将您的 AWS 账户 ID 存储在`AWS_ACCOUNT` 变量中。	AWS DevOps

Task	描述	所需技能

为 Lambda 代码创建 S3 存储桶。	要实施此解决方案，您必须创建三个用于不同用途的 Amazon S3 存储桶，如架构图所示。S3 存储桶用于存放 Lambda 代码、知识库和 OpenAPI 架构。
	
	要创建 Lambda 代码存储桶，请使用以下步骤：
	
	1. 要为 Lambda 代码创建 S3 存储桶，请运行以下命令：
	
	`aws s3 mb s3://bedrock-agent-lambda-artifacts-${AWS_ACCOUNT} --region us-east-1`
	
	2. 要安装 Lambda 代码依赖项，请运行以下命令：
	
	`cd eks-lambda`
	`npm install`
	`tsc`
	`cd .. && cd opensearch-lambda`
	`npm install`
	`tsc`
	`cd ..`
	
	3. 要打包代码并将其上传到 Lambda 的 S3 存储桶，请运行以下命令：
	
	`aws cloudformation package \`
		AWS DevOps

Task	描述	所需技能

	<pre>--template-file eks-access-controls.yaml \
	--s3-bucket bedrock-agent-lambda-artifacts-${AWS_ACCOUNT} \
	--output-template-file eks-access-controls-template.yaml \
	--region us-east-1</pre>
	package 命令创建一个包含以下内容的新 CloudFormation 模板 (`eks-access-controls-template.yaml`)：
	• 对存储在您的 S3 存储桶中的 Lambda 函数代码的引用。
	• 所有必需 AWS 基础设施的定义，包括 VPC、子网、Amazon Bedrock 代理和集合。 OpenSearch 您可以使用此模板部署完整的解决方案 CloudFormation。	

Task	描述	所需技能

为知识库创建 S3 存储桶。	要为知识库创建 Amazon S3 存储桶，请使用以下步骤：
	
	1. 要为知识库创建 Amazon S3 存储桶，请运行以下命令：
	
	`aws s3 mb s3://eks-knowledge-base-${AWS_ACCOUNT} --region us-east-1`
	
	2. 要下载 Amazon EKS 用户指南并将其存储在目录中，请运行以下命令：
	
	`mkdir dataSource`
	`cd dataSource`
	`curl https://docs.aws.amazon.com/pdfs/eks/latest/userguide/eks-ug.pdf -o eks-user-guide.pdf`
	
	3. 要将用户指南上传到您在步骤 1 中创建的 S3 存储桶，请运行以下命令：
	
	`aws s3 cp eks-user-guide.pdf s3://eks-knowledge-base-${AWS_ACCOUNT} \`
	`--region us-east-1 \`
	
	4. 要返回根目录，请运行以下命令：
	
	`cd ..`
		AWS DevOps

Task	描述	所需技能

为 OpenAPI 架构创建 S3 存储桶。	要为 OpenAPI 架构创建 Amazon S3 存储桶，请使用以下步骤：
	1. 要创建 S3 存储桶，请运行以下命令：
	`aws s3 mb s3://eks-openapi-schema-${AWS_ACCOUNT} --region us-east-1`
	2. 要将 OpenAPI 架构上传到 S3 存储桶，请运行以下命令：
	`aws s3 cp openapi-schema.yaml s3://eks-openapi-schema-${AWS_ACCOUNT} --region us-east-1`
		AWS DevOps

部署堆 CloudFormation 栈

Task	描述	所需技能
部署 CloudFormation 堆栈。	要部署 CloudFormation 堆栈，请使用之前创建`eks-access-controls-template.yaml` 的 CloudFormation 模板文件。有关更多详细说明，请参阅 CloudFormation 文档中的从 CloudFormation 控制台创建堆栈。	AWS DevOps

Task	描述	所需技能
	Note 使用 CloudFormation 模板配置 OpenSearch 索引大约需要 10 分钟。 创建堆栈后，记下`VPC_ID`和`PRIVATE_SUBNET_ID`s。	

Task	描述	所需技能

创建 Amazon EKS 集群。	要在 VPC 内创建 Amazon EKS 集群，请使用以下步骤：
	1. 创建`eks-config.yaml` 配置文件的副本，并将副本命名为`eks-deploy.yaml` 。
	2. 在文本编辑器中打开 `eks-deploy.yaml` 。然后，将以下占位符值替换为已部署堆栈中的值：`VPC_ID`、`PRIVATE_SUBNET1` 、和 `PRIVATE_SUBNET2`
	3. 要使用 eksctl 实用程序创建集群，请运行以下命令：
	`eksctl create cluster -f eks-deploy.yaml`
	Note
	此集群创建过程最多可能需要 15-20 分钟才能完成。
	4. 要验证集群是否已成功创建，请运行以下命令：
	`aws eks describe-cluster --name --query "cluster.status"`
	`aws eks update-kubeconfig --name --region`
		AWS DevOps

Task	描述	所需技能
	`kubectl get nodes` 预期结果如下： • 集群状态为ACTIVE。 • 该命令kubectl get nodes显示所有节点都处于Ready状态。	

连接 Lambda 函数和 Amazon EKS 集群

Task	描述	所需技能
在 Amazon EKS 集群和 Lambda 函数之间创建连接。	要设置网络和 IAM 权限以允许 Lambda 函数与 Amazon EKS 集群进行通信，请使用以下步骤： 1. 要识别附加到 Lambda 函数的 IAM 角色，请打开 AWS Management Console 并找到名为的 Lambda 函数。bedrock-agent-eks-access-control 记下 IAM 角色的亚马逊资源名称 (ARN)。 2. 要在 Amazon EKS 集群中为 Lambda 函数的 IAM 角色创建访问条目，请运行以下命令： `aws eks create-access-entry --cluster-name eks-testing-`	AWS DevOps

Task	描述	所需技能

	`cluster --principal-arn <principal-Role-ARN>`
	3. 要为该角色分配AmazonEKS ClusterAdminPolicy 权限，请运行以下命令：
	`aws eks associate-access-policy --cluster-name eks-testing-cluster --principal-arn <principal-Role-ARN> --policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy --access-scope type=cluster`
	有关更多信息，请参阅将访问策略与访问条目关联以及 Amazon EKS 文档 EKSClusterAdminPolicy中的 Amazon。
	4. 找到 Amazon EKS 集群的安全组。添加入站规则，允许从 Lambda 函数传入网络流量到 Amazon EKS 集群。
	为入站规则使用以下值：
	• 类型 – HTTPS
	• 端口范围 –443
	• 来源 — Lambda 安全组	

Task	描述	所需技能
	有关更多信息，请参阅 Amazon VPC 文档中的配置安全组规则。	

测试解决方案

Task	描述	所需技能
测试 Amazon Bedrock 代理。	在测试 Amazon Bedrock 代理之前，请务必执行以下操作： • 首先使用非生产角色进行测试。 • 记录对集群访问权限所做的任何更改。 • 制定计划，以便在需要时恢复更改。 要访问 Amazon Bedrock 代理，请使用以下步骤： 1. AWS Management Console 使用具有 Amazon Bedrock 权限的 IAM 角色登录，然后通过以下网址打开 Amazon Bedrock 控制台。https://console.aws.amazon.com/bedrock/ 2. 从左侧导航窗格中选择代理。然后，在 “代理” 部分选择您配置的代理。 3. 要测试代理，请尝试以下示例提示，其中将`Principal`	AWS DevOps

Task	描述	所需技能
	-ARN-OF-ROLE 替换为实际的 IAM 角色 ARN： • 要为您想要提供对 EKS 集群的访问权限的任何 IAM 角色创建访问条目，请使用以下提示：`Create an access entry in cluster eks-testing-new for a role whose principal arn is <Principal-ARN-OF-ROLE> with access policy as AmazonEKSAdminPolicy` 预期结果： • 代理应确认访问条目的创建。 • 要进行验证，请使用 AWS Management Console 或使用 Amazon EKS API 进行检查，然后运行以下命令：`aws eks list-access-entries --cluster-name ekscluster` • 要描述您创建的访问条目，请使用以下提示：`Describe an access entry in cluster eks-testing-new whose principal`	

Task	描述	所需技能
	`arn is <Principal-ARN-OF-ROLE>` 预期结果： • 代理应返回有关访问条目的详细信息。 • 详细信息应与您之前为访问条目配置的内容一致。 • 要删除您创建的访问条目，请使用以下提示：`Delete the access entry in cluster eks-testing-new whose principal arn is <Principal-ARN-OF-ROLE>` 预期结果： • 代理应确认删除访问条目。 • 要进行验证，请使用 AWS Management Console 或使用 Amazon EKS API 进行检查，然后运行以下命令：`aws eks list-access-entries --cluster-name ekscluster` 您也可以要求代理对 EKS Pod 身份关联执行操作。有关更多详细信息，请参阅 Amazon EKS 文档 AWS 服务中的了解 EKS Pod Identity 如何授予容器访问权限。	

清理

Task	描述	所需技能

清理资源。	要清理此模式创建的资源，请按以下步骤操作。等待每个删除步骤完成后再继续下一步。
	
	Warning
	此过程将永久删除这些堆栈创建的所有资源。在继续操作之前，请确保已备份所有重要数据。
	
	1. 要删除 Amazon EKS 集群，请运行以下命令：
	`eksctl delete cluster -f eks-deploy.yaml`
	
	Note
	此操作可能需要 15-20 分钟才能完成。
	
	2. 要删除 Amazon S3 存储桶，请运行以下命令：
	• 要清空 Lambda 存储桶，请执行以下操作：
	`aws s3 rm s3://bedrock-agent-lambda-a`
		AWS DevOps

Task	描述	所需技能
	rtifacts-${AWS_ACC OUNT} --recursive • 要清空知识库存储桶，请执行以下操作： aws s3 rm s3://eks-knowledge-base-${AWS_ACCOUNT} –recursive • 要清空 OpenAPI 架构存储桶，请执行以下操作： aws s3 rm s3://bedrock-agent-openapi-schema-${AWS_ACCOUNT} –recursive • 要删除空桶，请执行以下操作： aws s3 rb s3://bedrock-agent-lambda-artifacts-${AWS_ACCOUNT} aws s3 rb s3://eks-knowledge-base-${AWS_ACCOUNT} aws s3 rb s3://bedrock-agent-openapi-schema-${AWS_ACCOUNT} 3. 要删除 CloudFormation 堆栈，请运行以下命令： aws cloudformation delete-stack \ --stack-name	

Task	描述	所需技能

	4. 要验证是否已删除 Amazon EKS 集群，请运行以下命令：
	`eksctl get clusters`
	5. 要验证是否删除 Amazon S3 存储桶，请运行以下命令：
	• 要验证是否已删除 Lambda 存储桶，请执行以下操作：
	`aws s3 ls | grep "bedrock-agent-lambda-artifacts"`
	• 要验证知识库存储桶的删除，请执行以下操作：
	`aws s3 ls | grep "eks-knowledge-base"`
	• 要验证是否已删除 OpenAPI 架构存储桶，请执行以下操作：
	`aws s3 ls | grep "bedrock-agent-openapi-schema"`
	6. 要验证堆栈删除情况，请运行以下命令：
	`aws cloudformation list-stacks \--query 'StackSummaries[?StackName==``]'`	

Task	描述	所需技能
	如果无法删除堆栈，请参阅故障排除。	

故障排除

事务	解决方案
在环境设置期间返回非零错误代码。	运行任何命令部署此解决方案时，请验证您使用的文件夹是否正确。有关更多信息，请参阅此模式存储库中的 first_deploy.md 文件。
Lambda 函数无法执行该任务。	确保正确设置了从 Lambda 函数到 Amazon EKS 集群的连接。
代理提示无法识别 APIs。	重新部署解决方案。有关更多信息，请参阅此模式存储库中的 re_deploy.md 文件。
无法删除堆栈。	首次尝试删除堆栈可能会失败。之所以出现这种故障，可能是因为为知识库编制索引的 OpenSearch 集合创建的自定义资源存在依赖性问题。要删除堆栈，请通过保留自定义资源来重试删除操作。

相关资源

AWS 博客

- 深入探讨简化的 Amazon EKS 访问管理控制

亚马逊 Bedrock 文档

- 使用 AI 代理自动执行应用程序中的任务
- 亚马逊 Bedrock Agents 的工作原理
- 测试代理行为并对其进行故障排除

- 使用操作组定义代理要执行的操作

Amazon EKS 文档

- 了解访问控制在 Amazon EKS 中的工作原理

使用 Terraform 和 Amazon AWS Bedrock 部署 RAG 用例

由马丁·马里奇（AWS）、爱丽丝·莫拉诺（AWS）、朱利安·费迪南德·格鲁伯（AWS）、尼古拉斯·雅各布·贝尔（AWS）、奥利维尔·布里克（AWS）和尼古拉·奥拉齐奥（AWS）创作

摘要

AWS 提供了各种选项来构建支持检索增强生成 (RAG) 的生成式 AI 用例。这种模式为基于矢量存储且兼容 Amazon Aurora PostgreSQL 的基于 RAG 的应用程序提供了解决方案。 LangChain 您可以使用 Terraform 将此解决方案直接部署到您的中， AWS 账户 并实现以下简单的 RAG 用例：

1. 用户手动将文件上传到亚马逊简单存储服务 (Amazon S3) 存储桶，例如微软 Excel 文件或 PDF 文档。（有关支持的文件类型的更多信息，请参阅非结构化文档。）
2. 文件内容被提取并嵌入到基于无服务器 Aurora PostgreSQL 兼容的知识数据库中，该数据库支持近乎实时地将文档摄取到矢量存储中。这种方法使得 RAG 模型能够访问和检索相关信息，以应对低延迟至关重要的用例。
3. 当用户使用文本生成模型时，它通过从先前上传的文件中增强相关内容的检索来增强交互。

该模式使用 Amazon Titan 文本嵌入 v2 作为嵌入模型，使用 Anthropic Claude 3 Sonnet 作为文本生成模型，两者均可在亚马逊 Bedrock 上线。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- AWS Command Line Interface (AWS CLI) 已安装并配置为你的 AWS 账户。有关安装说明，请参阅 AWS CLI 文档 AWS CLI中的安装或更新到最新版本的。要查看您的 AWS 凭证和对账户的访问权限，请参阅 AWS CLI 文档中的配置和凭证文件设置。
- 在您 AWS 账户的 Amazon Bedrock 控制台中为所需的大型语言模型 (LLMs) 启用的模型访问权限。此模式需要以下内容 LLMs：
  - `amazon.titan-embed-text-v2:0`
  - `anthropic.claude-3-sonnet-20240229-v1:0`

限制

- 此示例架构不包括使用矢量数据库进行编程问答的接口。如果您的用例需要 API，可以考虑添加带有运行检索和问答任务 AWS Lambda 功能的 Amazon API Gat eway。
- 此示例架构不包括对已部署基础架构的监控功能。如果您的用例需要监控，请考虑添加AWS 监控服务。
- 如果您在短时间内将大量文档上传到 Amazon S3 存储桶，则 Lambda 函数可能会遇到速率限制。作为解决方案，您可以将 Lambda 函数与亚马逊简单队列服务 (Amazon SQS) Simple Queue SQUEE 队列分开，您可以在其中控制 Lambda 调用的速率。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

产品版本

- AWS CLI 版本 2 或更高版本
- Docker 版本 26.0.0 或更高版本
- 诗歌版本 1.7.1 或更高版本
- Python 版本 3.10 或更高版本
- Terraform 版本 1.8.4 或更高版本

架构

下图显示了此模式的工作流和体系结构组件。

此图说明了以下内容：

1. 在 Amazon S3 存储桶中创建对象时bedrock-rag-template-<account_id>，Ama zon S3 通知会调用 Lambda 函数。data-ingestion-processor
2. Lambda 函数基于存储在亚马逊弹性容器注册表 (Amazon ECR) Regist data-ingestion-processor ry 存储库中的 Docker 镜像。bedrock-rag-template

   该函数使用 LangChain S3 FileLoader 将文件作为LangChain 文档读取。然后，根据Amazon Titan 文本嵌入V2嵌入模型的最大令牌大小，给定a CHUNK_SIZE 和a对每个文档进行分LangChain RecursiveCharacterTextSplitter块。CHUNK_OVERLAP接下来，Lambda 函数在 Amazon Bedrock 上调用嵌入模型，将区块嵌入到数值向量表示中。最后，这些向量存储在 Aurora PostgreSQL 数据库中。要访问数据库，Lambda 函数首先从中检索用户名和密码。 AWS Secrets Manager

3. 在 Amazon SageMaker AI 笔记本实例上aws-sample-bedrock-rag-template，用户可以编写问题提示。该代码在 Amazon Bedrock 上调用 Claude 3，并将知识库信息添加到提示的上下文中。因此，Claude 3使用文件中的信息提供了答复。

这种模式的网络和安全方法如下：

- Lambda 函数data-ingestion-processor位于虚拟私有云 (VPC) 内的私有子网中。由于 Lambda 函数具有安全组，因此不允许向公共互联网发送流量。因此，流向 Amazon S3 和 Amazon Bedrock 的流量仅通过 VPC 终端节点路由。因此，流量不会通过公共互联网，从而减少了延迟，并在网络层面增加了额外的安全层。
- 只要适用，就会使用带别名的 AWS Key Management Service (AWS KMS) 密钥对所有资源和数据进行加密aws-sample/bedrock-rag-template。

自动化和扩缩

此模式使用 Terraform 将基础架构从代码存储库部署到. AWS 账户

工具

AWS 服务

- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩展 PostgreSQL 部署。在这种模式下，兼容 Aurora PostgreSQL 的 pgvector 插件使用 pgvector 插件作为矢量数据库。
- Amazon Bedrock 是一项完全托管的服务，它通过统一的 API 提供来自领先的人工智能初创公司和亚马逊的高性能基础模型 (FMs) 供您使用。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行 shell 中的命令进行交互。
- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。在这种模式下，Amazon ECR 托管 Lamb data-ingestion-processor da 函数的 Docker 镜像。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Key Management Service (AWS KMS) 帮助您创建和控制加密密钥以帮助保护您的数据。

- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。在这种模式下，Lambda 会将数据提取到矢量存储中。
- Amazon SageMaker AI 是一项托管机器学习 (ML) 服务，可帮助您构建和训练机器学习模型，然后将其部署到生产就绪的托管环境中。
- AWS Secrets Manager 有助于您通过对 Secrets Manager 的 API 调用来替换代码中的硬编码凭证（包括密码），以编程方式检索密钥。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon Virtual Private Cloud（亚马逊 VPC）可帮助您将 AWS 资源启动到您定义的虚拟网络中。该虚拟网络类似于您在数据中心中运行的传统网络，并具有使用 AWS的可扩展基础设施的优势。VPC 包括用于控制流量的子网和路由表。

其他工具

- Docker 是一组平台即服务（PaaS）产品，它们使用操作系统级别的虚拟化技术在容器中交付软件。
- HashiCorp Terraform 是一款基础设施即代码 (IaC) 工具，可帮助您使用代码来配置和管理云基础架构和资源。
- P@@ o etry 是一款在 Python 中进行依赖管理和打包的工具。
- Python 是通用的计算机编程语言。

代码存储库

此模式的代码可在-ama GitHub terraform-rag-template-usingzon- bedrock 存储库中找到。

最佳实践

- 尽管此代码示例可以部署到任何位置 AWS 区域，但我们建议您使用美国东部（弗吉尼亚北部）`us-east-1`或美国西部（加利福尼亚北部）`us-west-1`。该建议基于该模式发布时Amazon Bedrock中基础模型和嵌入模型的可用性。有关 Amazon Bedrock 基础模型支持的 up-to-date列表 AWS 区域，请参阅 Amazon Bedrock 文档 AWS 区域中的模型支持。有关将此代码示例部署到其他区域的信息，请参阅其他信息。
- 此模式仅提供 proof-of-concept（PoC）或试点演示。如果您想将代码投入生产，请务必使用以下最佳实践：
  - 为 Amazon S3 启用服务器访问日志记录。

- 为 Lambda 函数设置监控和警报。
- 如果您的用例需要 API，可以考虑添加带有运行检索和问答任务的 Lambda 函数的 Amazon API Gateway。

- 遵循最低权限原则，授予执行任务所需的最低权限。有关更多信息，请参阅 IAM 文档中的授予最低权限和安全最佳实践。

操作说明

将解决方案部署在 AWS 账户

Task	描述	所需技能
克隆存储库。	要克隆此模式提供的 GitHub 存储库，请使用以下命令： `git clone https://github.com/aws-samples/terraform-rag-template-using-amazon-bedrock`	AWS DevOps
配置变量。	要配置此模式的参数，请执行以下操作： 1. 在计算机上的 GitHub 存储库中，使用以下命令打开`terraform` 文件夹： `cd terraform` 2. 打开`commons.tfvars` 文件，然后根据需要自定义参数。	AWS DevOps
部署解决方案。	要部署该解决方案，请执行以下操作：	AWS DevOps

Task	描述	所需技能

	1. 在该`terraform` 文件夹中，使用以下命令运行 Terraform 并传入您自定义的变量：
	
	`terraform init`
	`terraform apply -var-file=commons.tfvars`
	
	2. 确认架构图中显示的资源已成功部署。
	
	基础设施部署在 VPC 内配置一个 SageMaker AI 实例，并具有访问 Aurora PostgreSQL 数据库的权限。	

测试解决方案

Task	描述	所需技能
运行演示。	成功部署之前的基础架构后，使用以下步骤在 Jupyter 笔记本中运行演示： 1. 登录到 AWS Management Console 基础架构的部署 AWS 账户 位置。 2. 打开 A SageMaker I 笔记本实例`aws-sample-bedrock-rag-template` 。 3. 使用拖放`rag_demo.ipynb` 操作将 Jupyter 笔记	常规 AWS

Task	描述	所需技能
	本移动到 SageMaker AI 笔记本实例上。 4. 在 SageMaker AI 笔记本实例`rag_demo.ipynb` 上打开，然后选择内`conda_python3` 核。 5. 要运行演示，请运行笔记本的单元。 Jupyter 笔记本将引导您完成以下过程： • 安装要求 • 嵌入定义 • 数据库连接 • 数据摄取 • 检索增强文本生成 • 相关文件查询	

清理基础设施

Task	描述	所需技能
清理基础架构。	要删除您在不再需要时创建的所有资源，请使用以下命令： `terraform destroy -var-file=commons.tfvars`	AWS DevOps

相关资源

AWS resources

- 使用 Python 构建 Lambda 函数
- 基础模型的推理参数
- 访问 Amazon Bedrock 基础模型
- 矢量数据库在生成式人工智能应用中的作用（AWS 数据库博客）
- 使用 Amazon Aurora PostgreSQL

其他资源

- pgvector 文档

其他信息

实现矢量数据库

这种模式使用兼容 Aurora PostgreSQL 的 Aurora 来实现 RAG 的矢量数据库。作为 Aurora PostgreSQL 的替代品 AWS ，为 RAG 提供其他功能和服务，例如亚马逊 Bedrock 知识库和亚马逊服务。 OpenSearch 您可以选择最适合您特定要求的解决方案：

- Amazon Ser OpenSearch v ice 提供分布式搜索和分析引擎，可用于存储和查询大量数据。
- Amazon Bedrock 知识库专为构建和部署知识库作为额外抽象而设计，以简化 RAG 摄取和检索过程。亚马逊 Bedrock 知识库可以与 Aurora PostgreSQL 和亚马逊服务一起使用。 OpenSearch

部署到其他 AWS 区域

如架构中所述，我们建议您使用美国东部（弗吉尼亚北部）`us-east-1`或美国西部（加利福尼亚北部）区域`us-west-1`来部署此代码示例。但是，除了`us-east-1`和之外，还有两种可能的方法可以将此代码示例部署到其他区域`us-west-1`。您可以在`commons.tfvars`文件中配置部署区域。对于跨区域基础模型访问，请考虑以下选项：

- 穿越公共互联网-如果流量可以穿过公共互联网，请将互联网网关添加到 VPC。然后，调整分配给 Lambda 函数`data-ingestion-processor`和 SageMaker AI 笔记本实例的安全组，以允许公共互联网的出站流量。
- 不穿越公共互联网-要将此示例部署到`us-east-1`或以外的任何区域`us-west-1`，请执行以下操作：

1. 在us-east-1或us-west-1区域中，为其创建一个 VPC，包括一个 VPC 终端节点bedrock-runtime。
2. 使用 VPC 对等连接或到应用程序 VPC 的传输网关创建对等连接。
3. 在或之外的任何 Lambda 函数中配置 b bedrock-runtime oto3 客户端时us-west-1，请将 in us-east-1 bedrock-runtime 或 us-east-1 us-west-1 的 VPC 终端节点的私有 DNS 名称传递给 boto3 客户端。endpoint_url

使用 Amazon 中的推理管道将预处理逻辑部署到单个终端节点的 ML 模型中 SageMaker

由 Mohan Gowda Purushothama (AWS)、Gabriel Rodriguez Garcia (AWS) 和 Mateusz Zaremba (AWS) 创建

摘要

此模式说明了如何使用 Amazon SageMaker 中的推理管道在单个终端节点中部署多个管道模型对象。管道模型对象表示不同的机器学习 (ML) 工作流程阶段，例如预处理、模型推断和后期处理。为了说明串行连接的管道模型对象的部署，此模式向您展示了如何部署预处理的 Scikit-Learn 容器和基于内置线性学习器算法的回归模型。 SageMaker部署托管在中的单个端点后面 SageMaker。

> **Note**
>
> 这种模式下的部署使用 ml.m4.2xlarge 实例类型。建议使用符合您的数据大小和工作流程复杂要求的实例类型。有关更多信息，请参阅 Amazon SageMaker 定价。此模式使用了预先构建的 Scikit-Learn Docker 映像，但您可以使用自己的 Docker 容器并将其集成至工作流程。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- Python 3.9
- 亚马逊 SageMaker Python 软件开发工具包和 Boto3 库
- AWS Identity and Access Management (AWS IAM) 角色具有基本 SageMaker 权限和亚马逊简单存储服务 (Amazon S3) S ervice 权限

产品版本

- 亚马逊 SageMaker Python SDK 2.49.2

架构

目标技术堆栈

- Amazon Elastic Container Registry (Amazon ECR)
- Amazon SageMaker
- 亚马逊 SageMaker Studio
- Amazon Simple Storage Service（Amazon S3）
- Amazon 的实时推理终端节点 SageMaker

目标架构

下图显示了部署 Amazon SageMaker 管道模型对象的架构。

图表显示了以下工作流：

1. SageMaker 笔记本部署管道模型。
2. S3 存储桶存储模型构件。
3. Amazon ECR 从 S3 存储桶获取源容器映像。

工具

AWS 工具

- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- Amazon SageMaker 是一项托管机器学习服务，可帮助您构建和训练机器学习模型，然后将其部署到可用于生产的托管环境中。
- Amazon SageMaker Studio 是一个基于 Web 的机器学习集成开发环境 (IDE)，允许您构建、训练、调试、部署和监控您的机器学习模型。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

代码

此模式的代码可在带有 Scikit-Learn 和 L GitHub inear Learner 存储库的推理管道中找到。

操作说明

准备数据集

Task	描述	所需技能

为回归任务准备数据集。	在 Amazon SageMaker Studio 中@@ 打开笔记本电脑。
	
	若要导入所有必要的库并初始化工作环境，请在笔记本中使用以下示例代码：
	
	`import sagemaker`
	`from sagemaker import get_execution_role`
	
	`sagemaker_session = sagemaker.Session()`
	
	`# Get a SageMaker-compatible role used by this Notebook Instance.`
	`role = get_execution_role()`
	
	`# S3 prefix`
	`bucket = sagemaker_session.default_bucket()`
	`prefix = "Scikit-LinearLearner-pipeline-abalone-example"`
	
	若要下载示例数据集，请将以下代码添加至您的笔记本：
	
	`! mkdir abalone_data`
	`! aws s3 cp s3://sagemaker-sample-files`
		数据科学家

Task	描述	所需技能
	`/datasets/tabular/uci_abalone/abalone.csv ./abalone_data` **Note** 此模式中的示例使用 UCI Machine Learning Repository 中的 Abalone 数据集。	

将数据集上传至 S3 存储桶。	在此前准备数据集的笔记本中，添加以下代码，以将示例数据上传至 S3 存储桶：
	
	`WORK_DIRECTORY = "abalone_data"`
	
	`train_input = sagemaker_session.upload_data( path="{}/{}".format(WORK_DIRECTORY, "abalone.csv"), bucket=bucket, key_prefix="{}/{}".format(prefix, "train"), )`
		数据科学家

使用创建数据预处理器 SKLearn

Task	描述	所需技能
准备 preprocessor.py 脚本。	1. 从 GitHub sklearn_abalone_featurizer.py 存储库中的 Python 文件中复	数据科学家

Task	描述	所需技能
	制预处理逻辑，然后将该代码粘贴到名`sklearn_abalone_featurizer.py` 为的单独的 Python 文件中。您可以修改代码，以适应您的自定义数据集和自定义工作流程。 2. 将`sklearn_abalone_featurizer.py` 文件保存在项目的根目录中（也就是说，与运行 SageMaker 笔记本的位置相同）。	

Task	描述	所需技能

创建 SKLearn 预处理器对象。	要创建可以合并到最终推理管道中的 SKLearn 预处理器对象（名为 E SKLearn stimator），请在笔记本中运行以下代码： SageMaker
	
	`from sagemaker.sklearn.estimator import SKLearn`
	
	`FRAMEWORK_VERSION = "0.23-1"`
	`script_path = "sklearn_abalone_featurizer.py"`
	
	`sklearn_preprocessor = SKLearn(`
	`entry_point=script_path,`
	`role=role,`
	`framework_version=FRAMEWORK_VERSION,`
	`instance_type="ml.c4.xlarge",`
	`sagemaker_session=sagemaker_session,`
	`)`
	`sklearn_preprocessor.fit({"train": train_input})`
		数据科学家

Task	描述	所需技能

测试预处理器推理。	要确认您的预处理器定义正确，请在 SageMaker 笔记本中输入以下代码来启动批处理转换作业：
	
	`# Define a SKLearn Transformer from the trained SKLearn Estimator`
	`transformer = sklearn_preprocessor.transformer(`
	`instance_count=1, instance_type="ml.m5.xlarge", assemble_with="Line", accept="text/csv"`
	`)`
	
	`# Preprocess training input`
	`transformer.transform(train_input, content_type="text/csv")`
	`print("Waiting for transform job: " + transformer.latest_transform_job.job_name)`
	`transformer.wait()`
	`preprocessed_train = transformer.output_path`	

验证机器学习模型

Task	描述	所需技能

创建模型对象。	要基于线性学习器算法创建模型对象，请在 SageMaker 笔记本中输入以下代码：
	
	`import boto3`
	`from sagemaker`
	`.image_uris import`
	` retrieve`
	
	`ll_image = retrieve(`
	`"linear-learner",`
	` boto3.Session().re`
	`gion_name)`
	`s3_ll_output_key`
	`_prefix = "ll_train`
	`ing_output"`
	`s3_ll_output_location`
	` = "s3://{}/{}/{}/{}"`
	`.format(`
	`    bucket, prefix,`
	` s3_ll_output_key_p`
	`refix, "ll_model"`
	`)`
	
	`ll_estimator =`
	` sagemaker.estimato`
	`r.Estimator(`
	`    ll_image,`
	`    role,`
	`    instance_count=1,`
	`    instance_type="ml.`
	`m4.2xlarge",`
	`    volume_size=20,`
	`    max_run=3600,`
	`    input_mode="File",`
	`    output_path=s3_ll_`
	`output_location,`
		数据科学家

Task	描述	所需技能

	<pre> sagemaker_session=
	sagemaker_session,
	)
	
	ll_estimator.s
	et_hyperparameters
	(feature_dim=10,
	 predictor_type="re
	gressor", mini_batc
	h_size=32)
	
	ll_train_data =
	 sagemaker.inputs.T
	rainingInput(
	    preprocessed_train
	,
	    distribution="Full
	yReplicated",
	    content_type="text
	/csv",
	    s3_data_type="S3Pr
	efix",
	)
	
	data_channels =
	 {"train": ll_train_
	data}
	ll_estimator.fit(inpu
	ts=data_channels,
	 logs=True)</pre>
	先前代码从公共 Amazon ECR 注册表中检索模型的相关 Amazon ECR Docker 映像，创建估算器对象，然后使用该对象训练回归模型。	

部署最终管道

Task	描述	所需技能

部署管道模型。	要创建管道模型对象（即预处理器对象）并部署该对象，请在 SageMaker 笔记本中输入以下代码：
	
	`from sagemaker.model import Model`
	`from sagemaker.pipeline import PipelineModel`
	`import boto3`
	`from time import gmtime, strftime`
	
	`timestamp_prefix = strftime("%Y-%m-%d-%H-%M-%S", gmtime())`
	
	`scikit_learn_inferencee_model = sklearn_preprocessor.create_model()`
	`linear_learner_model = ll_estimator.create_model()`
	
	`model_name = "inference-pipeline-" + timestamp_prefix`
	`endpoint_name = "inference-pipeline-ep-" + timestamp_prefix`
	`sm_model = PipelineModel(`
	`name=model_name, role=role, models=[scikit_learn_infe`
		数据科学家

Task	描述	所需技能
	rencee_model, linear_learner_model]) sm_model.deploy(init ial_instance_count =1, instance_type="ml. c4.xlarge", endpoint_ name=endpoint_name) **Note** 您可以调整模型对象中使用的实例类型以满足您的需求。	

Task	描述	所需技能

测试推理。	要确认端点是否正常工作，请在 SageMaker 笔记本中运行以下示例推理代码：
	
	`from sagemaker.predicto r import Predictor from sagemaker.serializ ers import CSVSerial izer`
	
	`payload = "M, 0.44, 0.365, 0.125, 0.516, 0.2155, 0.114, 0.155" actual_rings = 10 predictor = Predictor( endpoint_name=endp oint_name, sagemaker _session=sagemaker _session, serialize r=CSVSerializer() )`
	
	`print(predictor .predict(payload))`
		数据科学家

相关资源

- 在@@ 使用亚马逊 SageMaker 推理管道和 Scikit-Learn（AWS Machine Learning 博客）进行预测之前，对输入数据进行预处理
- 使用亚马逊进行端到端的 Machine Learn SageMaker ing (GitHub)

使用 RAG 和提示开发基于 AI 聊天的高级生成式 AI 助手 ReAct

由 Praveen Kumar Jeyarajan (AWS)、乔俊东 (AWS)、Kara Yang (AWS)、Kiowa Jackson (AWS)、Noah Hamilton (AWS) 和曹帅 (AWS) 创作

摘要

一家典型的公司将 70% 的数据存储在孤立的系统中。您可以使用基于人工智能的生成式聊天助手，通过自然语言交互来解锁这些数据孤岛之间的见解和关系。为了充分利用生成式人工智能，输出必须是可信的、准确的，并且包含可用的企业数据。成功的基于聊天的助手取决于以下几点：

- 生成式 AI 模型（例如 Anthropic Claude 2）
- 数据源矢量化
- 用于提示模型的高级推理技术，例如ReAct 框架

这种模式提供了从亚马逊简单存储服务 (Amazon S3) Simple S3 存储桶、AWS Glue 和亚马逊关系数据库服务 (Amazon RDS) 等数据源检索数据的方法。通过将检索增强生成 (RAG) 与 chain-of-thought 方法交织在一起，可以从这些数据中获得价值。结果支持基于聊天的复杂助手对话，这些对话利用了贵公司存储的全部数据。

这种模式以 Amazon SageMaker 手册和定价数据表为例，探索基于人工智能聊天的生成式助手的功能。您将构建一个基于聊天的助手，通过回答有关定价和 SageMaker 服务功能的问题来帮助客户评估服务。该解决方案使用Streamlit库来构建前端应用程序，并使用 LangChain 框架来开发由大型语言模型 (LLM) 支持的应用程序后端。

向基于聊天的助手提出的询问会得到初步的意图分类，以便路由到三个可能的工作流程之一。最复杂的工作流程将一般咨询指导与复杂的定价分析相结合。您可以调整模式以适应企业、企业和工业用例。

先决条件和限制

先决条件

- 已安装并@@ 配置 AWS 命令行接口 (AWS CLI) L ine Interface
- 已@@ 安装并配置 AWS Cloud Development Kit (AWS CDK) 工具包 2.114.1 或更高版本
- 对 Python 和 AWS CDK 有基本的熟悉程度
- Git 已安装

- 已@@ 安装 Docker
- 已安装并配置 Python 3.11 或更高版本（有关更多信息，请参阅 “工具” 部分）
- 使用 AWS CDK 启动的活跃 AWS 账户
- 亚马逊 Bedrock 服务启用了亚马逊 Titan 和 Anthropic Claude 模型访问权限
- 在终端环境中正确配置的 AWS 安全凭证，包括 `AWS_ACCESS_KEY_ID`

限制

- LangChain 不支持每个 LLM 进行直播。支持 Anthropic Claude 模型，但不支持 AI21 Labs 中的模型。
- 此解决方案部署至单个 Amazon Web Services account。
- 此解决方案只能部署在提供 Amazon Bedrock 和 Amazon Kendra 的 AWS 区域。有关供货情况的信息，请参阅 Amazon B edrock 和 Amazon Kendra 的文档。

产品版本

- Python 版本 3.11 或更高版本
- Streamlit 版本 1.30.0 或更高版本
- Streamlit-Chat 版本 0.1.1 或更高版本
- LangChain 版本 0.1.12 或更高版本
- AWS CDK 版本 2.132.1 或更高版本

架构

目标技术堆栈

- Amazon Athena
- Amazon Bedrock
- Amazon Elastic Container Service (Amazon ECS)
- AWS Glue
- AWS Lambda
- Amazon S3

- Amazon Kendra
- Elastic Load Balancing

目标架构

AWS CDK 代码将部署在 AWS 账户中设置基于聊天的助手应用程序所需的所有资源。下图所示的基于聊天的助手应用程序旨在回答用户的 SageMaker 相关查询。用户通过应用程序负载均衡器连接到包含托管 Streamlit 应用程序的 Amazon ECS 集群的 VPC。编排 Lambda 函数连接到应用程序。S3 存储桶数据源通过 Amazon Kendra 和 AWS Glue 向 Lambda 函数提供数据。Lambda 函数连接到 Amazon Bedrock，用于回答基于聊天的助理用户的查询（问题）。

1. 编排 Lambda 函数向 Amazon Bedrock 模型（Claude 2）发送 LLM 提示请求。
2. Amazon Bedrock 将 LLM 响应发回编排 Lambda 函数。

编排 Lambda 函数中的逻辑流

当用户通过 Streamlit 应用程序提问时，它会直接调用编排 Lambda 函数。下图显示了调用 Lambda 函数时的逻辑流程。

- 步骤 1 — 输入`query`（问题）分为三个意图之一：
  - 一般 SageMaker 指导问题
  - 一般 SageMaker 定价（训练/推理）问题
  - 与定价相关的 SageMaker 复杂问题
- 步骤 2-输入`query`启动以下三项服务之一：
  - `RAG Retrieval service`，它从 Amazon Kendra 矢量数据库中检索相关上下文，并通过 Amazon Bedrock 调用 LLM，将检索到的上下文汇总为响应。
  - `Database Query service`，它使用 LLM、数据库元数据和相关表中的样本行将`query`输入转换为 SQL 查询。数据库查询服务通过 Amazon A thena 对 SageMaker 定价数据库运行 SQL 查询，并将查询结果汇总为响应。
  - `In-context ReACT Agent service`，它会在提供响应之前`query`将输入分成多个步骤。代理使用`RAG Retrieval service`和`Database Query service`作为工具，在推理过程中检索相关信息。推理和操作过程完成后，代理生成最终答案作为响应。

- 第 3 步 — 来自编排 Lambda 函数的响应作为输出发送到 Streamlit 应用程序。

工具

Amazon Web Services

- Amazon Athena 是一种交互式查询服务，使您可使用标准 SQL 直接分析 Amazon Simple Storage Service（Amazon S3）中的数据。
- Amazon Bedrock 是一项完全托管的服务，它通过统一的 API 提供来自领先的人工智能初创公司和亚马逊的高性能基础模型 (FMs) 供您使用。
- AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，可帮助您在代码中定义和预置 Amazon Web Services Cloud 基础设施。
- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- Amazon Elastic Container Service (Amazon ECS)是一项快速且可扩展的容器管理服务，可帮助运行、停止和管理集群上的容器。
- AWS Glue 是一项完全托管的提取、转换、加载（ETL）服务。它可以帮助您在数据存储和数据流之间对数据进行可靠地分类、清理、扩充和移动。此模式使用 AWS Glue 爬网程序与 AWS Glue Data Catalog 表。
- Amazon Kendra 是一项智能搜索服务，它使用自然语言处理和高级机器学习算法，从您的数据中返回搜索问题的具体答案。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- 弹性负载均衡（ELB）将传入的应用程序或网络流量分配到多个目标。例如，您可以跨亚马逊弹性计算云 (Amazon EC2) 实例、容器以及一个或多个可用区中的 IP 地址分配流量。

代码存储库

此模式的代码可在 GitHub genai-bedrock-chatbot存储库中找到。

代码存储库包含以下文件和文件夹：

- `assets`文件夹-静态资产、架构图和公共数据集
- `code/lambda-container`文件夹 — 在 Lambda 函数中运行的 Python 代码

- `code/streamlit-app`文件夹 — 在 Amazon ECS 中作为容器镜像运行的 Python 代码
- `tests`文件夹 — 为对 AWS CDK 结构进行单元测试而运行的 Python 文件
- `code/code_stack.py`— AWS CDK 构造用于创建 AWS 资源的 Python 文件
- `app.py`— 用于在目标 AWS 账户中部署 AWS 资源的 AWS CDK 堆栈 Python 文件
- `requirements.txt`— 必须为 AWS CDK 安装的所有 Python 依赖项的列表
- `requirements-dev.txt`— AWS CDK 运行单元测试套件必须安装的所有 Python 依赖项的列表
- `cdk.json` — 用于提供启动资源所需值的输入文件

> **Note**
>
> AWS CDK 代码使用 L3（第 3 层）结构和 AWS 管理的 AWS 身份和访问管理 (IAM) 策略来部署解决方案。

最佳实践

- 此处提供的代码示例仅适用于 proof-of-concept (PoC) 或试点演示。如果您想将代码带到生产环境，请务必使用以下最佳实践：
  - Amazon S3 访问日志已启用。
  - VPC 流日志已启用。
  - 亚马逊 Kendra 企业版索引已启用。
- 为 Lambda 函数设置监控和警报。有关更多信息，请参阅 Lambda 函数监控和故障排除。有关使用 Lambda 函数时的一般最佳实践标准，请参阅 AWS 文档

操作说明

在本地机器上设置 AWS 凭证

Task	描述	所需技能
为要部署堆栈的账户和 AWS 区域导出变量。	要使用环境变量为 AWS CDK 提供 AWS 凭证，请运行以下命令。	DevOps 工程师，AWS DevOps

Task	描述	所需技能
	`export CDK_DEFAU LT_ACCOUNT=<12 Digit AWS Account Number> export CDK_DEFAU LT_REGION=<region>`	
设置 AWS CLI 配置文件。	要为账户设置 AWS CLI 配置文件，请按 AWS 文档中的说明进行操作。	DevOps 工程师，AWS DevOps

设置您的环境

Task	描述	所需技能
在本地机器克隆存储库。	要克隆存储库，请在终端中运行以下命令。 `git clone https://github.com/awslabs/genai-bedrock-chatbot.git`	DevOps 工程师，AWS DevOps
设置 Python 虚拟环境以及安装所需依赖项。	要设置 Python 虚拟环境，请运行以下命令。 `cd genai-bedrock-chatbot python3 -m venv .venv source .venv/bin/activate` 要设置所需的依赖关系，请运行以下命令。	DevOps 工程师，AWS DevOps

Task	描述	所需技能
	`pip3 install -r requirements.txt`	
设置 AWS CDK 环境和合成 AWS CDK 代码。	1. 要在您的 AWS 账户中设置 AWS CDK 环境，请运行以下命令。 `cdk bootstrap aws://ACCOUNT-NUMBER/REGION` 2. 要将代码转换为 AWS CloudFormation 堆栈配置，请运行命令`cdk synth`。	DevOps 工程师，AWS DevOps

配置和部署基于聊天的助手应用程序

Task	描述	所需技能
配置 Claude 模型访问权限。	要为您的 AWS 账户启用 Anthropic Claude 模型访问权限，请按照 A mazon Bed rock 文档中的说明进行操作。	AWS DevOps
在账户中部署资源。	要使用 AWS CDK 在 AWS 账户中部署资源，请执行以下操作： 1. 在克隆存储库的根目录中，在`cdk.json`文件中，为`logging`参数提供输入。示例值为`INFO`、`DEBUG`、`WARN`、和`ERROR`。	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	这些值定义了 Lambda 函数和 Streamlit 应用程序的日志级别消息。 2. 克隆存储库根目录中的`app.py`文件包含用于部署的 AWS CloudFormation 堆栈名称。默认堆栈名称为`chatbot-stack` 。 3. 要部署资源，请运行命令`cdk deploy`。 该`cdk deploy`命令使用 L3 结构创建多个 Lambda 函数，用于将文档和 CSV 数据集文件复制到 S3 存储桶。 4. 命令完成后，登录 AWS 管理控制台，打开控制 CloudFormation 台，查看堆栈是否已成功部署。 成功部署后，您可以使用 CloudFormation 输出部分中提供的 URL 访问基于聊天的助手应用程序。	

Task	描述	所需技能
运行 AWS Glue 爬网程序，并创建数据目录表。	AWS Glue 爬网程序用于保持数据架构的动态性。该解决方案通过按需运行爬虫来创建和更新 AWS Glue 数据目录表中的分区。将 CSV 数据集文件复制到 S3 存储桶后，运行 AWS Glue 爬网程序并创建用于测试的数据目录表架构： 1. 导航到 AWS Glue 控制台。 2. 在导航窗格中，在数据目录下，选择爬网程序。 3. 选择带有后缀`sagemaker-pricing-crawler` 的爬虫。 4. 运行爬网程序。 5. 爬网程序成功运行后，它会创建 AWS Glue Data Catalog 表。 **Note** AWS CDK 代码将 AWS Glue 爬网程序配置为按需运行，但您也可以将其安排为定期运行。	DevOps 工程师，AWS DevOps

Task	描述	所需技能
启动文档索引。	将文件复制到 S3 存储桶后，使用 Amazon Kendra 对其进行抓取和索引： 1. 导航到亚马逊 Kendra 控制台。 2. 选择带有后缀`chatbot-index` 的索引。 3. 在导航窗格中，选择数据源，然后选择带有后缀 `chatbot-index` 的数据源连接器。 4. 选择 “立即同步” 以启动索引编制过程。 **Note** AWS CDK 代码将 Amazon Kendra 索引同步配置为按需运行，但您也可以使用计划参数定期运行。	AWS DevOps， DevOps 工程师

清理解决方案中的所有 AWS 资源

Task	描述	所需技能
移除 AWS 资源。	测试解决方案后，清理资源： 1. 若要移除解决方案部署的 AWS 资源，请运行命令 `cdk destroy`。	DevOps 工程师，AWS DevOps

Task	描述	所需技能
	2. 从两个 S3 存储桶中删除所有对象，然后移除所有存储桶。 有关更多信息，请参阅删除存储桶。	

故障排除

事务	解决方案
AWS CDK 会返回错误信息。	要获得有关 AWS CDK 问题的帮助，请参阅常见 AWS CDK 问题疑难解答。

相关资源

- Amazon Bedrock：
  - 模型访问权限
  - 基础模型的推理参数
- 使用 Python 构建 Lambda 函数
- 开始使用 AWS CDK
- 在 Python 中使用 AWS CDK
- AWS 上的生成式 AI 应用程序生成器
- LangChain 文档
- Streamlit 文档

其他信息

AWS CDK 命令

使用 AWS CDK 时，切记以下有用的命令：

- 列出应用程序的所有堆栈

```
cdk ls
```

- 发出合成的 AWS 模板 CloudFormation

```
cdk synth
```

- 将堆栈部署至您的默认 Amazon Web Services account 和区域

```
cdk deploy
```

- 将已部署的堆栈与当前状态比较

```
cdk diff
```

- 打开 AWS CDK 文档

```
cdk docs
```

- 删除 CloudFormation 堆栈并移除 AWS 部署的资源

```
cdk destroy
```

使用 Amazon Bedrock 代理和知识库开发基于聊天的全自动助手

由乔俊东 (AWS)、Kara Yang (AWS)、Kiowa Jackson (AWS)、Noah Hamilton (AWS)、Praveen Kumar Jeyarajan (AWS) 和曹帅 (AWS) 创作

摘要

许多组织在创建能够协调各种数据源以提供全面答案的基于聊天的助手时面临挑战。这种模式为开发基于聊天的助手提供了一种解决方案，该助手能够回答来自文档和数据库的查询，并且部署简单。

从 A mazon Bedrock 开始，这项完全托管的生成式人工智能 (AI) 服务提供了各种高级基础模型 (FMs)。这有助于高效创建生成式 AI 应用程序，重点关注隐私和安全。在文档检索的背景下，检索增强生成 (RAG) 是一项关键功能。它使用知识库使用来自外部来源的上下文相关信息来扩充调频提示。A mazon OpenSearch Serverless 索引充当 Amazon Bedrock 知识库背后的矢量数据库。通过仔细的及时工程来增强这种集成，以最大限度地减少不准确之处，并确保答复以事实文档为基础。对于数据库查询，Amazon Bedrock 将文本查询转换为包含特定参数的结构化 SQL 查询。 FMs 这使得可以从 AWS Glue 数据库管理的数据库中精确检索数据。这些@@ 查询使用亚马逊 A thena。

要处理更复杂的查询，要获得全面的答案，就需要来自文档和数据库的信息。Amazon Bedrock 代理是一项生成式 AI 功能，可帮助您构建能够理解复杂任务的自主代理，并将其分解为更简单的任务进行编排。在 Amazon Bedrock 自主代理的推动下，将从简化任务中检索到的见解相结合，增强了信息的合成，从而得出了更全面和详尽的答案。此模式演示了如何使用 Amazon Bedrock 以及自动解决方案中的相关生成人工智能服务和功能来构建基于聊天的助手。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- Docker，已安装
- AWS Cloud Development Kit (AWS CDK)，已安装并引导到或 `us-east-1` AWS 区域 `us-west-2`
- AWS CDK Toolkit 版本 2.114.1 或更高版本，已安装
- AWS 命令行界面（AWS CLI），已安装并配置
- Python 版本 3.11 或更高版本，已安装
- 在 Amazon Bedrock 中，允许访问 Claude 2、Claude 2.1、Claude Instant 和 Titan Embeddings G1 — 文本

限制

- 此解决方案部署至单个 Amazon Web Services account。
- 此解决方案只能部署在支持 Amazon Bedrock 和 Amazon OpenSearch Serverless 的 AWS 区域。有关更多信息，请参阅 Amazon B edrock 和 Amazon OpenSearch Serverless 的文档。

产品版本

- llama-Index 版本 0.10.6 或更高版本
- Sqlalchemy 版本 2.0.23 或更高版本
- openSearch-py 版本 2.4.2 或更高版本
- requests_aws4Auth 版本 1.2.3 或更高版本
- 适用于 Python 的 AWS 开发工具包 (Boto3) SDK 版本 1.34.57 或更高版本

架构

目标技术堆栈

AWS Cloud Development Kit (AWS CDK) 是一个开源软件开发框架，用于在代码中定义云基础设施并通过 AWS CloudFormation 进行配置。此模式中使用的 AWS CDK 堆栈部署以下 AWS 资源：

- AWS Key Management Service（AWS KMS）
- Amazon Simple Storage Service（Amazon S3）
- AWS Glue 数据目录，适用于 AWS Glue 数据库组件
- AWS Lambda
- AWS Identity and Access Management (IAM)
- Amazon OpenSearch 无服务器
- Amazon Elastic Container Registry（Amazon ECR）
- Amazon Elastic Container Service（Amazon ECS）
- AWS Fargate
- Amazon Virtual Private Cloud（Amazon VPC）
- 应用程序负载均衡器

目标架构

该图显示了在单个 AWS 区域内使用多个 AWS 服务的全面的 AWS 云原生设置。基于聊天的助手的主要界面是托管在 Amazon ECS 集群上的 Streamlit 应用程序。App lication Load Balancer 管理可访问性。通过此接口进行的查询会激活 `Invocation` Lambda 函数，然后该函数与 Amazon Bedrock 的代理进行交互。该代理通过查阅 Amazon Bedrock 的知识库或调用 Lambda 函数`Agent executor`来回应用户的询问。此函数按照预定义的 API 架构触发一组与代理关联的操作。Amazon Bedrock 的知识库使用 OpenSearch 无服务器索引作为其矢量数据库的基础。此外，该`Agent executor`函数还会生成通过 Amazon Athena 对 AWS Glue 数据库执行的 SQL 查询。

工具

Amazon Web Services

- Amazon Athena 是一种交互式查询服务，使您可使用标准 SQL 直接分析 Amazon Simple Storage Service（Amazon S3）中的数据。
- Amazon Bedrock 是一项完全托管的服务，它通过统一的 API 提供来自领先的人工智能初创公司和亚马逊的高性能基础模型 (FMs) 供您使用。
- AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，可帮助您在代码中定义和预调配 Amazon Web Services Cloud 基础设施。
- AWS 命令行界面 (AWS CLI) L ine AWS CLI 是一款开源工具，可帮助您通过命令行外壳中的命令与 AWS 服务进行交互。
- Amazon Elastic Container Service (Amazon ECS)是一项快速且可扩展的容器管理服务，可帮助运行、停止和管理集群上的容器。
- 弹性负载均衡(ELB) 将传入的应用程序或网络流量分配到多个目标。例如，您可以跨亚马逊弹性计算云 (Amazon EC2) 实例、容器以及一个或多个可用区中的 IP 地址分配流量。
- AWS Glue 是一项完全托管的提取、转换、加载（ETL）服务。它可以帮助您在数据存储和数据流之间对数据进行可靠地分类、清理、扩充和移动。此模式使用 AWS Glue 爬网程序与 AWS Glue Data Catalog 表。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon OpenSearch Server less 是亚马逊 OpenSearch 服务的按需无服务器配置。在这种模式下， OpenSearch 无服务器索引充当 Amazon Bedrock 知识库的矢量数据库。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

其他工具

- Streamlit 是一个用于创建数据应用程序的开源 Python 框架。

代码存储库

此模式的代码可在 GitHub genai-bedrock-agent-chatbot存储库中找到。代码存储库包含以下文件和文件夹：

- `assets`文件夹-静态资产，例如架构图和公共数据集。
- `code/lambdas/action-lambda`文件夹 — 用作 Amazon Bedrock 代理操作的 Lambda 函数的 Python 代码。
- `code/lambdas/create-index-lambda`文件夹 — 用于创建 OpenSearch 无服务器索引的 Lambda 函数的 Python 代码。
- `code/lambdas/invoke-lambda`文件夹 — 调用 Amazon Bedrock 代理的 Lambda 函数的 Python 代码，该代理直接从 Streamlit 应用程序调用。
- `code/lambdas/update-lambda`文件夹 — Lambda 函数的 Python 代码，用于在通过 AWS CDK 部署 AWS 资源后更新或删除资源。
- `code/layers/boto3_layer`文件夹 — 创建在所有 Lambda 函数之间共享的 Boto3 层的 AWS CDK 堆栈。
- `code/layers/opensearch_layer`文件夹 — 创建 OpenSearch 无服务器层的 AWS CDK 堆栈，用于安装所有依赖项来创建索引。
- `code/streamlit-app`文件夹 — 在 Amazon ECS 中作为容器镜像运行的 Python 代码
- `code/code_stack.py`— AWS CDK 构建用于创建 AWS 资源的 Python 文件。
- `app.py`— 在目标 AWS 账户中部署 AWS 资源的 AWS CDK 堆栈 Python 文件。
- `requirements.txt`— 必须为 AWS CDK 安装的所有 Python 依赖项的列表。
- `cdk.json`— 提供创建资源所需值的输入文件。此外，在`context/config`字段中，您可以相应地自定义解决方案。有关自定义的更多信息，请参阅 “其他信息” 部分。

最佳实践

- 此处提供的代码示例仅用于 proof-of-concept (PoC) 或试运行目的。如果您想将代码投入生产，请务必使用以下最佳实践：
  - 启用 Amazon S3 访问日志

  - 启用 VPC 流日志
- 为 Lambda 函数设置监控和警报。有关更多信息，请参阅 Lambda 函数监控和故障排除。有关最佳实践，请参阅使用 AWS Lambda 函数的最佳实践。

操作说明

在您的本地工作站上设置 AWS 证书

Task	描述	所需技能
导出账户和地区的变量。	要使用环境变量为 AWS CDK 提供 AWS 凭证，请运行以下命令。 `export CDK_DEFAU LT_ACCOUNT=<12-digit AWS account number> export CDK_DEFAU LT_REGION=<Region>`	AWS DevOps， DevOps 工程师
设置名为 “配置文件” 的 AWS CLI。	要为账户设置名为 profile 的 AWS CLI，请按照配置和凭证文件设置中的说明进行操作。	AWS DevOps， DevOps 工程师

设置您的环境

Task	描述	所需技能
将存储库克隆到您的本地工作站。	要克隆存储库，请在终端中运行以下命令。 `git clone https://g ithub.com/awslabs/ genai-bedrock-agent- chatbot.git`	DevOps 工程师，AWS DevOps

Task	描述	所需技能

设置 Python 虚拟环境。	要设置 Python 虚拟环境，请运行以下命令。
	`cd genai-bedrock-agent-chatbot`
	`python3 -m venv .venv`
	`source .venv/bin/activate`
	要设置所需的依赖关系，请运行以下命令。
	`pip3 install -r requirements.txt`
		DevOps 工程师，AWS DevOps

设置 AWS CDK 环境。	要将代码转换为 AWS CloudFormation 模板，请运行命令`cdk synth`。	AWS DevOps， DevOps 工程师

配置和部署应用程序

Task	描述	所需技能
在账户中部署资源。	要使用 AWS CDK 在 AWS 账户中部署资源，请执行以下操作： 1. 在克隆存储库的根目录中，在`cdk.json`文件中，提供日志参数的输入。示例值为`INFO`、`DEBUG`、`WARN`、和`ERROR`。	DevOps 工程师，AWS DevOps

Task	描述	所需技能
	这些值定义了 Lambda 函数和 Streamlit 应用程序的日志级别消息。 2. 克隆存储库根目录中的`cdk.json`文件包含用于部署的 AWS CloudFormation 堆栈名称。默认堆栈名称为`chatbot-stack` 。默认 Amazon Bedrock 代理名称为`ChatbotBedrockAgent` ，默认 Amazon Bedrock 代理别名为。`Chatbot_Agent` 3. 要部署资源，请运行命令`cdk deploy`。 该`cdk deploy`命令使用第 3 层结构创建多个 Lambda 函数，用于将文档和 CSV 数据集文件复制到 S3 存储桶。它还为亚马逊 Bedrock 代理部署了 Amazon Bedrock 代理、知识库`Action group`和 Lambda 函数。 4. 登录 AWS 管理控制台，然后在上打开 CloudFormation 控制台https://console.aws.amazon.com/cloudformation/。 5. 确认堆栈已成功部署。有关说明，请参阅在 AWS CloudFormation 控制台上查看您的堆栈。	

Task	描述	所需技能
	成功部署后，您可以使用控制台输出选项卡上提供的 URL 访问基于聊天的助手应用程序。CloudFormation	

清理解决方案中的所有 AWS 资源

Task	描述	所需技能
移除 AWS 资源。	测试解决方案后，要清理资源，请运行命令`cdk destroy`。	AWS DevOps， DevOps 工程师

相关资源

AWS 文档

- Amazon Bedrock 资源：
  - 模型访问权限
  - 基础模型的推理参数
  - Amazon Bedrock 的代理
  - Amazon Bedrock 知识库
- 使用 Python 构建 Lambda 函数
- AWS CDK 资源：
  - 开始使用 AWS CDK
  - 解决常见的 AWS CDK 问题
  - 在 Python 中使用 AWS CDK
- AWS 上的生成式 AI 应用程序生成器

其他 AWS 资源

- 适用于 Amazon OpenSearch 无服务器的矢量引擎

其他资源

- LlamaIndex 文档
- Streamlit 文档

其他信息

使用您自己的数据自定义基于聊天的助手

要整合用于部署解决方案的自定义数据，请遵循以下结构化指南。这些步骤旨在确保无缝高效的集成流程，使您能够使用定制数据有效地部署解决方案。

用于知识库数据集成

数据准备

1. 找到该`assets/knowledgebase_data_source/`目录。
2. 将您的数据集放在此文件夹中。

配置调整

1. 打开 `cdk.json`文件。
2. 导航到该`context/configure/paths/knowledgebase_file_name`字段，然后相应地对其进行更新。
3. 导航到该`bedrock_instructions/knowledgebase_instruction`字段，然后对其进行更新以准确反映新数据集的细微差别和上下文。

用于结构数据集成

数据组织

1. 在该`assets/data_query_data_source/`目录中，创建一个子目录，例如。`tabular_data`
2. 将您的结构化数据集（可接受的格式包括 CSV、JSON、ORC 和 Parquet）放入这个新创建的子文件夹。
3. 如果您要连接到现有数据库，请更新`create_sql_engine()`中的函数`code/lambda/action-lambda/build_query_engine.py`以连接到您的数据库。

配置和代码更新

1. 在cdk.json文件中，更新context/configure/paths/athena_table_data_prefix字段以使其与新的数据路径保持一致。
2. code/lambda/action-lambda/dynamic_examples.csv通过加入与您的数据集对应的新text-to-SQL示例进行修改。
3. 修改code/lambda/action-lambda/prompt_templates.py以反映结构化数据集的属性。
4. 在cdk.json文件中，更新context/configure/bedrock_instructions/action_group_description字段以解释 Action group Lambda 函数的目的和功能。
5. 在assets/agent_api_schema/artifacts_schema.json文件中，解释您的 Lambd Action group a 函数的新功能。

一般更新

在cdk.json文件中，考虑到新整合的数据，在该context/configure/bedrock_instructions/agent_instruction部分中，全面描述了Amazon Bedrock代理的预期功能和设计目的。

使用 Amazon Bedrock 和 Amazon Transcribe 从语音输入中记录机构知识

由 Praveen Kumar Jeyarajan (AWS)、乔俊东 (AWS)、Megan Wu (AWS) 和 Rajiv Upadhyay (AWS) 创作

摘要

获取机构知识对于确保组织成功和应变能力至关重要。机构知识代表了员工随着时间的推移积累的集体智慧、见解和经验，这些知识本质上通常是默许的，是非正式地传下来的。这些丰富的信息包括独特的方法、最佳实践和解决错综复杂的问题的解决方案，这些问题可能无法在其他地方记录下来。通过正式化和记录这些知识，公司可以保留机构记忆，促进创新，增强决策流程，并加快新员工的学习曲线。此外，它还促进协作，赋予个人权力，培养持续改进的文化。最终，利用机构知识可以帮助公司利用其最有价值的资产（员工的集体智慧）来应对挑战，推动增长，并在动态的商业环境中保持竞争优势。

这种模式解释了如何通过高级员工的录音来获取机构知识。它使用 Amazon Transcribe 和 A mazon Bedrock 进行系统的记录和验证。通过记录这些非正式知识，您可以将其保留并与随后的员工群共享。这项努力通过纳入通过直接经验获得的实践知识来支持卓越运营，并提高培训计划的有效性。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- Docker，已安装
- AWS Cloud Development Kit (AWS CDK) 版本 2.114.1 或更高版本，已安装并引导到或 AWS 区域 `us-east-1` `us-west-2`
- AWS CDK Toolkit 版本 2.114.1 或更高版本，已安装
- AWS 命令行界面（AWS CLI），已安装并配置
- Python 版本 3.12 或更高版本，已安装
- 创建 Amazon Transcribe、Amazon Bedrock、亚马逊简单存储服务 (Amazon S3) Service 和 AWS Lambda 资源的权限

限制

- 此解决方案部署至单个 Amazon Web Services account。

- 此解决方案只能部署在提供 Amazon Bedrock 和 Amazon Transcribe 的 AWS 区域。有关供货情况的信息，请参阅 Amazon B edrock 和 Amazon Transc ri be 的文档。
- 音频文件必须采用 Amazon Transcribe 支持的格式。有关支持的格式列表，请参阅 Transcribe 文档中的媒体格式。

产品版本

- 适用于 Python 的 AWS 开发工具包 (Boto3) SDK 版本 1.34.57 或更高版本
- LangChain 版本 0.1.12 或更高版本

架构

该架构代表了 AWS 上的无服务器工作流程。AWS Step F unctions 编排 Lambda 函数，用于音频处理、文本分析和文档生成。下图显示了 Step Functions 工作流程，也称为状态机。

状态机中的每个步骤都由一个不同的 Lambda 函数处理。以下是文档生成过程中的步骤：

1. `preprocess`Lambda 函数验证传递给 Step Functions 的输入，并列出所提供的 Amazon S3 URI 文件夹路径中存在的所有音频文件。工作流程中的下游 Lambda 函数使用文件列表来验证、汇总和生成文档。
2. `transcribe`Lambda 函数使用 Amazon Transcribe 将音频文件转换为文本脚本。此 Lambda 函数负责启动转录过程并将语音准确地转换为文本，然后将其存储起来以供后续处理。
3. `validate`Lambda 函数分析文本记录，确定答案与初始问题的相关性。通过使用 Amazon Bedrock 的大型语言模型 (LLM)，它可以识别主题答案和题外答案并将其区分开来。
4. `summarize`Lambda 函数使用 Amazon Bedrock 生成连贯而简洁的主题答案摘要。
5. `generate`Lambda 函数将摘要汇编成结构良好的文档。它可以根据预定义的模板格式化文档，并包括任何其他必要的内容或数据。
6. 如果任何 Lambda 函数失败，您将通过亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 收到一封电子邮件通知。

在整个过程中，AWS Step Functions 确保每个 Lambda 函数都按正确的顺序启动。该状态机具有并行处理能力以提高效率。Amazon S3 存储桶充当中央存储库，通过管理所涉及的各种媒体和文档格式来支持工作流程。

工具

Amazon Web Services

- Amazon Bedrock 是一项完全托管的服务，它通过统一的 API 提供来自领先的人工智能初创公司和亚马逊的高性能基础模型 (FMs) 供您使用。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Step Functions 是一项无服务器编排服务，可让您搭配使用 AWS Lambda 函数和其他 Amazon Web Services 来构建业务关键型应用程序。
- Amazon Transcribe 是一项自动语音识别服务，它使用机器学习模型将音频转换为文本。

其他工具

- LangChain是一个用于开发由大型语言模型支持的应用程序的框架 (LLMs)。

代码存储库

此模式的代码可在 GitHub genai-knowledge-capture存储库中找到。

代码存储库包含以下文件和文件夹：

- `assets`folder-解决方案的静态资产，例如架构图和公共数据集
- `code/lambdas`文件夹 — 所有 Lambda 函数的 Python 代码
  - `code/lambdas/generate`文件夹-根据 S3 存储桶中的汇总数据生成文档的 Python 代码
  - `code/lambdas/preprocess`文件夹-处理 Step Functions 状态机输入的 Python 代码
  - `code/lambdas/summarize`文件夹-使用 Amazon Bedrock 服务汇总转录数据的 Python 代码
  - `code/lambdas/transcribe`文件夹-使用 Amazon Transcribe 将语音数据（音频文件）转换为文本的 Python 代码
  - `code/lambdas/validate`folder-验证所有答案是否都与同一个主题有关的 Python 代码
- `code/code_stack.py`— AWS CDK 构造用于创建 AWS 资源的 Python 文件

- `app.py`— 用于在目标 AWS 账户中部署 AWS 资源的 AWS CDK 应用程序 Python 文件
- `requirements.txt`— 必须为 AWS CDK 安装的所有 Python 依赖项的列表
- `cdk.json`— 用于提供创建资源所需的值的输入文件

最佳实践

提供的代码示例仅用于 proof-of-concept (PoC) 或试运行目的。如果您想将解决方案投入生产，请使用以下最佳实践：

- 启用 Amazon S3 访问日志
- 启用 VPC 流日志

操作说明

在您的本地工作站上设置 AWS 证书

Task	描述	所需技能

导出账户和 AWS 区域的变量。	要使用环境变量为 AWS CDK 提供 AWS 凭证，请运行以下命令。
	`export CDK_DEFAULT_ACCOUNT=<12-digit AWS account number>`
	`export CDK_DEFAULT_REGION=<Region>`
		AWS DevOps， DevOps 工程师

设置名为 “配置文件” 的 AWS CLI。	要为账户设置名为 profile 的 AWS CLI，请按照配置和凭证文件设置中的说明进行操作。	AWS DevOps， DevOps 工程师

设置您的环境

Task	描述	所需技能
将存储库克隆到您的本地工作站。	要克隆genai-knowledge-capture存储库，请在终端中运行以下命令。 `git clone https://github.com/aws-samples/genai-knowledge-capture`	AWS DevOps， DevOps 工程师
（可选）替换音频文件。	要自定义示例应用程序以包含您自己的数据，请执行以下操作： 1. 导航到克隆存储库中的`assets/audio_samples` 文件夹。 2. 删除包含示例音频文件的文件夹。 3. 为要分析的每个主题创建一个文件夹。 4. 将您的音频文件传输到各自的文件夹。	AWS DevOps， DevOps 工程师

设置 Python 虚拟环境。	要设置 Python 虚拟环境，请运行以下命令。
	
	`cd genai-knowledge-capture`
	`python3 -m venv .venv`
	`source .venv/bin/activate`
	`pip install -r requirements.txt`
		AWS DevOps， DevOps 工程师

Task	描述	所需技能
合成 AWS CDK 代码。	要将代码转换为 AWS CloudFormation 堆栈配置，请运行以下命令。 `cdk synth`	AWS DevOps， DevOps 工程师

配置和部署解决方案

Task	描述	所需技能
配置基础模型访问权限。	允许您的 AWS 账户访问 Anthropic Claude 3 Sonnet 模型。有关说明，请参阅 Bedrock 文档中的添加模型访问权限。	AWS DevOps
在账户中部署资源。	要使用 AWS CDK 在 AWS 账户中部署资源，请执行以下操作： 1. （可选）在克隆存储库的根目录中，在`app.py`文件中，更新 AWS CloudFormation 堆栈名称。默认堆栈名称为`genai-knowledge-capture-stack` 。 2. 要部署资源，请运行命令`cdk deploy`。 该`cdk deploy`命令使用第 3 层结构来创建一组 Lambda 函数、一个 S3 存储桶、一个 Amazon SNS 主题和一个 Step Functions	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	状态机。部署期间，`assets/audio_samples` 文件夹中的音频文件会被复制到 S3 存储桶中。 3. 登录 AWS 管理控制台，然后在上打开 CloudFormation 控制台https://console.aws.amazon.com/cloudformation/。 4. 确认堆栈已成功部署。有关说明，请参阅在 AWS CloudFormation 控制台上查看您的堆栈。	
订阅 Amazon SNS 主题。	要订阅 Amazon SNS 主题以获取通知，请执行以下操作： 1. 在 CloudFormation 控制台的导航窗格中，选择 Stacks。 2. 选择`genai-knowledge-capture-stack` 堆栈。 3. 选择输出选项卡。 4. 找到带有密钥的 Amazon SNS 主题名称。`SNSTopicName` 5. 按照将电子邮件地址订阅 Amazon SNS 主题中的说明配置为接收通知的电子邮件地址。	常规 AWS

测试解决方案

Task	描述	所需技能
运行状态机。	1. 打开 Step Functions 控制台。 2. 在状态机页面上，选择 genai-knowledge-capture-stack-state- machine。 3. 选择启动执行。 4. （可选）在“名称”框中，输入执行的名称。 5. 在输入区域中，通过替换占位符文本来输入以下 JSON 对象，其中： • <Name>就是你要给文档起的名字。 • <S3 bucket name>是包含音频文件的 Amazon S3 存储桶的名称。 • <Folder path>是包含音频文件的目录。 `{ "documentName": "<Name>", "audioFileFolderUri": "s3://<S3 bucket name>/<Folder path>" }` 6. 选择启动执行。 7. 在执行详细信息页面上，查看结果并等待执行完成。	应用程序开发人员，常规 AWS

清理解决方案中的所有 AWS 资源

Task	描述	所需技能
移除 AWS 资源。	测试解决方案后，清理资源： 1. 从 S3 存储桶中删除所有对象，然后删除该存储桶。有关更多信息，请参阅删除存储桶。 2. 从克隆的存储库中运行命令`cdk destroy`。	AWS DevOps， DevOps 工程师

相关资源

AWS 文档

- Amazon Bedrock 资源：
  - 模型访问权限
  - 基础模型的推理参数
- AWS CDK 资源：
  - 开始使用 AWS CDK
  - 在 Python 中使用 AWS CDK
  - 解决常见的 AWS CDK 问题
  - 工具包命令
- AWS Step Functions 资源：
  - AWS Step Functions 入门
  - 故障排除
- 使用 Python 构建 Lambda 函数
- AWS 上的生成式 AI 应用程序生成器

其他资源

- LangChain 文档

使用 Amazon Personalize 生成个性化和重新排名的推荐

创建者：Mason Cahill (AWS)、Matthew Chasse (AWS) 和 Tayo Olajide (AWS)

摘要

此模式向您展示如何使用 Amazon Personalize 根据从这些用户那里摄取的实时用户互动数据，为用户生成个性化推荐（包括重新排名的推荐）。此模式中使用的示例场景基于宠物收养网站，该网站根据用户的互动（例如，用户访问的宠物）为其用户生成推荐。通过遵循示例场景，您将学习使用 Amazon Kinesis Data Streams 提取互动数据，使用 AWS Lambda 生成推荐并对推荐进行重新排名，以及使用 Amazon Data Firehose 将数据存储在亚马逊简单存储服务 (Amazon S3) 存储桶中。您还将学习使用 AWS Step Functions 来构建状态机以管理生成推荐的解决方案版本（即经过训练的模型）。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account，其中包含已引导的 AWS Cloud Development Kit（AWS CDK）
- AWS 命令行界面（AWS CLI），带有配置的凭证
- Python 3.9

产品版本

- Python 3.9
- AWS CDK 2.23.0 或更高版本
- AWS CLI 2.7.27 或更高版本

架构

技术堆栈

- Amazon Data Firehose
- Amazon Kinesis Data Streams
- Amazon Personalize
- Amazon Simple Storage Service（Amazon S3）

- AWS Cloud Development Kit (AWS CDK)
- AWS 命令行界面（AWS CLI）
- AWS Lambda
- AWS Step Functions

目标架构

下图说明了将实时数据摄取到 Amazon Personalize 的管线。然后，该管线使用这些数据为用户生成个性化和重新排名的推荐。

图表显示了以下工作流：

1. Kinesis Data Streams 提取实时用户数据（例如，拜访过的宠物之类的事件），由 Lambda 和 Firehose 处理。
2. Lambda 函数处理来自 Kinesis Data Streams 的记录，并发出 API 调用，将记录中的用户互动添加到 Amazon Personalize 中的事件跟踪器中。
3. 基于时间的规则调用 Step Functions 状态机，并使用 Amazon Personalize 中事件跟踪器中的事件为推荐和重新排名模型生成新的解决方案版本。
4. Amazon Personalize 活动由状态机更新，以使用新的解决方案版本。
5. Lambda 通过调用 Amazon Personalize 重新排名活动对推荐项目列表进行重新排名。
6. Lambda 通过调用 Amazon Personalize 推荐活动对推荐项目列表进行检索。
7. Firehose 将事件保存到 S3 存储桶中，在那里可以将其作为历史数据进行访问。

工具

AWS 工具

- AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，可帮助您在代码中定义并预置 Amazon Web Services Cloud 基础设施。
- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- Amazon Data Firehose 可帮助您将实时流数据传输到其他 AWS 服务、自定义 HTTP 终端节点以及受支持的第三方服务提供商拥有的 HTTP 终端节点。

- Amazon Kinesis Data Streams 可帮助您实时收集和处理大型数据记录流。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Personalize 是一项完全托管的机器学习（ML）服务，可帮助您根据数据为用户生成项目推荐。
- AWS Step Functions 是一项无服务器编排服务，可帮助您搭配使用 Lambda 函数和其他 Amazon Web Services 来构建业务关键型应用程序。

其他工具

- pytest 是一个 Python 框架，用于编写可读的小型测试。
- Python 是通用的计算机编程语言。

代码

此模式的代码可在 GitHub 动物推荐器存储库中找到。您可以使用此存储库 CloudFormation 中的 AWS 模板来部署示例解决方案的资源。

> **Note**
>
> Amazon Personalize 解决方案版本、事件跟踪器和活动由基于原生资源的自定义资源（在基础设施内）CloudFormation 提供支持。

操作说明

创建基础设施

Task	描述	所需技能
创建一个隔离的 Python 环境。	Mac/Linux 安装程序 1. 要手动创建虚拟环境，请从终端运行该 `$ python3 -m venv .venv` 命令。 2. 初始化过程完成后，运行 `$ source .venv/bin/`	DevOps 工程师

Task	描述	所需技能
	`activate` 命令以激活虚拟环境。 Windows 设置 要手动创建虚拟环境，请从终端运行该 `% .venv\Scripts\activate.bat` 命令。	

Task	描述	所需技能
合成 CloudFormation 模板。	1. 要安装所需依赖项，请从终端运行该 `$ pip install -r requirements.txt` 命令。 2. 在 AWS CLI 中，设置以下环境变量： • `export ACCOUNT_ID=123456789` • `export CDK_DEPLOY_REGION=us-east-1` • `export CDK_ENVIRONMENT=dev` 3. 在 `config/{env}.yml` 文件中，更新 `vpcId` 以匹配虚拟私有云（VPC）ID。 4. 要合成此代码的 CloudFormation 模板，请运行`$ cdk synth`命令。 **Note** 在步骤 2 中，`CDK_ENVIRONMENT` 引用该`config/{env}.yml` 文件。	DevOps 工程师

Task	描述	所需技能
部署资源并创建基础设施。	要部署解决方案资源，请从终端运行该 `./deploy.sh` 命令。 此命令安装所有必需的 Python 依赖项。Python 脚本创建 S3 存储桶和 AWS Key Management Service（AWS KMS）密钥，然后添加用于创建初始模型的种子数据。最后，该脚本运行 `cdk deploy` 以创建剩余的基础设施。 **Note** 初始模型训练发生在堆栈创建期间。堆栈最长可能需要两个小时才能完成创建。	DevOps 工程师

相关资源

- 动物推荐者 () GitHub
- AWS CDK 参考文档
- Boto3 文档
- 使用 Amazon Personalize 针对您选择的业务指标优化个性化推荐（AWS 机器学习博客）

其他信息

有效负载和响应示例

推荐 Lambda 函数

要检索推荐，请使用以下格式的有效负载向推荐 Lambda 函数提交请求：

```
{
  "userId": "3578196281679609099",
  "limit": 6
}
```

以下示例响应包含动物组列表：

```
[{"id": "1-domestic short hair-1-1"},
{"id": "1-domestic short hair-3-3"},
{"id": "1-domestic short hair-3-2"},
{"id": "1-domestic short hair-1-2"},
{"id": "1-domestic short hair-3-1"},
{"id": "2-beagle-3-3"},
```

如果省略该 `userId` 字段，则该函数将返回一般建议。

重新排名 Lambda 函数

要使用重新排名，请向重新排名 Lambda 函数提交请求。有效载荷包含所有 IDs 要重新排名的项目及其元数据。`userId`以下示例数据使用 Oxford Pets 类表示 `animal_species_id`（1=猫，2=狗），使用整数 1-5 表示 `animal_age_id` 和 `animal_size_id`：

```
{
   "userId":"12345",
   "itemMetadataList":[
      {
         "itemId":"1",
         "animalMetadata":{
            "animal_species_id":"2",
            "animal_primary_breed_id":"Saint_Bernard",
            "animal_size_id":"3",
            "animal_age_id":"2"
         }
      },
      {
         "itemId":"2",
         "animalMetadata":{
            "animal_species_id":"1",
            "animal_primary_breed_id":"Egyptian_Mau",
            "animal_size_id":"1",
            "animal_age_id":"1"
         }
```

```
      },
      {
         "itemId":"3",
         "animalMetadata":{
            "animal_species_id":"2",
            "animal_primary_breed_id":"Saint_Bernard",
            "animal_size_id":"3",
            "animal_age_id":"2"
         }
      }
   ]
}
```

Lambda 函数对这些商品进行重新排名，然后返回包含该商品和来自 Amazon Personalize 的直接回复的排序列表。 IDs 这是项目所属动物群及其分数的排名列表。Amazon Personalize 使用 User-Personalization 和 Personalized-Ranking 食谱，在推荐中包含每件项目的分数。这些分数表示 Amazon Personalize 对于用户接下来将选择哪个项目的相对确定性。分数越高，意味着确定性越大。

```
{
   "ranking":[
      "1",
      "3",
      "2"
   ],
   "personalizeResponse":{
      "ResponseMetadata":{
         "RequestId":"a2ec0417-9dcd-4986-8341-a3b3d26cd694",
         "HTTPStatusCode":200,
         "HTTPHeaders":{
            "date":"Thu, 16 Jun 2022 22:23:33 GMT",
            "content-type":"application/json",
            "content-length":"243",
            "connection":"keep-alive",
            "x-amzn-requestid":"a2ec0417-9dcd-4986-8341-a3b3d26cd694"
         },
         "RetryAttempts":0
      },
      "personalizedRanking":[
         {
            "itemId":"2-Saint_Bernard-3-2",
            "score":0.8947961
         },
         {
```

```
            "itemId":"1-Siamese-1-1",
            "score":0.105204
         }
      ],
      "recommendationId":"RID-d97c7a87-bd4e-47b5-a89b-ac1d19386aec"
   }
}
```

Amazon Kinesis 有效负载

发送到 Amazon Kinesis 的有效负载采用以下格式：

```
{
    "Partitionkey": "randomstring",
    "Data": {
        "userId": "12345",
        "sessionId": "sessionId4545454",
        "eventType": "DetailView",
        "animalMetadata": {
            "animal_species_id": "1",
            "animal_primary_breed_id": "Russian_Blue",
            "animal_size_id": "1",
            "animal_age_id": "2"
        },
        "animal_id": "98765"

    }
}
```

> **Note**
>
> 对于未经身份验证的用户，该`userId`字段已被删除。

使用 SageMaker AI 和 Hydra 简化从本地开发到可扩展实验的机器学习工作流程

由 David Sauerwein (AWS)、Julian Ferdinand Grueber (AWS) 和 Marco Geiger (AWS) 创作

摘要

这种模式为在 Amazon A SageMaker I 上配置和运行从本地测试到生产的机器学习 (ML) 算法提供了一种统一的方法。机器学习算法是这种模式的焦点，但其方法延伸到特征工程、推理和整个 ML 管道。此模式通过示例用例演示了从本地脚本开发到 SageMaker AI 训练作业的过渡。

典型的机器学习工作流程是在本地计算机上开发和测试解决方案，在云端运行大规模实验（例如，使用不同的参数），然后在云端部署经批准的解决方案。然后，必须对部署的解决方案进行监控和维护。如果没有统一的工作流程方法，开发人员通常需要在每个阶段重构代码。如果解决方案依赖于大量参数，而这些参数在此工作流程的任何阶段都可能发生变化，那么保持井井有条和一致性就会变得越来越困难。

这种模式解决了这些挑战。首先，它提供了一个统一的工作流程，无论是在本地计算机、容器中还是在人工智能上运行，都能保持一致，从而无需在环境之间进行 SageMaker 代码重构。其次，它通过Hydra的配置系统简化了参数管理，在该系统中，参数是在单独的配置文件中定义的，可以轻松修改和组合，并自动记录每次运行的配置。有关此模式如何解决这些挑战的更多详细信息，请参阅其他信息。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 用于部署和启动 A SageMaker I 训练作业的 AWS Identity and Access Management (IAM) 用户角色
- AWS Command Line Interface (AWS CLI) 已安装并配置 2.0 或更高版本
- 已安装@@ 诗歌版本 1.8 或更高版本，但低于 2.0
- 已@@ 安装 Docker
- Python 版本 3.10.x

限制

- 该代码目前仅针对 A SageMaker I 训练作业。将其扩展到处理任务和整个 SageMaker AI 管道非常简单。

- 对于完全生产化的 SageMaker AI 设置，需要准备其他细节。示例可以是用于计算和存储的自定义 AWS Key Management Service (AWS KMS) 密钥，也可以是网络配置。您还可以通过在文件夹的专用子文件夹中使用 Hydra 来配置这些其他选项。`config`
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅按地区划分的AWS 服务。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

架构

下图描述了解决方案的架构。

图表显示了以下工作流：

1. 数据科学家无需使用 Docker 或 SageMaker AI 即可在本地环境中对算法进行小规模迭代，调整参数并快速测试训练脚本。（有关更多详细信息，请参阅 Epics 中的 “在本地运行以进行快速测试” 任务。）
2. 一旦对算法感到满意，数据科学家就会构建 Docker 镜像并将其推送到名为的亚马逊弹性容器注册表 (Amazon ECR) Container Registry 存储库。`hydra-sm-artifact`（有关更多详细信息，请参阅 Epics 中的 “在 SageMaker AI 上运行工作流程”。）
3. 数据科学家使用 Python 脚本启动 SageMaker AI 训练作业或超参数优化 (HPO) 作业。对于常规训练作业，调整后的配置将写入名为的亚马逊简单存储服务 (Amazon S3) 存储桶。`hydra-sample-config`对于 HPO 作业，将应用位于`config`文件夹中的默认配置集。
4. SageMaker AI 训练作业提取 Docker 镜像，从 Amazon S3 存储桶读取输入数据`hydra-sample-data`，然后从 Amazon S3 存储桶获取配置`hydra-sample-config`或使用默认配置。训练结束后，该任务会将输出数据保存到 Amazon S3 存储桶`hydra-sample-data`。

自动化和扩缩

- 对于自动训练、再训练或推理，您可以将 AWS CLI 代码与、或 Amazon 等AWS Lambda服务集成。AWS CodePipeline EventBridge
- 可以通过更改实例大小的配置或为分布式训练添加配置来实现扩展。

工具

AWS 服务

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行 shell 中的命令进行交互。对于这种模式， AWS CLI 对于初始资源配置和测试都很有用。
- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- Amazon SageMaker AI 是一项托管机器学习 (ML) 服务，可帮助您构建和训练机器学习模型，然后将其部署到生产就绪的托管环境中。 SageMaker AI Training 是 SageMaker AI 中一项完全托管的机器学习服务，支持大规模训练机器学习模型。该工具利用内置的可扩展性以及与其他 AWS 服务模型的集成，可以有效地处理训练模型的计算需求。 SageMaker AI Training 还支持自定义算法和容器，使其灵活适用于各种机器学习工作流程。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

其他工具

- Docker 是一组平台即服务（PaaS）产品，它们使用操作系统级别的虚拟化技术在容器中交付软件。在这种模式中，它用于确保从开发到部署的各个阶段的环境保持一致，并可靠地打包依赖关系和代码。Docker 的容器化允许在整个工作流程中轻松扩展和版本控制。
- Hydra 是一款配置管理工具，它为处理多种配置和动态资源管理提供了灵活性。它在管理环境配置方面发挥了重要作用，允许在不同的环境中进行无缝部署。有关 Hydra 的更多详细信息，请参阅其他信息。
- Python 是通用的计算机编程语言。Python 用于编写机器学习代码和部署工作流程。
- P@@ o etry 是一款在 Python 中进行依赖管理和打包的工具。

代码存储库

此模式的代码可在-w GitHub configuring-sagemaker-training-jobsith-h ydra 存储库中找到。

最佳实践

- 选择一个 IAM 角色来部署和启动 SageMaker AI 训练作业，该角色遵循最低权限原则，并授予执行任务所需的最低权限。有关更多信息，请参阅 IAM 文档中的授予最低权限和安全最佳实践。
- 使用临时证书访问终端中的 IAM 角色。

操作说明

设置环境

Task	描述	所需技能

创建并激活虚拟环境。	要创建和激活虚拟环境，请在存储库的根目录中运行以下命令：
	`poetry install`
	`poetry shell`
		常规 AWS

部署基础设施。	要使用部署基础架构 CloudFormation，请运行以下命令： `aws cloudformation deploy --template-file infra/hydra-sagemaker-setup.yaml --stack-name hydra-sagemaker-setup --capabilities CAPABILITY_NAMED_IAM`	通用 AWS， DevOps 工程师
下载示例数据。	要将输入数据从 openml 下载到本地计算机，请运行以下命令： `python scripts/download_data.py`	常规 AWS
在本地运行以进行快速测试。	要在本地运行训练代码进行测试，请运行以下命令： `python mypackage/train.py data.train_data_path=data/t`	数据科学家

Task	描述	所需技能
	`rain.csv evaluatio n.base_dir_path=data` 所有执行的日志按执行时间存储在名为的文件夹中`outputs`。有关更多信息，请参阅GitHub 存储库中的 “输出” 部分。 通过使用该`--multirun` 功能，您还可以使用不同的参数并行执行多项训练。有关更多详细信息，请参阅 Hydra 文档。	

在 SageMaker AI 上运行工作流程

Task	描述	所需技能

设置环境变量。	要在 SageMaker AI 上运行作业，请设置以下环境变量，并提供您 AWS 区域 和您的 AWS 账户 ID：
	
	`export ECR_REPO_ NAME=hydra-sm-arti fact`
	`export image_tag =latest`
	`export AWS_REGIO N="<your_aws_regio n>" # for instance, us- east-1`
	`export ACCOUNT_I D="<your_account_id>"`
		常规 AWS

Task	描述	所需技能

	`export BUCKET_NAME_DATA=hydra-sample-data-$ACCOUNT_ID`
	`export BUCKET_NAME_CONFIG=hydra-sample-config-$ACCOUNT_ID`
	`export AWS_DEFAULT_REGION=$AWS_REGION`
	`export ROLE_ARN=arn:aws:iam::${ACCOUNT_ID}:role/hydra-sample-sagemaker`
	`export INPUT_DATA_S3_PATH=s3://$BUCKET_NAME_DATA/hydra-on-sm/input/`
	`export OUTPUT_DATA_S3_PATH=s3://$BUCKET_NAME_DATA/hydra-on-sm/output/`	

Task	描述	所需技能

创建并推送 Docker 镜像。	要创建 Docker 镜像并将其推送到 Amazon ECR 存储库，请运行以下命令：
	`chmod +x scripts/create_and_push_image.sh`
	`scripts/create_and_push_image.sh $ECR_REPO_NAME $image_tag $AWS_REGION $ACCOUNT_ID`
	此任务假设您在自己的环境中拥有有效的证书。Docker 映像被推送到上一个任务中环境变量中指定的 Amazon ECR 存储库，用于激活将在其中运行训练作业的 SageMaker AI 容器。
		机器学习工程师，通用 AWS

将输入数据复制到 Amazon S3。	A SageMaker I 训练作业需要获取输入数据。要将输入数据复制到 Amazon S3 存储桶以获取数据，请运行以下命令： `aws s3 cp data/train.csv "${INPUT_DATA_S3_PATH}train.csv"`	数据工程师，通用 AWS

Task	描述	所需技能
提交 SageMaker AI 训练作业。	要简化脚本的执行，请在`default.yaml` 文件中指定默认配置参数。除了确保运行间的一致性外，这种方法还提供了根据需要轻松覆盖默认设置的灵活性。请参见以下示例： `python scripts/start_sagemaker_training_job.py sagemaker.role_arn=$ROLE_ARN sagemaker.config_s3_bucket=$BUCKET_NAME_CONFIG sagemaker.input_data_s3_path=$INPUT_DATA_S3_PATH sagemaker.output_data_s3_path=$OUTPUT_DATA_S3_PATH`	AWS 常规、机器学习工程师、数据科学家

Task	描述	所需技能
运行 SageMaker AI 超参数调整。	运行 SageMaker AI 超参数调整与提交 SageMaker AII 训练作业类似。但是，正如你在 start_sagemaker_hpo_job.py 文件中看到的那样，执行脚本在一些重要方面有所不同。要调整的超参数必须通过 boto3 有效载荷传递，而不是传递到训练作业的通道。 要启动超参数优化 (HPO) 作业，请运行以下命令： `python scripts/start_sagemaker_hpo_job.py sagemaker.role_arn=$ROLE_ARN sagemaker.config_s3_bucket=$BUCKET_NAME_CONFIG sagemaker.input_data_s3_path=$INPUT_DATA_S3_PATH sagemaker.output_data_s3_path=$OUTPUT_DATA_S3_PATH`	数据科学家

故障排除

事务	解决方案
令牌已过期	导出新的 AWS 凭证。
缺乏 IAM 权限	请务必导出具有部署 CloudFormation 模板和启动 A SageMaker I 训练任务所需的所有 IAM 权限的 IAM 角色的证书。

相关资源

- 使用 Amazon A SageMaker I 训练模型（AWS 文档）
- 什么是超参数调整？

其他信息

这种模式解决了以下挑战：

从本地开发到大规模部署的一致性 — 通过这种模式，开发人员可以使用相同的工作流程，无论他们是在使用本地 Python 脚本、运行本地 Docker 容器、在 SageMaker AI 上进行大型实验，还是在生产中在 AI 上 SageMaker 部署。这种一致性很重要，原因如下：

- 更快的迭代 — 它允许快速进行局部实验，而无需在扩大规模时进行重大调整。
- 无需重构 — 可以无缝过渡到更大规模的 SageMaker AI 实验，无需对现有设置进行全面改革。
- 持续改进 — 开发新功能和不断改进算法非常简单，因为代码在不同环境中保持不变。

配置管理-此模式利用配置管理工具 Hydra 来提供以下好处：

- 参数在配置文件中定义，与代码分开。
- 不同的参数集可以轻松交换或组合。
- 由于每次运行的配置都会自动记录，因此简化了实验跟踪。
- 云实验可以使用与本地运行相同的配置结构，从而确保一致性。

使用 Hydra，您可以有效地管理配置，启用以下功能：

- 划分配置 — 将您的项目配置分成更小、易于管理的部分，可以独立修改。这种方法可以更轻松地处理复杂的项目。
- 轻松调整默认值 — 快速更改基准配置，让测试新想法变得更加简单。
- 对齐 CLI 输入和配置文件 — 将命令行输入与配置文件顺畅地组合在一起。这种方法可以减少混乱和混乱，使您的项目随着时间的推移更易于管理。

在 Amazon 上训练和部署支持 GPU 的自定义机器学习模型 SageMaker

创建者：Ankur Shukla（AWS）

摘要

训练和部署支持图形处理单元 (GPU) 的机器学习 (ML) 模型需要对某些环境变量进行初始设置和初始化，才能充分发挥 NVIDIA GPUs 的优势。但是，在亚马逊网络服务 (AWS) 云上设置环境并使其与亚马逊 SageMaker 架构兼容可能很耗时。

此模式可帮助您使用 Amazon 训练和构建支持 GPU 的自定义机器学习模型。 SageMaker它提供了训练和部署基于开源 Amazon Reviews 数据集构建的自定义 CatBoost 模型的步骤。然后，您可以在`p3.16xlarge`亚马逊弹性计算云 (Amazon EC2) 实例上对其性能进行基准测试。

如果您的组织想要在上部署支持 GPU 的现有机器学习模型，则此模式非常有用。 SageMaker数据科学家可以按照此模式中的步骤创建支持 NVIDIA GPU 的容器，并在这些容器上部署 ML 模型。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- Amazon Simple Storage Service（Amazon S3）源存储桶，用于存储模型构件和预测。
- 了解 SageMaker 笔记本实例和 Jupyter 笔记本。
- 了解如何创建 AWS 身份和访问管理 (IAM) 角色，该角色具有基本 SageMaker 角色权限、S3 存储桶访问和更新权限以及亚马逊弹性容器注册表 (Amazon ECR) Container Registry (Amazon ECR) Container Registry 的额外权限。

限制

- 此模式适用于以 Python 编写的训练和部署代码的受监管机器学习工作负载。

架构

技术堆栈

- SageMaker
- Amazon ECR

工具

工具

- Amazon ECR – Amazon Elastic Container Registry (Amazon ECR) 是一项 AWS 托管容器映像注册表服务，它安全、可扩展且可靠。
- Amazon SageMaker — SageMaker 是一项完全托管的机器学习服务。
- Docker – 是软件平台，可以快速构建、测试和部署应用程序。
- Python – Python 是一种编程语言。

代码

此模式的代码可在使用 Catboost 和 SageMaker存储库 GitHub 实现审查分类模型中找到。

操作说明

准备数据

Task	描述	所需技能
创建 IAM 角色并向其附加所需策略。	登录 AWS 管理控制台，打开 IAM 控制台并创建新的 IAM 角色。附加以下策略到 IAM 角色： • `AmazonEC2ContainerRegistryFullAccess` • `AmazonS3FullAccess` • `AmazonSageMakerFullAccess`	数据科学家

Task	描述	所需技能
	有关这方面的更多信息，请参阅 Amazon SageMaker 文档中的创建笔记本实例。	
创建 SageMaker 笔记本实例。	打开 SageMaker 控制台，选择笔记本实例，然后选择创建笔记本实例。对于 IAM 角色，选择您之前创建的 IAM 角色。根据要求配置笔记本实例，然后选择创建笔记本实例。 有关详细步骤和说明，请参阅 Amazon SageMaker 文档中的创建笔记本实例。	数据科学家
克隆存储库。	在 SageMaker 笔记本实例中打开终端，运行以下命令克隆使用 Catboost 和 SageMaker存储库 GitHub实现评论分类模型： `git clone https://github.com/aws-samples/review-classification-using-catboost-sagemaker.git`	
启动 Jupyter 笔记本。	启动 `Review classification model with Catboost and SageMaker.ipynb` Jupyter 笔记本，其中包含预定义的步骤。	数据科学家

特征工程

Task	描述	所需技能
在 Jupyter 笔记本中运行命令。	打开 Jupyter 笔记本并运行以下情节中的命令，准备用于训练 ML 模型的数据。	数据科学家

从 S3 存储桶读取数据。	`import pandas as pd`
	`import csv`
	`fname = 's3://ama`
	`zon-reviews-pds/ts`
	`v/amazon_reviews_u`
	`s_Digital_Video_Do`
	`wnload_v1_00.tsv.gz'`
	`df = pd.read_csv(fname,`
	`sep='\t',delimiter`
	`='\t',error_bad_li`
	`nes=False)`
		数据科学家

预处理数据。	`import numpy as np`
	`def pre_process(df):`
	
	`  df.fillna(value={'`
	`review_body': '',`
	`'review_headline':`
	`''}, inplace=True)`
	`df.fillna( value={'v`
	`erified_purchase':`
	`'Unk'}, inplace=True)`
	
	`df.fillna(0,`
	`inplace=True)`
	`return df`
	`df = pre_process(df)`
	`df.review_date =`
	`pd.to_datetime(df.`
	`review_date)`
	`df['target'] =`
	`np.where(df['star_`
	`rating']>=4,1,0)`
		数据科学家

Task	描述	所需技能
	Note 此代码将中的空值替换为空字符串，并将该`'verified_purchase'` 列替换为`'Unk'`，这意味着“未知”。`'review_body'`	

Task	描述	所需技能

将数据拆分为训练、验证和测试数据集。	要保持目标标签在拆分集中的分布相同，必须使用 scikit-learn library 对采样进行分层。
	
	`from sklearn.model_selection import StratifiedShuffleSplit`
	`sss = StratifiedShuffleSplit(n_splits=2, test_size=0.10, random_state=0)`
	`sss.get_n_splits(df, df['target'])`
	`for train_index, test_index in sss.split(df, df['target']):`
	`X_train_vallid , X_test = df.iloc[train_index], df.iloc[test_index]`
	
	`sss.get_n_splits(X_train_vallid, X_train_vallid['target'])`
	`for train_index, test_index in sss.split(X_train_vallid, X_train_vallid['target']):`
	`X_train , X_valid = X_train_vallid.iloc[train_index], X_train_vallid.iloc[test_index]`
		数据科学家

构建、运行 Docker 映像并将其推送到 Amazon ECR

Task	描述	所需技能
准备并推送 Docker 映像。	在 Jupyter 笔记本中，运行以下情节中的命令来准备 Docker 映像并将其推送到 Amazon ECR。	机器学习工程师

在 Amazon ECR 中创建存储库。	`%%sh`
	
	`algorithm_name=s agemaker-catboost- github-gpu-img`
	
	`chmod +x code/train chmod +x code/serve`
	
	`account=$(aws sts get- caller-identity -- query Account --output text)`
	
	`# Get the region defined in the current configuration (default to us-west-2 if none defined) region=$(aws configure get region) region=${region:-us- east-1}`
	
	`fullname="${accou nt}.dkr.ecr.${regi on}.amazonaws.com/ ${algorithm_name}: latest"`
	
	`aws ecr create-re pository --repository-`
		机器学习工程师

Task	描述	所需技能
	`name "${algorithm_name} " > /dev/nul`	
在本地构建 Docker 映像。	`docker build -t "${algorithm_name}" . docker tag ${algorithm_name} ${fullname}`	机器学习工程师
运行 Docker 映像并将其推送到 Amazon ECR。	`docker push ${fullname}`	机器学习工程师

训练

Task	描述	所需技能
创建 SageMaker 超参数调整作业。	在 Jupyter 笔记本中，运行以下故事中的命令，使用你的 Docker 镜 SageMaker 像创建超参数调整作业。	数据科学家
创建 SageMaker 估算器。	使用 Docker 镜像的名称创建SageMaker 估算器。 `import sagemaker as sage from time import gmtime, strftime sess = sage.Session() from sagemaker.tuner import IntegerParameter, CategoricalParameter, ContinuousParameter, HyperparameterTuner  account = sess.boto_session.client('s`	数据科学家

Task	描述	所需技能
	ts').get_caller_id entity()['Account'] region = sess.boto _session.region_name image = '{}.dkr.e cr.{}.amazonaws.co m/sagemaker-catboo st-github-gpu-img: latest'.format(acc ount, region) tree_hpo = sage.esti mator.Estimator(im age, role, 1, 'ml.p3.16xlarge', train_volume_size = 100, output_path="s3:// {}/sagemaker/DEMO- GPU-Catboost/outpu t".format(bucket), sagemaker_session= sess)	

Task	描述	所需技能

创建 HPO 作业。	使用参数范围创建超参数优化（HPO）调整作业，并将训练集和验证集作为参数传递给函数。
	
	`hyperparameter_ranges = {'iterations': IntegerParameter(80000, 130000),`
	
	`'max_depth': IntegerParameter(6, 10),`
	
	`'max_ctr_complexity': IntegerParameter(4, 10),`
	
	`'learning_rate': ContinuousParameter(0.01, 0.5)}`
	
	`objective_metric_name = 'auc'`
	`metric_definitions = [{'Name': 'auc',`
	
	`'Regex': 'auc: ([0-9\.]+)'}]`
	
	`tuner = HyperparameterTuner(tree_hpo,`
	
	`objective_metric_name,`
	
	`hyperparameter_ranges,`
	
	`metric_definitions,`
		数据科学家

Task	描述	所需技能

	`objective_type='Maximize',`
	`max_jobs=50,`
	`max_parallel_jobs=2)`	

运行 HPO 作业。	`train_location = 's3://'+bucket+'/sagemaker/DEMO-GPU-Catboost/data/train/'`
	`valid_location = 's3://'+bucket+'/sagemaker/DEMO-GPU-Catboost/data/valid/'`
	
	`tuner.fit({'train': train_location, 'validation': valid_location })`
		数据科学家

接收表现最好的训练作业。	`import sagemaker as sage`
	`from time import gmtime, strftime`
	`sess = sage.Session()`
	
	`best_job =tuner.best_training_job()`
		数据科学家

批量转换

Task	描述	所需技能
对测试数据创建 SageMaker 批量转换作业，以进行模型预测。	在 Jupyter 笔记本中，运行以下故事中的命令，通过 SageMaker 超参数调整作业创建模型，并提交对测试数据的 SageMaker 批量转换作业以进行模型预测。	数据科学家

创建 SageMaker 模型。	使用最佳训练作业在 SageMaker 模型中创建模型。
	`attached_estimator = sage.estimator.Estimator.attach(best_job)`
	`output_path ='s3://'+bucket+'/sagemaker/DEMO-GPU-Catboost/data/test-predictions/'`
	`input_path ='s3://'+bucket+'/sagemaker/DEMO-GPU-Catboost/data/test/'`
	`transformer = attached_estimator.transformer(instance_count=1,`
	`instance_type='ml.p3.16xlarge',`
	`assemble_with='Line',`
		数据科学家

Task	描述	所需技能
	accept='text/csv', max_payload=1, output_path=output_path, env = {'SAGEMAKER_MODEL_SERVER_TIMEOUT' : '3600' })	
创建批量转换作业。	在测试数据集上创建批量转换作业。 transformer.transform(input_path, content_type='text/csv', split_type='Line')	数据科学家

分析结果

Task	描述	所需技能
阅读结果并评估模型的性能。	在 Jupyter 笔记本中，运行以下情节中的命令，阅读结果并评估模型在 ROC 曲线下面积（ROC-AUC）模型和精确率-召回率曲线下面积（PR-AUC）模型指标的性能。	数据科学家

Task	描述	所需技能
	有关这方面的更多信息，请参阅 Amazon Machine Learning (Amazon ML) 文档中的 Amazon 机器学习关键概念。	

阅读批量转换作业结果。	将批量转换作业结果读入数据框。
	
	`file_name = 's3://'+bucket+'/sagemaker/DEMO-GPU-Catboost/data/test-predictions/file_1.out'`
	
	`results = pd.read_csv(file_name, names=['review_id','target','score'] ,sep='\t',escapechar ='\\' , quoting=csv.QUOTE_NONE,`
	
	`lineterminator='\n',quotechar='"').dropna()`
		数据科学家

Task	描述	所需技能

评估性能指标。	在 ROC-AUC 和 PR-AUC 上评估模型的性能。
	
	`from sklearn import metrics`
	`import matplotlib`
	`import pandas as pd`
	`matplotlib.use('agg', warn=False, force=True)`
	`from matplotlib import pyplot as plt`
	
	`%matplotlib inline`
	
	`def analyze_results(labels, predictions):`
	`precision, recall, thresholds = metrics.precision_recall_curve(labels, predictions)`
	`auc = metrics.auc(recall, precision)`
	
	`fpr, tpr, _ = metrics.roc_curve(labels, predictions)`
	`roc_auc_score = metrics.roc_auc_score(labels, predictions)`
	
	`print('Neural-Nets: ROC auc=%.3f' % ( roc_auc_score))`
	
	`plt.plot(fpr, tpr, label="data 1, auc=" + str(roc_auc_score))`
		数据科学家

Task	描述	所需技能

	plt.xlabel('1-Specificity')
	plt.ylabel('Sensitivity')
	plt.legend(loc=4)
	plt.show()
	
	lr_precision, lr_recall, _ = metrics.precision_recall_curve(labels, predictions)
	lr_auc = metrics.auc(lr_recall, lr_precision)
	# summarize scores
	print('Neural-Nets: PR auc=%.3f' % (lr_auc))
	# plot the precision-recall curves
	no_skill = len(labels[labels==1.0]) / len(labels)
	plt.plot([0, 1], [no_skill, no_skill], linestyle='--', label='No Skill')
	
	plt.plot(lr_recall, lr_precision, marker='.', label='Neural-Nets')
	# axis labels
	plt.xlabel('Recall')
	plt.ylabel('Precision')
	# show the legend
	plt.legend()
	# show the plot	

Task	描述	所需技能
	plt.show() return auc analyze_results(results['target'].values,results['score'].values)	

相关资源

- SageMaker 通过构建 Scikit Docker 容器在亚马逊中训练和托管 Scikit-Learn 模型

其他信息

以下列表显示了 Dockerfile 的不同元素，这些元素在构建、运行 Docker 映像并将其推送到 Amazon ECR操作说明中运行。

使用 aws-cli 安装 Python。

```
FROM amazonlinux:1

 RUN yum update -y && yum install -y python36 python36-devel python36-libs python36-
tools python36-pip && \
 yum install gcc tar make wget util-linux kmod man sudo git -y && \
 yum install wget -y && \
 yum install aws-cli -y && \
 yum install nginx -y && \
yum install gcc-c++.noarch -y && yum clean all
```

安装 Python 软件包

```
RUN pip-3.6 install --no-cache-dir --upgrade pip && \pip3 install --no-cache-dir --
upgrade setuptools && \
pip3 install Cython && \
```

```
pip3 install --no-cache-dir numpy==1.16.0 scipy==1.4.1 scikit-learn==0.20.3
 pandas==0.24.2 \
flask gevent gunicorn boto3 s3fs  matplotlib joblib catboost==0.20.2
```

安装 CUDA 和 CuDNN

```
RUN wget https://developer.nvidia.com/compute/cuda/9.0/Prod/local_installers/
cuda_9.0.176_384.81_linux-run \
&& chmod u+x cuda_9.0.176_384.81_linux-run \
&& ./cuda_9.0.176_384.81_linux-run --tmpdir=/data --silent --toolkit --override \
&& wget https://custom-gpu-sagemaker-image.s3.amazonaws.com/installation/cudnn-9.0-
linux-x64-v7.tgz \
&& tar -xvzf cudnn-9.0-linux-x64-v7.tgz \
&& cp /data/cuda/include/cudnn.h /usr/local/cuda/include \
&& cp /data/cuda/lib64/libcudnn* /usr/local/cuda/lib64 \

&& chmod a+r /usr/local/cuda/include/cudnn.h /usr/local/cuda/lib64/libcudnn* \
&&  rm -rf /data/*
```

为创建所需的目录结构 SageMaker

```
RUN mkdir /opt/ml /opt/ml/input /opt/ml/input/config /opt/ml/input/data /opt/ml/input/
data/training /opt/ml/model /opt/ml/output /opt/program
```

设置 NVIDIA 环境变量

```
ENV PYTHONPATH=/opt/program
ENV PYTHONUNBUFFERED=TRUE
ENV PYTHONDONTWRITEBYTECODE=TRUE
ENV PATH="/opt/program:${PATH}"

# Set NVIDIA mount environments
ENV LD_LIBRARY_PATH=/usr/local/nvidia/lib:/usr/local/nvidia/lib64:$LD_LIBRARY_PATH
ENV NVIDIA_VISIBLE_DEVICES="all"
ENV NVIDIA_DRIVER_CAPABILITIES="compute,utility"
ENV NVIDIA_REQUIRE_CUDA "cuda>=9.0"
```

将训练和推理文件复制到 Docker 映像中

```
COPY code/* /opt/program/
WORKDIR /opt/program
```

将自然语言翻译成查询 DSL OpenSearch 和 Elasticsearch 查询

由 Tabby Ward (AWS)、Nicholas Switzer (AWS) 和 Breanne Warner (AWS) 创作

摘要

此模式演示了如何使用大型语言模型 (LLMs) 将自然语言查询转换为查询域特定的语言（查询 DSL），这使用户无需对查询语言有大量了解即可更轻松地与搜索服务（例如 OpenSearch 和 Elasticsearch）进行交互。对于希望通过自然语言查询功能增强基于搜索的应用程序，从而最终改善用户体验和搜索功能的开发人员和数据科学家来说，此资源特别有价值。

该模式说明了快速工程、迭代完善和整合专业知识的技术，所有这些对于合成数据生成都至关重要。尽管这种方法主要侧重于查询转换，但它隐含地证明了数据增强和可扩展合成数据生成的潜力。这一基础可以扩展到更全面的合成数据生成任务，以突出将非结构化自然语言输入与结构化、特定于应用程序的输出联系起来的力量。 LLMs

此解决方案不涉及传统意义上的迁移或部署工具。相反，它侧重于演示通过使用将自然语言查询转换为查询 DSL 的概念验证 (PoC)。 LLMs

- 该模式使用 Jupyter 笔记本作为设置环境和实现转换的 step-by-step指南。 text-to-query
- 它使用 Amazon Bedrock 进行访问 LLMs，这对于解释自然语言和生成适当的查询至关重要。
- 该解决方案专为与 Amazon OpenSearch 服务配合使用而设计。您可以在 Elasticsearch 上遵循类似的流程，生成的查询有可能适用于类似的搜索引擎。

查询 DSL 是一种灵活的、基于 JSON 的搜索语言，用于在 Elasticsearch 和. OpenSearch 它使您能够在搜索操作的查询参数中指定查询，并支持各种查询类型。DSL 查询包括树叶查询和复合查询。Leaf 查询在某些字段中搜索特定值，包括全文查询、术语级查询、地理查询、联接查询、跨度查询和专业查询。复合查询充当多个叶子句或复合子句的封装器，并合并它们的结果或修改其行为。Query DSL 支持创建复杂的搜索，从简单的全匹配查询到生成高度具体结果的复杂多子句查询。Query DSL 对于需要高级搜索功能、灵活的查询结构和基于 JSON 的查询结构的项目特别有价值。

此模式使用诸如少镜头提示、系统提示、结构化输出、提示链接、上下文配置和特定于任务的 DSL 转换提示等技术。 text-to-query有关这些技术的定义和示例，请参阅 “其他信息” 部分。

先决条件和限制

先决条件

要有效地使用 Jupyter 笔记本将自然语言查询转换为查询 DSL 查询，您需要：

- 熟悉 Jupyter 笔记本电脑。基本了解如何在 Jupyter 笔记本环境中浏览和运行代码。
- Python 环境。一个工作的 Python 环境，最好是 Python 3.x，并安装了必要的库。
- 弹性搜索或知识。 OpenSearch Elasticsearch 或的基础知识 OpenSearch，包括其架构和如何执行查询。
- AWS 账户。 已激活 AWS 账户 访问 Amazon Bedrock 和其他相关服务。
- 库和依赖关系。安装笔记本中提到的特定库，例如`boto3`用于 AWS 交互的库，以及 LLM 集成所需的任何其他依赖项。
- 在 Amazon Bedrock 中访问模型。这个模式使用了 Anthropic 中的三个 Clau LLMs de。打开 Amazon Bedrock 控制台并选择模型访问权限。在下一个屏幕上，选择 “启用特定模型”，然后选择以下三个模型：
  - 库罗德 3 十四行诗
  - 克劳德 3.5 十四行诗
  - Claude 3 俳句
- 正确的 IAM 策略和 IAM 角色。要在中运行笔记本 AWS 账户，您的 AWS Identity and Access Management (IAM) 角色需要`SagemakerFullAccess`策略以及其他信息部分中提供的策略，您可以命名`APGtext2querydslpolicy`。本政策包括订阅列出的三款 Claude 模型。

具备这些先决条件可确保您在使用笔记本电脑和实现 text-to-query功能时获得流畅的体验。

限制

- 概念验证状态。该项目主要用作概念验证 (PoC)。它展示了使用将自然语言查询 LLMs 转换为查询 DSL 的潜力，但它可能尚未完全优化或尚未准备好投入生产。
- 型号限制：

  上下文窗口约束。使用 Amazon Bedrock 上提供的内容时，请注意上下文窗口的限制： LLMs

  克劳德模型（截至 2024 年 9 月）：
  - Claude 3 Opus：20 万个代币
  - Claude 3 十四行诗：20 万个代币
  - Claude 3 Haiku：200,000

  Amazon Bedrock 上的其他型号可能具有不同的上下文窗口大小。请务必查看最新文档以获取最新信息。

型号可用性。Amazon Bedrock 上特定型号的供应情况可能有所不同。在实施此解决方案之前，请确保您可以访问所需的模型。

- 其他限制
  - 查询复杂性。自然语言对查询 DSL 转换的有效性可能因输入查询的复杂性而异。
  - 版本兼容性。生成的查询 DSL 可能需要根据特定的 Elasticsearch 版本或您 OpenSearch 使用的版本进行调整。
  - 性能。这种模式提供了 PoC 实现，因此查询生成速度和准确性可能不是大规模生产使用的最佳选择。
  - 成本。 LLMs 在 Amazon Bedrock 中使用会产生成本。请注意所选型号的定价结构。有关更多信息，请参阅 Amazon Bedrock 定价。
  - 维护。为了跟上 LLM 技术的进步和查询 DSL 语法的变化，可能需要定期更新提示和模型选择。

产品版本

此解决方案已在 Amazon OpenSearch 服务中进行了测试。如果您想使用 Elasticsearch，则可能需要进行一些更改才能复制此模式的确切功能。

- OpenSearch 版本兼容性。 OpenSearch 在主要版本中保持向后兼容性。例如：
  - OpenSearch 1.x 客户端通常与 OpenSearch 1.x 集群兼容。
  - OpenSearch 2.x 客户端通常与 OpenSearch 2.x 集群兼容。

  但是，如果可能，最好对客户端和集群使用相同的次要版本。
- OpenSearch API 兼容性。 OpenSearch 在大多数操作中都保持 API 与 Elasticsearch OSS 7.10.2 的兼容性。但是，存在一些差异，尤其是在较新的版本中。
- OpenSearch 升级注意事项：
  - 不支持直接降级。如果需要，可使用快照进行回滚。
  - 升级时，请查看兼容性表和发行说明，了解是否有任何重大更改。

弹性搜索注意事项

- 弹性搜索版本。您正在使用的 Elasticsearch 的主要版本至关重要，因为主要版本之间的查询语法和功能可能会发生变化。目前，最新的稳定版本是 Elasticsearch 8.x。请确保您的查询与您的特定的 Elasticsearch 版本兼容。

- Elasticsearch 查询 DSL 库版本。如果你使用的是 Elasticsearch 查询 DSL Python 库，请确保其版本与你的 Elasticsearch 版本相匹配。例如：
  - 对于 Elasticsearch 8.x，请使用大于或等于 8.0.0 但小于 9.0.0 的`elasticsearch-dsl`版本。
  - 对于 Elasticsearch 7.x，请使用大于或等于 7.0.0 但小于 8.0.0 的`elasticsearch-dsl`版本。
- 客户端库版本。无论您使用的是官方的 Elasticsearch 客户端还是特定语言的客户端，都要确保它与您的 Elasticsearch 版本兼容。
- 查询 DSL 版本。查询 DSL 会随着 Elasticsearch 版本的变化而发展。某些查询类型或参数可能会在不同的版本中被弃用或引入。
- 映射版本。为索引定义映射的方式以及在版本之间发生变化。请务必查看您的特定 Elasticsearch 版本的映射文档。
- 分析工具版本。如果您使用的是分析器、分词器或其他文本分析工具，则它们的行为或可用性可能会在不同版本之间发生变化。

架构

目标架构

下图阐明了该模式的架构。

其中：

1. 用户输入和系统提示以及少量提示示例。该过程从提供自然语言查询或架构生成请求的用户开始。
2. Amazon Bedrock。输入内容将发送到 Amazon Bedrock，后者充当访问 Claude LLM 的接口。
3. Claude 3 Sonnet LLM Amazon Bedrock 使用 Claude 3 系列中的 Claude 3 Sonnet LLMs 来处理输入。它解释并生成相应的 Elasticsearch 或 OpenSearch 查询 DSL。对于架构请求，它会生成合成的 Elasticsearch 或 OpenSearch 映射。
4. 查询 DSL 生成。对于自然语言查询，应用程序会获取 LLM 的输出并将其格式化为有效的 Elasticsearch 或 OpenSearch 服务查询 DSL。
5. 合成数据生成。该应用程序还使用架构来创建合成的 Elasticsearch，或者将 OpenSearch 数据加载到 OpenSearch 无服务器集合中进行测试。
6. OpenSearch 或 Elasticsearch 生成的 Query DSL 将针对所有索引上的 OpenSearch 无服务器集合进行查询。JSON 输出包含位于 OpenSearch 无服务器集合中的数据的相关数据和命中次数。

自动化和扩缩

此模式提供的代码严格按照 PoC 目的构建。以下列表为进一步自动化和扩展解决方案以及将代码移至生产环境提供了一些建议。这些增强功能超出了此模式的范围。

- 容器化：
  - 对应用程序进行 Dockerize 以确保不同环境之间的一致性。
  - 使用诸如亚马逊弹性容器服务 (Amazon ECS) 或 Kubernetes 之类的容器编排平台进行可扩展的部署。
- 无服务器架构：
  - 将核心功能转换为 AWS Lambda 函数。
  - 使用 Amazon API Gateway 为自然语言查询输入创建 RESTful 终端节点。
- 异步处理：
  - 实施亚马逊简单队列服务 (Amazon SQS) Simple Queue Service，对传入的查询进行排队。
  - AWS Step Functions 用于协调处理查询和生成查询 DSL 的工作流程。
- 缓存：
  - 实现一种缓存提示的机制。
- 监控和记录：
  - 使用 Amazon CloudWatch 进行监控和提醒。
  - 使用 Amazon L CloudWatch ogs 或 Amazon OpenSearch 服务实现集中日志记录以进行日志分析。
- 安全增强功能：
  - 实施 IAM 角色以实现精细的访问控制。
  - 用于安全 AWS Secrets Manager 地存储和管理 API 密钥和证书。
- 多区域部署：
  - 考虑跨多个部署解决方案， AWS 区域 以改善延迟和灾难恢复。
  - 使用 Amazon Route 53 进行智能请求路由。

通过实施这些建议，您可以将此 PoC 转变为强大、可扩展且可用于生产的解决方案。我们建议您在完全部署之前对每个组件和整个系统进行全面测试。

工具

工具

- Amazon SageMaker AI 笔记本是用于机器学习开发的完全托管 Jupyter 笔记本电脑。这种模式使用笔记本作为交互式环境，在 Amazon A SageMaker I 中进行数据探索、模型开发和实验。笔记本电脑可与其他 SageMaker 人工智能功能无缝集成， AWS 服务.
- Python 是通用的计算机编程语言。这种模式使用 Python 作为核心语言来实现解决方案。
- Amazon Bedrock 是一项完全托管的服务，它通过统一的 API 提供来自领先的人工智能初创公司和亚马逊的高性能基础模型 (FMs) 供您使用。Amazon Bedrock 提供 LLMs 对自然语言处理的访问权限。此模式使用 Anthropic Claude 3 模型。
- 适用于 Python (Boto3) 的 AWS SDK是一款软件开发套件，可帮助您将 Python 应用程序、库或脚本与 AWS 服务（包括 Amazon Bedrock）集成。
- Amazon S OpenSearch er vice 是一项托管服务，可帮助您在 AWS 云中部署、运营和扩展 OpenSearch 服务集群。此模式使用 OpenSearch 服务作为生成查询 DSL 的目标系统。

代码存储库

此模式的代码可在 Text-to-QueryDSL使用 Claude 3 模型的 GitHub 提示工程存储库中找到。该示例使用健康社交媒体应用程序，该应用程序为用户创建帖子以及与健康应用程序关联的用户个人资料。

最佳实践

使用此解决方案时，请考虑以下几点：

- 需要适当的 AWS 凭证和权限才能访问 Amazon Bedrock
- 与使用 AWS 服务 和相关的潜在成本 LLMs
- 了解查询 DSL 对于验证并可能修改生成的查询非常重要

操作说明

设置环境并准备数据

Task	描述	所需技能
设置开发环境。	**Note** 有关此模式中此步骤以及其他步骤的详细说明	Python、pip、AWS 开发工具包

Task	描述	所需技能
	和代码，请参阅GitHub存储库中的综合演练。 1. 使用 p ip 安装所需的 Python 软件包，包括boto3numpyawscliopensearch-py 、、、和requests-aws4auth 。 2. 导入必要的模块boto3，例如jsonos、、opensearchpy 、opensearch RequestsHttpConnection bulk从Opensearchpy 、opensearchpy.helpers 、、sagemaker time、random、re、和AWS4Auth从requests_aws4auth 。	
设置 AWS 访问权限。	设置 Amazon Bedrock 客户端和 SageMaker AI 会话。检索 A SageMaker I 执行角色的 Amazon 资源名称 (ARN)，以便日后在创建 OpenSearch 无服务器集合时使用。	IAM、AWS CLI、亚马逊 Bedrock、亚马逊 SageMaker
加载健康应用程序架构。	从预定义文件中读取和解析健康帖子和用户个人资料的 JSON 架构。将架构转换为字符串，以便以后在提示中使用。	DevOps 工程师、通用 AWS、Python、JSON

生成合成数据

Task	描述	所需技能

创建基于 LLM 的数据生成器。	实现 g enerate_data () 函数，使用 Claude 3 模型调用 Amazon Bedrock Converse API。为 Sonnet、 IDs Sonnet 3.5 和 Haiku 设置模型：
	`model_id_sonnet3_5 = "anthropic.claude-3-5-sonnet-20240620-v1:0"`
	`model_id_sonnet = "anthropic.claude-3-sonnet-20240229-v1:0"`
	`model_id_haiku = "anthropic.claude-3-haiku-20240307-v1:0"`
		Python、Amazon Bedrock API、LLM 提示

创建合成生命值帖子。	使用带有特定消息提示的 g enerate_data () 函数，根据提供的架构创建合成健康帖子条目。函数调用如下所示： `health_post_data = generate_data(bedrock_rt, model_id_sonnet, system_prompt, message_healthpost, inference_config)`	Python，json
创建合成用户配置文件。	使用带有特定消息提示的 g enerate_data () 函数，根据提供的架构创建综合用户配置文件条目。这与生成生命值帖子类似，但使用的提示不同。	Python，json

设置 OpenSearch 和摄取数据

Task	描述	所需技能
设置 OpenSearch 无服务器集合。	使用 Boto3 创建具有适当加密、网络和访问策略的 OpenSearch 无服务器集合。集合的创建如下所示： `collection = aoss_client.create_collection(name=es_name, type='SEARCH')` 有关 OpenSearch 无服务器的更多信息，请参阅AWS 文档。	OpenSearch 无服务器、IAM
定义 OpenSearch 索引。	根据预定义的架构映射，使用 OpenSearch 客户端为健康帖子和用户配置文件创建索引。索引的创建如下所示： `response_health = oss_client.indices.create(healthpost_index, body=healthpost_body)`	OpenSearch，JSON
将数据加载到 OpenSearch。	运行 ingest_data () 函数，将合成健康帖子和用户配置文件批量插入各自的索引中。OpenSearch 该函数使用来自`opensearch-py` 以下内容的批量助手： `success, failed = bulk(oss_client, actions)`	Python、 OpenSearch API、批量数据操作

生成和运行查询

Task	描述	所需技能
设计少量提示示例。	使用 Claude 3 模型生成示例查询和相应的自然语言问题，作为查询生成的简短示例。系统提示包括以下示例： `system_prompt_query_generation = [{"text": f"""You are an expert query dsl generator. ... Examples: {example_prompt} ..."""}]`	LLM 提示，查询 DSL
创建一个 text-to-query DSL 转换器。	实现系统提示符，包括架构、数据和少量示例，用于生成查询。使用系统提示将自然语言查询转换为查询 DSL。函数调用如下所示： `query_response = generate_data(bedrock_rt, model_id, system_prompt_query_generation, query, inference_config)`	Python、Amazon Bedrock API、LLM 提示
开启测试查询 DSL。OpenSearch	运行 query_oss () 函数，对 OpenSearch 无服务器集合运行生成的查询 DSL 并返回结果。该函数使用 OpenSearch 客户端的搜索方法：	Python、 OpenSearch API、查询 DSL

Task	描述	所需技能
	`response = oss_clien t.search(index="_a ll", body=temp)`	

测试和评估

Task	描述	所需技能
创建测试查询集。	使用 Claude 3 根据综合数据和架构生成一组不同的试题： `test_queries = generate_data(bedr ock_rt, model_id_ sonnet, query_sys tem_prompt, query_pro mpt, inference_config)`	法学硕士提示
评估查询 DSL 转换的准确性。	通过运行查询 OpenSearch 并分析返回结果的相关性和准确性，测试生成的查询 DSL。这包括运行查询和检查命中： `output = query_oss (response1) print("Re sponse after running query against Opensearc h") print(output)`	Python、数据分析、查询 DSL
基准 Claude 3 模型。	比较不同的 Claude 3 模型（Haiku、Sonnet、Sonnet 3.5）在准确性和延迟方面的查询生成性能。要进行比较，请更改调用 `model_id` g	Python，性能基准测试

Task	描述	所需技能
	enerate_data () 的时间，然后测量执行时间。	

清理和记录

Task	描述	所需技能
制定清理流程。	使用后从 OpenSearch 无服务器集合中删除所有索引。	Python、AWS 开发工具包、OpenSearch API

相关资源

- 查询 DSL（OpenSearch 文档）
- 亚马逊 OpenSearch 服务文档
- OpenSearch 无服务器集合
- 亚马逊 Bedrock 文档
- 亚马逊 SageMaker AI 文档
- 适用于 Python (Boto3) 的 AWS SDK 文档

其他信息

IAM 策略

以下是此模式中使用的 IAM 角色的`APGtext2querydslpolicy`策略：

```
{
  "Version": "2012-10-17",
  "Statement": [
    { "Effect": "Allow",
      "Action": [
        "bedrock:InvokeModel",
        "bedrock:InvokeModelWithResponseStream"
      ],
      "Resource": "*"
    },
```

```
    { "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::sagemaker-*",
        "arn:aws:s3:::sagemaker-*/*"
      ]
    },
    { "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": "arn:aws:logs:*:*:log-group:/aws/sagemaker/*"
    },
    { "Effect": "Allow",
      "Action": [
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    },
    { "Effect": "Allow",
      "Action": [
        "aoss:*"
      ],
      "Resource": "*"
    },
    { "Effect": "Allow",
      "Action": [
        "iam:PassRole",
        "sagemaker:*"
      ],
      "Resource": [
        "arn:aws:iam::*:role/*", "*"
      ],
      "Condition": {
        "StringEquals": {
          "iam:PassedToService": "sagemaker.amazonaws.com"
```

```
          }
      }
    },
    { "Effect": "Allow",
      "Action": [
        "codecommit:GetBranch",
        "codecommit:GetCommit",
        "codecommit:GetRepository",
        "codecommit:ListBranches",
        "codecommit:ListRepositories"
      ],
      "Resource": "*"
    },
    { "Effect": "Allow",
      "Action": [
        "aws-marketplace:Subscribe"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws-marketplace:ProductId": [
            "prod-6dw3qvchef7zy",
            "prod-m5ilt4siql27k",
            "prod-ozonys2hmmpeu"
          ]
        }
      }
    },
    { "Effect": "Allow",
      "Action": [
        "aws-marketplace:Unsubscribe",
        "aws-marketplace:ViewSubscriptions"
      ],
      "Resource": "*"
    },
    { "Effect": "Allow",
      "Action": "iam:*",
      "Resource": "*"
    }
  ]
}
```

Anthropic Claude 3 模型的提示技巧

此模式演示了使用 Claude 3 模型进行 text-to-query DSL 转换的以下提示技巧。

- Few-shot 提示：Few-shot 提示是一种强有力的技术，可以提高 Amazon Bedrock 上的 Claude 3 机型的性能。这种方法包括为模型提供少量示例，这些示例演示了说明任务所需的input/output behavior before asking it to perform a similar task. When you use Claude 3 models on Amazon Bedrock, few-shot prompting can be particularly effective for tasks that require specific formatting, reasoning patterns, or domain knowledge. To implement this technique, you typically structure your prompt with two main components: the example section and the actual query. The example section contains one or more input/output配对，而查询部分则显示了您想要响应的新输入。此方法有助于 Claude 3 了解上下文和预期的输出格式，并且通常会得到更准确和一致的响应。

  示例：

```
"query": {
  "bool": {
    "must": [
      {"match": {"post_type": "recipe"}},
      {"range": {"likes_count": {"gte": 100}}},
      {"exists": {"field": "media_urls"}}
    ]
  }
}
Question: Find all recipe posts that have at least 100 likes and include media URLs.
```

- 系统提示：除了少量镜头提示外，Amazon Bedrock 上的 Claude 3 机型还支持使用系统提示。系统提示是一种在向模型呈现特定用户输入之前向模型提供整体上下文、说明或指南的方法。它们对于设定基调、定义模特的角色或为整个对话设定约束特别有用。要在 Amazon Bedrock 上对 Claude 3 使用系统提示，请将其包含在 API `system` 请求的参数中。这与用户消息是分开的，适用于整个互动。详细的系统提示用于设置上下文并为模型提供指导。

  示例：

```
You are an expert query dsl generator. Your task is to take an input question and
 generate a query dsl to answer the question. Use the schemas and data below to
 generate the query.

Schemas: [schema details]
Data: [sample data]
Guidelines:
- Ensure the generated query adheres to DSL query syntax
```

```
- Do not create new mappings or other items that aren't included in the provided
 schemas.
```

- 结构化输出：您可以指示模型以特定格式（例如 JSON 或 XML 标签）提供输出。

  示例：

  ```
  Put the query in json tags
  ```

- 提示链接：笔记本使用一个 LLM 调用的输出作为另一个调用的输入，例如使用生成的合成数据来创建示例问题。
- 上下文提供：提示中提供了相关的上下文，包括架构和样本数据。

  示例：

  ```
  Schemas: [schema details]
  Data: [sample data]
  ```

- 特定任务的提示：针对特定任务精心制作了不同的提示，例如生成合成数据、创建示例问题以及将自然语言查询转换为查询 DSL。

  生成测试题的示例：

  ```
  Your task is to generate 5 example questions users can ask the health app based on
   provided schemas and data. Only include the questions generated in the response.
  ```

使用 Amazon Q Developer 作为编码助手来提高工作效率

创建者：Ram Kandaswamy (AWS)

摘要

此模式使用 tic-tac-toe游戏来演示如何将 Amazon Q Developer 应用于一系列开发任务。它以单页应用程序 (SPA) 的形式为 tic-tac-toe游戏生成代码，增强其用户界面，并创建用于部署应用程序的脚本。AWS

Amazon Q Developer 充当编码助手，可帮助加快软件开发工作流程，提高开发人员和非开发人员的工作效率。无论您的技术专业知识如何，它都可以帮助您创建架构和设计业务问题的解决方案，启动您的工作环境，帮助您实现新功能，并生成用于验证的测试用例。它使用自然语言指令和 AI 功能来确保代码的一致性、高质量并缓解编码挑战，无论您的编程技能如何。

Amazon Q Developer 的主要优势在于它能够将您从重复的编码任务中解放出来。当您使用`@workspace`注释时，Amazon Q Developer 会提取集成开发环境 (IDE) 中的所有代码文件、配置和项目结构并将其编入索引，并提供量身定制的响应以帮助您专注于创造性地解决问题。您可以将更多时间投入到设计创新的解决方案和增强用户体验上。如果您不是技术人员，则可以使用 Amazon Q Developer 来简化工作流程，更有效地与开发团队协作。Amazon Q Developer Explain 代码功能提供了详细的说明和摘要，因此您可以浏览复杂的代码库。

此外，Amazon Q Developer 还提供了一种与语言无关的方法，可以帮助初级和中级开发者扩展他们的技能。您可以专注于核心概念和业务逻辑，而不是特定于语言的语法。当你切换技术时，这会缩短学习曲线。

先决条件和限制

先决条件

- 安装了 Amazon Q Developer 插件的 IDE（例如， WebStorm 或 Visual Studio Code）。有关说明，请参阅 Amazon Q 开发者文档中的在 IDE 中安装 Amazon Q 开发者扩展或插件。
- Amazon Q Developer 的活跃 AWS 账户 设置。有关说明，请参阅 Amazon Q 开发者文档中的入门指南。
- npm 已安装。有关说明，请参阅 npm 文档。此模式已在 npm 版本 10.8 中进行了测试。
- AWS Command Line Interface (AWS CLI) 已安装。有关说明，请参阅AWS CLI 文档。

限制

- Amazon Q 开发者一次只能执行一项开发任务。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额页面，然后选择服务的链接。

工具

- 这种模式需要一个 IDE，比如 Visual Studio Code 或 WebStorm。有关支持的列表 IDEs，请参阅 Amazon Q 开发者文档。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行 shell 中的命令进行交互。

最佳实践

请参阅《 AWS 规范性指南》中的 Amazon Q 开发人员的最佳编码实践。此外：

- 当您向 Amazon Q 开发者提供提示时，请确保您的说明清晰明了。在提示中添加代码片段和注释，为提示提供更多上下文。`@workspace`
- 包括相关的库并导入它们，以避免系统发生冲突或错误猜测。
- 如果生成的代码不准确或不符合预期，请使用提供反馈并重新生成选项。尝试将提示分解成较小的指令。

操作说明

设置工作环境

Task	描述	所需技能
创建新 项目	要在工作环境中创建新项目，请运行以下命令并接受所有问题的默认设置： `npx create-next-app@latest`	应用程序开发人员、程序员、软件开发人员
测试基础应用程序。	运行以下命令并确认基本应用程序已成功加载到浏览器中：	应用程序开发人员、程序员、软件开发人员

Task	描述	所需技能
	`npm run dev`	
清理基本代码。	导航到`src/app`文件夹中的`page.tsx`文件并删除默认内容以获得空白页面。删除后，文件应如下所示： `export default function Home() { return (<div></div> ); }`	应用程序开发人员、程序员、软件开发人员

使用 Amazon Q 开发者设计 tic-tac-toe游戏项目

Task	描述	所需技能
获取步骤概述。	1. 在您的 IDE 中，打开您的项目，然后选择 Amazon Q 图标以打开聊天面板。 2. 在 Amazon Q 开发者聊天面板中，询问创建单页应用程序 (SPA) 的概述。例如： `I would like to create a single-page application involving Next.js React framework for tic-tac-toe game. What are the steps?`	应用程序开发人员、程序员、软件开发人员
为生成代码 tic-tac-toe。	在聊天面板中，使用`/dev`命令和任务描述启动开发任务。例如：	应用程序开发人员、程序员、软件开发人员

Task	描述	所需技能

	`/dev Create a React-based single-page application written in TypeScript for a tic-tac-toe game with the following specifications:`
	`1. Design an aesthetically pleasing interface with the game grid centered vertically and horizontally on the page.`
	`2. Include a heading and clear instructions on how to play the game.`
	`3. Implement color-coding for X and O marks to distinguish them easily.`
	
	Amazon Q 开发者会根据您的说明生成代码。	

检查并接受生成的代码。	直观地检查代码，然后选择 “接受代码” 以自动替换`page.tsx`文件。 如果您遇到问题，请选择 “提供反馈并重新生成”，然后描述您遇到的问题。	应用程序开发人员、程序员、软件开发人员

Task	描述	所需技能
修复 lint 错误。	示例 tic-tac-toe游戏包括一个网格。Amazon Q Developer 生成的代码可能使用默认类型any。您可以通过提示 Amazon Q 开发者来增加类型安全，如下所示： `/dev Ensure proper TypeScript typing for the onSquare Click event handler to resolve any 'any' type issues.`	应用程序开发人员、程序员、软件开发人员

Task	描述	所需技能

增加视觉吸引力。	您可以将原始要求分解为较小的片段。例如，您可以在开发任务中使用以下提示来改善游戏界面。此提示增强了层叠样式表 (CSS) 样式并导出应用程序以进行部署。
	
	`/dev Debug and fix any CSS issues to correctly display the game grid and overall layout.`
	
	`Simplify the code by removing game history functionality and related components.`
	
	`Implement static file export to an 'out' directory for easy deployment. The solution should be fully functional, visually appealing, and free of typing errors or layout issues.`
		应用程序开发人员、程序员、软件开发人员

Task	描述	所需技能

再次测试。	1. 现在您已经完成了开发生命周期，请测试代码以确认其是否按预期运行。要在本地运行应用程序，请使用以下命令：
	`npm run dev`
	2. 如果应用程序按预期运行，请使用build命令将整个应用程序导出到输出文件夹，为部署做准备：
	`npm run build`
		应用程序开发人员、程序员、软件开发人员

将应用程序部署到 AWS Cloud

Task	描述	所需技能

创建用于部署的文件夹和文件。	在工作环境中的项目中，创建一个部署文件夹，并在其中创建两个文件：pushtos3.sh 和cloudformation.yml ：
	`mkdir deployment && cd deployment`
	`touch pushtos3.sh && chmod +x pushtos3.sh`
	`touch cloudformation.yml`
		应用程序开发人员、程序员、软件开发人员

生成自动化代码。	1. 在 Amazon Q 开发者的聊天面板中，提供以下提示：	AWS 管理员、AWS DevOps、应用程序开发者

Task	描述	所需技能
	``` /dev Generate a Cloudformation template that creates two resources: tictactoe artifact bucket and CloudFron t. CloudFront should be configure d with this bucket as origin. Add cache policy appropriate for Amazon S3 and default root object as index.html. Ensure that origin access control is used and no public bucket is created. Output all the resources and their ARNs. ``` 2. 查看并接受生成的代码。现在，您之前创建`cloudformation.yml` 的文件应该填充为创建资源的 AWS CloudFormation 脚本 AWS Cloud。	

Task	描述	所需技能
生成脚本内容。	要创建部署脚本，请使用以下提示： `/dev Modify the pushtos3 shell script so that it can use AWS CLI commands to create a CloudFormation stack named tictactoe-stack if it does not exist already, and use cloudformation.yml as the source template. Wait for the stack to complete and sync the contents from the out folder to the S3 bucket. Perform invalidation of the CloudFront origin.`	应用程序开发人员、程序员、软件开发人员
将应用程序部署到 AWS Cloud。	1. 使用有效 AWS 凭据配置工作环境。 2. 运行 shell 脚本将功能齐全的 tic-tac-toe游戏部署到 AWS Cloud。	AWS 管理员、AWS DevOps、云架构师、应用程序开发人员

故障排除

事务	解决方案
该版本不会创建单页应用程序，也不会将其导出到输出文件夹。	查看`next.config.mjs` 文件内容。 如果代码具有以下默认配置：

事务	解决方案

	`const nextConfig = {};`
	
	按如下方式对其进行修改：
	
	`const nextConfig = {`
	`  output: 'export',`
	`  distDir: 'out',`
	`};`

相关资源

- 创建一个新的 React 项目（React 文档）
- Amazon Q 开发者概述（AWS 文档）
- Amazon Q 开发者最佳实践（AWS 规范性指导）
- 通过以下方式@@ 安装、配置和使用 Amazon Q Devel o YouTube p JetBrains IDEs er（视频）
- 为命令行安装 Amazon Q（AWS 文档）

使用 Processing 对 TB 级机器学习 SageMaker 数据集进行分布式特征工程

由 Chris Boomhower (AWS) 编写

摘要

许多 TB 级或更大的数据集通常由分层文件夹结构构成，数据集中的文件有时会共享相互依存关系。因此，机器学习 (ML) 工程师和数据科学家必须做出深思熟虑的决策，为模型训练和推理准备此类数据。此模式演示了如何将手动宏分片和微分片技术与 Amazon Processing 和虚拟 CPU (vCPU) 并行化相结合，为复杂的大数据 ML 数据集高效扩展功能工程 SageMaker 流程。

这种模式将宏分片定义为在多台计算机上拆分数据目录进行处理，将微分片定义为将每台计算机上的数据分割至多个处理线程中。该模式通过使用 Amazon 和 PhysioNet MIM IC-II SageMaker I 数据集中的时间序列波形记录样本来演示这些技术。通过采用这种模式技术，您可以最大限度地减少特征工程的处理时间和成本，同时最大限度地提高资源利用率和吞吐量效率。无论数据类型如何，这些优化都依赖于亚马逊弹性计算云 (Amazon EC2) 实例上的分布式 SageMaker 处理，CPUs 对类似的大型数据集进行 v。

先决条件和限制

先决条件

- 如果您想为自己的数据集实现此模式，则可以访问 SageMaker 笔记本实例或 SageMaker Studio。如果您是首次使用亚马逊 SageMaker ，请参阅 AWS 文档 SageMaker中的亚马逊入门。
- SageMaker Studio，如果你想用 PhysioNet MIMIC-III 样本数据实现这种模式。
- 该模式使用 SageMaker 处理，但不需要任何运行 SageMaker 处理作业的经验。

限制

- 这种模式非常适合包含相互依赖文件的机器学习数据集。手动宏分片和并行运行多个单实例处理 SageMaker 作业对这些相互依赖关系的益处最大。对于不存在此类相互依赖关系的数据集，Processing 中的`ShardedByS3Key` SageMaker 功能可能是宏分片的更好替代方案，因为它会将分片数据发送到由同一 Processing 作业管理的多个实例。但是，你可以在两种情况下实现这种模式的微分片策略，以最好地利用实例 v。CPUs

产品版本

- 亚马逊 SageMaker Python 软件开发工具包版本 2

架构

目标技术堆栈

- Amazon Simple Storage Service（Amazon S3）
- Amazon SageMaker

目标架构

宏分片和分布式实例 EC2

该架构中代表的 10 个并行进程反映了 MIMIC-III 数据集结构。（为了简化逻辑示意图，流程用省略号表示。）当您使用手动宏分片时，类似架构适用于任何数据集。就 MIMIC-III 而言，您可以毫不费力地单独处理每个患者组文件夹，从而充分利用数据集的原始结构。下图中的记录组块出现在左侧 (1)。鉴于数据分布性质，按患者群体进行分片是有意义的。

但是，按患者组手动分片意味着每个患者组文件夹都需要单独的处理作业，如您在图 (2) 的中间部分所见，而不是具有多个 EC2 实例的单个处理作业。由于 MIMIC-III 的数据包括二进制波形文件和匹配的基于文本的头文件，并且需要依赖 wfdb 库提取二进制数据，因此特定患者的所有记录都必须在同一个实例上可用。要确保每个二进制波形文件的关联头文件也存在，唯一的方法是实现手动分片，以在自己的处理作业中运行每个分片，并指定 `s3_data_distribution_type='FullyReplicated'` 何时定义处理作业输入。或者，如果所有数据都位于一个目录中，并且文件之间不存在依赖关系，则更合适的选择可能是启动具有多个 EC2 实例并`s3_data_distribution_type='ShardedByS3Key'`指定的单个处理作业。指定`ShardedByS3Key` 为 Amazon S3 数据分配类型 SageMaker 可自动管理跨实例的数据分片。

为每个文件夹启动 Processing 作业是经济高效的预处理数据的方式，因为同时运行多个实例可以节省时间。为进一步节省成本和时间，您可以在每个处理作业中使用微分片。

微分片和 parallel v CPUs

在每个 Processing 作业中，对分组的数据进行进一步划分，以最大限度地利用 SageMaker 完全托管 EC2 实例CPUs 上的所有可用 v。图 (2) 中间部分的方块描述了每个主要处理任务中的情况。患者记录文件夹的内容是扁平化的，并根据实例上可用 v 的数量CPUs 进行平均分割。分割文件夹内容后，大小

均匀的一组文件将分布在所有 v 中CPUs 进行处理。处理完成后，每个 vCPU 结果将合并到每个处理作业的单个数据文件中。

在随附的代码中，这些概念在 src/feature-engineering-pass1/preprocessing.py 文件的下一节中进行了介绍。

```
def chunks(lst, n):
    """
    Yield successive n-sized chunks from lst.

    :param lst: list of elements to be divided
    :param n: number of elements per chunk
    :type lst: list
    :type n: int
    :return: generator comprising evenly sized chunks
    :rtype: class 'generator'
    """
    for i in range(0, len(lst), n):
        yield lst[i:i + n]


# Generate list of data files on machine
data_dir = input_dir
d_subs = next(os.walk(os.path.join(data_dir, '.')))[1]
file_list = []
for ds in d_subs:
    file_list.extend(os.listdir(os.path.join(data_dir, ds, '.')))
dat_list = [os.path.join(re.split('_|\.', f)[0].replace('n', ''), f[:-4]) for f in
 file_list if f[-4:] == '.dat']

# Split list of files into sub-lists
cpu_count = multiprocessing.cpu_count()
splits = int(len(dat_list) / cpu_count)
if splits == 0: splits = 1
dat_chunks = list(chunks(dat_list, splits))

# Parallelize processing of sub-lists across CPUs
ws_df_list = Parallel(n_jobs=-1, verbose=0)(delayed(run_process)(dc) for dc in
 dat_chunks)

# Compile and pickle patient group dataframe
ws_df_group = pd.concat(ws_df_list)
ws_df_group = ws_df_group.reset_index().rename(columns={'index': 'signal'})
```

```
ws_df_group.to_json(os.path.join(output_dir, group_data_out))
```

chunks 函数首先被定义为使用给定列表，方法是将其分成大小均匀的长度 n ，然后将这些结果作为生成器返回。接下来，通过编译所有存在的二进制波形文件列表，在患者文件夹中对数据进行扁平化。完成此操作后，将获得 EC2 实例上CPUs 可用的 v 数。CPUs 通过调用，将二进制波形文件列表平均分成这些 vchunks，然后使用 joblib 的 Parallel 类在自己的 vCPU 上处理每个波形子列表。处理任务会自动将结果合并到一个数据帧列表中，然后处理任务会进一步处理， SageMaker 然后在任务完成后将其写入 Amazon S3。在此示例中，处理任务有 10 个文件写入 Amazon S3（每个任务一个）。

当所有初始处理任务完成后，辅助处理任务（如图 (3) 右侧的方块所示）将合并每个主处理任务生成的输出文件，并将合并后的输出写入 Amazon S3 (4)。

工具

工具

- Python — 用于此模式的示例代码是 Python（版本 3）。
- SageMaker Stud io — Amazon SageMaker Studio 是一个基于 Web 的机器学习集成开发环境 (IDE)，允许您构建、训练、调试、部署和监控您的机器学习模型。您可以在 Studio 中 SageMaker 使用 Jupyter 笔记本来运行 SageMaker 处理作业。
- SageMaker 处理 — Amazon P SageMaker rocessing 提供了一种运行数据处理工作负载的简化方法。在这种模式中，特征工程代码是使用 SageMaker 处理作业大规模实现的。

代码

随附的 .zip 文件提供了此模式的完整代码。以下部分介绍为此模式构建架构的步骤。附件中的示例代码介绍了每个步骤。

操作说明

设置你的 SageMaker 工作室环境

Task	描述	所需技能
访问亚马逊 SageMaker 工作室。	按照亚马逊 SageMaker 文档中提供的说明使用您的 AWS 账户登录 SageMaker Studio。	数据科学家、机器学习工程师

Task	描述	所需技能
安装 wget 实用程序。	如果您已使用新的 SageMaker Studio 配置或以前从未在 Studio 中 SageMaker 使用过这些实用程序，请安装 wget。 要进行安装，请在 SageMaker Studio 控制台中打开终端窗口并运行以下命令： `sudo yum install wget`	数据科学家、机器学习工程师

下载并解压缩示例代码。	在附件部分下载 `attachments.zip` 文件。在终端窗口，导航至下载文件并提取其内容的文件夹：
	
	`unzip attachment.zip`
	
	导航到提取 .zip 文件的文件夹，然后提取 `Scaled-Processing.zip` 文件的内容。
	
	`unzip Scaled-Processing.zip`
		数据科学家、机器学习工程师

从 physionet.org 下载示例数据集，并将其上传到 Amazon S3。	在包含 `get_data.ipynb` 文件的文件夹中运行 Jupyter 笔记本 `Scaled-Processing`。此笔记本从 physionet.org 下载示例 MIMIC-III 数据集，然后将其上传到亚马逊 S3 中的 Studio 会话存储桶 SageMaker。	数据科学家、机器学习工程师

配置第一项预处理脚本

Task	描述	所需技能

扁平化所有子目录的文件层次结构。	在 MIMIC-III 等大型数据集，文件通常分布在多个子目录中，即使在逻辑父组中也是如此。您的脚本应配置为扁平化所有子目录中所有组文件，如以下代码所示。
	`# Generate list of .dat files on machine`
	`data_dir = input_dir`
	`d_subs = next(os.walk(os.path.join(data_dir, '.')))[1]`
	`file_list = []`
	`for ds in d_subs:`
	`  file_list.extend(os.listdir(os.path.join(data_dir, ds, '.')))`
	`dat_list = [os.path.join(re.split('_|\.', f)[0].replace('n', ''), f[:-4]) for f in file_list if f[-4:] == '.dat']`
	Note
	此长篇故事中的示例代码片段来自该`src/feature-engineering-pass1/preprocessing.p`
		数据科学家、机器学习工程师

Task	描述	所需技能
	y 文件，该文件在附件中提供。	

根据 vCPU 数量将文件划分至子组。	应根据运行脚本的实例上CPUs存在的 v 数，将文件分成大小均匀的子组或块。在此步骤中，可实现类似于以下代码的代码。
	`# Split list of files into sub-lists`
	`cpu_count = multiprocessing.cpu_count()`
	`splits = int(len(dat_list) / cpu_count)`
	`if splits == 0: splits = 1`
	`dat_chunks = list(chunks(dat_list, splits))`
		数据科学家、机器学习工程师

在 v 上并行处理子组。CPUs	应将脚本逻辑配置为并行处理所有的子组。为此，请按如下方式使用 Joblib 库 的 `Parallel` 类和 `delayed` 方法。
	`# Parallelize processing of sub-lists across CPUs`
	`ws_df_list = Parallel(n_jobs=-1, verbose=0)(delayed(run_process)(dc) for dc in dat_chunks)`
		数据科学家、机器学习工程师

Task	描述	所需技能

将单个文件组的输出保存至 Amazon S3。	并行 vCPU 处理完成后，应合并每个 vCPU 的结果，并将其上传到文件组的 S3 存储桶路径。在此步骤，可以使用类似于以下代码的代码。
	`# Compile and pickle patient group dataframe`
	`ws_df_group = pd.concat(ws_df_list)`
	`ws_df_group = ws_df_group.reset_index().rename(columns={'index': 'signal'})`
	`ws_df_group.to_json(os.path.join(output_dir, group_data_out))`
		数据科学家、机器学习工程师

配置第二预处理脚本

Task	描述	所需技能
合并运行第一个脚本的所有 Processing 作业所生成的数据文件。	前面的脚本为 SageMaker 处理数据集中的一组文件的每个处理作业输出一个文件。 接下来，您需要将这些输出文件合并至一个对象，并将单个输出数据集写入 Amazon S3。文件中对此进行了演示，此 `src/feature-engineering-pass1p5/preprocessing.py` 文件载于附件，如下所示。	数据科学家、机器学习工程师

Task	描述	所需技能

	`def write_parquet(wavs_df, path):`
	`    """`
	`    Write waveform summary dataframe to S3 in parquet format.`
	`    :param wavs_df: waveform summary dataframe`
	`    :param path: S3 directory prefix`
	`    :type wavs_df: pandas dataframe`
	`    :type path: str`
	`    :return: None`
	`    """`
	`    extra_args = {"ServerSideEncryption": "aws:kms"}`
	`    wr.s3.to_parquet(`
	`        df=wavs_df,`
	`        path=path,`
	`        compression='snappy',`
	`        s3_additional_kwargs=extra_args)`
	
	`def combine_data():`
	`    """`
	`    Get combined data and write to parquet.`
	`    :return: waveform summary dataframe`
	`    :rtype: pandas dataframe`
	`    """`	

Task	描述	所需技能

	<pre> wavs_df = get_data(
	)
	    wavs_df = normalize
	_signal_names(wavs
	_df)
	    write_parquet(wavs
	_df, "s3://{}/{}/
	{}".format(buck
	et_xform, dataset_p
	refix, pass1p5ou
	t_data))
	
	    return wavs_df
	
	wavs_df = combine_d
	ata()</pre>	

运行处理作业

Task	描述	所需技能

运行第一项处理作业。	若要执行宏分片，请为每个文件组运行单独的处理作业。Microsharding 是在每个 Processing 作业中执行，因为每个作业都会运行您的第一个脚本。以下代码演示了如何为以下片段（包含在 `notebooks/FeatExtract_Pass1.ipynb` ）中的每个文件组目录启动处理作业。
	<pre>pat_groups = list(rang
	e(30,40))
	ts = str(int(time.time(
	)))</pre>
		数据科学家、机器学习工程师

Task	描述	所需技能

	for group in pat_groups:
	    sklearn_processor = SKLearnProcessor(framework_version='0.20.0',
	role=role,
	instance_type='ml.m5.4xlarge',
	instance_count=1,
	volume_size_in_gb=5)
	    sklearn_processor.run(
	code='../src/feature-engineering-pass1/preprocessing.py',
	job_name='-'.join(['scaled-processing-p1', str(group), ts]),
	arguments=[
	"input_path", "/opt/ml/processing/input",
	"output_path", "/opt/ml/processing/output",
	"group_data_out", "ws_df_group.json"
	],
	inputs=
	[	

Task	描述	所需技能

	<pre> ProcessingInput(
	
	 source=f's3://{sess.default_bucket()}/data_inputs/{group}',
	
	 destination='/opt/ml/processing/input',
	
	 s3_data_distribution_type='FullyReplicated'
	            )
	        ],
	        outputs=
	        [
	            ProcessingOutput(
	
	 source='/opt/ml/processing/output',
	
	 destination=f's3://{sess.default_bucket()}/data_outputs/{group}'
	            )
	        ],
	        wait=False
	    )</pre>	

Task	描述	所需技能

运行第二个处理作业。	要合并第一组处理作业生成的输出并执行任何其他计算以进行预处理，请使用单 SageMaker 个 Processing 作业运行第二个脚本。以下代码演示了这一点（包含在notebooks/FeatExtract_Pass1p5.ipynb ）。
	`ts = str(int(time.time()))`
	`bucket = sess.default_bucket()`
	
	`sklearn_processor = SKLearnProcessor(framework_version='0.20.0',`
	
	`role=role,`
	
	`instance_type='ml.t3.2xlarge',`
	
	`instance_count=1,`
	
	`volume_size_in_gb=5)`
	`sklearn_processor.run(`
	`code='../src/feature-engineering-pass1p5/preprocessing.py',`
	`job_name='-'.join(['scaled-processing', 'p1p5', ts]),`
	`arguments=['bucket', bucket,`
		数据科学家、机器学习工程师

Task	描述	所需技能

	<pre>'pass1out
	_prefix', 'data_out
	puts',
	'pass1out
	_data', 'ws_df_gr
	oup.json',
	'pass1p5o
	ut_data', 'waveform
	_summary.parquet',
	'statsdat
	a_name', 'signal_s
	tats.csv'],
	wait=True
	)</pre>	

相关资源

- 使用快速入门登录 Amazon SageMaker Studio（SageMaker 文档）
- 流程数据（SageMaker 文档）
- 使用 scikit-learn 进行数据处理（文档）SageMaker
- joblib.Parallel 文档
- Moody, B., Moody, G., Villarroel, M., Clifford, G. D., & Silva, I. (2020). MIMIC-III 波形数据库（版本 1.0）。 PhysioNet。
- Johnson, A. E. W., Pollard, T. J., Shen, L., Lehman, L. H., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Celi, L. A., & Mark, R. G. (2016). MIMIC-III，可免费访问的重症监护数据库。Scientific Data, 3, 160035.
- MIMIC-III Waveform Database 许可证

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 Flask 和 AWS Elastic Beanstalk 查看人工智能/机器学习 (AI/ML) 模型结果

由 Chris Caudill (AWS) 和 Durga Sury (AWS) 编写

摘要

可视化人工智能和机器学习 (AI/ML) 服务输出通常需要复杂的 API 调用，这些调用必须由开发人员和工程师自定义。如果您的分析师想快速探索新数据集，这可能是一个缺点。

您可以使用基于 Web 的用户界面 (UI) 增强服务的可访问性并提供更具交互性的数据分析形式，该界面使用户能够上传自己的数据并在控制面板中可视化模型结果。

这种模式使用 Flask 和 Plotly，将 Amazon Comprehend 与自定义 Web 应用程序集成，并可视化用户提供的数据中的观点和实体。该模式还提供了通过 AWS Elastic Beanstalk 部署应用程序的步骤。您可以使用 A mazon Web Services (AWS) AI 服务或托管在终端节点（例如，亚马逊 SageMaker 终端节点）上的自定义训练模型，来调整应用程序。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- AWS 命令行界面（AWS CLI）已在本地计算机上安装和配置。有关这方面的更多信息，请参阅 AWS CLI 文档中的配置基础知识。您也可以使用 AWS Cloud9 集成式开发环境（IDE）；有关这方面的更多信息，请参阅 AWS Cloud9 文档中的 AWS Cloud9 的 Python 教程和预览 AWS Cloud9 IDE 中的运行应用程序。

  注意： AWS Cloud9 不再向新客户开放。的现有客户 AWS Cloud9 可以继续照常使用该服务。了解更多
- 了解 Flask 的网络应用程序框架。有关 Flask 的更多信息，请参阅 Flask 文档中的 Quickstart。
- Python 版本 3.6 或更高版本，已安装并已配置。您可按照 AWS Elastic Beanstalk 文档中设置 Python 开发环境中的说明安装 Python。
- Elastic Beanstalk 命令行界面 (EB CLI)，已安装并配置。有关这方面的更多信息，请参阅 AWS Elastic Beanstalk 文档中的安装 EB CLI 和配置 EB CLI。

限制

- 此模式的 Flask 应用程序，旨在处理使用单个文本列且限制在 200 行以内的 .csv 文件。可调整应用程序代码以处理其他文件类型和数据量。
- 该应用程序不考虑数据留存，将继续汇总上传的用户文件，直到手动删除这些文件。您可以将应用程序与 Amazon Simple Storage Service (Amazon S3) 集成以实现永久对象存储，也可以使用 Amazon DynamoDB 等数据库进行无服务器键值存储。
- 该应用程序仅适用英文文档。但是，您可使用 Amazon Comprehend 来检测文档的主要语言。有关每个操作支持的语言的更多信息，请参见 Amazon Comprehend 文档中的 API 参考。
- 其他信息部分提供了包含常见错误及其解决方案的故障排除列表。

架构

Flask 应用程序架构

Flask 是轻量级框架，用于在 Python 中开发 Web 应用程序。它旨在将 Python 的强大数据处理功能与丰富的 Web 用户界面相结合。该模式的 Flask 应用程序向您展示了如何构建 Web 应用程序，该应用程序使用户能够上传数据，将数据发送至 Amazon Comprehend 进行推理，然后将结果可视化。 应用程序具有以下结构：

- `static`— 包含所有支持 Web UI 的静态文件（例如 JavaScript，CSS 和图像）
- `templates` – 包含应用程序的所有 HTML 页面
- `userData` –存储上传的用户数据
- `application.py` – Flask 应用程序文件
- `comprehend_helper.py` – 用于对 Amazon Comprehend 进行 API 调用的函数
- `config.py` – 应用程序配置文件
- `requirements.txt` – 应用程序所需 Python 依赖项

`application.py` 脚本包含 Web 应用程序的核心功能，该功能由四个 Flask 路由组成。下图介绍了这些 Flask 路由。

- `/` 是应用程序根目录，可将用户定向至 `upload.html` 页面（存储在`templates`目录中）。
- `/saveFile` 是在用户上传文件后调用的路由。此路由通过 HTML 表单接收 `POST` 请求，其中包含用户上传的文件。文件保存在 `userData` 目录中，路由会将用户重定向至 `/dashboard` 路由。

- `/dashboard` 将用户发送到 `dashboard.html` 页面。在此页面的 HTML 中，它运行中的 JavaScript 代码`static/js/core.js`，从`/data`路径中读取数据，然后为该页面构建可视化效果。
- `/data` 是 JSON API，用于显示要在控制面板中可视化的数据。此路由读取用户提供的数据，并使用 `comprehend_helper.py` 中的函数将用户数据发送到 Amazon Comprehend，用于观点分析和命名实体识别 (NER)。Amazon Comprehend 的响应已格式化，并作为 JSON 对象返回。

部署架构

有关在 AWS 云上使用 Elastic Beanstalk 部署的应用程序的设计注意事项的更多信息，请参阅 AWS Elastic Beanstalk 文档。

设计注意事项

技术堆栈

- Amazon Comprehend
- Elastic Beanstalk
- Flask

自动化和扩缩

Elastic Beanstalk 部署会自动设置负载均衡器以及自动扩缩组。有关更多配置选项，请参阅 AWS Elastic Beanstalk 文档中的配置 Elastic Beanstalk 环境。

工具

- AWS Command Line Interface (AWS CLI) 是一个统一的工具，它为与 AWS 的所有部分进行交互提供了一致的接口。
- Amazon C omprehend 使用自然语言处理 (NLP) 来提取有关文档内容的见解，而无需特殊的预处理。
- AWS E lastic Beanstalk 可帮助您在 AWS 云中快速部署和管理应用程序，而无需了解运行这些应用程序的基础设施。
- Elastic Beanstalk CLI (EB CLI) 是 AWS Elastic Beanstalk 的命令行界面，它提供交互式命令，可简化从本地存储库创建、更新和监控环境的过程。

- Flask 框架使用 Python 执行数据处理和 API 调用，并通过 Plotly 提供交互式网页可视化。

代码

这种模式的代码可在使用 Flask 和 AWS Elastic Beanstalk 存储库的 Visuali GitHub ze AI/ML 模型结果中找到。

操作说明

设置 Flask 应用程序

Task	描述	所需技能
克隆 GitHub 存储库。	运行以下命令，使用 Flask 和 AWS Elastic Beanstalk 存储库从 Visuali GitHub ze AI/ML 模型结果中提取应用程序代码： `git clone git@github.com:aws-samples/aws-comprehend-elasticbeanstalk-for-flask.git` **Note** 确保使用配置 SSH 密钥 GitHub。	开发人员
安装 Python 模块。	克隆存储库后，将创建新的本地 `aws-comprehend-elasticbeanstalk-for-flask` 目录。在该目录中，`requirements.txt` 文件包含运行该应用程序的 Python 模块和版本。使用以下命令安装模块：	Python 开发人员

Task	描述	所需技能

	`cd aws-comprehend-elasticbeanstalk-for-flask`
	
	`pip install -r requirements.txt`	

Task	描述	所需技能

在本地测试应用程序。	运行以下命令以启动 Flask 服务器：
	
	`python application.py`
	
	这将返回有关正在运行的服务器的信息。您应该能够通过打开浏览器和访问 http://localhost:5000 来访问该应用程序
	
	Note
	如果您在 AWS Cloud9 IDE 中运行应用程序，则需要将`application.py` 文件中的`application.run()` 命令替换为以下行：
	
	`application.run(host=os.getenv('IP', '0.0.0.0'),port=int(os.getenv('PORT', 8080)))`
	
	在部署之前，您必须恢复此更改。
		Python 开发人员

将应用程序部署到 Elastic Beanstalk

Task	描述	所需技能
启动 Elastic Beanstalk 应用程序。	若要将您的项目作为 Elastic Beanstalk 应用程序启动，请从应用程序的根目录运行以下命令： `eb init -p python-3.6 comprehend_flask --region us-east-1` **⚠ Important** - `comprehend_flask` 是 Elastic Beanstalk 应用程序的名称，可以根据您的要求进行更改。 - 您可以用自己选择的区域替换 AWS 区域。如果您未指定区域，则使用 AWS CLI 默认区域。 - 该应用程序通过 Python 版本 3.6 构建。如果您使用其他版本 Python，则可能会遇到错误。 运行 `eb init -i` 命令，以获取更多部署配置选项。	架构师、开发人员
部署 Elastic Beanstalk 环境。	从应用程序的根目录运行以下命令：	架构师、开发人员

Task	描述	所需技能
	`eb create comprehend-flask-env` **Note** `comprehend-flask-env` 是 Elastic Beanstalk 环境的名称，可以根据您的要求进行更改。名称仅可包含字母、数字和短划线。	

Task	描述	所需技能
授权部署使用 Amazon Comprehend。	尽管您的应用程序可能已成功部署，但您还应为部署提供访问 Amazon Comprehend 的权限。`ComprehendFullAccess` 是 AWS 托管式策略，它为已部署的应用程序提供对 Amazon Comprehend 进行 API 调用的权限。 通过运行以下命令将 `ComprehendFullAccess` 策略附加到`aws-elasticbeanstalk-ec2-role`（此角色是为部署的亚马逊弹性计算云 (Amazon EC2) 实例自动创建的）： `aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/ComprehendFullAccess --role-name aws-elasticbeanstalk-ec2-role` **Important** `aws-elasticbeanstalk-ec2-role` 是在您的应用程序部署时创建的。您必须先完成部署过程，然后才能附加 AWS Identity and	开发人员、安全架构师

Task	描述	所需技能
	Access Management (IAM) policy。	
访问您所部署的应用程序。	成功部署应用程序后，您可通过运行 `eb open` 命令访问。 您也可以运行 `eb status` 命令来接收有关您的部署的详细信息。部署 URL 列在 `CNAME` 下面。	架构师、开发人员

（可选）根据您的机器学习模型自定义应用程序

Task	描述	所需技能
授权 Elastic Beanstalk 访问新的模型。	确保 Elastic Beanstalk 拥有您的新模型端点所需访问权限。例如，如果您使用 Amazon SageMaker 终端节点，则您的部署需要拥有调用该终端节点的权限。 有关这方面的更多信息，请参阅 Amazon SageMaker 文档InvokeEndpoint中的。	开发人员、安全架构师
将用户数据发送至新模型。	若要更改此应用程序中的底层 ML 模型，必须更改以下文件： • `comprehend_helper.py` – 这是与 Amazon Comprehend 连接、处理响应并将最终结果返回给应用程序的 Python 脚本。在此脚本中，您可以将数据路由	数据科学家

Task	描述	所需技能
	至 Amazon Web Services Cloud 上的其他 AI 服务，也可以将数据发送到自定义模型端点。我们建议您同时格式化此脚本结果，以实现逻辑分离和此模式的可重复使用。 • `application.py` — 如果您更改 `comprehend_helper.py` 脚本或函数的名称，则需要更新应用程序 `application.py` 脚本以反映这些更改。	
更新控制面板可视化效果。	通常，合并新的机器学习模型，意味着必须更新可视化效果以反映新的结果。这些更改在以下文件中进行： • `templates/dashboard.html` — 预构建的应用程序只考虑两个基本的可视化。可在此文件中调整页面的整个布局。 • `static/js/core.js` — 此脚本捕获 Flask 服务器 `/data` 路由的格式化输出，并使用 Plotly 创建可视化效果。您可添加或更新页面的图表。	网页开发人员

（可选）部署更新的应用程序

Task	描述	所需技能
更新应用程序需求文件。	在向 Elastic Beanstalk 发送更改之前，请在应用程序的根目录中运行以下命令，以更新 `requirements.txt` 文件，反映任何新的 Python 模块： `pip freeze > requirements.txt`	Python 开发人员
重新部署 Elastic Beanstalk 环境。	为确保您的应用程序更改反映在 Elastic Beanstalk 部署中，请导航到应用程序的根目录并运行以下命令： `eb deploy` 这会将应用程序代码的最新版本发送至您现有的 Elastic Beanstalk 部署中。	系统管理员、架构师

相关资源

- 使用 Amazon API Gateway 和 AWS Lambda 调用亚马逊 SageMaker 模型终端节点
- 将 Flask 应用程序部署到 Elastic Beanstalk
- EB CLI 命令参考
- 设置 Python 开发环境

其他信息

故障排除列表

以下是六种常见错误及其解决方法。

错误 1

```
Unable to assume role "arn:aws:iam::xxxxxxxxxx:role/aws-elasticbeanstalk-ec2-role".
 Verify that the role exists and is configured correctly.
```

解决方案：如果您在运行 `eb create` 时出现此错误，请在 Elastic Beanstalk 控制台创建一个示例应用程序，以创建默认实例配置文件。有关此内容的更多信息，请参阅 AWS Elastic Beanstalk 文档中的创建 Elastic Beanstalk 环境。

错误 2

```
Your WSGIPath refers to a file that does not exist.
```

解决方案：部署日志中会出现此错误，因为 Elastic Beanstalk 希望将 Flask 代码命名为 `application.py`。如果您选择了其他名称，请运行`eb config`并编辑， WSGIPath 如以下代码示例所示：

```
aws:elasticbeanstalk:container:python:
     NumProcesses: '1'
     NumThreads: '15'
     StaticFiles: /static/=static/
     WSGIPath: application.py
```

务必将 `application.py` 更换为您的文件名。

您还可利用 Gunicorn 与 Procfile。有关此方法的更多信息，请参阅 AWS Elastic Beanstalk 文档中的通过 Procfile 配置 WSGI 服务器。

错误 3

```
Target WSGI script '/opt/python/current/app/application.py' does not contain WSGI
 application 'application'.
```

解决方案：Elastic Beanstalk 希望将代表您的 Flask 应用程序变量命名为 `application`。确保 `application.py` 文件使用 `application` 作为变量名：

```
application = Flask(__name__)
```

错误 4

```
The EB CLI cannot find your SSH key file for keyname
```

解决方案：使用 EB CLI 指定要使用的密钥对，或者为部署的 EC2 实例创建密钥对。要解决错误，请运行 `eb init -i`，其中一个选项将询问：

```
Do you want to set up SSH for your instances?
```

以 Y响应，创建密钥对或指定现有密钥对。

错误 5

我已更新了我的代码并进行了重新部署，但是我的部署没有反映我的更改。

解决方案：如果您在部署时使用 Git 存储库，请确保在重新部署之前添加并提交更改。

错误 6

您正在通过 AWS Cloud9 IDE 预览 Flask 应用程序，但遇到了错误。

解决方案：有关这方面的更多信息，请参阅 AWS Cloud9 文档中的在 AWS Cloud9 IDE 中预览正在运行的应用程序。

使用 Amazon Comprehend 进行自然语言处理

通过选择使用 Amazon Comprehend，您现在可以通过运行实时分析或异步批处理作业来检测单个文本文档中的自定义实体。Amazon Comprehend 还允许训练自定义实体识别和文本分类模型，这些模型可以通过创建端点来实时使用。

这种模式使用异步批处理作业，检测包含多个文档的输入文件中的观点和实体。此模式提供的示例应用程序旨在让用户上传包含单列且每行一个文本文档的 .csv 文件。使用 Flask 和 AWS Elastic Beanstalk 存储库 GitHub 可视化 AI/ML 模型结果`comprehend_helper.py`的文件读取输入文件并将输入发送到 Amazon Comprehend 进行处理。

BatchDetectEntities

Amazon Comprehend 检查一批文档的文本以查找命名实体，并返回检测到的实体、位置、实体类型，以及表明 Amazon Comprehend 可信度等级的分数。一次 API 调用最多可发送 25 个文档，每个文档的大小小于 5,000 字节。您可根据用例筛选结果以仅显示某些实体。例如，您可以跳过 `‘quantity’` 实体类型，为检测到的实体设置阈值分数 (例如 0.75)。我们建议，您在选

择阈值之前，先浏览特定用例的结果。有关这方面的更多信息，请参阅亚马逊 Comprehend 文档BatchDetectEntities中的。

BatchDetectSentiment

Amazon Comprehend 会检查一批传入的文档，并返回每份文档的普遍观点（POSITIVE、NEUTRAL、MIXED 或 NEGATIVE）。一次 API 调用最多可发送 25 个文档，每个文档的大小小于 5,000 字节。分析观点非常简单，您可选择分数最高的观点以显示在最终结果中。有关这方面的更多信息，请参阅亚马逊 Comprehend 文档BatchDetectSentiment中的。

Flask 配置处理

Flask 服务器使用一系列配置变量控制服务器的运行方式。这些变量可以包含调试输出、会话令牌或者其他应用程序设置。您还可定义在应用程序运行时访问的自定义变量。设置配置变量的方法有很多种。

在这种模式中，配置是在 `config.py` 中定义，且在 `application.py` 中继承的。

- > **Note**
  >
  > `config.py` 包含在应用程序启动时设置的配置变量。在此应用程序中，定义了 DEBUG 变量，告诉应用程序以调试模式运行服务器。：在生产环境中运行应用程序时，不应使用调试模式。 UPLOAD_FOLDER是一个自定义变量，定义为稍后在应用程序中引用，并告知应用程序应将上传的用户数据存储在哪里。
- `application.py` 启动 Flask 应用程序并继承 `config.py` 中定义的配置设置。这通过以下代码执行：

```
application = Flask(__name__)
application.config.from_pyfile('config.py')
```

更多模式

- 使用 Amazon Bedrock 实现 AWS 基础设施运营自动化
- 部署 ChatOps 解决方案来管理 SAST 扫描结果，方法是在聊天应用程序中使用 Amazon Q Developer 自定义操作和 AWS CloudFormation
- 使用 AWS 大型机现代化和 Amazon Q 生成数据见解 QuickSight
- 使用和 AWS Mainframe Modernization Amazon Q 生成 Db2 z/OS 数据见解 QuickSight
- 为 SageMaker 笔记本实例提供对另一个 AWS 账户中 CodeCommit 存储库的临时访问权限
- SageMaker 使用 AWS 开发人员工具将 ML 构建、训练和部署工作负载迁移到 Amazon
- 使用 Amazon Q 开发人员对 CardDemo 大型机应用程序进行现代化改造
- 使用 Amazon Redshift ML 执行高级分析
- 使用自动化工作流程简化 Amazon Lex 机器人的开发和部署
- 使用 Amazon Bedroc AWS Step Functions k 对中的状态进行故障排除

Analytics

主题

- 在 Microsoft SQL Server Analysis Services 中分析 Amazon Redshift 数据
- 使用亚马逊 Athena 和亚马逊分析和可视化嵌套的 JSON 数据 QuickSight
- 自动将数据从 Amazon S3 AWS Data Exchange 中提取数据
- 使用 AWS CloudFormation 模板在 AWS Glue 中自动执行加密
- 使用 AWS DataOps 开发套件构建数据管道，以提取、转换和分析 Google Analytics（分析）数据
- 通过 Amazon Kinesis Video Streams 和 AWS Fargate 构建视频处理管道
- 使用 AWS Glue 构建 ETL 服务管道以增量方式将数据从 Amazon S3 加载到 Amazon Redshift
- 使用 Amazon 构建企业数据网格 DataZone AWS CDK，以及 AWS CloudFormation
- 通过 Amazon Web Services 计算风险价值 (VaR)
- 使用 Amazon Athena 来配置对共享 AWS Glue Data Catalog 的跨账户存取
- 将 Teradata 标准化时态功能转换为 Amazon Redshift SQL
- 将 Teradata RESET WHEN 功能转换为 Amazon Redshift SQL
- 使用基础设施即代码，在 Amazon Web Services Cloud 上部署和管理无服务器数据湖
- 启动时强制标记 Amazon EMR 集群
- 确保在启动时启用 Amazon EMR 日志记录到 Amazon S3
- 使用 AWS Glue 作业和 Python 生成测试数据
- 使用 Amazon IoT Greengrass 将物联网数据直接摄取至 Amazon S3，经济实惠
- 使用 Lambda 函数在瞬态 EMR 集群中启动 Spark 作业
- 使用 AWS Glue 将 Apache Cassandra 工作负载迁移到亚马逊密钥空间
- 使用迁移器将 Hadoop 数据迁移到 Amazon S3 WANdisco LiveData
- 将 Oracle 商业智能 12c 从本地服务器迁移到 Amazon Web Services Cloud
- 使用将本地 Apache Kafka 集群迁移到亚马逊 MSK MirrorMaker
- 将 ELK 堆栈迁移至 Elastic Cloud on AWS
- 使用 Starburs AWS Cloud t 将数据迁移到
- 优化 AWS 输入文件大小的 ETL 摄取
- 使用 AWS Step Functions 编排 ETL 管道，包含验证、转换和分区
- 使用 Amazon Redshift ML 执行高级分析

- 使用 Amazon Athena 使用 SQL 查询亚马逊 DynamoDB 表
- 使用 Athena 访问、查询和联接 Amazon DynamoDB 表
- 设置最小可行数据空间，以便在组织之间共享数据
- 使用标量 Python UDF 为 Amazon Redshift 查询结果设置特定于语言的排序
- 将 Lambda 函数，以接收来自不同 AWS 区域中的 S3 存储桶的事件通知
- 三种用于将数据转换为 Apache Parquet 的 AWS Glue ETL 作业类型
- 使用亚马逊 Athena 和亚马逊可视化 Amazon Redshift 审计日志 QuickSight
- 使用 Amazon 可视化所有 AWS 账户的 IAM 凭证报告 QuickSight
- 更多模式

在 Microsoft SQL Server Analysis Services 中分析 Amazon Redshift 数据

由 Sunil Vora (AWS) 创建

摘要

此模式描述了如何使用 Intellisoft OLE 数据库提供程序或 CData ADO.NET 提供程序进行数据库访问来连接和分析微软 SQL Server 分析服务中的 Amazon Redshift 数据。

Amazon Redshift 是云中一种完全托管的 PB 级数据仓库服务。SQL Server Analysis Services 是一种联机分析处理（OLAP）系统工具，可用于分析来自数据集市和数据仓库（如 Amazon Redshift）的数据。可以使用SQL Server Analysis Services从数据创建 OLAP 多维数据集，以便进行快速、高级的数据分析。

先决条件和限制

假设

- 此模式描述了如何在亚马逊弹性计算云（亚马逊）实例上为亚马逊 Redshift 设置 SQL Server 分析服务和 Intellisof CData t OLE 数据库提供商或 ADO.NET 提供商。 EC2或者，您可以将两者安装在公司数据中心的主机上。

先决条件

- 一个有效的 Amazon Web Services account
- 具有凭证的 Amazon Redshift 集群

架构

源技术堆栈

- Amazon Redshift 集群

目标技术堆栈

- Microsoft SQL Server Analysis Services

源架构和目标架构

工具

- Microsoft Visual Studio 2019 (Community Edition)
- 适用于亚马逊 Redshift 的 Intellisoft OLE 数据库提供商（试用版）或亚马逊 Redshift 的 ADO.NET CData 提供商（试用版）

操作说明

分析表

Task	描述	所需技能
分析要导入的表和数据。	确定要导入的 Amazon Redshift 表及其大小。	数据库管理员

设置 EC2 实例并安装工具

Task	描述	所需技能
设置 EC2 实例。	在您的 AWS 账户中，在私有或公有子网中创建 EC2 实例。	系统管理员
安装用于数据库访问的工具。	下载并安装适用于亚马逊 Redshift 的 Intellisoft OLE 数据库提供程序（或亚马逊 Redshift 的 ADO.NET CData 提供商）。	系统管理员
安装 Visual Studio。	下载并安装 Visual Studio 2019（社区版）。	系统管理员

Task	描述	所需技能
安装扩展。	在 Visual Studio 中安装 Microsoft Analysis Services 项目扩展。	系统管理员
创建项目。	在 Visual Studio 中创建新的表格模型项目以存储 Amazon Redshift 数据。在 Visual Studio 中，创建项目时选择 Analysis Services 表格项目选项。	数据库管理员

创建数据来源和导入表

Task	描述	所需技能
创建 Amazon Redshift 数据来源。	使用适用于亚马逊 Redshift 的 Intellisoft OLE 数据库提供商（或亚马逊 Redshift 的 ADO.NET 提供商 CData ）和您的亚马逊 Redshift 凭证创建亚马逊 Redshift 数据源。	Amazon Redshift、数据库管理员
导入表。	从 Amazon Redshift 选择表和视图并将其导入到 SQL Server Analysis Services 项目中。	Amazon Redshift、数据库管理员

迁移后清理

Task	描述	所需技能
删除实 EC2 例。	删除您之前启动的 EC2 实例。	系统管理员

相关资源

- Amazon Redshift（AWS 文档）
- 安装 SQL Server Analysis Services（Microsoft 文档）
- 表格模型设计器（Microsoft 文档）
- 用于高级分析的 OLAP 多维数据集概述（Microsoft 文档）
- Microsoft Visual Studio 2019 (Community Edition)
- 适用于 Amazon Redshift 的 Intellisoft OLE DB Provider（试用版）
- CData 亚马逊 Redshift 的 ADO.NET 提供商（试用版）

使用亚马逊 Athena 和亚马逊分析和可视化嵌套的 JSON 数据 QuickSight

由 Anoop Singh (AWS) 创建

摘要

此模式说明了如何使用 Amazon Athena 将嵌套的、JSON 格式的数据结构转换为表格视图，然后在 Amazon 中可视化数据。 QuickSight

您可以将 JSON 格式的数据用于来自操作系统的 API 驱动的数据源，以创建数据产品。这些数据还可以帮助您更好地了解客户及其与产品的互动，因此您可以量身定制用户体验并预测结果。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 表示嵌套数据结构的 JSON 文件（此模式提供了示例文件）

限制:

- JSON 功能与 Athena 中现有的面向 SQL 的函数很好地集成。但是，它们不兼容 ANSI SQL，并且 JSON 文件应将每条记录放在单独的行中。您可能需要使用 Athena 中的`ignore.malformed.json`属性来指示是否应将格式错误的 JSON 记录转换为空字符或生成错误。 有关更多信息，请参阅 Athena 文档中的读取 JSON 数据的最佳实践。
- 此模式仅考虑简单且少量 JSON 格式的数据。如果您想大规模使用这些概念，可以考虑应用数据分区并将数据整合到更大的文件中。

架构

下图显示了此模式的架构和工作流程。嵌套的数据结构以 JSON 格式存储在亚马逊简单存储服务 (Amazon S3) 中。在 Athena 中，JSON 数据映射到 Athena 数据结构。然后，您可以创建一个视图来分析数据，并在中可视化数据结构 QuickSight。

工具

Amazon Web Services

- Amazon Simple Storage Service(Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。此模式使用 Amazon S3 来存储 JSON 文件。
- Amazon Athena 是一种交互式查询服务，可帮助您使用标准 SQL 直接在 Amazon S3 中分析数据。此模式使用 Athena 来查询和转换 JSON 数据。只需在中执行一些操作 AWS Management Console，您就可以将 Athena 指向您在 Amazon S3 中的数据，然后使用标准 SQL 来运行一次性查询。Athena 是无服务器的，因此无需设置或管理基础架构，您只需为运行的查询付费。Athena 会自动扩展并行运行查询，因此即使使用大型数据集和复杂查询，也能快速获得结果。
- Amazon QuickSight 是一项云规模的商业智能 (BI) 服务，可帮助您在单个控制面板上可视化、分析和报告数据。 QuickSight 允许您轻松创建和发布包含机器学习 (ML) 见解的交互式仪表板。您可以从任何设备访问这些仪表板，并将其嵌入到您的应用程序、门户和网站中。

示例代码

以下 JSON 文件提供了可以在此模式中使用的嵌套数据结构。


```
{
  "symbol": "AAPL",
  "financials": [
    {
      "reportDate": "2017-03-31",
      "grossProfit": 20591000000,
      "costOfRevenue": 32305000000,
      "operatingRevenue": 52896000000,
      "totalRevenue": 52896000000,
      "operatingIncome": 14097000000,
      "netIncome": 11029000000,
      "researchAndDevelopment": 2776000000,
      "operatingExpense": 6494000000,
      "currentAssets": 101990000000,
      "totalAssets": 334532000000,
      "totalLiabilities": 200450000000,
      "currentCash": 15157000000,
      "currentDebt": 13991000000,
      "totalCash": 67101000000,
      "totalDebt": 98522000000,
      "shareholderEquity": 134082000000,
```

```
      "cashChange": -1214000000,
      "cashFlow": 12523000000,
      "operatingGainsLosses": null
    }
  ]
}
```

操作说明

设置 S3 存储桶

Task	描述	所需技能
创建 S3 存储桶。	要创建用于存储 JSON 文件的存储桶，请登录并打开 Amazon S3 控制台，然后选择创建存储桶。 AWS Management Console有关更多信息，请参阅 Amazon S3 文档中的创建存储桶。	系统管理员
添加嵌套的 JSON 数据。	将您的 JSON 文件上传到 S3 存储桶。有关 JSON 文件的示例，请参阅上一节。有关说明，请参阅 Amazon S3 文档中的上传对象。	系统管理员

在 Athena 中分析数据

Task	描述	所需技能
创建用于映射 JSON 数据的表。	1. 打开 Athena 控制台。 2. 按照 A thena 文档中的说明创建数据库。 3. 从 “数据库” 菜单中，选择您创建的数据库。	开发人员

Task	描述	所需技能
	4. 在查询编辑器中，输入如下CREATE TABLE语句： `CREATE EXTERNAL TABLE financials_json (symbol string, financials array< struct<reportdate: string, grossprofit: bigint, totalrevenue: bigint, totalcash: bigint, totaldebt: bigint, researchanddevelopment: bigint>>) ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe' LOCATION 's3://s3bucket-for-athena/'` 其中，LOCATION指定包含 JSON 文件的 S3 存储桶的位置。 5. 选择 Run 创建表。 有关创建表的更多信息，请参阅 Athena 文档。	

Task	描述	所需技能
创建用于数据分析的视图。	1. 打开 Athena 控制台。 2. 按照 A thena 文档中的说明创建数据库。 3. 从 “数据库” 菜单中，选择您创建的数据库。 4. 在查询编辑器中，输入如下CREATE VIEW语句： `CREATE OR REPLACE VIEW financial_json_view AS SELECT symbol, financials[1].reportdate one_report_date, -- indexes start with 1 financials[1].totalrevenue one_total_revenue, financials[1].reportdate another_report_date, financials[1].totalrevenue another_total_revenue FROM financials_json where symbol='AAPL' ORDER BY 1` 5. 选择 Run（运行）以创建视图。 有关创建视图的更多信息，请参阅 Athena 文档。	开发人员

Task	描述	所需技能
分析和验证数据。	1. 打开 Athena 控制台。 2. 在查询编辑器中，使用您在上一步中创建的视图运行查询。 3. 对照 JSON 文件验证数据，以确认列名和数据类型映射正确。	开发人员

在中可视化数据 QuickSight

Task	描述	所需技能
将 Athena 设置为中的数据源。 QuickSight	1. 打开 QuickSight 管理控制台。 2. 选择数据集、新数据集。 3. 选择 Athena 作为数据源。 4. 选择包含您创建的视图的数据库。 5. 选择要为其创建数据集的视图。 6. 在 “完成数据集创建” 页面上，选择 “直接查询您的数据”。 7. 选择 Visualize。	系统管理员
可视化中的数据 QuickSight。	1. 可视化数据集后，从左侧窗格中选择视觉对象，然后为数据集选择字段。有关更多信息，请参阅 QuickSight 文档中的教程。 2. 保存对分析的更改。	数据分析人员

Task	描述	所需技能
	3. 选择 “发布仪表板” 以发布您创建的视觉效果。	

相关资源

- Amazon Athena 文档
- 亚马逊 QuickSight 教程
- 使用嵌套 JSON（博客文章）

自动将数据从 Amazon S3 AWS Data Exchange 中提取数据

由 Adnan Alvee (AWS) 和 Manikanta Gona (AWS) 创作

摘要

此模式提供了一个 AWS CloudFormation 模板，使您能够自动将数据从亚马逊简单存储服务 (Amazon S3) 中的数据提取 AWS Data Exchange 到您的数据湖中。

AWS Data Exchange 是一项服务，可让您轻松安全地在 AWS 云中交换基于文件的数据集。 AWS Data Exchange 数据集是基于订阅的。作为订阅用户，您还可以在提供程序发布新数据时访问数据集修订版。

该 AWS CloudFormation 模板在 Amazon Events 中创建了一个 CloudWatch 事件和一个 AWS Lambda 函数。该事件将监视您所订阅的数据集是否有任何更新。如果有更新，则 CloudWatch 启动 Lambda 函数，该函数会将数据复制到您指定的 S3 存储桶。成功复制数据后，Lambda 将向您发送 Amazon Simple Notification Service(Amazon SNS) 通知。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 订阅中的数据集 AWS Data Exchange

限制

- 必须为中的每个订阅数据集单独部署 AWS CloudFormation 模板。 AWS Data Exchange

架构

目标技术堆栈

- AWS Lambda
- Amazon S3
- AWS Data Exchange
- Amazon CloudWatch

- Amazon SNS

目标架构

自动化和扩缩

对于要导入到数据湖中的数据集，您可以多次使用该 AWS CloudFormation 模板。

工具

- AWS Data Exchange使 AWS 客户可以轻松安全地交换中基于文件的数据集。 AWS Cloud作为订阅用户，您可查找和订阅来自合格数据提供商的数百种产品。然后，您可以快速下载数据集或将其复制到 Amazon S3，以便在各种 AWS 分析和机器学习服务中使用。任何拥有的人 AWS 账户 都可以成为 AWS Data Exchange 订阅者。
- 利用 AWS Lambda，您可以运行代码，而无需预置或管理服务器。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。您只需按消耗的计算时间付费；代码未运行时不产生费用。借助 Lambda，您几乎可以为任何类型的应用程序或后端服务运行代码，无需任何管理。Lambda 在高可用性计算基础设施上运行您的代码，并管理所有计算资源，包括服务器和操作系统维护、容量配置和自动扩展、代码监控和日志记录。
- Amazon S3 为互联网提供存储空间。您可以通过 Amazon S3 随时在 Web 上的任何位置存储和检索的任意大小的数据。
- Amazon CloudWatch Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。使用可以快速设置的简单规则，您可以匹配事件并将它们路由到一个或多个目标函数或流。CloudWatch 事件在发生时就会意识到操作变化。其可响应这些操作更改并在必要时采取纠正措施，方式是发送消息以响应环境、激活函数、进行更改并捕获状态信息。您还可以使用 Ev CloudWatch ents 来安排自动操作，这些操作在特定时间使用 cron 或速率表达式自行启动。
- 亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 使应用程序、最终用户和设备能够立即从云端发送和接收通知。Amazon SNS 为高吞吐量、基于推送的消息传递提供主题（通信渠道）。 many-to-many使用亚马逊 SNS 主题，发布者可以向大量订阅者分发消息以进行并行处理，包括亚马逊简单队列服务 (Amazon SQS) Simple Queue Service 队列、Lambda 函数和 HTTP/S 网络挂钩。您还可以使用 Amazon SNS，通过移动推送、SMS 和电子邮件向最终用户发送通知。

操作说明

订阅数据集

Task	描述	所需技能
订阅数据集。	在 AWS Data Exchange 控制台中，订阅数据集。有关说明，请参阅 AWS 文档 AWS Data Exchange中的订阅数据产品。	常规 AWS
注意数据集的属性。	记下数据集的 AWS 区域、ID 和修订版 ID。在下一步中，你将需要这个作为 AWS CloudFormation 模板。	常规 AWS

部署 AWS CloudFormation 模板

Task	描述	所需技能
创建 S3 存储桶和文件夹。	如果您在 Amazon S3 中已经有数据湖，请创建一个文件夹来存储要从中 AWS Data Exchange提取的数据。如果您要为测试部署模板，请创建新的 S3 存储桶，并记下存储桶名称和文件夹前缀，以供下一步使用。	常规 AWS
部署 AWS CloudFormation 模板。	部署作为该模式附件提供的 AWS CloudFormation 模板。有关说明，请参阅AWS CloudFormation 文档。 将以下参数配置为与您的 AWS 账户、数据集和	常规 AWS

Task	描述	所需技能
	S3 存储桶设置相对应：数据集 AWS 区域、数据集 ID、修订版 ID、S3 存储桶名称（例如`DOC-EXAMPLE-BUCKET`）、文件夹前缀（例如`myfolder/`）和用于 SNS 通知的电子邮件。您可以将数据集名称参数设置为任何名称。部署模板时，它会运行 Lambda 函数，以自动摄取数据集中可用的第一组数据。随后，当新数据到达数据集时，后续摄取将自动进行。	

相关资源

- 在 AWS Data Exchange（AWS Data Exchange 文档）上订阅数据产品

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS CloudFormation 模板在 AWS Glue 中自动执行加密

由 Diogo Guedes(AWS) 编写

摘要

此模式向您展示如何使用 AWS CloudFormation 模板在 AWS Glue 中设置和自动执行加密。该模板创建强制加密所需所有配置和资源。这些资源包括初始配置、由亚马逊 EventBridge 规则创建的预防性控制和 AWS Lambda 函数。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 部署 CloudFormation 模板及其资源的权限

限制

这种安全控制为区域性的。您必须在每个要对 AWS Glue 中设置加密强制性的AWS 区域 部署安全控件。

架构

目标技术堆栈

- 亚马逊 CloudWatch 日志（来自 AWS Lambda）
- 亚马逊 EventBridge 规则
- AWS CloudFormation 堆栈
- AWS CloudTrail
- AWS Identity and Access Management (IAM) 托管角色和策略
- AWS Key Management Service (AWS KMS)
- AWS KMS 别名
- AWS Lambda 函数
- AWS Systems Manager Parameter Store

目标架构

下图显示了如何在 AWS Glue 中自动执行加密。

图表显示了以下工作流：

1. CloudFormation 模板可创建所有资源，包括在 AWS Glue 中强制执行加密的初始配置和侦测控制。
2. EventBridge 规则检测到加密配置中的状态变化。
3. 通过 CloudWatch 日志调用 Lambda 函数进行评估和记录。对于不合规性检测，通过 AWS KMS 密钥的 Amazon 资源名称（ARN）恢复 Parameter Store。该服务已修复为启用加密的合规状态。

自动化和扩缩

如果您使用的是 AWS Org anizations，则可以使用 AWS 将此模板部署 CloudFormation StackSets到要在 AWS Glue 中启用加密强制执行的多个账户中。

工具

- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和在 AWS 上运行的应用程序的指标。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如，Lambda 函数、使用 API 目标的 HTTP 调用端点或其他 Amazon Web Services account 中的事件总线。
- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。
- AWS CloudTrail 可帮助您对 AWS 账户进行运营和风险审计、监管和合规。
- AWS Glue 是一项完全托管的提取、转换、加载（ETL）服务。它可以帮助您在数据存储和数据流之间对数据进行可靠地分类、清理、扩充和移动。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥，以帮助保护您的数据。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Systems Manager 可帮助您管理在 Amazon Web Services Cloud 中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。

代码

此模式的代码可在 GitHub aws-custom-guardrail-event驱动的存储库中找到。

最佳实践

AWS Glue 支持静态数据加密，用于在 AWS Glue 中编写作业和使用开发端点开发脚本。

考虑下面的最佳实践：

- 配置 ETL 作业和开发端点，以使用 AWS KMS 密钥写入加密的静态数据。
- 您可通过 AWS KMS 托管的密钥，对存储在 AWS Glue Data Catalog 中的元数据加密。
- 使用AWS KMS 密钥来加密作业书签以及爬网程序和 ETL 作业生成的日志。

操作说明

启动 CloudFormation 模板

Task	描述	所需技能
部署 CloudFormation 模板。	从 GitHub 存储库下载`aws-custom-guardrail-event-driven.yaml` 模板，然后部署该模板。`CREATE_COMPLETE` 状态表示您的模板已成功部署。 **Note** 该模板不需要任何输入参数。	云架构师

验证 AWS Glue 中的加密设置

Task	描述	所需技能
检查 AWS KMS 密钥配置。	1. 登录到 AWS 管理控制台，然后打开 AWS Glue 控制台。	云架构师

Task	描述	所需技能
	2. 在导航窗格的数据目录下，选择目录设置。 3. 确认已标记元数据加密和加密连接密码设置，并将其配置为使用 `KMSKeyGlue` 。	

测试加密强制执行

Task	描述	所需技能
在中标识加密设置 CloudFormation。	1. 登录 AWS 管理控制台，然后打开CloudFormation 控制台。 2. 在导航窗格中，选择 Stacks(堆栈)，然后选择您的堆栈。 3. 选择资源选项卡。 4. 在资源表，按逻辑 ID 查找加密设置。	云架构师
将已配置的基础设施切换至不合规状态。	1. 登录 AWS 管理控制台，然后打开 AWS Glue 控制台。 2. 在导航窗格的数据目录下，选择目录设置。 3. 清除元数据加密复选框。 4. 清除加密连接密码复选框。 5. 选择 Save(保存)。 6. 刷新 AWS Glue 控制台。 在您清除复选框后，防护机制会检测 AWS Glue 中的不合规状态，然后通过自动修复加密	云架构师

Task	描述	所需技能
	错误配置来强制合规。因此刷新页面后，应再次选中加密复选框。	

相关资源

- 在 AWS CloudFormation 控制台上创建堆栈（AWS CloudFormation 文档）
- 使用 AWS 创建在 AWS API 调用时触发 CloudWatch 的事件规则 CloudTrail（亚马逊 CloudWatch 文档）
- 在 AWS Glue 中设置加密（AWS Glue 文档）

使用 AWS DataOps 开发套件构建数据管道，以提取、转换和分析 Google Analytics（分析）数据

创建者：Anton Kukushkin (AWS) 和 Rudy Puig (AWS)

摘要

此模式描述了如何使用 AWS DataOps 开发套件 (AWS DDK) 等构建数据管道来提取、转换和分析 Google Analytics 数据。 AWS 服务 AWS DDK 是一个开源开发框架，可帮助您在上 AWS面构建数据工作流程和现代数据架构。 AWS DDK 的主要目标之一是为您节省通常用于劳动密集型数据管道任务的时间和精力，例如协调管道、构建基础设施和创建基础架构。 DevOps 您可以将这些劳动密集型任务卸载到 AWS DDK，这样您就可以专注于编写代码和其他高价值的活动。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 已配置用于谷歌分析的 Amazon AppFlow 连接器
- Python 和 pip（Python 的包管理器）
- Git，已安装和配置
- AWS Command Line Interface (AWS CLI)，已安装并配置
- AWS Cloud Development Kit (AWS CDK)，已安装

产品版本

- Python 3.7 或更高版本
- pip 9.0.3 或更高版本

架构

技术堆栈

- Amazon AppFlow
- Amazon Athena

- Amazon CloudWatch
- Amazon EventBridge
- Amazon Simple Storage Service（Amazon S3）
- Amazon Simple Queue Service(Amazon SQS)
- AWS DataOps 开发套件 (AWS DDK)
- AWS Lambda

目标架构

下图显示了摄取、转换和分析 Google Analytics 数据的事件驱动流程。

图表显示了以下工作流：

1. 亚马逊 CloudWatch 计划的事件规则会调用亚马逊。 AppFlow
2. 亚马逊将谷 AppFlow 歌分析数据提取到 S3 存储桶中。
3. 在 S3 存储桶提取数据后，将生成中的 EventBridge 事件通知，由 CloudWatch 事件规则捕获，然后将其放入 Amazon SQS 队列中。
4. Lambda 函数使用来自 Amazon SQS 队列的事件，读取相应的 S3 对象，将对象转换为 Apache Parquet 格式，将转换后的对象写入 S3 存储桶，然后创建或更新表定义。 AWS Glue Data Catalog
5. Athena 查询针对此表运行。

工具

AWS 工具

- Amazon AppFlow 是一项完全托管的集成服务，使您能够在软件即服务 (SaaS) 应用程序之间安全地交换数据。
- Amazon Athena 是一种交互式查询服务，可帮助您使用标准 SQL 直接在 Amazon S3 中分析数据。
- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和运行的应用程序 AWS 的指标。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如， AWS Lambda 函数、使用 API 目的地的 HTTP 调用端点或其他 AWS 账户目的地的事件总线。

- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon Simple Queue Service (Amazon SQS) 提供了一个安全、持久且可用的托管队列，它可帮助您集成和分离分布式软件系统与组件。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Cloud Development Kit (AWS CDK)是一个框架，用于在代码中定义云基础架构并通过它进行配置 AWS CloudFormation。
- AWS DataOps 开发套件 (AWS DDK) 是一个开源开发框架，可帮助您在上 AWS面构建数据工作流程和现代数据架构。

代码

此模式的代码可在 GitHub AWS DataOps 开发套件 (AWS DDK) 和使用亚马逊 AppFlow、Amazon Athena AWS DataOps 和开发套件存储库分析谷歌分析数据中找到。

操作说明

准备环境

Task	描述	所需技能
克隆源代码。	要克隆源代码，请运行以下命令： `git clone https://github.com/aws-samples/aws-ddk-examples.git`	DevOps 工程师
创建虚拟环境。	导航到源代码目录，然后运行以下命令创建虚拟环境： `cd google-analytics-data-using-appflow/python && python3 -m venv .venv`	DevOps 工程师

Task	描述	所需技能
安装依赖项。	要激活虚拟环境并安装依赖项，请运行以下命令： `source .venv/bin/activate && pip install -r requirements.txt`	DevOps 工程师

部署使用您数据管线的应用程序

Task	描述	所需技能
引导环境。	1. 确认 AWS CLI 已使用您的有效凭据进行设置 AWS 账户。有关更多信息，请参阅文档中的使用命名配置 AWS CLI 文件。 2. 运行 `cdk bootstrap --profile [AWS_PROFILE]` 命令。	DevOps 工程师
部署数据。	要部署数据管线，请运行 `cdk deploy --profile [AWS_PROFILE]` 命令。	DevOps 工程师

测试部署

Task	描述	所需技能
验证堆栈状态。	1. 打开 AWS CloudFormation 管理控制台。 2. 在堆栈页面，确认堆栈 DdkAppflowAthenaSt	DevOps 工程师

Task	描述	所需技能
	ack 的状态为 CREATE_COMPLETE 。	

故障排除

事务	解决方案
如果在创建 AWS::AppFlow::Flow 资源期间部署失败，您会收到以下错误：Connector Profile with name ga-connection does not exist	确认您已为 Google Analytics（分析）创建了亚马逊 AppFlow 连接器并将其命名ga-connection 。 有关说明，请参阅 Amazon AppFlow 文档中的 Google 分析。

相关资源

- AWS DataOps 开发套件 (AWS DDK) (GitHub)
- AWS DDK 示例 () GitHub

其他信息

AWS DDK 数据管道由一个或多个阶段组成。在以下代码示例中，您使用 AppFlowIngestionStage 从 Google Analytics 摄取数据，使用 SqsToLambdaStage 处理数据转换，使用 AthenaSQLStage 运行 Athena 查询。

首先，创建数据转换和摄取阶段，如以下代码示例所示：

```
        appflow_stage = AppFlowIngestionStage(
            self,
            id="appflow-stage",
            flow_name=flow.flow_name,
        )
        sqs_lambda_stage = SqsToLambdaStage(
            self,
            id="lambda-stage",
```

```
            lambda_function_props={
                "code": Code.from_asset("./ddk_app/lambda_handlers"),
                "handler": "handler.lambda_handler",
                "layers": [
                    LayerVersion.from_layer_version_arn(
                        self,
                        id="layer",
                        layer_version_arn=f"arn:aws:lambda:
{self.region}:336392948345:layer:AWSDataWrangler-Python39:1",
                    )
                ],
                "runtime": Runtime.PYTHON_3_9,
            },
        )
        # Grant lambda function S3 read & write permissions
        bucket.grant_read_write(sqs_lambda_stage.function)
        # Grant Glue database & table permissions
        sqs_lambda_stage.function.add_to_role_policy(
            self._get_glue_db_iam_policy(database_name=database.database_name)
        )
        athena_stage = AthenaSQLStage(
            self,
            id="athena-sql",
            query_string=[
                (
                    "SELECT year, month, day, device, count(user_count) as cnt "
                    f"FROM {database.database_name}.ga_sample "
                    "GROUP BY year, month, day, device "
                    "ORDER BY cnt DESC "
                    "LIMIT 10; "
                )
            ],
            output_location=Location(
                bucket_name=bucket.bucket_name, object_key="query-results/"
            ),
            additional_role_policy_statements=[
                self._get_glue_db_iam_policy(database_name=database.database_name)
            ],
        )
```

接下来，使用该`DataPipeline`构造通过使用 EventBridge 规则将各个阶段 “连接” 在一起，如以下代码示例所示：

```
        (
            DataPipeline(self, id="ingestion-pipeline")
            .add_stage(
                stage=appflow_stage,
                override_rule=Rule(
                    self,
                    "schedule-rule",
                    schedule=Schedule.rate(Duration.hours(1)),
                    targets=appflow_stage.targets,
                ),
            )
            .add_stage(
                stage=sqs_lambda_stage,
                # By default, AppFlowIngestionStage stage emits an event after the flow
run finishes successfully
                # Override rule below changes that behavior to call the the stage when
data lands in the bucket instead
                override_rule=Rule(
                    self,
                    "s3-object-created-rule",
                    event_pattern=EventPattern(
                        source=["aws.s3"],
                        detail={
                            "bucket": {"name": [bucket.bucket_name]},
                            "object": {"key": [{"prefix": "ga-data"}]},
                        },
                        detail_type=["Object Created"],
                    ),
                    targets=sqs_lambda_stage.targets,
                ),
            )
            .add_stage(stage=athena_stage)
        )
```

有关更多代码示例，请参阅使用亚马逊、Amazon AppFlow Athena AWS DataOps 和开发套件 GitHub 分析谷歌分析数据存储库。

通过 Amazon Kinesis Video Streams 和 AWS Fargate 构建视频处理管道

由 Piotr Chotkowski (AWS) 和 Pushparaju Thangavel (AWS) 编写

摘要

此模式演示了如何使用Amazon Kinesis Video Streams 和 AWS Fargate 从视频流中提取帧并将其存储为图像文件，以便 Amazon Simple Storage Service (Amazon S3) 进行进一步处理。

该模式以 Java Maven 项目的形式提供了示例应用程序。此应用程序使用 AWS Cloud Development Kit (AWS CDK) 定义 AWS 基础设施。帧处理逻辑和基础设施定义均采用 Java 编程语言编写。您可将此示例应用程序用作开发自己的实时视频处理管道或构建机器学习管道的视频预处理步骤的基础。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- Java SE Development Kit (JDK) 11，已安装
- Apache Maven，已安装
- AWS Cloud Development Kit (AWS CDK)，已安装
- AWS 命令行界面（AWS CLI）版本 2，已安装
- Docker（构建 Docker 映像以在 AWS Fargate 任务定义中使用）

限制

该模式旨在作为概念证明，或作为进一步开发的基础。不应在生产部署中以当前形式使用它。

产品版本

- 此模式已在 AWS CDK 版本 1.77.0 中进行了测试（参见 AWS CDK 版本）
- JDK 11
- AWS CLI 版本 2

架构

目标技术堆栈

- Amazon Kinesis Video Streams
- AWS Fargate 任务
- Amazon Simple Queue Service (Amazon SQS) 队列
- Amazon S3 存储桶

目标架构

用户创建 Kinesis 视频流、上传视频并将包含有关输入 Kinesis 视频流和输出 S3 存储桶的详细信息的 JSON 消息发送到 SQS 队列。AWS Fargate 在容器中运行主应用程序，从 SQS 队列中提取消息并开始提取帧。每帧都保存在图像文件中，并存储在目标 S3 存储桶中。

自动化和扩缩

该示例应用程序可以在单个 AWS 区域内进行水平和垂直扩展。可以通过增加从 SQS 队列读取的已部署 AWS Fargate 任务的数量来实现水平扩展。垂直缩放可以通过增加应用程序中的帧分割和图像发布线程的数量来实现。在 AWS CDK 的QueueProcessingFargateService资源定义中，这些设置作为环境变量传递给应用程序。由于 AWS CDK 堆栈部署的性质，您无需额外努力即可在多个 AWS 区域和账户中部署此应用程序。

工具

工具

- AWS CDK 是一个软件开发框架，用于使用编程语言（例如、、Python TypeScript JavaScript、Java 和 C#/Net）来定义您的云基础设施和资源。
- Amazon Kinesis Video Streams 是一项完全托管的 Amazon Web Services，您可使用该服务将实时视频从各个设备流式传输到 Amazon Web Services Cloud，或者构建应用程序以进行实时视频处理或进行面向批处理的视频分析。
- AWS Fargate 是适用于容器的无服务器计算引擎。Fargate 无需配置和管理服务器，让您可专注于开发应用程序。
- Amazon S3 是一种对象存储服务，提供可扩展性、数据可用性、安全性和性能。

- Amazon SQS 是一种完全托管的消息队列服务，使您能够分离和扩展微服务、分布式系统和无服务器应用程序。

代码

- 随附示例应用程序项目 (`frame-splitter-code.zip`) 的 .zip 文件。

操作说明

部署基础设施

Task	描述	所需技能
启动 Docker 守护程序。	在本地系统上启动 Docker 进程守护程序。AWS CDK 使用 Docker 构建 AWS Fargate 任务中使用的映象。继续执行下一步操作之前，必须运行 Docker。	开发人员、 DevOps 工程师
构建 项目。	下载 `frame-splitter-code` 示例应用程序（附件）并将其内容解压缩到本地计算机上的文件夹中。在部署基础设施之前，必须要先构建 Java Maven 项目。在命令提示符处，导航至项目的根目录，然后通过运行以下命令来生成项目： `mvn clean install`	开发人员、 DevOps 工程师
引导 AWS CDK。	（仅限首次使用 AWS CDK 的用户）如果这是您首次使用 AWS CDK，则可能需要通过运行 AWS CLI 命令来引导环境：	开发人员、 DevOps 工程师

Task	描述	所需技能
	`cdk bootstrap --profile "$AWS_PROFILE_NAME"` 其中 `$AWS_PROFILE_NAME` 包含您的 AWS 凭证中的 AWS 个人资料的名称。或者，您可删除此参数以使用默认配置文件。有关更多信息，请参阅 AWS CDK 文档。	

Task	描述	所需技能
部署 AWS CDK 堆栈。	在此步骤中，您将在 Amazon Web Services account 中创建所需基础设施资源（SQS 队列、S3 存储桶、AWS Fargate 任务定义），构建 AWS Fargate 任务所需 Docker 映像，然后部署应用程序。在命令提示符处，导航至项目的根目录，然后运行以下命令： `cdk deploy --profile "$AWS_PROFILE_NAME" --all` 其中 `$AWS_PROFILE_NAME` 包含您的 AWS 凭证中的 AWS 个人资料的名称。或者，您可删除此参数以使用默认配置文件。确认部署。注意 CDK 部署输出中的QueueUrl和存储桶值；您将在以后的步骤中使用这些值。AWS CDK 创建资产，将其上传至您的 Amazon Web Services account，然后创建所有基础设施资源。您可以在 AWS CloudFormation 控制台中观察资源创建过程。有关更多信息，请参阅 AWS CloudFormation 文档和 AWS CDK 文档。	开发人员、 DevOps 工程师

Task	描述	所需技能
创建视频流	在此步骤中，您将创建 Kinesis 视频流，该视频流将用作视频处理的输入流。确保您已经安装并配置 AWS CLI。在 AWS CLI 中运行： `aws kinesisvideo --profile "$AWS_PROFILE_NAME" create-stream --stream-name "$STREAM_NAME" --data-retention-in-hours "24"` 其中，`$AWS_PROFILE_NAME` 包含您的 AWS 凭证中的 AWS 配置文件名称（或删除此参数以使用默认配置文件），`$STREAM_NAME` 是任何有效的直播名称。 或者，您可按照 Kinesis Video Streams 文档中的步骤使用 Kinesis 控制台创建视频流。记下创建的流的 AWS Resource Name (ARN)；稍后您会需要它。	开发人员、 DevOps 工程师

运行示例

Task	描述	所需技能
将视频上传至流数据。	在示例 `frame-splitter-code` 应用程序的项目文件夹中，打开 `src/test/`	开发人员、 DevOps 工程师

Task	描述	所需技能
	`java/amazon/awscdk/examples/splitter` 文件夹中的 `ProcessingTaskTest.java` 文件。将 `profileName` 和 `streamName` 变量替换为您在前面的步骤中使用的值。若要将示例视频上传到您在上一步中创建的 Kinesis 视频流，请运行： `amazon.awscdk.examples.splitter.ProcessingTaskTest#testExample test` 或者，您可使用 Kinesis Video Streams 文档中描述的方法之一上传视频。	

Task	描述	所需技能

启动视频处理。	现在，您已将视频上传到 Kinesis 视频流，您可开始处理该视频了。要启动处理逻辑，您须向 AWS CDK 在部署期间创建的 SQS 队列发送一条包含详细信息的消息。要使用 AWS CLI 发送消息，请运行：
	`aws sqs --profile "$AWS_PROFILE_NAME" send-message --queue-url QUEUE_URL --message-body MESSAGE`
	其中$AWS_PROFILE_NAME 包含您的 AWS 凭证中的 AWS 配置文件名称（删除此参数以使用默认配置文件），QUEUE_URL 是 AWS CDK 输出中的QueueUrl值，MESSAGE是以下格式的 JSON 字符串：
	`{ "streamARN": "STREAM_ARN", "bucket": "BUCKET_NAME", "s3Directory": "test-output" }`
	其中，STREAM_ARN 是您在前面步骤中创建的视频流的 ARN，BUCKET_NAME 也是 AWS CDK 输出中的存储桶值。
		开发人员、 DevOps 工程师

Task	描述	所需技能
	发送此消息将启动视频处理。或者，您也可以使用 Amazon SQS 控制台发送消息，如 Amazon SQS 文档中所述。	
查看视频帧图像。	您可在 S3 输出存储桶 `s3://BUCKET_NAME/test-output` 中看到生成的图像，其中 `BUCKET_NAME` 是 AWS CDK 输出中的存储桶值。	开发人员、 DevOps 工程师

相关资源

- AWS CDK 文档
- AWS CDK API 参考
- AWS CDK 入门研讨会
- Amazon Kinesis Video Streams 文档
- 示例：使用识别视频流中的对象 SageMaker
- 示例：解析和渲染 Kinesis 视频流片段
- 使用 Amazon Kinesis Video Streams 和亚马逊实时大规模分析直播视频（AWS Machin SageMaker e Learning 博客文章）
- AWS Fargate 入门

其他信息

选择 IDE

我们建议您使用自己喜欢的 Java IDE 构建和探索此项目。

清理

运行完此示例后，请移除所有已部署的资源，以免产生额外的 AWS 基础设施成本。

要删除基础设施和视频流，请在 AWS CLI 中使用以下两个命令：

```
cdk destroy --profile "$AWS_PROFILE_NAME" --all
```

```
aws kinesisvideo --profile "$AWS_PROFILE_NAME" delete-stream --stream-arn "$STREAM_ARN"
```

或者，您可以手动删除资源，方法是使用 AWS CloudFormation 控制台移除 AWS CloudFormation 堆栈，使用 Kinesis 控制台移除 Kinesis 视频流。请注意，`cdk destroy` 不会移除输出 S3 存储桶或 Amazon Elastic Container Registry (Amazon ECR) 存储库 (`aws-cdk/assets`) 中的图像。您必须手动将其移除。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS Glue 构建 ETL 服务管道以增量方式将数据从 Amazon S3 加载到 Amazon Redshift

由 Rohan Jamadagni (AWS) 和 Arunabha Datta (AWS) 编写

摘要

此模式提供有关如何配置 Amazon Simple Storage Service (Amazon S3) 以获得最佳数据湖性能，然后使用 AWS Glue 将增量数据更改从 Amazon S3 加载到 Amazon Redshift 中，执行提取、转换、加载（ETL）操作的指导 。

Amazon S3 中的源文件可以具有不同的格式，包括逗号分隔值 (CSV)、XML 和 JSON 文件。本示例介绍了如何使用 AWS Glue 将源文件转换为成本优化且性能优化的格式，例如 Apache Parquet。您可以直接从 Amazon Athena 和 Amazon Redshift Spectrum 查询 Parquet 文件。您还可以将 Parquet 文件加载到 Amazon Redshift 中，对其进行聚合，然后与消费者共享聚合数据，或者使用亚马逊对数据进行可视化。 QuickSight

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 具有正确权限并包含 CSV、XML 或 JSON 文件的 S3 源存储桶。

假设

- CSV、XML 或 JSON 源文件已加载到 Amazon S3 中，可以从配置 AWS Glue 和 Amazon Redshift 的账户进行访问。
- 如 Amazon Redshift 文档中所述，遵循加载文件、拆分文件、压缩和使用清单的最佳实践。
- 源文件结构保持不变。
- 源系统能够按照 Amazon S3 中定义的文件夹结构将数据提取至 Amazon S3 中。
- Amazon Redshift 集群可跨越单个可用区。(这种架构很合适，因为 AWS Lambda、AWS Glue 和 Amazon Athena 为无服务器。) 为了实现高可用性，定期拍摄集群快照。

限制

- 文件格式仅限于 AWS Glue 当前支持的格式。

- 不支持实时下游报告。

架构

源技术堆栈

- 包含 CSV、XML 或 JSON 文件的 S3 存储桶

目标技术堆栈

- S3 数据湖（带有分区 Parquet 文件存储）
- Amazon Redshift

目标架构

数据流

工具

- Amazon S3 — Amazon Simple Storage Service（Amazon S3）是一项可扩展的数据存储服务。Amazon S3 可用于各种存储解决方案，包括网站、移动应用程序、备份和数据湖。
- AWS Lambda – AWS Lambda 让您无需预置或管理服务器即可运行代码。AWS Lambda 是一项事件驱动的服务，您可以将代码设置为自动从其他 Amazon Web Services 启动。
- Amazon Redshift — Amazon Redshift 是一种完全托管的 PB 级数据仓库服务。借助 Amazon Redshift，您可以使用标准 SQL 跨数据仓库和数据湖查询 PB 级的结构化和半结构化数据。
- AWS Glue — AWS Glue 是一项完全托管的 ETL 服务，可让客户轻松准备和加载数据以进行分析。AWS Glue 发现您的数据并将关联的元数据（例如表定义和架构）存储在 AWS Glue Data Catalog 中。分类后的数据立即变得可搜索、可查询和可用于 ETL。
- AWS Secrets Manager — AWS Secrets Manager 有助于保护和集中管理应用程序或服务访问所需密钥。该服务存储数据库凭证、API 密钥和其他机密，并且无需以明文格式对敏感信息进行硬编码。Secrets Manager 还提供密钥轮换，以满足安全和合规性需求。它内置了 Amazon Redshift、Amazon Relational Database Service（Amazon RDS）和 Amazon DocumentDB 的集

成。你可以使用 Secrets Manager 控制台、命令行界面 (CLI) 或 Secrets Manager API 来存储和集中管理密钥。 SDKs

- Amazon Athena — Amazon Athena 是一种交互式查询服务，可让您轻松分析存储在 Amazon S3 中的数据。Athena 是无服务器的，可与 AWS Glue 集成，因此它可以直接查询使用 AWS Glue 编目的数据。Athena 可弹性扩展以提供交互式查询性能。

操作说明

创建 S3 存储桶与文件夹结构

Task	描述	所需技能
分析源系统的数据结构与属性。	对参与 Amazon S3 数据湖的每个数据来源执行此任务。	数据工程师
定义分区与访问策略。	该策略应基于数据捕获的频率、增量处理以及消耗需求。确保 S3 存储桶不向公众开放，并且访问仅由基于特定服务角色的策略控制。有关更多信息，请参阅 Amazon S3 文档。	数据工程师
为每种数据来源类型创建单独的 S3 存储桶，为每个数据来源创建单独的 S3 存储桶 (Parquet)，为每个数据来源创建单独的 S3 存储桶。	为每个源创建一个单独的存储桶，然后根据源系统的数据提取频率创建文件夹结构，例如 `s3://source-system-name/date/hour` 。对于已处理的 (转换为 Parquet 格式) 文件，请创建类似的结构，例如 `s3://source-processed-bucket/date/hour` 。有关创建 S3 存储桶的更多信息，请参阅 Amazon S3 文档。	数据工程师

在 Amazon Redshift 中创建数据仓库

Task	描述	所需技能
使用适当的参数组以及维护和备份策略启动 Amazon Redshift 集群。	创建 Amazon Redshift 集群时，使用 Secrets Manager 数据库密钥作为管理员用户凭证。有关创建和调整 Amazon Redshift 集群规模的信息，请参阅 Amazon Redshift 文档和 Sizing Cloud Data Warehouses 白皮书。	数据工程师
创建 IAM 服务角色，并将其附加至 Amazon Redshift 集群。	AWS Identity and Access Management (IAM) 服务角色确保对 Secrets Manager 和源 S3 存储桶的访问。有关更多信息，请参阅有关授权和添加角色的 AWS 文档。	数据工程师
创建数据库架构。	遵循设计表的 Amazon Redshift 最佳实践。根据用例，选择适当的排序和分配键，以及最佳的压缩编码。有关最佳实践，请参阅 AWS 文档。	数据工程师
配置工作负载管理。	根据您的要求配置工作负载管理 (WLM) 队列、短查询加速 (SQA) 或者并发扩展。有关更多信息，请参阅 Amazon Redshift 文档中的实施工作负载管理。	数据工程师

在 Secrets Manager 中创建密钥

Task	描述	所需技能
创建一个新密钥以将 Amazon Redshift 登录凭证存储在 Secrets Manager 中。	此机密存储管理员用户以及各个数据库服务用户的凭证。有关说明，请参阅 Secrets Manager 文档。选择 Amazon Redshift 集群作为密钥类型。此外，在密钥轮换页面，打开轮换。这将在 Amazon Redshift 集群中创建适当的用户，并按定义的时间间隔轮换密钥。	数据工程师
创建 IAM policy 以限制 Secrets Manager 访问权限。	将 Secrets Manager 的访问权限限制为仅限 Amazon Redshift 管理员和 AWS Glue。	数据工程师

配置 AWS Glue

Task	描述	所需技能
在 AWS Glue Data Catalog 中，为 Amazon Redshift 添加连接。	有关说明，请参阅 AWS Glue 文档。	数据工程师
创建并附加 AWS Glue 的 IAM 服务角色，以访问 Secrets Manager、Amazon Redshift 和 S3 存储桶。	有关更多信息，请参阅 AWS Glue 文档。	数据工程师
为数据源定义 AWS Glue Data Catalog。	此步骤涉及在 AWS Glue Data Catalog 中创建数据库和所需表。您可使用爬网程序对 AWS Glue 数据库中的表进行编目，	数据工程师

Task	描述	所需技能
	也可以将其定义为 Amazon Athena 外部表。您还可以通过 AWS Glue Data Catalog 访问在 Athena 中定义的外部表。有关在 Athena 中定义数据目录和创建外部表的更多信息，请参阅 AWS 文档。	
创建 AWS Glue 作业来处理源数据。	AWS Glue 任务可以是 Python 外壳，也可以 PySpark 用于对源数据文件进行标准化、重复数据删除和清理。为了优化性能并避免查询整个 S3 源存储桶，请按日期对 S3 存储桶进行分区，按年、月、日和小时细分，作为 AWS Glue 作业的下推谓词。有关更多信息，请参阅 AWS Glue 文档。将处理和转换后的数据以 Parquet 格式加载至已处理的 S3 存储桶分区。您可从 Athena 查询 Parquet 文件。	数据工程师
创建 AWS Glue 作业以将数据加载至 Amazon Redshift 中。	AWS Glue 任务可以是 Python 外壳，也可以 PySpark 通过更新数据然后完全刷新来加载数据。有关详细信息，请参阅 AWS Glue 文档和其他信息部分。	数据工程师

Task	描述	所需技能
（可选）必要时使用触发器安排 AWS Glue 作业。	增量数据加载主要由 Amazon S3 事件驱动，该事件导致 AWS Lambda 函数调用 AWS Glue 作业。对于需要基于时间而不是基于事件的调度的任何数据加载，使用基于 AWS Glue 触发器的调度。	数据工程师

创建 Lambda 函数

Task	描述	所需技能
为 AWS Lambda 创建并附加 IAM 服务相关角色，以访问 S3 存储桶和 AWS Glue 作业。	为 AWS Lambda 创建 IAM 服务相关角色，并使用读取 Amazon S3 对象和存储桶的策略以及访问 AWS Glue API 以启动 AWS Glue 作业的策略。有关更多信息，请参阅知识中心。	数据工程师
创建 Lambda 函数，以根据定义的 Amazon S3 事件运行 AWS Glue 任务。	Lambda 函数应通过创建 Amazon S3 清单文件来启动。Lambda 函数应将 Amazon S3 文件夹位置（例如 source_bucket/year/month/date/hour）作为参数传递给 AWS Glue 作业。AWS Glue 作业将使用此参数作为下推谓词，以优化文件访问和作业处理性能。有关更多信息，请参阅 AWS Glue 文档。	数据工程师
创建 Amazon S3 PUT 对象事件以检测对象创建，并调用相应的 Lambda 函数。	Amazon S3 PUT 对象事件只能通过创建清单文件启动。清单文件控制 Lambda 函数和	数据工程师

Task	描述	所需技能
	AWS Glue 作业并发性，并批量处理负载，而不是处理到达 S3 源存储桶的特定分区的单个文件。有关更多信息，请参阅 Lambda 文档。	

相关资源

- Amazon S3 文档
- AWS Glue 文档
- Amazon Redshift 文档
- AWS Lambda
- Amazon Athena
- AWS Secrets Manager

其他信息

更新插入和完全刷新的详细方法

更新插入：这适用于需要历史汇总的数据集，具体取决于业务用例。根据您的业务需求，按照更新和插入新数据（Amazon Redshift 文档）中描述的方法之一进行操作。

完全刷新：适用于不需要历史聚合的小数据集。请遵循以下方法之一：

1. 截断 Amazon Redshift 表。
2. 从暂存区加载当前分区

或者：

1. 创建包含当前分区数据的临时表。
2. 删除目标 Amazon Redshift 表。
3. 将临时表重命名至目标表。

使用 Amazon 构建企业数据网格 DataZone AWS CDK，以及 AWS CloudFormation

由 Dhrubajyoti Mukherjee (AWS)、Adjoa Taylor (AWS)、Ravi Kumar (AWS) 和 Weizhou Sun (AWS) 创作

摘要

在 Amazon Web Services (AWS) 上，客户明白，数据是加速创新和提高企业商业价值的关键。要管理这些海量数据，你可以采用分散式架构，例如数据网格。数据网格架构促进了产品思维，这种思维方式将客户、目标和市场考虑在内。数据网格还有助于建立联合治理模型，提供对数据的快速、安全访问。

构建基于数据网格的企业解决方案的策略 AWS讨论了如何使用数据网格策略框架为组织制定和实施数据网格策略。通过使用数据网格策略框架，您可以优化团队的组织及其互动，从而加快数据网格之旅。

本文档提供了有关如何使用 Amazon 构建企业数据网格的指导 DataZone。Amazon DataZone 是一项数据管理服务，用于编目、发现、共享和管理存储在本地和第三方来源的数据。 AWS该模式包括代码工件，可帮助您使用和部署基于数据网格的数据解决方案基础架构。 AWS Cloud Development Kit (AWS CDK) AWS CloudFormation此模式适用于云架构师和 DevOps 工程师。

有关此模式的目标和解决方案范围的信息，请参阅 “其他信息” 部分。

先决条件和限制

先决条件

- 至少有两个活跃账户 AWS 账户：一个用于中央治理账户，另一个用于成员账户
- AWS 开发环境中中央治理账户的管理员凭证
- AWS Command Line Interface (AWS CLI) 已安装，用于 AWS 服务 从命令行管理你的
- 安装了 Node.js 和 Node Package Manager (npm) 来管理应用程序 AWS CDK
- AWS CDK 使用 npm 在全球范围内安装工具包，用于合成和部署应用程序 AWS CDK

```
npm install -g aws-cdk
```

- 你的开发环境中安装了 Python 3.12 版本
- TypeScript 安装在您的开发环境中或使用 npm 编译器全局安装：

```
npm install -g typescript
```

- 在你的开发环境中安装了 Docker
- 使用版本控制系统（例如 Git）来维护解决方案的源代码（推荐）
- 支持 Python 和（强烈推荐）的集成开发环境 TypeScript (IDE) 或文本编辑器

限制

- 该解决方案仅在运行 Linux 或 macOS 的计算机上进行了测试。
- 在当前版本中，该解决方案默认不支持Amazon DataZone AWS IAM Identity Center 的集成。但是，您可以将其配置为支持此集成。

产品版本

- Python 版本 3.12

架构

下图显示了数据网格参考架构。该架构基于亚马逊，使用亚马逊 DataZone 简单存储服务 (Amazon S3) Simple Service AWS Glue Data Catalog 和作为数据源。根据贵组织的要求 AWS 服务 ，您在数据网格实施 DataZone 中与 Amazon 搭配使用的可能有所不同。

1. 在生产者账户中，原始数据要么适合以当前形式使用，要么通过使用进行转换以供使用 AWS Glue。数据的技术元数据存储在 Amazon S3 中，并使用 AWS Glue 数据爬网程序进行评估。数据质量是使用AWS Glue 数据质量来衡量的。数据目录中的源数据库已注册为亚马逊 DataZone 目录中的资产。Amazon DataZone 目录使用亚马逊 DataZone 数据源任务托管在中央管理账户中。
2. 中央管理账户托管亚马逊 DataZone 域名和亚马逊 DataZone 数据门户。数据生产者和使用者与 Amazon DataZone 域名相关联。 AWS 账户 数据生产者和消费者的亚马逊 DataZone 项目是在相应的亚马逊 DataZone 域名单位下组织的。
3. 数据资产的最终用户使用其 AWS Identity and Access Management (IAM) 证书或单点登录（通过 IAM 身份中心集成）登录亚马逊 DataZone 数据门户。他们搜索、筛选和查看 Amazon 数据目录中的资产信息（例如，数据质量信息或业务和技术元 DataZone 数据）。
4. 最终用户找到他们想要的数据资产后，他们会使用 Amazon DataZone 订阅功能来请求访问权限。创建者团队的数据所有者会收到通知，并在 Amazon DataZone 数据门户中评估订阅请求。数据所有者根据订阅请求的有效性批准或拒绝订阅请求。
5. 在批准并完成订阅请求后，可以在消费者账户中访问该资产，用于以下活动：

- 使用 Amazon AI 开发人工智能/机器学习模型 SageMaker
- 使用亚马逊 Athena 和亚马逊进行分析和报告 QuickSight

工具

AWS 服务

- Amazon Athena 是一种交互式查询服务，可帮助您通过使用标准 SQL 直接分析 Amazon Simple Storage Service（Amazon S3）中的数据。
- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，可帮助您在代码中定义和配置 AWS Cloud 基础架构。
- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。
- Amazon DataZone 是一项数据管理服务，可帮助您编目、发现、共享和管理存储在本地和第三方来源中的数据。 AWS
- Amazon QuickSight 是一项云规模的商业智能 (BI) 服务，可帮助您在单个控制面板中可视化、分析和报告数据。
- Amazon SageMaker AI 是一项托管机器学习 (ML) 服务，可帮助您构建和训练机器学习模型，然后将其部署到生产就绪的托管环境中。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon Simple Queue Service (Amazon SQS) 提供了一个安全、持久且可用的托管队列，它可帮助您集成和分离分布式软件系统与组件。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

代码存储库

该解决方案可在 GitHub data-mesh-datazone-cdk-cloudformation 存储库中找到。

操作说明

设置环境

Task	描述	所需技能
克隆存储库。	要克隆存储库，请在本地开发环境（Linux 或 macOS）中运行以下命令： `git clone https://github.com/aws-samples/data-mesh-data zone-cdk-cloudform ation`	云架构师、 DevOps 工程师

创建环境。	要创建 Python 虚拟环境，请运行以下命令：
	`python3 -m venv .venv`
	`source .venv/bin/activate`
	`pip install -r requirements.txt`
		云架构师、 DevOps 工程师

引导账户。	要使用引导中央治理帐户 AWS CDK，请运行以下命令： `cdk bootstrap aws://<GOVERNANCE_ACCOUNT_ID>/<AWS_REGION>` 登录 AWS Management Console，打开中央治理账户控制台，获取 AWS CDK 执行角色的 Amazon 资源名称 (ARN)。	云架构师、 DevOps 工程师

Task	描述	所需技能
构造DzDataMeshMemberStackSet.yaml 文件。	要从存储库的根目录中构造DzDataMeshMemberStackSet.yaml 文件，请启动以下 bash 脚本： `./lib/scripts/create_dz_data_mesh_member_stack_set.sh`	云架构师、 DevOps 工程师
确认模板创建。	确保 AWS CloudFormation 模板文件是在该lib/cfn-templates/DzDataMeshMemberStackSet.yaml 位置创建的。	云架构师、 DevOps 工程师

在中央治理账户中部署资源

Task	描述	所需技能

修改配置。	在该config/Config.ts 文件中，修改以下参数：
	`DZ_APPLICATION_NAME - Name of the application.`
	`DZ_STAGE_NAME - Name of the stage.`
	`DZ_DOMAIN_NAME - Name of the Amazon DataZone domain`
	`DZ_DOMAIN_DESCRIPTION - Description of the Amazon DataZone domain`
	`DZ_DOMAIN_TAG - Tag of the Amazon DataZone domain`
		云架构师、 DevOps 工程师

Task	描述	所需技能
	DZ_ADMIN_PROJECT_NAM E - Name of the Amazon DataZone project for administrators DZ_ADMIN_PROJECT_DESC RIPTION - Description of the Amazon DataZone project for administr ators CDK_EXEC_ROLE_ARN - ARN of the cdk execution role DZ_ADMIN_ROLE_ARN - ARN of the administr ator role 将其余参数留空。	

Task	描述	所需技能
更新 Amazon DataZone 词汇表配置。	要更新`lib/utils/glossary_config.json` 文件中的 Amazon DataZone 词汇表配置，请使用以下示例配置： `{ "GlossaryName": "PII Data", "GlossaryDescription": "If data source contains PII attributes", "GlossaryTerms": [{ "Name": "Yes", "ShortDescription": "Yes", "LongDescription": "Yes Glossary Term" }, { "Name": "No", "ShortDescription": "No", "LongDescription": "No Glossary Term" } ] }`	云架构师、 DevOps 工程师

Task	描述	所需技能
更新 Amazon DataZone 元数据表单配置。	要更新中的 Amazon DataZone 元数据表单配置`lib/utils/metadata_form_config.json file` ，请使用以下示例配置： `{ "FormName": "ScheduleDataRefresh", "FormDescription": "Form for data refresh schedule", "FormSmithyModel": "@amazon.datazone#displayname(defaultName: \"Data Refresh Schedule\")\nstructure ScheduleDataRefresh {\n @documentation(\"Schedule of Data Refresh\")\n @required\n @amazon.datazone#searchable\n @amazon.datazone#displayname(defaultName: \"Data Refresh Schedule\")\n data_refresh_schedule: String\n}" }`	云架构师、 DevOps 工程师

Task	描述	所需技能
导出 AWS 凭证。	要将具有管理权限的 IAM 角色的 AWS 证书导出到您的开发环境，请使用以下格式： `export AWS_ACCES S_KEY_ID= export AWS_SECRE T_ACCESS_KEY= export AWS_SESSI ON_TOKEN=`	云架构师、 DevOps 工程师
合成模板。	要合成 AWS CloudFormation 模板，请运行以下命令： `npx cdk synth`	云架构师、 DevOps 工程师
部署解决方案。	要部署解决方案，请运行以下命令： `npx cdk deploy --all`	云架构师、 DevOps 工程师

配置新的成员账户

Task	描述	所需技能
部署 模板。	使用以下输入参数部署`lib/cfn-templates/DzDataMeshCfnStackSetExecutionRole.yaml` 位于成员账户中的 AWS CloudFormation 模板： • `GovernanceAccountID` – 治理账户的账户 ID	云架构师、 DevOps 工程师

Task	描述	所需技能
	• `DataZoneKMSKeyID` -用于加密 Amazon DataZone 元数据的 AWS Key Management Service (AWS KMS) 密钥的 ID • `NotificationQueueName` – 管理账户中的 Amazon SQS 通知队列的名称	
更新 ARNs.	要更新成员账户的 AWS CloudFormation StackSet 执行角色 ARNs 列表，请使用以下代码： `DZ_MEMBER_STACK_SET_EXEC_ROLE_LIST - List of Stack set execution role arns for the member accounts.`	云架构师、 DevOps 工程师

合成并部署。	要合成 AWS CloudFormation 模板并部署解决方案，请运行以下命令：
	`npx cdk synth`
	`npx cdk deploy --all`
		云架构师、 DevOps 工程师

Task	描述	所需技能
关联成员账户。	要将成员账户与中央治理账户关联，请执行以下操作： 1. 登录中央治理账户的主机，然后在以下网址打开 Amazon DataZone 控制台https://console.aws.amazon.com/datazone/。 2. 选择您创建的域。 3. 滚动到 “关联账户” 选项卡，然后选择 “请求关联”。 4. 提供 AWS 账户 ID，然后选择AWSRAMPermissionDataZonePortalReadWrite作为 RAM 策略。 5. 选择 “请求关联”。 6. 请等待，直到您收到一封电子邮件通知，告知您的帐户已成功启动。	云架构师、 DevOps 工程师

Task	描述	所需技能
更新参数。	要更新配置文件中特定于成员账户的参数`config/Config.ts` ，请使用以下格式： `export const DZ_MEMBER_ACCOUNT_CONFIG: memberAccountConfig = { '123456789012' : { PROJECT_NAME: 'TEST-PROJECT-123456789012', PROJECT_DESCRIPTION: 'TEST-PROJECT-123456789012', PROJECT_EMAIL: 'user@xyz.com' } }`	云架构师、 DevOps 工程师

合成并部署模板。	要合成 AWS CloudFormation 模板并部署解决方案，请运行以下命令：
	
	`npx cdk synth`
	`npx cdk deploy --all`
		云架构师、 DevOps 工程师

添加成员账户。	要在数据解决方案中创建和配置其他成员帐户，请对每个成员帐户重复上述步骤。 该解决方案不区分数据生产者和消费者。	云架构师、 DevOps 工程师

清理

Task	描述	所需技能
取消关联成员账户。	要取消关联账户，请执行以下操作： 1. 登录控制台并打开 Amazon DataZone 控制台。 2. 选择查看域名。 3. 选择您创建的域。 4. 选择 “账户关联” 选项卡。 5. 选择要取消关联的成员账户。 6. 选择 “取消关联”，然后输入`disassociate` 进行确认。 7. 对所有成员账户重复步骤 3-6。	云架构师、 DevOps 工程师
删除堆栈实例。	要删除 AWS CloudFormation 堆栈实例，请执行以下操作： 1. 打开 AWS CloudFormation 控制台，网址为https://console.aws.amazon.com/cloudformation/。 2. 在导航窗格中，选择 StackSets。 3. 选择名为 StackSet-DataZone-DataMesh-Member 的堆栈集，然后选择堆栈实例选项卡。 4. 复制您要从成员资格中移除的成员账户的 AWS 账户 ID。	云架构师、 DevOps 工程师

Task	描述	所需技能
	5. 选择 “操作”，选择 “从中删除堆栈” StackSet，然后保留默认选项。 6. 在账号字段中，输入账户 ID。 7. 在 “指定区域” 下拉列表中，选择 AWS 区域。 8. 选择 “下一步”，然后选择 “提交”。 9. 在 “操作” 选项卡上，确认操作已成功。删除堆栈可能需要一些时间。 10对所有成员账户重复步骤 2-9。	
摧毁所有资源。	要销毁资源，请在本地开发环境（Linux 或 macOS）中执行以下步骤： 1. 导航到存储库的根目录。 2. 导出您用于创建 AWS CDK 堆栈的 IAM 角色的 AWS 证书。 3. 要销毁云资源，请运行以下命令： `npx cdk destroy --all`	云架构师、 DevOps 工程师

相关资源

- Amazon DataZone 数据快速入 AWS Glue 门
- 教程：创建你的第一个 AWS CDK 应用程序
- 入门 AWS CloudFormation

- 在上构建基于数据网格的企业解决方案的策略 AWS

其他信息

目标

实现此模式可实现以下效果：

- 分散的数据所有权——将数据所有权从中央团队转移到代表组织源系统、业务部门或用例的团队。
- 产品思维 – 在考虑组织中的数据资产时，引入一种基于产品的思维方式，包括客户、市场和其他因素。
- 联合治理-改善组织数据产品的安全护栏、控制和合规性。
- 多账户和多项目支持 – 支持组织各业务部门或项目之间高效、安全的数据共享和协作。
- 集中监控和通知-使用 Amazon 监控数据网格的云资源 CloudWatch，并在关联新成员账户时通知用户。
- 可扩展性和可扩展性 – 随着组织的发展，向数据网格中添加新的用例。

解决方案范围

使用此解决方案时，您可以从小规模开始，并随着数据网格之旅的进展进行扩展。通常，当成员账户采用数据解决方案时，它包含特定于组织、项目或业务部门的账户配置。该解决方案通过支持以下功能来适应这些不同的 AWS 账户 配置：

- AWS Glue 数据目录作为亚马逊的数据源 DataZone
- 管理 Amazon DataZone 数据域和相关数据门户
- 管理在基于数据网格的数据解决方案中添加成员账户
- Amazon DataZone 项目和环境的管理
- 管理 Amazon DataZone 词汇表和元数据表单
- 管理与基于数据网格的数据解决方案用户相对应的 IAM 角色
- 通知基于数据网格的数据解决方案用户
- 监控已配置的云基础架构

  该解决方案使用 AWS CDK 和 AWS CloudFormation 来部署云基础架构。它 AWS CloudFormation 用于执行以下操作：

  - 在较低的抽象级别上定义和部署云资源。

- 部署来自的云资源 AWS Management Console。通过使用这种方法，您可以在没有开发环境的情况下部署基础架构。

数据网格解决方案用于 AWS CDK 在更高的抽象级别上定义资源。因此，该解决方案通过选择相关工具来部署云资源，提供了一种解耦的、模块化和可扩展的方法。

后续步骤

您可以联系 AWS专家，获取有关使用 Amazon 构建数据网格的指导 DataZone。

该解决方案的模块化性质支持使用不同的架构（例如数据结构和数据湖）构建数据管理解决方案。此外，根据贵组织的要求，您可以将解决方案扩展到其他 Amazon DataZone 数据源。

通过 Amazon Web Services 计算风险价值 (VaR)

由 Sumon Samanta (AWS) 编写

摘要

此示例介绍了如何使用 Amazon Web Services 实现风险价值 (VaR) 计算系统。在本地环境中，大多数 VaR 系统使用大型专用基础设施以及内部或商用电网调度软件来运行批处理流程。这种模式提供了一种简单、可靠且可扩展的架构，用于处理 Amazon Web Services Cloud 中的 VaR 处理。它构建了一个无服务器架构，该架构使用亚马逊Kinesis Data Streams作为流媒体服务，使用亚马逊简单队列服务 (Amazon SQS) Simple Queue Service作为托管队列服务，使用 ElastiCache 亚马逊作为缓存服务，使用AWS Lambda来处理订单和计算风险。

VaR 是一种统计指标，交易者和风险经理使用它来估算其投资组合中超过一定信心水平的潜在损失。大多数 VaR 系统都需要运行大量数学和统计计算并存储结果。这些计算需要大量计算资源，因此 VaR 批处理过程必须分解为较小的计算任务集。可以将大批次拆分为较小的任务，因为这些任务大多是独立的（也就是说，一项任务的计算不依赖于其他任务）。

VaR 架构的另一重要要求是计算可扩展性。这种模式采用无服务器架构，可根据计算负载自动向内或向外扩展。由于批处理或在线计算需求难以预测，因此需要动态扩展才能在服务水平协议（SLA）规定的时间计划内完成该过程。此外，成本优化架构应该能够在每个计算资源的任务完成后，立即缩减该资源的规模。

Amazon Web Services 非常适合 VaR 计算，因为它们提供可扩展计算和存储容量、用于以成本优化的方式处理的分析服务，以及用于运行风险管理工作流的不同类型的调度程序。此外，您只需为在 AWS 上使用的计算与存储资源付费。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 输入文件，具体取决于业务需求。典型的用例涉及以下输入文件：
  - 市场数据文件（输入到 VaR 计算引擎）
  - 交易数据文件（除非交易数据通过数据流传输）。
  - 配置数据文件（模型和其他静态配置数据）
  - 计算引擎模型文件（定量库）
  - 时间序列数据文件（用于历史数据，例如过去五年的股价）

- 如果市场数据或其他输入通过直播传入，则设置 Amazon Kinesis Data Streams，并将 Amazon Identity and Access Management (IAM) 权限配置为写入该流。

这种模式构建了一个架构，在此架构中，交易数据从交易系统写入到 Kinesis 数据流。您可以不使用 Storage 服务，而是将您的交易数据存储在小批量文件中，将它们存储在 Amazon Simple Storage Service (Amazon S3) 存储桶中，然后调用事件开始处理数据。

限制

- Kinesis 数据流排序在每个分片都有保证，因此不能保证写入多个分片的交易订单的交付顺序与写入操作的顺序相同。
- AWS Lambda 的运行时间限制目前为 15 分钟。（有关更多信息，请参阅 Lambda 常见问题。）

架构

目标架构

以下架构图介绍了风险评测系统的 Amazon Web Services 和工作流。

该图阐释了以下内容：

1. 交易从订单管理系统流入。
2. 票据头寸净额结算 Lambda 函数处理订单，并将每个指示器的合并消息写入到 Amazon SQS 中的风险队列中。
3. 风险计算引擎 Lambda 函数处理来自亚马逊 SQS 的消息，执行风险计算，并更新亚马逊风险缓存中的 VaR 损益 (PnL) 信息。 ElastiCache
4. 读取 ElastiCache 数据 Lambda 函数从中检索风险结果并将其存储在数据库 ElastiCache 和 S3 存储桶中。

有关这些服务和步骤的更多信息，请参见操作说明部分。

自动化和扩缩

您可以使用 AWS Cloud Development Kit (AWS CDK) 或 AWS CloudFormation 模板部署整个架构。该架构可以支持批处理和盘中（实时）处理。

架构内置了扩展功能。随着越来越多的交易被写入 Kinesis 数据流并等待处理，可以调用其他 Lambda 函数处理这些交易，然后可以在处理完成后缩小规模。也可以选择通过多个 Amazon SQS 风险计算队列处理。如果需要对队列进行严格的排序或者整合，则无法并行处理。但是，对于 end-of-the-day批次或小型盘中批次，Lambda 函数可以并行处理并将最终结果存储在中。 ElastiCache

工具

Amazon Web Services

- Amazon Aurora MySQL 兼容版是一个完全托管的、与 MySQL 兼容的关系数据库引擎，可帮助您建立、运行和扩展 MySQL 部署。此示例以 MySQL 为例，但您可以使用任何 RDBMS 系统来存储数据。
- Amazon ElastiCache 可帮助您在 AWS 云中设置、管理和扩展分布式内存缓存环境。
- Amazon Kinesis Data Streams 可帮助您实时收集和处理大型数据记录流。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Queue Service (Amazon SQS) 提供了一个安全、持久且可用的托管队列，它可帮助您集成和分离分布式软件系统与组件。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

代码

此模式为 Amazon Web Services Cloud 中的 VaR 系统提供了示例架构，并描述了如何使用 Lambda 函数进行 VaR 计算。要创建您的 Lambda 函数，请参阅 Lambda 文档中的代码示例。如果需要帮助，请联系 AWS Professional Services。

最佳实践

- 让每个 VaR 计算任务尽可能小巧。在每个计算任务中尝试不同数量交易，看看哪个任务在计算时间和成本方面最为优化。
- 在 Amazon 中存储可重复使用的对象 ElastiCache。使用 Apache Arrow 等框架来减少序列化和反序列化。
- 考虑 Lambda 的时间限制。如果您认为自己的计算任务可能会超过 15 分钟，请尝试将其分解为较小的任务，以避免 Lambda 超时。如果无法实现，您可以考虑使用 AWS Fargate、Amazon Elastic Container Service (Amazon ECS) 以及 Amazon Elastic Kubernetes Service (Amazon EKS) 等容器编排解决方案。

操作说明

交易流向风险系统

Task	描述	所需技能
开始写入交易。	新建、已结算或部分结算的交易从订单管理系统写入风险流。这种模式使用 Amazon Kinesis 作为托管流媒体服务。交易订单代码哈希值用于在多个分片上放置交易订单。	Amazon Kinesis

运行 Lambda 函数进行订单处理

Task	描述	所需技能
使用 Lambda 开始执行风险处理。	为新订单运行 AWS Lambda 函数。根据待处理交易订单数量，Lambda 将自动扩缩。每个 Lambda 实例都有一个或多个订单，并从亚马逊检索每张股票的最新头寸。ElastiCache（您可以使用CUSIP ID、Curve名称或其他金融衍生产品的指数名称作为存储和检索数据的密钥 ElasticCache。）在中 ElastiCache，总持仓（数量）和键值对 < 股票代码、净头寸 >（其中净持仓是缩放因子）为每个股票更新一次。	亚马逊 Kinesis、AWS Lambda、亚马逊 ElastiCache

将每个指示器消息写入队列

Task	描述	所需技能
将合并的消息写入至风险队列。	将消息写入至队列。此模式使用 Amazon SQS 提供托管队列服务。单个 Lambda 实例可能在任何给定时间收到一小批交易订单，但只会为每个指示器写一条消息至 Amazon SQS。计算比例系数：（旧净头寸 + 当前头寸）/旧净头寸。	Amazon SQS、AWS Lambda

调用风险引擎

Task	描述	所需技能
开始风险计算。	调用风险引擎 lambda 的 Lambda 函数。每个位置都由单个 Lambda 函数处理。但是，出于优化目的，每个 Lambda 函数都可以处理来自 Amazon SQS 的多条消息。	Amazon SQS、AWS Lambda

在缓存中检索风险结果

Task	描述	所需技能
检索与更新风险缓存。	Lambda 从中检索每个股票的当前净头寸。 ElastiCache它还会从中检索每个股票的VaR损益 (PnL) 数组。 ElastiCache 如果 pNL 数组已经存在，则 Lambda 函数会使用扩展更新数组和 VaR，该扩展来自净	亚马逊 SQS、AWS Lambda、亚马逊 ElastiCache

Task	描述	所需技能
	额结算 Lambda 函数所编写的 Amazon SQS 消息。如果盈亏数组不存在 ElasticCache，则使用模拟股票价格序列数据计算出新的盈亏和VaR。	

更新 Elastic Cache 中的数据，并存储在数据库中

Task	描述	所需技能
存储风险结果。	更新 VaR 和 PnL 数字后 ElastiCache，每五分钟调用一次新的 Lambda 函数。此函数读取所有存储的数据，并将其存储在兼容 Aurora MySQL 的数据库和 S3 存储桶中。ElastiCache	AWS Lambda、亚马逊 ElastiCache

相关资源

- Basel VaR Framework

使用 Amazon Athena 来配置对共享 AWS Glue Data Catalog 的跨账户存取

创建者：Denis Avdonin (AWS)

摘要

此模式提供了使用AWS Glue数据目录配置存储在亚马逊简单存储服务 (Amazon S3) 存储桶中的数据集的跨账户共享的 step-by-step说明，包括AWS Identity and Access Management (IAM) 策略示例。您可将数据集存储在 S3 存储桶中。元数据由 AWS Glue 爬网程序收集并放入 AWS Glue Data Catalog 中。S3 存储桶和 AWS Glue Data Catalog 位于称为数据账户的 Amazon Web Services account 中。您可向另一个 Amazon Web Services account（称为使用者账户）中的 IAM 主体提供访问权限。用户可以使用 Amazon Athena 无服务器查询引擎查询使用者账户中的数据。

先决条件和限制

先决条件

- 两个活动 Amazon Web Services account
- 一个 Amazon Web Services account 中的 S3 存储桶
- Athena 引擎版本 2
- 已安装并配置 AWS 命令行接口 (AWS CLI) Line In terface（或者用于运行 AW CloudShell S CLI 命令的 AWS）

产品版本

此模式仅适用于 Athena 引擎版本 2 和 Athena 引擎版本 3。我们建议您升级到 Athena 引擎版本 3。如果您无法从 Athena 引擎版本 1 升级到 Athena 引擎版本 3，请按照 AWS 大数据博客中使用亚马逊 AWS Athena 跨账户 AWS Glue 数据目录访问中讨论的方法进行操作。

架构

目标技术堆栈

- Amazon Athena
- Amazon Simple Storage Service（Amazon S3）

- AWS Glue
- AWS Identity and Access Management (IAM)
- AWS Key Management Service (AWS KMS)

下图显示了一种架构，该架构使用 IAM 权限通过 AWS Glue Data Catalog 与另一个 Amazon Web Services account（使用者账户）共享一个 Amazon Web Services account（数据账户）中 S3 存储桶中的数据。

图表显示了以下工作流：

1. 数据账户中的 S3 存储桶策略向使用者账户中的 IAM 角色以及数据账户中的 AWS Glue 爬网程序服务角色授予权限。
2. 数据账户中的 AWS KMS 密钥政策向使用者账户中的 IAM 角色以及数据账户中的 AWS Glue 爬网程序服务角色授予权限。
3. 数据账户中的 AWS Glue 爬网程序会发现存储在 S3 存储桶中的数据的架构。
4. 数据账户中 AWS Glue Data Catalog 的资源策略授予对使用者账户中 IAM 角色的访问权限。
5. 用户通过 AWS CLI 命令在使用者账户中创建命名目录引用。
6. IAM policy 授予使用者账户中的 IAM 角色对数据账户中资源的访问权限。IAM 角色的信任策略允许消费者账户中的用户担任 IAM 角色。
7. 使用者账户中的用户承担 IAM 角色并使用 SQL 查询访问数据目录中的对象。
8. Athena 无服务器引擎运行 SQL 查询。

> **Note**
>
> IAM 最佳实践建议您向 IAM 角色授予权限并使用联合身份验证。

工具

- Amazon Athena 是一种交互式查询服务，它可帮助您通过使用标准 SQL 直接在 Amazon S3 中分析数据。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

- AWS Glue 是一项完全托管的提取、转换、加载（ETL）服务。它可以帮助您在数据存储和数据流之间对数据进行可靠地分类、清理、扩充和移动。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥以保护您的数据。

操作说明

设置数据账户权限

Task	描述	所需技能
授予对 S3 存储桶中数据的访问权限。	根据以下模板创建 S3 存储桶策略，并将该策略分配给存储数据的存储桶。 `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "AWS": [ "arn:aws:iam::<consumer account id>:role/<role name>", "arn:aws:iam::<data account id>:role/service-role/AWSGlueServiceRole-data-bucket-crawler" ] }, "Action": "s3:GetObject",`	云管理员

Task	描述	所需技能
	"Resource": "arn:aws:s3:::data-bucket/*" }, { "Effect": "Allow", "Principal": { "AWS": ["arn:aws:iam::<consumer account id>:role/<role name>", "arn:aws:iam::<data account id>:role/service-role/AWSGlueServiceRole-data-bucket-crawler"] }, "Action": "s3:ListBucket", "Resource": "arn:aws:s3:::data-bucket" }] } 存储桶策略向使用者账户中的 IAM 角色和数据账户中的 AWS Glue 爬网程序服务角色授予权限。	

Task	描述	所需技能
(如果需要)授予对数据加密密钥的访问权限。	如果 S3 存储桶由 AWS KMS 密钥加密，请向使用者账户中的 IAM 角色和数据账户中的 AWS Glue 爬网程序服务角色授予密钥 `kms:Decrypt` 权限。 使用以下语句更新密钥政策： `{ "Effect": "Allow", "Principal": { "AWS": [ "arn:aws:iam::<consumer account id>:role/<role name>", "arn:aws:iam::<data account id>:role/service-role/AWSGlueServiceRole-data-bucket-crawler" ] }, "Action": "kms:Decrypt", "Resource": "arn:aws:kms:<region>:<data account id>:key/<key id>" }`	云管理员

Task	描述	所需技能
授予爬网程序对数据的访问权限。	将以下 IAM policy 附加到爬网程序的服务角色： `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": "s3:GetObject", "Resource": "arn:aws:s3:::data-bucket/*" }, { "Effect": "Allow", "Action": "s3:ListBucket", "Resource": "arn:aws:s3:::data-bucket" } ] }`	云管理员

Task	描述	所需技能

（如果需要）授予爬网程序访问数据加密密钥的权限。	如果 S3 存储桶由 AWS KMS 密钥加密，则通过向爬网程序的服务角色附加以下策略，向其授予密钥 `kms:Decrypt` 权限：
	
	`{`
	`  "Effect": "Allow",`
	`  "Action": "kms:Decrypt",`
	`  "Resource": "arn:aws:kms:<region>:<data account id>:key/<key id>"`
	`}`
		云管理员

Task	描述	所需技能

向使用者账户中的 IAM 角色和爬网程序授予对数据目录的访问权限。	1. 登录 AWS 管理控制台，然后打开 AWS Glue 控制台。
	2. 在导航窗格的数据目录下方，选择设置。
	3. 在权限部分中添加以下语句，然后选择保存。
	
	{
	"Version" :
	"2012-10-17",
	"Statement" : [
	{
	"Effect" :
	"Allow",
	"Principa
	l" : {
	"AWS" :
	[
	"arn:aws:iam::<con
	sumer account id>:role/
	<role name>",
	"arn:aws:iam::<dat
	a account id>:role/
	service-role/AWSGl
	ueServiceRole-data-
	bucket-crawler"
	]
	},
	"Action" :
	"glue:*",
	"Resource
	" : [
	"arn:aws:glue:<reg
	ion>:<data account
	id>:catalog",
		云管理员

Task	描述	所需技能

	<pre>"arn:aws:glue:<reg
	ion>:<data account
	id>:database/*",
	
	"arn:aws:glue:<reg
	ion>:<data account
	id>:table/*"
	]
	}
	]
	}</pre> 此策略允许对数据账户中的所有数据库和表执行所有 AWS Glue 操作。您可对策略进行自定义，使其仅向使用者主体授予所需权限。例如，您可提供对数据库中特定表或视图的只读访问权限。	

通过使用者账户访问数据

Task	描述	所需技能

数据目录命名引用。	要创建命名的数据目录引用，请使用CloudShell或本地安装的 AWS CLI 运行以下命令：
	
	<pre>aws athena create-da
	ta-catalog --name
	<shared catalog name>
	--type GLUE --paramet
	ers catalog-id=<data
	account id></pre>
		云管理员

Task	描述	所需技能
授予使用者账户中的 IAM 角色对数据的访问权限。	将以下策略附加到使用者账户中的 IAM 角色，以授予该角色跨账户存取数据的权限： { "Version": "2012-10-17", "Statement": [{ "Effect": "Allow", "Action": "s3:GetObject", "Resource": "arn:aws:s3:::data-bucket/*" }, { "Effect": "Allow", "Action": "s3:ListBucket", "Resource": "arn:aws:s3:::data-bucket" }, { "Effect": "Allow", "Action": "glue:*", "Resource": ["arn:aws:glue:<region>:<data account id>:catalog", "arn:aws:glue:<region>:<data account id>:database/*",	云管理员

Task	描述	所需技能

	<pre>"arn:aws:glue:<region>:<data account id>:table/*"
	            ]
	        }
	    ]
	}</pre>
	接下来，使用以下模板指定哪些用户可以按其信任策略接受 IAM 角色：
	<pre>{
	    "Version": "2012-10-17",
	    "Statement": [
	       {
	            "Effect": "Allow",
	            "Principal": {
	                "AWS": "arn:aws:iam::<consumer account id>:user/<IAM user>"
	            },
	            "Action": "sts:AssumeRole"
	        }
	    ]
	}</pre>
	最后，通过将相同的策略附加到用户组所属的用户组，授予其代入 IAM 角色的权限。	

Task	描述	所需技能
（如果需要）授予使用者账户中的 IAM 角色访问数据加密密钥的权限。	如果 S3 存储桶由 AWS KMS 密钥加密，请向使用者账户中的 IAM 角色授予密钥 `kms:Decrypt` 权限，方法是向该角色附加以下策略： `{ "Effect": "Allow", "Action": "kms:Decrypt", "Resource": "arn:aws:kms:<region>:<data account id>:key/<key id>" }`	云管理员
切换到使用者账户中的 IAM 角色以访问数据。	作为数据使用者，切换至 IAM 角色以访问数据账户中的数据。	数据使用者

Task	描述	所需技能

访问数据。	使用 Athena 查询数据。例如，打开 Athena 查询编辑器，并运行以下查询：
	
	`SELECT * FROM <shared catalog name>.<database name>.<table name>`
	
	您还可以使用 Amazon 资源名称（ARN）引用该目录，而不必使用命名目录引用。
	
	Note
	如果您在查询或视图中使用动态目录引用，请用转义的双引号 (\") 将该引用括起来。例如：
	
	`SELECT * FROM \"glue:arn:aws:glue:<region>:<data account id>:catalog\".<database name>.<table name>`
	
	有关更多信息，请参阅 Amazon Athena 用户指南中的跨账户存取 AWS Glue Data Catalog。
		数据使用者

相关资源

- 跨账户存取 AWS Glue Data Catalog（Athena 文档）
- (AWS CLI) create-data-catalog（AWS CLI 命令参考）
- 使用 Amazon Athena 跨账户存取 AWS Glue Data Catalog（AWS 大数据博客）
- IAM 中的安全最佳实践（IAM 文档）

其他信息

使用 Lake Formation 作为共享跨账户替代方案

您还可使用 AWS Lake Formation 在账户之间共享对 AWS Glue 目录对象的访问权限。Lake Formation 提供列和行级别的精细访问控制、基于标签的访问控制、ACID 事务的受控表以及其他功能。尽管 Lake Formation 与 Athena 集成良好，但与此模式的仅 IAM 方法相比，它确实需要额外的配置。我们建议您在整体解决方案架构的更广泛背景下考虑决定使用 Lake Formation 或仅 IAM 访问控制。需要考虑的因素包括涉及哪些其他服务以及它们如何与这两种方法集成。

将 Teradata 标准化时态功能转换为 Amazon Redshift SQL

由 Po Hong (AWS) 编写

摘要

NORMALIZE 是 ANSI SQL 标准的 Teradata 扩展。当 SQL 表包含具有 PERIOD 数据类型的列时，NORMALIZE 会组合该列中相交或重叠的值，形成一个整合多个单独周期值的单个周期。要使用 NORMALIZE，SQL SELECT 列表中至少有一列必须是 Teradata 的时态周期数据类型。有关 NORMALIZE 的更多信息，请参阅 Teradata 文档。

Amazon Redshift 不支持 NORMALIZE，但您可以使用原生 SQL 语法和 Amazon Redshift 中的 LAG 窗口函数来实现此功能。这种模式侧重于使用带有 ON MEETS OR OVERLAPS 条件的 Teradata NORMALIZE 扩展，这是最流行的格式。它解释了该功能在 Teradata 中的工作原理，以及如何将其转换为 Amazon Redshift 原生 SQL 语法。

先决条件和限制

先决条件

- 基本 Teradata SQL 知识和经验
- Amazon Redshift 知识和经验

架构

源技术堆栈

- Teradata 数据仓库

目标技术堆栈

- Amazon Redshift

目标架构

有关将 Teradata 数据库迁移至 Amazon Redshift 的高级架构，请参阅使用 AWS SCT 数据提取代理将 Teradata 数据库迁移至 Amazon Redshift 模式。迁移不会自动将 Teradata NORMALIZE 短语转换为 Amazon Redshift SQL。您可按照此模式中的指导方针转换此 Teradata 扩展。

工具

代码

为了说明 NORMALIZE 的概念和功能，请考虑 Teradata 中的下表定义：

```
CREATE TABLE systest.project
     (    emp_id        INTEGER,
          project_name  VARCHAR(20),
          dept_id       INTEGER,
          duration      PERIOD(DATE)
     );
```

运行以下 SQL 代码，以将示例数据插入表中：

```
BEGIN TRANSACTION;

INSERT INTO systest.project VALUES (10, 'First Phase', 1000,  PERIOD(DATE '2010-01-10',
 DATE '2010-03-20') );
INSERT INTO systest.project VALUES (10, 'First Phase', 2000,  PERIOD(DATE '2010-03-20',
 DATE '2010-07-15') );

INSERT INTO systest.project VALUES (10, 'Second Phase', 2000,  PERIOD(DATE
 '2010-06-15', DATE '2010-08-18') );
INSERT INTO systest.project VALUES (20, 'First Phase', 2000,  PERIOD(DATE '2010-03-10',
 DATE '2010-07-20') );

INSERT INTO systest.project VALUES (20, 'Second Phase', 1000,  PERIOD(DATE
 '2020-05-10', DATE '2020-09-20') );

END TRANSACTION;
```

结果：

```
select * from systest.project order by 1,2,3;

 *** Query completed. 4 rows found. 4 columns returned.
 *** Total elapsed time was 1 second.

     emp_id  project_name              dept_id  duration
-----------  --------------------  -----------  ------------------------
         10  First Phase                  1000  ('10/01/10', '10/03/20')
```

```
         10  First Phase                  2000  ('10/03/20', '10/07/15')
         10  Second Phase                 2000  ('10/06/15', '10/08/18')
         20  First Phase                  2000  ('10/03/10', '10/07/20')
         20  Second Phase                 1000  ('20/05/10', '20/09/20')
```

Teradata NORMALIZE 用例

现在将 Teradata NORMALIZE SQL 子句添加至 SELECT 语句：

```
SELECT NORMALIZE ON MEETS OR OVERLAPS emp_id, duration
FROM systest.project
ORDER BY 1,2;
```

NORMALIZE 运算是在单列 (emp_id) 上执行的。对于 emp_id=10，持续时间中的三个重叠周期值合并至一个周期值，如下所示：

```
     emp_id  duration
-----------  ------------------------
         10  ('10/01/10', '10/08/18')
         20  ('10/03/10', '10/07/20')
         20  ('20/05/10', '20/09/20')
```

以下 SELECT 语句对 project_name 和 dept_id 执行 NORMALIZE 运算。请注意，SELECT 列表仅包含一个 PERIOD 列，即持续时间。

```
SELECT NORMALIZE project_name, dept_id, duration
FROM systest.project;
```

输出：

```
project_name              dept_id  duration
--------------------  -----------  ------------------------
First Phase                  1000  ('10/01/10', '10/03/20')
Second Phase                 1000  ('20/05/10', '20/09/20')
First Phase                  2000  ('10/03/10', '10/07/20')
Second Phase                 2000  ('10/06/15', '10/08/18')
```

Amazon Redshift 等效 SQL

Amazon Redshift 目前不支持表中的 PERIOD 数据类型。相反，您需要将 Teradata PERIOD 数据字段分为两部分：start_date、end_date，如下所示：

```
CREATE TABLE systest.project
     (    emp_id        INTEGER,
          project_name  VARCHAR(20),
          dept_id       INTEGER,
          start_date  DATE,
          end_date    DATE
     );
```

在表中插入示例数据：

```
BEGIN TRANSACTION;

INSERT INTO systest.project VALUES (10, 'First Phase', 1000,  DATE '2010-01-10', DATE
 '2010-03-20' );
INSERT INTO systest.project VALUES (10, 'First Phase', 2000,  DATE '2010-03-20', DATE
 '2010-07-15');

INSERT INTO systest.project VALUES (10, 'Second Phase', 2000,  DATE '2010-06-15', DATE
 '2010-08-18' );
INSERT INTO systest.project VALUES (20, 'First Phase', 2000,  DATE '2010-03-10', DATE
 '2010-07-20' );

INSERT INTO systest.project VALUES (20, 'Second Phase', 1000,  DATE '2020-05-10', DATE
 '2020-09-20' );

END TRANSACTION;
```

输出：

```
 emp_id | project_name | dept_id | start_date |  end_date
--------+--------------+---------+------------+------------
     10 | First Phase  |    1000 | 2010-01-10 | 2010-03-20
     10 | First Phase  |    2000 | 2010-03-20 | 2010-07-15
     10 | Second Phase |    2000 | 2010-06-15 | 2010-08-18
     20 | First Phase  |    2000 | 2010-03-10 | 2010-07-20
     20 | Second Phase |    1000 | 2020-05-10 | 2020-09-20
(5 rows)
```

要重写 Teradata 的 NORMALIZE 子句，您可以使用 Amazon Redshift 中的 LAG 窗口函数。此函数返回位于分区中当前行的上方（之前）的某个给定偏移量位置的行的值。

您可使用 LAG 函数通过确定一个时段是否与前一时段相交或重叠来识别开始新时段的每一行（如果是，则为 0，如果不是，则为 1）。累积汇总此标志后，它会提供组标识符，该标识符可用于外部 Group By 子句中，从而在 Amazon Redshift 中得出所需结果。

以下是使用 LAG() 的 Amazon Redshift SQL 语句示例：

```
SELECT emp_id, start_date, end_date,
            (CASE WHEN start_date <= LAG(end_date) OVER (PARTITION BY emp_id ORDER BY
 start_date, end_date) THEN 0 ELSE 1 END) AS GroupStartFlag
FROM systest.project
ORDER BY 1,2;
```

输出：

```
 emp_id | start_date |  end_date  | groupstartflag
--------+------------+------------+----------------
     10 | 2010-01-10 | 2010-03-20 |              1
     10 | 2010-03-20 | 2010-07-15 |              0
     10 | 2010-06-15 | 2010-08-18 |              0
     20 | 2010-03-10 | 2010-07-20 |              1
     20 | 2020-05-10 | 2020-09-20 |              1
(5 rows)
```

以下 Amazon Redshift SQL 语句仅在 emp_id 列上标准化：

```
SELECT T2.emp_id, MIN(T2.start_date) as new_start_date, MAX(T2.end_date) as
 new_end_date
FROM
( SELECT T1.*, SUM(GroupStartFlag) OVER (PARTITION BY emp_id ORDER BY start_date ROWS
 UNBOUNDED PRECEDING) As GroupID
FROM ( SELECT emp_id, start_date, end_date,
            (CASE WHEN start_date <= LAG(end_date) OVER (PARTITION BY emp_id ORDER BY
 start_date, end_date) THEN 0 ELSE 1 END) AS GroupStartFlag
FROM systest.project ) T1
) T2
GROUP BY T2.emp_id, T2.GroupID
ORDER BY 1,2;
```

输出：

```
 emp_id | new_start_date | new_end_date
```

```
--------+----------------+-------------------------------------
     10 | 2010-01-10     | 2010-08-18
     20 | 2010-03-10     | 2010-07-20
     20 | 2020-05-10     | 2020-09-20
(3 rows)
```

以下 Amazon Redshift SQL 语句在 project_name 和 dept_id 列上执行标准化：

```
SELECT T2.project_name, T2.dept_id, MIN(T2.start_date) as new_start_date,
 MAX(T2.end_date) as new_end_date
FROM
( SELECT T1.*, SUM(GroupStartFlag) OVER (PARTITION BY project_name, dept_id ORDER BY
 start_date ROWS UNBOUNDED PRECEDING) As GroupID
FROM ( SELECT project_name, dept_id, start_date, end_date,
            (CASE WHEN start_date <= LAG(end_date) OVER (PARTITION BY project_name,
 dept_id ORDER BY start_date, end_date) THEN 0 ELSE 1 END) AS GroupStartFlag
FROM systest.project ) T1
) T2
GROUP BY T2.project_name, T2.dept_id, T2.GroupID
ORDER BY 1,2,3;
```

输出：

```
 project_name | dept_id | new_start_date | new_end_date
--------------+---------+----------------+--------------
 First Phase  |    1000 | 2010-01-10     | 2010-03-20
 First Phase  |    2000 | 2010-03-10     | 2010-07-20
 Second Phase |    1000 | 2020-05-10     | 2020-09-20
 Second Phase |    2000 | 2010-06-15     | 2010-08-18
(4 rows)
```

操作说明

将 NORMALIZE 转换为 Amazon Redshift SQL

Task	描述	所需技能
创建您的 Teradata SQL 代码。	根据自身需求使用 NORMALIZE 短语。	SQL Developer

Task	描述	所需技能
将代码转换为 Amazon Redshift SQL。	若要转换您的代码，请按照此模式的“工具”部分中的指南进行操作。	SQL Developer
在 Amazon Redshift 中运行代码。	创建您的表，将数据加载至表中，然后在 Amazon Redshift 中运行您的代码。	SQL Developer

相关资源

参考

- Teradata NORMALIZE 时态功能（Teradata 文档）
- LAG 窗口函数（Amazon Redshift 文档）
- 迁移至 Amazon Redshift（AWS 网站）
- 使用 AWS SCT 数据提取代理，将 Teradata 数据库迁移至 Amazon Redshift (AWS Prescriptive Guidance)
- 将 Teradata RESET WHEN 功能转换为 Amazon Redshift SQL (AWS Prescriptive Guidance)

工具

- AWS Schema Conversion Tool (AWS SCT)

合作伙伴

- AWS Migration Competency Partners

将 Teradata RESET WHEN 功能转换为 Amazon Redshift SQL

由 Po Hong (AWS) 编写

摘要

RESET WHEN是 SQL 分析窗口函数中使用的 Teradata 功能。它是 ANSI SQL 标准的扩展。RESET WHEN 根据某些指定条件确定 SQL 窗口函数在其上运行的分区。如果条件计算结果为 TRUE，则会在现有窗口分区内创建新的动态子分区。有关 RESET WHEN 的更多信息，请参阅 Teradata 文档。

Amazon Redshift 不支持 SQL 窗口函数中的 RESET WHEN。若要实现此功能，您必须将 RESET WHEN 转换为 Amazon Redshift 中的原生 SQL 语法，并使用多个嵌套函数。此模式演示了如何使用 Teradata RESET WHEN 功能以及如何将其转换为 Amazon Redshift SQL 语法。

先决条件和限制

先决条件

- Teradata 数据仓库及其 SQL 语法的基础知识
- 对 Amazon Redshift 及其 SQL 语法有很好的了解

架构

源技术堆栈

- Teradata 数据仓库

目标技术堆栈

- Amazon Redshift

架构

有关将 Teradata 数据库迁移至 Amazon Redshift 的高级架构，请参阅使用 AWS SCT 数据提取代理将 Teradata 数据库迁移至 Amazon Redshift模式。迁移不会自动将 Teradata NORMALIZE 短语转换为 Amazon Redshift SQL。您可以按照下一节中的指导原则转换此 Teradata 扩展。

工具

代码

为了说明 RESET WHEN 的概念，请考虑 Teradata 中的下表定义：

```
create table systest.f_account_balance
( account_id integer NOT NULL,
  month_id integer,
  balance integer )
unique primary index (account_id, month_id);
```

运行以下 SQL 代码，以将示例数据插入表中：

```
BEGIN TRANSACTION;
Insert Into systest.f_account_balance values (1,1,60);
Insert Into systest.f_account_balance values (1,2,99);
Insert Into systest.f_account_balance values (1,3,94);
Insert Into systest.f_account_balance values (1,4,90);
Insert Into systest.f_account_balance values (1,5,80);
Insert Into systest.f_account_balance values (1,6,88);
Insert Into systest.f_account_balance values (1,7,90);
Insert Into systest.f_account_balance values (1,8,92);
Insert Into systest.f_account_balance values (1,9,10);
Insert Into systest.f_account_balance values (1,10,60);
Insert Into systest.f_account_balance values (1,11,80);
Insert Into systest.f_account_balance values (1,12,10);
END TRANSACTION;
```

该示例表具有以下数据：

account_id	month_id	balance
1	1	60
1	2	99
1	3	94
1	4	90
1	5	80

1	6	88
1	7	90
1	8	92
1	9	10
1	10	60
1	11	80
1	12	10

对于每个账户，假设您想要分析每月余额连续增加的顺序。当一个月的余额小于或等于上个月的余额时，要求将计数器重置为零，并重新启动。

Teradata RESET WHEN 用例

若要分析此数据，Teradata SQL 使用了带有嵌套聚合和 RESET WHEN 短语的窗口函数，如下所示：

```
SELECT account_id, month_id, balance,
 ( ROW_NUMBER() OVER (PARTITION BY account_id ORDER BY month_id
RESET WHEN balance <= SUM(balance) over (PARTITION BY account_id ORDER BY month_id ROWS
 BETWEEN 1 PRECEDING AND 1 PRECEDING) ) -1 ) as balance_increase
FROM systest.f_account_balance
ORDER BY 1,2;
```

输出：

account_id	month_id	balance	balance_increase
1	1	60	0
1	2	99	1
1	3	94	0
1	4	90	0
1	5	80	0

1	6	88	1
1	7	90	2
1	8	92	3
1	9	10	0
1	10	60	1
1	11	80	2
1	12	10	0

在 Teradata 中，查询处理方式如下：

1. SUM（余额）汇总函数计算给定账户在给定月份内所有余额的总和。
2. 我们会检查给定月份（给定账户）的余额是否大于上个月的余额。
3. 如果余额增加，我们将跟踪累积计数值。如果 RESET WHEN 条件的计算结果为 false，这意味着余额在连续几个月中有所增加，则我们会继续增加计数。
4. ROW_NUMBER () 有序分析函数计算计数值。当我们一个月的余额小于或等于前一个月的余额时，RESET WHEN 条件的计算结果为 true。如果是这样，我们启动一个新分区，ROW_NUMBER() 从 1 重新开始计数。我们使用 ROWS BETWEEN 1 PRECEDING AND 1 PRECEDING 来访问前一行的值。
5. 我们减去 1，以确保计数值以 0 开头。

Amazon Redshift 等效 SQL

Amazon Redshift 不支持 SQL 分析窗口函数中的 RESET WHEN 短语。 若要生成相同的结果，您必须使用 Amazon Redshift 原生 SQL 语法和嵌套子查询重写 Teradata SQL，如下所示：

```
SELECT account_id, month_id, balance,
   (ROW_NUMBER() OVER(PARTITION BY account_id, new_dynamic_part ORDER BY month_id) -1)
 as balance_increase
FROM
( SELECT account_id, month_id, balance, prev_balance,
SUM(dynamic_part) OVER (PARTITION BY account_id ORDER BY month_id ROWS BETWEEN
 UNBOUNDED PRECEDING AND CURRENT ROW) As new_dynamic_part
FROM ( SELECT account_id, month_id, balance,
```

```
SUM(balance) over (PARTITION BY account_id ORDER BY month_id ROWS BETWEEN 1 PRECEDING
 AND 1 PRECEDING) as prev_balance,
(CASE When balance <= prev_balance Then 1 Else 0 END) as dynamic_part
FROM systest.f_account_balance ) A
) B
ORDER BY 1,2;
```

由于 Amazon Redshift 不支持单个 SQL 语句的 SELECT 子句中的嵌套窗口函数，因此您必须使用两个嵌套的子查询。

- 在内部子查询(别名 A)中，创建并填充动态分区指示器 (dynamic_part)。如果一个月的余额小于或等于前一个月的余额，则 dynamic_part 设置为 1；否则，设置为 0。
- 在下一层(别名 B)中，将生成 new_dynamic_part 属性作为 SUM 窗口函数的结果。
- 最后，将 new_dynamic_part 作为新的分区属性 (dynamic partition) 添加到现有分区属性 (account_id) 中，并应用与 Teradata 中相同的 ROW_NUMBER() 窗口函数(减一)。

进行上述更改后，Amazon Redshift SQL 生成的输出与 Teradata 相同。

操作说明

将 RESET WHEN 转换为 Amazon Redshift SQL

Task	描述	所需技能
创建 Teradata 窗口函数。	根据需求使用嵌套聚合与 RESET WHEN 短语。	SQL Developer
将代码转换为 Amazon Redshift SQL。	若要转换您的代码，请按照此模式的“工具”部分中的指南进行操作。	SQL Developer
在 Amazon Redshift 中运行代码。	创建您的表，将数据加载至表中，然后在 Amazon Redshift 中运行您的代码。	SQL Developer

相关资源

参考

- RESET WHEN 短语（Teradata 文档）
- RESET WHEN 解释（堆栈溢出）
- 迁移至 Amazon Redshift（AWS 网站）
- 使用 AWS SCT 数据提取代理，将 Teradata 数据库迁移至 Amazon Redshift (AWS Prescriptive Guidance)
- 将 Teradata RESET WHEN 时态功能转换至 Amazon Redshift SQL (AWS Prescriptive Guidance)

工具

- AWS Schema Conversion Tool (AWS SCT)

合作伙伴

- AWS Migration Competency Partners

使用基础设施即代码，在 Amazon Web Services Cloud 上部署和管理无服务器数据湖

由 Kirankumar Chandrashekar (AWS) 和 Abdel Jaidi (AWS) 创建

摘要

注意： CodeCommit AWS 不再向新客户开放。AWS 的现有客户 CodeCommit 可以继续照常使用该服务。了解更多

此示例介绍了如何使用无服务器计算和基础设施即代码（IaC）在 Amazon Web Services (AWS) Cloud 上实施和管理数据湖。这种模式基于 AWS 开发的无服务器数据湖框架 (SDLF) 研讨会。

SDLF 是一组可重复使用的资源，可加速企业数据湖在 Amazon Web Services Cloud 上的交付，并有助于更快地部署到生产环境。它用于通过遵循最佳实践来实施数据湖的基础结构。

SDLF 使用 AWS、AWS 和 AWS 等 CodePipeline AWS 服务，在整个代码和基础设施部署过程中实施持续集成/持续部署 (CI/CD) 流程。 CodeBuild CodeCommit

此模式使用多个 AWS 无服务器服务来简化数据湖管理。其中包括用于存储的亚马逊简单存储服务 (Amazon S3) 和亚马逊 DynamoDB、用于计算的 AWS Lambda 和 AWS Glue，以及用于编排的亚马逊活动、亚马逊简单队列服务 (Amazon SQS) Simple Queue Service CloudWatch 和 AWS Step Functions。

AWS CloudFormation 和 AWS 代码服务充当 IaC 层，提供可重现且快速的部署，且操作和管理非常简单。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 已安装并配置 AWS 命令行界面（AWS CLI）。
- Git 客户端，已安装和配置。
- SDLF 研讨会，在 Web 浏览器窗口中打开并随时可以使用。

架构

该架构图通过以下步骤说明了事件驱动的流程。

1. 将文件添加到原始数据 S3 存储桶后，Amazon S3 事件通知将放置在 SQS 队列中。每个通知均以 JSON 文件形式传递，其中包含 S3 存储桶名称、对象键或时间戳等元数据。
2. 此通知由 Lambda 函数使用，该函数根据元数据将事件路由到正确的提取、转换和加载 (ETL) 流程。Lambda 函数还可以使用存储在 Amazon DynamoDB 表中的上下文配置。此步骤可以解耦并扩展到数据湖中的多个应用程序。
3. 该事件被路由至 ETL 流程中的第一个 Lambda 函数，该函数将数据从原始数据区域转换并移动到数据湖的暂存区域。第一步是更新综合目录。这是 DynamoDB 表，其中包含数据湖的所有文件元数据。此表中的每一行都包含关于存储在 Amazon S3 中的单个对象的操作元数据。同步调用 Lambda 函数，该函数对 S3 对象执行轻量变换，这是一种计算成本低廉的操作（例如将文件从一种格式转换为另一种格式）。由于新对象已添加到暂存 S3 存储桶中，因此综合目录会更新，并且消息会发送到 SQS 队列，等待 ETL 中的下一阶段。
4. CloudWatch 事件规则每 5 分钟触发一个 Lambda 函数。此函数检查消息是否从上一个 ETL 阶段传送至 SQS 队列。如果消息已传送，则 Lambda 函数将在 ETL 流程中启动 AWS Step Functions 中的第二个函数。
5. 然后对批量文件进行大量转换。这种繁重的转换是一项计算成本很高的操作，例如同步调用 AWS Glue 作业、AWS Fargate 任务、亚马逊 EMR 步骤或亚马逊笔记本。 SageMaker 使用 AWS Glue 爬网程序从输出文件中提取表元数据，这会更新 AWS Glue 目录。文件元数据也将添加至 DynamoDB 的综合目录表中。最后，还利用 Deequ 运行数据质量步骤。

技术堆栈

- 亚马逊 CloudWatch 活动
- AWS CloudFormation
- AWS CodePipeline
- AWS CodeBuild
- AWS CodeCommit
- Amazon DynamoDB
- AWS Glue
- AWS Lambda
- Amazon S3
- Amazon SQS
- AWS Step Functions

工具

- Amazon CloudWatch Ev CloudWatch ents — Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。
- AWS CloudFormation — CloudFormation 帮助以可预测的方式重复创建和配置 AWS 基础设施部署。
- AWS CodeBuild — CodeBuild 是一项完全托管的构建服务，可编译您的源代码、运行单元测试并生成可随时部署的工件。
- AWS CodeCommit — CodeCommit 是一项由 AWS 托管的版本控制服务，您可以使用它来私下存储和管理资产（例如源代码和二进制文件）。
- AWS CodePipeline — CodePipeline 是一项持续交付服务，您可以使用它对持续发布软件变更所需的步骤进行建模、可视化和自动化。
- Amazon DynamoDB – DynamoDB 是一种全托管 NoSQL 数据库服务，提供快速而可预测的性能，能够实现扩展。
- AWS Glue – AWS Glue 是一项完全托管的 ETL 服务，可让客户轻松准备和加载数据以进行分析。
- AWS Lambda – Lambda 支持无需预调配或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一种高度可扩展的对象存储服务。Amazon S3 可用于各种存储解决方案，包括网站、移动应用程序、备份和数据湖。
- AWS Step Functions - AWS Step Functions 是一款无服务器函数编排工具，它可以轻松地将 AWS Lambda 函数和多个 Amazon Web Services 排序到关键业务应用程序中。
- Amazon SQS – Amazon Simple Queue Service (Amazon SQS) 是一种全托管消息队列服务，可帮助您解耦和扩展微服务、分布式系统和无服务器应用程序。
- Deequ – Deequ 是一款工具，可帮助您计算大型数据集的数据质量指标，定义和验证数据质量限制，并随时了解数据分布的变化。

代码存储库

SDLF 的源代码和资源可在 AWS 实验室 GitHub 存储库中找到。

操作说明

设置 CI/CD 管道以配置 IaC

Task	描述	所需技能
设置 CI/CD 管道，以管理数据湖的 IaC。	登录 AWS 管理控制台，按照 SDLF 研讨会的初始设置部分中的步骤进行操作。这将创建初始 CI/CD 资源，例如为 CodeCommit 数据湖配置和管理 IaC 的存储库、 CodeBuild 环境和 CodePipeline 管道。	DevOps 工程师

IaC 版本控制

Task	描述	所需技能
在本地计算机上克隆 CodeCommit 存储库。	按照 SDLF 研讨会部署基础部分中的步骤进行操作。这可以帮助您将托管 IaC 的 Git 存储库克隆至本地环境中。 有关更多信息，请参阅 CodeCommit 文档中的连接 CodeCommit 存储库。	DevOps 工程师
修改 CloudFormation 模板。	使用本地工作站和代码编辑器根据您的用例或要求修改 CloudFormation 模板。将它们提交到本地克隆 Git 存储库。 有关更多信息，请参阅 AWS CloudFormation 文档中的使用 AWS CloudFormation 模板。	DevOps 工程师
将更改推送到 CodeCommit 存储库。	您的基础设施代码现在处于版本控制之下，并且会跟踪对代	DevOps 工程师

Task	描述	所需技能
	码库的修改。当您将更改推送到 CodeCommit 存储库时，CodePipeline 会自动将其应用于您的基础架构并将其交付给 CodeBuild。 **Important** 如果您在中使用 AWS SAM CLI CodeBuild，请运行`sam package`和`sam deploy`命令。如果您使用 AWS CLI，请运行 `aws cloudformation package` 和 `aws cloudformation deploy` 命令。	

相关资源

设置 CI/CD 管道以配置 IaC

- SDLF 研讨会 — 初始设置

IaC 版本控制

- SDLF 研讨会 — 部署基础
- 正在连接 CodeCommit 存储库
- 使用 AWS CloudFormation 模板

其他资源

- AWS 无服务器数据分析管道参考架构
- SDLF 文档

启动时强制标记 Amazon EMR 集群

由 Priyanka Chaudhary (AWS) 编写

摘要

此模式提供了一种安全控制，可确保 Amazon EMR 集群在创建时被标记。

Amazon EMR 是一项 Amazon Web Services (AWS) 服务，用于处理和分析大量数据。Amazon EMR 提供可扩展、低配置的服务，作为运行内部集群计算的更简单的替代方案。您可以使用标记以不同的方式对 AWS 资源进行分类，例如按用途、所有者或环境。例如，您可通过将自定义元数据分配给各个 Amazon EMR 集群，为 Amazon EMR 集群添加标签。标签由您定义的键和值组成。我们建议您创建一组一致的标签以满足您的组织要求。当您向 Amazon EMR 集群添加标签时，该标签还会传播到与该集群关联的每个活动的亚马逊弹性计算云 (Amazon EC2) 实例。同样，当您从 Amazon EMR 集群中移除标签时，该标签也会从每个关联的活动 EC2 实例中删除。

侦探控件监控 API 调用并启动RunJobFlow、AddTags、和CreateTags APIs的 Amazon Event RemoveTagss CloudWatch 事件。该事件调用 AWS Lambda，后者运行 Python 脚本。Python 函数从事件的 JSON 输入中获取 Amazon EMR 集群 ID 并执行以下检查：

- 检查 Amazon EMR 集群是否配置了您指定的标签名称。
- 如果没有，请向用户发送包含相关信息的 Amazon Simple Notification Service（Amazon SNS）通知：Amazon EMR 集群名称、违规详细信息、AWS 区域、Amazon Web Services account 以及此通知源自 Lambda 的 Amazon 资源名称（ARN）。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 用于上传提供的 Lambda 代码的 Amazon Simple Storage Service （Amazon S3）存储桶。或者，您可为此目的创建 S3 存储桶，如操作说明部分所述。
- 您希望接收违规通知的有效电子邮件地址。
- 您要检查的必填标签列表。

限制

- 这种安全控制为区域性的。您必须将其部署在要监控的每个 AWS 区域。

产品版本

- Amazon EMR 发行版 4.8.0 及更高版本。

架构

工作流程架构

自动化和扩缩

- 如果您使用的是 AWS Organ izations，则可以使用 AWS Cloudformation StackSets 在要监控的多个账户中部署此模板。

工具

Amazon Web Services

- AWS CloudFormation — AWS CloudFormation 可帮助您建模和设置 AWS 资源，快速一致地配置这些资源，并在资源的整个生命周期中对其进行管理。您可以使用模板来描述资源及其依赖关系，然后将它们作为堆栈一起启动和配置，而不必单独管理资源。您可以跨多个 Amazon Web Services account 和 AWS 区域管理和预置堆栈。
- Amazon CloudWatch Events-Amazon CloudWatch Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。
- Amazon EMR — Amazon EMR 是一项 Web 服务，可简化大数据框架的运行并高效处理大量数据。
- AWS Lambda – AWS Lambda 是一项计算服务，支持无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项对象存储服务。您可以通过 Amazon S3 随时在 Web 上的任何位置存储和检索的任意大小的数据。
- Amazon SNS — Amazon Simple Notification Service (Amazon SNS) 可协调和管理发布者和客户端之间消息的传送或发送，包括 Web 服务器和电子邮件地址。订阅用户接收所有发布至他们所订阅主题的消息，并且一个主题的所有订阅用户收到的消息都相同。

代码

此模式包括以下附件：

- `EMRTagValidation.zip` — 用于安全控制的 Lambda 代码。
- `EMRTagValidation.yml`— 用于设置事件和 Lambda 函数的 CloudFormation 模板。

操作说明

设置 S3 存储桶

Task	描述	所需技能
定义 S3 存储桶。	在 Amazon S3 控制台，选择或创建 S3 存储桶来托管 Lambda 代码 .zip 文件。此 S3 存储桶必须与您想要监视的 Amazon EMR 集群位于相同的 AWS 区域中。Amazon S3 存储桶名称是全局唯一的，并且命名空间由所有 Amazon Web Services account 共享。S3 存储桶名称不得包含前导斜杠。	云架构师
上传 Lambda 代码。	将附件部分中提供的 Lambda 代码 .zip文件上传至 S3 存储桶。	云架构师

部署 AWS CloudFormation 模板

Task	描述	所需技能
启动 AWS CloudFormation 模板。	在与 S3 存储桶相同的 AWS 区域中打开 AWS CloudFormation 控制台并部署模板。有关部署 AWS CloudFormation 模板的更多信息，请参阅 CloudFormation 文档中的在	云架构师

Task	描述	所需技能
	AWS CloudFormation 控制台上创建堆栈。	

Task	描述	所需技能
填写模板中的参数。	启动模板时，系统将会提示输入以下信息： • S3 存储桶：指定您在首个操作说明中创建或选择的存储桶。这是您上传所附的 Lambda 代码（文件）的地方。 • S3 密钥：指定 Lambda .zip 文件在您的 S3 存储桶中的位置（例如，filename.zip 或 controls/filename.zip）。切勿纳入前导斜字符。 • 通知电子邮件：提供有效的电子邮件地址以接收 Amazon SNS 通知。 • 标记键名称：以逗号分隔的列表形式提供要检查的标签（例如、`ApplicationID` 、`Environment` 、`Owner`）。Events CloudWatch 事件会监控集群中是否有这些标签，如果找不到这些标签，则会发送通知。 • Lambda 日志级别： 指定 Lambda 函数的日志记录级别和频率。使用信息记录有关进度的详细信息消息，使用错误记录仍然允许部署继续的错误事件，使用警告记录潜在的有害情况。	云架构师

确认订阅

Task	描述	所需技能
确认订阅。	CloudFormation 模板成功部署后，它会向您提供的电子邮件地址发送一封订阅电子邮件。您必须确认此电子邮件订阅，才能开始接收违规通知。	云架构师

相关资源

- AWS Lambda 开发人员指南
- 为 Amazon EMR 集群添加标签

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

确保在启动时启用 Amazon EMR 日志记录到 Amazon S3

由 Priyanka Chaudhary (AWS) 编写

摘要

此模式提供了一种安全控制，用于监控在 Amazon Web Services（AWS）上运行的 Amazon EMR 集群的日志记录配置。

Amazon EMR 是用于大数据处理和分析的 AWS 工具。Amazon EMR 提供可扩展的低配置服务，作为运行内部集群计算的替代方法。Amazon EMR 提供两类 EMR 集群。

- 临时 Amazon EMR 集群：临时 Amazon EMR 集群在处理完成后自动关闭并停止产生成本。
- 永久 Amazon EMR 集群：数据处理任务完成后，永久性 Amazon EMR 集群将继续运行。

Amazon EMR 和 Hadoop 都可以生成日志文件，报告集群上的状态。默认情况下，它们会写入/mnt/var/log/目录中的主节点。根据在启动集群时配置集群的方式，您还可将这些日志保存到 Amazon Simple Storage Service (Amazon S3)，并通过图形调试工具进行查看。请注意，只有在启动集群时才可指定 Amazon S3 日志记录。通过此配置，日志每 5 分钟从主节点发送到 Amazon S3 位置。对于瞬态集群，Amazon S3 日志记录非常重要，因为处理完成后集群就会消失，并且这些日志文件可用于调试任何失败的作业。

该模式使用 AWS CloudFormation 模板部署安全控 CloudWatch 件，用于监控 API 调用，并在 “” 上启动 Amazon Events RunJobFlow。 触发器调用 AWS Lambda，后者运行 Python 脚本。Lambda 函数从事件 JSON 输入中检索 EMR 集群 ID，并检查 Amazon S3 日志 URI。如果未找到 Amazon S3 URI，Lambda 函数将发送 Amazon Simple Notification Service (Amazon SNS) 通知，详细说明 EMR 集群名称、违规详细信息、AWS 区域、Amazon Web Services account 以及通知来源的 Lambda Amazon 资源名称（ARN）。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 存放 Lambda 代码 .zip 文件的 S3 存储桶
- 接收违规通知的电子邮件地址

限制

- 此检测控制是区域性的，必须部署在您要监控的 AWS 区域中。

产品版本

- Amazon EMR 发行版 4.8.0 及以上版本

架构

目标技术堆栈

- 亚马逊 CloudWatch 活动活动
- Amazon EMR
- Lambda 函数
- S3 存储桶
- Amazon SNS

目标架构

自动化和扩缩

- 如果您使用的是 AWS Organizations CloudFormation StackSets，则可以使用 AWS 在要监控的多个账户中部署此模板。

工具

工具

- AWS CloudFormation — AWS CloudFormation 可帮助您使用基础设施即代码建模和设置 AWS 资源。
- AWS Cloudwatch Ev CloudWatch ents — AWS Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。
- Amazon EMR — Amazon EMR 是托管集群平台，可简化大数据框架的运行。
- AWS Lambda — AWS Lambda 支持无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。

- Amazon S3 — Amazon S3 是一个 Web 服务接口，可用于从 Web 上的任何位置存储和检索任意数量的数据。
- Amazon SNS — Amazon SNS 是一项 Web 服务，可协调和管理发布者和客户端之间消息的传送或发送，包括 Web 服务器和电子邮件地址。

代码

- 该项目的 .zip 文件作为附件提供。

操作说明

删除 S3 存储桶

Task	描述	所需技能
定义 S3 存储桶。	要托管 Lambda 代码 .zip 文件，请选择或创建一个具有不包含前导斜杠的唯一名称的 S3 存储桶。S3 存储桶名称是全局唯一的，并且命名空间由所有 Amazon Web Services account 共享。您的 S3 存储桶需要与正在评估的 Amazon EMR 集群位于同一 AWS 区域。	云架构师

将 Lambda 代码上传至 S3 存储桶

Task	描述	所需技能
将 Lambda 代码上传至 S3 存储桶。	将“附件”部分中提供的 Lambda 代码.zip 文件上传到 S3 存储桶。S3 存储桶需要与正在评估的 Amazon EMR 集群位于同一区域。	云架构师

部署 AWS CloudFormation 模板

Task	描述	所需技能
部署 AWS CloudFormation 模板。	在 AWS CloudFormation 控制台上，在与 S3 存储桶相同的区域中，部署作为该模式附件提供的 AWS CloudFormation 模板。在下一个操作说明中，提供参数的值。有关部署 AWS CloudFormation 模板的更多信息，请参阅 “相关资源” 部分。	云架构师

填写 AWS CloudFormation 模板中的参数

Task	描述	所需技能
命名 S3 存储桶。	输入您在第一个操作说明中创建的 S3 存储桶的名称。	云架构师
提供 Amazon S3 密钥。	提供 Lambda 代码 .zip 文件在 S3 存储桶中的位置，不带前导斜杠（例如，<directory>/<file-name>.zip）。	云架构师
提供电子邮箱地址。	提供有效的电子邮件地址以接收 Amazon SNS 通知。	云架构师
定义日志记录级别。	定义 Lambda 函数的日志记录级别与频率。“信息”表示有关应用程序进度的详细信息消息。“错误”表示仍可能允许应用程序继续运行的错误事件。“警告”表示潜在的有害情况。	云架构师

确认订阅

Task	描述	所需技能
确认订阅。	成功部署模板后，它将向提供的电子邮件地址发送订阅电子邮件。您必须确认此电子邮件订阅才能接收违规通知。	云架构师

相关资源

- AWS Lambda
- Amazon EMR 日志记录
- 部署 AWS CloudFormation 模板

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS Glue 作业和 Python 生成测试数据

由 Moinul Al-Mamun (AWS) 创建

摘要

此模式向您展示如何通过创建用 Python 编写的 AWS Glue 作业，快速轻松地同时生成数百万个示例文件。示例文件存储在 Amazon Simple Storage Service（Amazon S3）存储桶中。能够快速生成大量示例文件对于在 Amazon Web Services Cloud 中测试或评估服务非常重要。例如，您可以通过对数百万个带有 Amazon S3 前缀的小文件进行数据分析来测试 AWS Glue Studio 或 AWS Glue DataBrew 任务的性能。

尽管您可以使用其他 Amazon Web Services 生成示例数据集，但我们建议您使用 AWS Glue。您无需管理任何基础设施，因为 AWS Glue 是一项无服务器数据处理服务。您只需带上代码并在 AWS Glue 集群中运行即可。此外，AWS Glue 还预调配、配置和扩展运行作业所需资源。您只需为作业运行时使用的资源付费。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- AWS 命令行界面（AWS CLI），已安装并配置以使用 Amazon Web Services account

产品版本

- Python 3.9
- AWS CLI 版本 2

限制

每个触发器的最大 AWS Glue 作业数是 50。有关更多信息，请参阅 AWS Glue 端点和限额。

架构

下图描绘了一个以 AWS Glue 作业为中心的示例架构，该作业将其输出（即示例文件）写入 S3 存储桶。

图表包括以下工作流程：

1. 您可以使用 AWS CLI、AWS 管理控制台 或 API 启动 AWS Glue 作业。AWS CLI 或 API 使您能够自动并行化调用的作业，并缩短生成示例文件的运行时系统。
2. AWS Glue 作业随机生成文件内容，将内容转换为 CSV 格式，然后将内容作为 Amazon S3 对象存储在通用前缀下。每个文件小于 1 千字节。AWS Glue 作业接受两个用户定义的任务参数：`START_RANGE` 和 `END_RANGE`。您可以使用这些参数来设置文件名和每次作业运行在 Amazon S3 中生成的文件数。您可以并行运行此作业的多个实例（例如，100 个实例）。

工具

- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- AWS Glue 是一项完全托管的提取、转换、加载（ETL）服务。它可以帮助您在数据存储和数据流之间对数据进行可靠地分类、清理、扩充和移动。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。

最佳实践

在实施此模式时，请考虑以下 AWS Glue 最佳实践：

- 使用正确的 AWS Glue Worker 类型来降低成本。我们建议您了解 Worker 类型的不同属性，然后根据 CPU 和内存要求为工作负载选择正确的 Worker 类型。对于这种模式，我们建议您使用 Python Shell 作业作为作业类型，以最大限度地减少 DPU 并降低成本。有关更多信息，请参阅 AWS Glue 开发人员指南中的在 AWS Glue 中添加作业。
- 使用正确的并发限制来扩展作业。我们建议您根据自己的时间要求和所需文件数量来确定 AWS Glue 作业的最大并发度。
- 首先开始生成少量文件。为了在构建 AWS Glue 作业时降低成本并节省时间，请从少量文件（例如 1,000 个）开始。这样可以更轻松地进行故障排除。如果成功生成少量文件，则可以扩展到更多数量的文件。

- 首先在本地运行。为了在构建 AWS Glue 作业时降低成本并节省时间，请在本地开始开发并测试代码。有关设置 Docker 容器以帮助您在 Shell 和集成式开发环境（IDE）中测试提取、转换、加载（ETL）作业的说明，请参阅 AWS 大数据博客文章使用容器在本地开发 AWS Glue ETL 作业。

有关更多 AWS Glue 最佳实践，请参阅 AWS Glue 文档中的最佳实践。

操作说明

创建目标 S3 存储桶和 IAM 角色

Task	描述	所需技能
创建 S3 存储桶以存储文件。	创建 S3 存储桶及其中的前缀。 **Note** 此模式使用该`s3://{your-s3-bucket-name}/small-files/` 位置进行演示。	应用程序开发人员
创建和配置 IAM 角色。	您必须创建 IAM 角色，这样 AWS Glue 作业可以使用该角色写入 S3 存储桶。 1. 创建 IAM 角色（例如，名为 `"AWSGlueServiceRole-smallfiles"` ）。 2. 选择 AWS Glue 作为策略的受信任实体。 3. 将名为 `"AWSGlueServiceRole"` 的 AWS 托管式策略附加到该角色。 4. 根据以下配置创建名为 `"s3-small-file-acc`	应用程序开发人员

Task	描述	所需技能
	ess" 的内联策略或客户管理型策略。将 "{bucket}" 替换为存储桶名称。 `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "s3:GetObject", "s3:PutObject" ], "Resource": [ "arn:aws:s3:::{bucket}/small-files/input/*" ] } ] }` 5. 将 "s3-small-file-access" 策略附加到角色。	

创建和配置 AWS Glue 作业以处理并发运行

Task	描述	所需技能
创建 AWS Glue 作业。	您必须创建 AWS Glue 作业来生成内容并将其存储在 S3 存储桶中。 创建 AWS Glue 作业，然后通过完成以下步骤来配置作业： 1. 登录 AWS 管理控制台，然后打开 AWS Glue 控制台。 2. 在导航窗格中的数据集成和 ETL下方，选择作业。 3. 在创建作业部分中，选择 Python Shell 脚本编辑器。 4. 在选项部分中，选择使用样板代码创建新脚本，然后选择创建。 5. 选择作业详细信息。 6. 在名称中，输入 create_small_files。 7. 对于 IAM 角色，选择您之前创建的 IAM 角色。 8. 在此作业运行部分中，选择要由您创作的新脚本。 9. 选择高级属性。 10对于最大并发度，请输入 100 以供演示。注意：最大并发度定义您可以并行运行的作业实例数量。 11选择保存。	应用程序开发人员
更新作业代码。	1. 打开 AWS Glue 控制台。	应用程序开发人员

Task	描述	所需技能

	2. 在导航窗格中，选择作业。
	3. 在您的作业部分中，选择您之前创建的作业。
	4. 选择脚本选项卡，然后根据以下代码更新脚本。使用您的值更新 BUCKET_NAME 、PREFIX 和 text_str 变量。
	
	`from awsglue.utils import getResolvedOptions`
	`import sys`
	`import boto3`
	`from random import randrange`
	
	`# Two arguments`
	`args = getResolvedOptions(sys.argv, ['START_RANGE', 'END_RANGE'])`
	
	`START_RANGE = int(args['START_RANGE'])`
	`END_RANGE = int(args['END_RANGE'])`
	
	`BUCKET_NAME = '{BUCKET_NAME}'`
	`PREFIX = 'small-files/input/'`
	`s3 = boto3.resource('s3')`
	`for x in range(START_RANGE, END_RANGE):`	

Task	描述	所需技能

	`    # generate file name`
	`    file_name = f"input_{x}.txt"`
	`    # generate text`
	`    text_str = str(randrange(100000))+","+str(randrange(100000))+", " + str(randrange(10000000)) + "," + str(randrange(10000))`
	`    # write in s3`
	`s3.Object(BUCKET_NAME, PREFIX + file_name).put(Body=text_str)`
	5. 选择保存。	

从命令行或控制台运行 AWS Glue 作业

Task	描述	所需技能

从命令行运行 AWS Glue 作业。	要从 AWS CLI 运行 AWS Glue 作业，请使用您的值运行以下命令：
	`cmd:~$ aws glue start-job-run --job-name create_small_files --arguments '{"--START_RANGE":"0","--END_RANGE":"1000000"}'`
	`cmd:~$ aws glue start-job-run --job-name create_small_files --arguments '{"--STAR`
		应用程序开发人员

Task	描述	所需技能

```
T_RANGE":"1000000"
,"--END_RANGE":"20
00000"}'
```

> **Note**
>
> 有关从 AWS 管理控制台运行 AWS Glue 任务的说明，请参阅 AWS 管理控制台中按此模式运行 AWS Glue 任务的故事。

> **Tip**
>
> 如果您想使用不同的参数同时运行多个执行，我们建议您使用 AWS CLI 运行 AWS Glue 任务，如上面的示例所示。

要生成所有 AWS CLI 命令（在使用特定的并行化系数生成指定数量的文件时所需），请运行以下 bash 代码（使用您的值）：

```
# define parameters
NUMBER_OF_FILES=
10000000;
PARALLELIZATION=50;

# initialize
```

Task	描述	所需技能

	<pre>_SB=0;
	
	# generate commands
	for i in $(seq 1
	 $PARALLELIZATION);
	do
	      echo aws glue
	 start-job-run --
	job-name create_sm
	all_files --argumen
	ts "'"'{"--START_RANG
	E":"'$(((NUMBER_OF
	_FILES/PARALLELIZA
	TION) * (i-1) +
	 _SB))'","--END_RAN
	GE":"'$(((NUMBER_O
	F_FILES/PARALLELIZ
	ATION) * (i)))'"}'"'";
	      _SB=1;
	done</pre>
	如果使用上述脚本，请考虑以下事项：
	• 该脚本简化了大规模调用和生成小文件的过程。
	• 使用您的值更新 NUMBER_OF_FILES 和 PARALLELIZATION 。
	• 上述脚本打印了必须运行的命令列表。复制这些输出命令，然后在终端中运行它们。
	• 如果要直接从脚本中运行命令，请删除第 11 行的 echo 语句。	

Task	描述	所需技能
	Note 要查看上述脚本的输出示例，请参阅此模式的 “其他信息” 部分中的 Shell 脚本输出。	
在 AWS 管理控制台 中运行 AWS Glue 作业。	1. 登录 AWS 管理控制台，然后打开 AWS Glue 控制台。 2. 在导航窗格中的数据集成和 ETL下方，选择作业。 3. 在您的作业部分中，选择作业。 4. 在参数（可选）部分中，更新参数。 5. 选择操作，然后选择运行作业。 6. 根据需要，多次重复执行步骤 3-5。例如，要创建 1000 万个文件，请重复此过程 10 次。	应用程序开发人员
检查 AWS Glue 作业的状态。	1. 打开 AWS Glue 控制台。 2. 在导航窗格中，选择作业。 3. 在您的作业部分中，选择您之前创建的作业（即：`create_small_files`）。 4. 要深入了解文件的进度和生成情况，请查看运行 ID、运行状态和其他列。	应用程序开发人员

相关资源

参考

- Registry of Open Data on AWS
- 用于分析的数据集
- 在 AWS 上打开数据
- 在 AWS Glue 中添加作业
- AWS Glue 入门

指南和模式

- AWS Glue 最佳实践
- 加载测试应用程序

其他信息

基准测试

作为基准测试的一部分，该模式用于使用不同的并行化参数生成 1000 万个文件。下表显示测试输出：

并行化	作业运行生成的文件数	作业时长	Speed
10	1000000	6 小时 40 分钟	很慢
50	200,000	80 分钟	中
100	100000	40 minutes	快速

如果要加快处理速度，可以在作业配置中配置更多的并发运行。您可以根据自己的要求轻松调整作业配置，但请记住，AWS Glue 服务限额存在限制。有关更多信息，请参阅 AWS Glue 端点和限额。

Shell 脚本输出

以下示例显示了此模式中从命令行运行 AWS Glue 作业情节的 Shell 脚本的输出。

```
user@MUC-1234567890 MINGW64 ~
  $ # define parameters
```

```
  NUMBER_OF_FILES=10000000;
  PARALLELIZATION=50;
  # initialize
  _SB=0;

  # generate commands
  for i in $(seq 1 $PARALLELIZATION);
   do
         echo aws glue start-job-run --job-name create_small_files --arguments
 "'"'{"--START_RANGE":"'$(((NUMBER_OF_FILES/PARALLELIZATION)  (i-1) + SB))'","--
ENDRANGE":"'$(((NUMBER_OF_FILES/PARALLELIZATION)  (i)))'"}'"'";
         _SB=1;
   done

  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"0","--END_RANGE":"200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"200001","--END_RANGE":"400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"400001","--END_RANGE":"600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"600001","--END_RANGE":"800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"800001","--END_RANGE":"1000000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"1000001","--END_RANGE":"1200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"1200001","--END_RANGE":"1400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"1400001","--END_RANGE":"1600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"1600001","--END_RANGE":"1800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"1800001","--END_RANGE":"2000000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"2000001","--END_RANGE":"2200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"2200001","--END_RANGE":"2400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"2400001","--END_RANGE":"2600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"2600001","--END_RANGE":"2800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"2800001","--END_RANGE":"3000000"}'
```

```
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"3000001","--END_RANGE":"3200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"3200001","--END_RANGE":"3400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"3400001","--END_RANGE":"3600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"3600001","--END_RANGE":"3800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"3800001","--END_RANGE":"4000000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"4000001","--END_RANGE":"4200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"4200001","--END_RANGE":"4400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"4400001","--END_RANGE":"4600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"4600001","--END_RANGE":"4800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"4800001","--END_RANGE":"5000000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"5000001","--END_RANGE":"5200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"5200001","--END_RANGE":"5400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"5400001","--END_RANGE":"5600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"5600001","--END_RANGE":"5800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"5800001","--END_RANGE":"6000000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"6000001","--END_RANGE":"6200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"6200001","--END_RANGE":"6400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"6400001","--END_RANGE":"6600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"6600001","--END_RANGE":"6800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"6800001","--END_RANGE":"7000000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"7000001","--END_RANGE":"7200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"7200001","--END_RANGE":"7400000"}'
```

```
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"7400001","--END_RANGE":"7600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"7600001","--END_RANGE":"7800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"7800001","--END_RANGE":"8000000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"8000001","--END_RANGE":"8200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"8200001","--END_RANGE":"8400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"8400001","--END_RANGE":"8600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"8600001","--END_RANGE":"8800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"8800001","--END_RANGE":"9000000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"9000001","--END_RANGE":"9200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"9200001","--END_RANGE":"9400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"9400001","--END_RANGE":"9600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"9600001","--END_RANGE":"9800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"9800001","--END_RANGE":"10000000"}'

  user@MUC-1234567890 MINGW64 ~
```

常见问题解答

我应该使用多少个并发运行或并行作业？

并发运行和并行作业的数量取决于您的时间要求和所需测试文件数量。我们建议您检查正在创建的文件的大小。首先，检查 AWS Glue 作业生成所需数量的文件需要多长时间。然后，使用正确的并发运行次数来实现您的目标。例如，如果您假设 100,000 个文件需要 40 分钟才能完成运行，但目标时间为 30 分钟，则必须增加 AWS Glue 作业的并发设置。

我可以使用这种模式创建什么类型的内容？

您可以创建任何类型的内容，例如具有不同分隔符的文本文件（例如 PIPE、JSON 或 CSV）。此模式使用 Boto3 写入文件，然后将文件保存到 S3 存储桶中。

我需要在 S3 存储桶中获得什么级别的 IAM 权限？

您必须具有基于身份的策略，该策略允许对 S3 存储桶中的对象进行 `Write` 访问。有关更多信息，请参阅 Amazon S3 文档中的 Amazon S3：允许对 S3 存储桶中的对象进行读写访问。

使用 Amazon IoT Greengrass 将物联网数据直接摄取至 Amazon S3，经济实惠

由 Sebastian Viviani (AWS) 和 Rizwan Syed (AWS) 创建

摘要

本文向您介绍了如何使用 Amazon IoT Greengrass Version 2 设备经济高效地将物联网 (IoT) 数据直接摄取至 Amazon Simple Storage Service (Amazon S3) 存储桶中。设备运行自定义组件，用于读取物联网数据，并将数据保存在永久存储（即本地磁盘或卷）中。然后，设备将物联网数据压缩为 Apache Parquet 文件，并定期将数据上传至 S3 存储桶。

您采集的物联网数据的数量和速度仅受边缘硬件功能以及网络带宽的限制。您可使用 Amazon Athena 经济高效地分析您摄取的数据。Athena 支持使用 Amazon Managed Grafana 进行压缩 Apache Parquet 文件和数据可视化。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 在 Amazon IoT Greengrass Version 2 上运行的边缘网关手机传感器数据。（数据来源和数据收集过程超出了此模式的范围，但您几乎可以使用任何类型的传感器数据。本示例采用本地 MQTT 代理，其带传感器或网关，可在本地发布数据）。
- Amazon IoT Greengrass 组件、角色和 SDK 依赖项
- 用于将数据上传至 S3 存储桶的流管理器组件
- 适用于 Java 的 AW S 开发工具包、适用于 JavaScript Java 的 A WS 开发工具包或适用于 Python 的 AWS 开发工具包 (Boto3) 来运行 APIs

限制

- 这种模式中的数据不会实时上传至 S3 存储桶。有延迟期，您可配置延迟时间。数据在边缘设备中临时缓冲，然后在到期后上传。
- SDK 仅可采用 Java、Node.js 和 Python 语言。

架构

目标技术堆栈

- Amazon S3
- Amazon IoT Greengrass
- MQTT 代理
- 流管理器组件

目标架构

下图显示了一种架构，该架构旨在摄取物联网传感器数据，并将该数据存储至 S3 存储桶。

图表显示了以下工作流：

1. 多个传感器（例如温度和阀门）更新会发布到本地 MQTT 代理。
2. 订阅这些传感器的 Parquet 文件压缩器会更新主题并接收这些更新。
3. Parquet 文件压缩器将更新项存储在本地。
4. 期限过后，存储的文件被压缩为 Parquet 文件，然后传递至流管理器，以上传到指定的 S3 存储桶。
5. 流管理器会将 Parquet 文件上传至 S3 存储桶。

> **Note**
>
> 直播管理器 (`StreamManager`) 是一个托管组件。有关如何将数据导出至 Amazon S3 的示例，请参阅 Amazon IoT Greengrass 文档中的流管理器。你可以使用本地 MQTT 代理作为组件，也可以使用其他代理，比如 Eclipse Mosquitto。

工具

AWS 工具

- Amazon Athena 是一种交互式查询服务，可帮助您使用标准 SQL 直接在 Amazon S3 中分析数据。

- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon IoT Greengrass 是一项开源 IoT 边缘运行时和云服务，可帮助您在设备上构建、部署和管理 IoT 应用程序。

其他工具

- Apache Parquet 是一种专为存储和检索而设计的开源列式数据文件格式。
- MQTT（消息队列遥测传输）是一种轻量级消息协议，专为受限的设备而设计。

最佳实践

对上传的数据使用正确分区格式

对 S3 存储桶中的根前缀名称没有具体要求（例如 "myAwesomeDataSet/" 或 "dataFromSource"），但我们建议您使用有意义的分区和前缀，以便于理解数据集的用途。

我们还建议您在 Amazon S3 中使用正确分区，以便查询在数据集上以最佳方式运行。在以下示例中，数据以 HIVE 格式分区，以便优化每个 Athena 查询扫描的数据量。这可以提高性能并降低成本。

```
s3://<ingestionBucket>/<rootPrefix>/year=YY/month=MM/day=DD/
HHMM_<suffix>.parquet
```

操作说明

设置您的环境

Task	描述	所需技能
创建 S3 存储桶。	1. 创建一个 S3 存储桶或使用现有存储桶。 2. 为要从中摄取物联网数据的 S3 存储桶创建有意义前缀（例如s3:\\<bucket>\<prefix> ）。 3. 记录您的前缀以备后续使用。	应用程序开发人员

Task	描述	所需技能
添加 IAM 权限至 S3 存储桶。	要向用户授予您之前创建的 S3 存储桶和前缀写入权限，请将以下 IAM policy 添加至您的 Amazon IoT Greengrass 角色： `{ "Version": "2012-10-17", "Statement": [ { "Sid": "S3DataUpload", "Effect": "Allow", "Action": [ "s3:List*", "s3:Put*" ], "Resource": [ "arn:aws:s3:::<ingestionBucket>", "arn:aws:s3:::<ingestionBucket>/<prefix>/*" ] } ] }` 有关更多信息，请参阅 Aurora 文档中的创建 Amazon S3 资源 IAM policy。	应用程序开发人员

Task	描述	所需技能
	接下来，更新 S3 存储桶的资源策略（如果需要），以允许使用正确的 AWS 主体写入。	

构建和部署 Amazon IoT Greengrass 组件

Task	描述	所需技能
更新组件的配方。	当您根据以下示例创建部署时更新组件配置： `{ "region": "<region>", "parquet_period": <period>, "s3_bucket": "<s3Bucket>", "s3_key_prefix": "<s3prefix>" }` 将 `<region>` 替换为您的 AWS 区域、将 `<period>` 替换为定期间隔、将 `<s3Bucket>` 替换为 S3 存储桶，将 `<s3prefix>` 替换为前缀。	应用程序开发人员
创建组件。	请执行以下操作之一： • 创建组件。 • 将该组件添加到 CI/CD 管道（如果存在）。请务必将构件从构件存储库复制到 Amazon IoT Greengrass 构件存储桶。然后创建或更新您的	应用程序开发人员

Task	描述	所需技能
	Amazon IoT Greengrass 组件。 • **Note** 将 MQTT 代理添加为组件，或稍后手动添加。：此决定会影响您可以与代理一起使用的身份验证方案。手动添加代理会使代理与 Amazon IoT Greengrass 分离，启用该代理支持的任何身份验证方案。AWS 提供的代理组件具有预定义身份验证方案。欲了解更多信息，请参阅 MQTT 3.1.1 代理 (Moquette) 和 MQTT 5 代理 (EMQX)。	
更新 MQTT 客户端。	示例代码不使用身份验证，因为该组件在本地连接至代理。如果您的场景不同，请按需要更新 MQTT 客户端部分。此外，执行下列操作： 1. 更新订阅中的 MQTT 主题。 2. 根据需要更新 MQTT 消息解析器，因为每个来源的消息可能有所不同。	应用程序开发人员

将该组件添加至 Amazon IoT Greengrass Version 2 核心设备中

Task	描述	所需技能
更新核心设备部署。	如果 Amazon IoT Greengrass Version 2 核心设备的部署已经存在，请修改部署。如果部署不存在，请创建新部署。 要为组件指定正确的名称，请根据以下内容更新新组件的日志管理器配置（如果需要）： `{ "logsUploaderConfiguration": { "systemLogsConfiguration": { ... }, "componentLogsConfigurationMap": { "<com.iot.ingest.parquet>": { "minimumLogLevel": "INFO", "diskSpaceLimit": "20", "diskSpaceLimitUnit": "MB", "deleteLogFileAfterCloudUpload": "false" } ... } }, "periodicUploadIntervalSec": "300" }`	应用程序开发人员

Task	描述	所需技能
	最后，完成对 Amazon IoT Greengrass 核心设备部署的修订。	

验证是否将数据摄取至 S3 存储桶

Task	描述	所需技能
查看 Amazon IoT Greengrass 卷日志。	检查以下各项： • MQTT 客户端已成功连接至本地 MQTT 代理。 • MQTT 客户端订阅正确的主题。 • 关于 MQTT 主题的传感器更新消息将发送自豪代理。 • 每隔一段时间就会发生 Parquet 压缩。	应用程序开发人员
检查 S3 存储桶。	验证数据是否正在上传至 S3 存储桶。您可以看到每个时间段在上传的文件。 您还可以通过查询下一部分中的数据，验证数据是否已上传至 S3 存储桶。	应用程序开发人员

设置来自 Athena 的查询

Task	描述	所需技能
创建数据库和表。	1. 创建 AWS Glue 数据库（如需要）。	应用程序开发人员

Task	描述	所需技能
	2. 在 AWS Glue 中手动创建表格，或者在 AWS Glue 中运行爬网程序创建表。	
授予 Athena 数据访问权限。	1. 更新权限，以允许 Athena 访问 S3 存储桶。有关更多信息，请参阅 Athena 文档中的精细访问 AWS Glue Data Catalog 中的数据库和表。 2. 在数据库中查询表格。	应用程序开发人员

故障排除

事务	解决方案
MQTT 客户端无法连接	• 验证 MQTT 代理权限。如果你有来自 AWS 的 MQTT 代理，请参阅 MQTT 3.1.1 代理 (Moquette) 和 MQTT 5 代理 (EMQX)。 • 在 MQTT 客户端验证凭证。如果你有来自 AWS 的 MQTT 代理，请参阅 MQTT 3.1.1 代理 (Moquette) 和 MQTT 5 代理 (EMQX)。
MQTT 客户端订阅失败	验证 MQTT 代理权限。如果你有来自 AWS 的 MQTT 代理，请参阅 MQTT 3.1.1 代理 (Moquette) 和 MQTT 5 代理 (EMQX)。
无法创建 Parquet 文件	• 验证 MQTT 主题是否正确。 • 验证来自传感器的 MQTT 消息格式是否正确。
对象未上传至 S3 存储桶	• 确认您有互联网连接与端点连接。 • 验证您的 S3 存储桶的资源策略的正确性。

事务	解决方案
	• 验证 Amazon IoT Greengrass Version 2 核心设备角色的权限。

相关资源

- DataFrame（Pandas 文档）
- Apache Parquet 文档（Parquet 文档）
- 开发 Amazon IoT Greengrass 组件（Amazon IoT Greengrass 开发人员指南，第 2 版）
- 将 Amazon IoT Greengrass 组件部署至设备（Amazon IoT Greengrass 开发人员指南，第 2 版）
- 与本地物联网设备互动（Amazon IoT Greengrass 开发人员指南，第 2 版）
- MQTT 3.1.1 代理 (Moquette)（Amazon IoT Greengrass 开发人员指南，第 2 版）
- MQTT 5 代理 (EMQX)（Amazon IoT Greengrass 开发人员指南，第 2 版）

其他信息

成本分析

以下成本分析场景演示了此模式中涵盖的数据摄取方法如何影响 Amazon Web Services Cloud 中的数据摄取成本。此场景中的定价示例基于发布价格。价格可能会发生变化。此外，您的费用可能会有所不同，具体取决于您的 AWS 区域、AWS 服务限额以及与云环境相关的其他因素。

输入信号集

该分析使用以下一组输入信号为基础，将物联网摄取成本与其他可用替代方案进行比较。

信号数量	Frequency	每个信号的数据
125	25 Hz	8 字节

在此情况下，系统接收 125 个信号。每个信号为 8 字节，每 40 毫秒 (25 Hz) 会出现一次。这些信号可单独发出，也可以分组至公共有效载荷中。您可根据需要选择拆分和打包这些信号。您还可确定延迟。延迟由接收、累积和摄取数据时间段组成。

为了便于比较，此场景的摄取操作基于 us-east-1 AWS 区域。成本比较仅适用 Amazon Web Services。硬件或连接等其他成本未计入分析。

成本比较

下表显示了每种摄取方法的每月费用（以美元为单位）。

方法	月度成本
AWS 物联网 SiteWise *	331.77 美元
带有数据处理包的 AWS IoT SiteWise Edge（将所有数据保存在边缘）	200 美元
AWS IoT Core 和 Amazon S3 访问原始数据规则	84.54 美元
边缘 Parquet 文件压缩并上传至 Amazon S3	0.5 美元

*必须对数据进行缩减采样，才能符合服务限额。这意味着使用此方法会丢失数据。

替代方法

本节显示了以下替代方法等效成本：

- AWS IoT SiteWise — 每个信号都必须以单独的消息形式上传。因此，每月的消息总数为 125×25×3600×24×30，相当于每月 81 亿条消息。但是，AWS IoT 每个属性每秒 SiteWise 只能处理 10 个数据点。假设将数据缩减到 10 Hz，则每月的消息数量将减少到125×10×3600×24×30，即 32.4 亿条。如果您使用发布者组件，该组件以 10 个为一组（每百万封邮件 1 美元）打包，则每月的费用为每月 324 美元。假设每条消息为 8 字节 (1 Kb/125)，则为 25.92 Gb 的数据存储空间。这增加了每月 7.77 美元的费用。第一个月的总费用为 331.77 美元，每月增加 7.77 美元。
- 带有数据处理包的 AWS IoT SiteWise Edge，包括在边缘完全处理的所有模型和信号（即无需云端接入）— 您可以使用数据处理包作为替代方案，以降低成本并配置在边缘计算的所有模型。即使没有进行实际计算，也可以仅用于存储和可视化。在这种情况下，必须为边缘网关使用强大硬件。每月的固定费用为 200 美元。
- MQTT 直接接入 AWS IoT Core by MQTT 以及物联网规则，以将原数据存储至 Amazon S3 — 假设所有信号都发布在公共负载中，则发布到 AWS IoT Core 的消息总数为 25×3600×24×30，即每月 6,480 万条。按每百万条消息 1 美元算，每月费用为 64.8 美元。按每百万条规则激活 0.15 美元，

每条消息一条规则，每月增加 19.44 美元的费用。按照 Amazon S3 中每 Gb 存储空间 0.023 美元成本，每月再增加 1.5 美元（每月增加以反映新数据）。第一个月的总费用为 84.54 美元，每月增加 1.5 美元。

- 在边缘压缩 Parquet 文件中的数据并上传至 Amazon S3（建议的方法）— 压缩率取决于数据的类型。使用针对 MQTT 测试的相同工业数据，整个月的总产出数据为 1.2 Gb。每月的费用为 0.03 美元。其他基准测试中描述的压缩率（使用随机数据）约为 66%（更接近最坏的情况）。总数据量为 21 Gb，每月花费 0.5 美元。

Parquet 文件生成器

以下代码示例显示了用 Python 编写的 Parquet 文件生成器结构。该代码示例仅用于说明，如果粘贴到您的环境中则不起作用。

```
import queue
import paho.mqtt.client as mqtt
import pandas as pd

#queue for decoupling the MQTT thread
messageQueue = queue.Queue()
client = mqtt.Client()
streammanager = StreamManagerClient()

def feederListener(topic, message):
    payload = {
        "topic" : topic,
        "payload" : message,
    }
    messageQueue.put_nowait(payload)

def on_connect(client_instance, userdata, flags, rc):
    client.subscribe("#",qos=0)

def on_message(client, userdata, message):
    feederListener(topic=str(message.topic),
 message=str(message.payload.decode("utf-8")))


filename = "tempfile.parquet"
streamname = "mystream"
destination_bucket= "amzn-s3-demo-bucket"
keyname="mykey"
```

```
period= 60

client.on_connect = on_connect
client.on_message = on_message
streammanager.create_message_stream(
            MessageStreamDefinition(name=streamname,
 strategy_on_full=StrategyOnFull.OverwriteOldestData)
        )


while True:
   try:
       message = messageQueue.get(timeout=myArgs.mqtt_timeout)
   except (queue.Empty):
       logger.warning("MQTT message reception timed out")

   currentTimestamp = getCurrentTime()
   if currentTimestamp >= nextUploadTimestamp:
       df = pd.DataFrame.from_dict(accumulator)
       df.to_parquet(filename)
       s3_export_task_definition = S3ExportTaskDefinition(input_url=filename,
 bucket=destination_bucket, key=key_name)
       streammanager.append_message(streamname,
 Util.validate_and_serialize_to_json_bytes(s3_export_task_definition))
       accumulator = {}
       nextUploadTimestamp += period
   else:
        accumulator.append(message)
```

使用 Lambda 函数在瞬态 EMR 集群中启动 Spark 作业

由 Dhrubajyoti Mukherjee (AWS) 创建

摘要

此模式使用 Amazon EMR RunJobFlow API 操作启动临时集群，以便通过 Lambda 函数运行 Spark 作业。瞬态 EMR 集群设计为在作业完成或发生任何错误时立即终止。临时集群可以节省成本，因为它只在计算期间运行，而且它在云环境中提供了可扩展性和灵活性。

瞬态 EMR 集群是在 Lambda 函数中使用 Boto3 API 和 Python 编程语言启动的。用 Python 编写的 Lambda 函数提供了更大的灵活性，可在需要时启动集群。

为了演示批处理计算和输出示例，此模式将在 EMR 集群中通过 Lambda 函数启动 Spark 作业，并针对一家虚构公司的示例销售数据运行批处理计算。Spark 作业的输出将是 Amazon Simple Storage Service (Amazon S3)中的逗号分隔值 (CSV) 文件。输入数据文件、Spark.jar 文件、代码片段以及用于运行计算的虚拟私有云 (VPC) 和 AWS Identity and Access Management (IAM) 角色的 AWS CloudFormation 模板作为附件提供。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account

限制

- 每次只能从代码中启动一个 Spark 作业。

产品版本

- 已在 Amazon EMR 6.0.0 上进行了测试

架构

目标技术堆栈

- Amazon EMR

- AWS Lambda
- Amazon S3
- Apache Spark

目标架构

自动化和扩缩

要实现 Spark-EMR 批处理计算的自动化，您可以使用以下任一选项。

- 实施可在 cron 计划中启动 Lambda 函数的亚马逊 EventBridge 规则。有关更多信息，请参阅教程：使用安排 AWS Lambda 函数。 EventBridge
- 配置Amazon S3 事件通知以在文件到达时启动 Lambda 函数。
- 通过事件主体和 Lambda 环境变量将输入参数传递给 AWS Lambda 函数。

工具

Amazon Web Services

- Amazon EMR 是一个托管集群平台，可简化在 AWS 上运行大数据框架以处理和分析海量数据的操作。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

其他工具

- Apache Spark 是用于大规模数据处理的多语言分析引擎。

操作说明

创建 Amazon EMR 和 Lambda IAM 角色以及 VPC

Task	描述	所需技能
创建 IAM 角色和 VPC。	如果您已经拥有 AWS Lambda 和 Amazon EMR IAM 角色以及 VPC，则可以跳过此步骤。要运行代码，EMR 集群和 Lambda 函数都需要 IAM 角色。EMR 集群还需要一个具有公有子网的 VPC 或带有 NAT 网关的私有子网。要自动创建所有 IAM 角色和 VPC，请按原样部署附加的 AWS CloudFormation 模板，或者您可以按照其他信息部分中的指定手动创建角色和 VPC。	云架构师
注意 AWS CloudFormation 模板的输出密钥。	成功部署 CloudFormation 模板后，在 AWS CloudFormation 控制台中导航至 “输出” 选项卡。请注意五个输出键： • `S3Bucket` • `LambdaExecutionRole` • `ServiceRole` • `JobFlowRole` • `Ec2SubnetId` 在创建 Lambda 函数时，您将使用这些键中的值。	云架构师

上传 Spark .jar 文件

Task	描述	所需技能
上传 Spark .jar 文件。	将 Spark .jar 文件上传到 AWS CloudFormation 堆栈创建的 S3 存储桶。存储桶名称与输出键 `S3Bucket` 相同。	常规 AWS

创建 Lambda 函数来启动 EMR 集群

Task	描述	所需技能
创建一个 Lambda 函数。	在 Lambda 控制台上，创建一个具有执行角色的 Python 3.9+ Lambda 函数。执行角色策略必须允许 Lambda 启动 EMR 集群。（参见随附的 AWS CloudFormation 模板。）	数据工程师、云工程师
复制并粘贴代码。	用此模式其他信息部分中的代码替换 `lambda_function.py` 文件中的代码。	数据工程师、云工程师
更改代码中的参数。	请根据代码中的注释更改参数值，以匹配您的 Amazon Web Services account。	数据工程师、云工程师
启动函数以启动集群。	启动函数以使用提供的 Spark .jar 文件启动瞬态 EMR 集群的创建。它将运行 Spark 作业，并在作业完成后自动终止。	数据工程师、云工程师
检查 EMR 集群状态。	EMR 集群启动后，它会显示在 Amazon EMR 控制台的集群选	数据工程师、云工程师

Task	描述	所需技能
	项卡下。可以相应地检查启动集群或运行作业时出现的任何错误。	

设置并运行示例演示

Task	描述	所需技能
上传 Spark .jar 文件。	从附件部分下载 Spark .jar 文件并将其上传到 S3 存储桶。	数据工程师、云工程师
上传输入数据集。	将附件 `fake_sales_data.csv` 文件上传到 S3 存储桶。	数据工程师、云工程师
粘贴 Lambda 代码并更改参数。	复制工具部分中的代码，然后将代码粘贴到 Lambda 函数中，替换代码 `lambda_function.py` 文件。更改参数值以匹配您的账户。	数据工程师、云工程师
启动函数并验证输出。	在 Lambda 函数使用提供的 Spark 作业启动集群后，它会在 S3 存储桶中生成一个 .csv 文件。	数据工程师、云工程师

相关资源

- Building Spark
- Apache Spark 和 Amazon EMR
- Boto3 Docs run_job_flow 文档
- Apache Spark 信息和文档

其他信息

代码

```
"""
Copy paste the following code in your Lambda function. Make sure to change the
 following key parameters for the API as per your account

-Name (Name of Spark cluster)
-LogUri (S3 bucket to store EMR logs)
-Ec2SubnetId (The subnet to launch the cluster into)
-JobFlowRole (Service role for EC2)
-ServiceRole (Service role for Amazon EMR)

The following parameters are additional parameters for the Spark job itself. Change the
 bucket name and prefix for the Spark job (located at the bottom).

-s3://your-bucket-name/prefix/lambda-emr/SparkProfitCalc.jar (Spark jar file)
-s3://your-bucket-name/prefix/fake_sales_data.csv (Input data file in S3)
-s3://your-bucket-name/prefix/outputs/report_1/ (Output location in S3)
"""
import boto3

client = boto3.client('emr')


def lambda_handler(event, context):
    response = client.run_job_flow(
        Name='spark_job_cluster',
        LogUri='s3://your-bucket-name/prefix/logs',
        ReleaseLabel='emr-6.0.0',
        Instances={
            'MasterInstanceType': 'm5.xlarge',
            'SlaveInstanceType': 'm5.large',
            'InstanceCount': 1,
            'KeepJobFlowAliveWhenNoSteps': False,
            'TerminationProtected': False,
            'Ec2SubnetId': 'subnet-XXXXXXXXXXXXX'
        },
        Applications=[{'Name': 'Spark'}],
        Configurations=[
            {'Classification': 'spark-hive-site',
             'Properties': {
```

```
                'hive.metastore.client.factory.class':
'com.amazonaws.glue.catalog.metastore.AWSGlueDataCatalogHiveClientFactory'}
            }
        ],
        VisibleToAllUsers=True,
        JobFlowRole='EMRLambda-EMREC2InstanceProfile-XXXXXXXXX',
        ServiceRole='EMRLambda-EMRRole-XXXXXXXXX',
        Steps=[
            {
                'Name': 'flow-log-analysis',
                'ActionOnFailure': 'TERMINATE_CLUSTER',
                'HadoopJarStep': {
                    'Jar': 'command-runner.jar',
                    'Args': [
                        'spark-submit',
                        '--deploy-mode', 'cluster',
                        '--executor-memory', '6G',
                        '--num-executors', '1',
                        '--executor-cores', '2',
                        '--class', 'com.aws.emr.ProfitCalc',
                        's3://your-bucket-name/prefix/lambda-emr/SparkProfitCalc.jar',
                        's3://your-bucket-name/prefix/fake_sales_data.csv',
                        's3://your-bucket-name/prefix/outputs/report_1/'
                    ]
                }
            }
        ]
    )
```

IAM 角色和 VPC 创建

要在 Lambda 函数中启动 EMR 集群，需要一个 VPC 和 IAM 角色。您可以使用此模式的 “附件” 部分中的 AWS CloudFormation 模板设置 VPC 和 IAM 角色，也可以使用以下链接手动创建它们。

运行 Lambda 和 Amazon EMR 需要以下 IAM 角色。

Lambda 执行角色

AWS Lambda 函数的执行角色授予该函数访问 Amazon Web Services 和资源的权限。

Amazon EMR 的服务角色

Amazon EMR 角色定义了 Amazon EMR 在配置资源和执行非在集群内运行的亚马逊弹性计算云 (Amazon EC2) 实例环境中执行的服务级别任务时允许执行的操作。例如，服务角色用于在集群启动时配置 EC2 实例。

EC2 实例的服务角色

集群 EC2 实例的服务角色（也称为 Amazon EMR 的 EC2 实例配置文件）是一种特殊类型的服务角色，该角色在 EC2 实例启动时分配给 Amazon EMR 集群中的每个实例。在 Apache Hadoop 上运行的应用程序进程代入该角色来获得与其它 Amazon Web Services 交互的权限。

VPC 和子网创建

您可以从 VPC 控制台创建 VPC。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS Glue 将 Apache Cassandra 工作负载迁移到亚马逊密钥空间

创建者：Nikolai Kolesnikov (AWS)、Karthiga Priya Chandran (AWS) 和 Samir Patel (AWS)

摘要

此模式向您展示了如何在 AWS Glue 上使用将现有 Apache Cassandra 工作负载迁移到亚马逊密钥空间（适用于 Apache Cassandra）。 CQLReplicator 您可以 CQLReplicator 在 AWS Glue 上使用来最大限度地减少将工作负载迁移的复制延迟缩至几分钟。您还将学习如何使用 Amazon Simple Storage Service (Amazon S3)存 储桶来存储迁移所需数据，包括 Apache Parquet 文件、配置文件和脚本。此模式假设您的 Cassandra 工作负载托管在虚拟私有云 (VPC EC2) 中的亚马逊弹性计算云 (Amazon) 实例上。

先决条件和限制

先决条件

- 带源表的 Cassandra 集群
- Amazon Keyspaces 中的目标表，用于复制工作负载
- 用于存储包含增量数据更改的中间 Parquet 文件的 S3 存储桶
- 用于存储作业配置文件和脚本的 S3 存储桶

限制

- CQLReplicator 在 AWS Glue 上，需要一些时间来为 Cassandra 工作负载配置数据处理单元 (DPUs)。Cassandra 集群与 Amazon Keyspaces 中的目标键空间和表之间的复制延迟可能只会持续几分钟。

架构

源技术堆栈

- Apache Cassandra
- DataStax 服务器
- scyllaDB

目标技术堆栈

- Amazon Keyspaces

迁移架构

下图显示了一个示例架构，其中 Cassandra 集群托管在 EC2 实例上并分布在三个可用区。Cassandra 节点托管在私有子网。

图表显示了以下工作流：

1. 自定义服务角色提供对 Amazon Keyspaces 和 S3 存储桶的访问权限。
2. AWS Glue 任务读取 S3 存储桶中的任务配置和脚本。
3. AWS Glue 作业通过端口 9042 连接以从 Cassandra 集群读取数据。
4. AWS Glue 作业通过端口 9142 连接，将数据写入至 Amazon Keyspaces。

工具

Amazon Web Services 和工具

- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- AWS CloudShell 是一个基于浏览器的外壳，您可以使用 AWS 命令行界面 (AWS CLI) Line CLI 和一系列预装的开发工具来管理 AWS 服务。
- AWS Glue 是一项完全托管的 ETL 服务，它可帮助您在各种数据存储和数据流之间可靠地对数据进行分类、清理、扩充和移动。
- Amazon Keyspaces（Apache Cassandra 兼容）是一项托管数据库服务，可帮助您在 Amazon Web Services Cloud 中迁移、运行和扩展 Cassandra 工作负载。

代码

此模式的代码可在 GitHub CQLReplicator存储库中找到。

最佳实践

- 要确定迁移所需的 AWS Glue 资源，请估计源 Cassandra 表中的行数。例如，使用 84 GB 的磁盘，每 0.25 DPU（2 vCPUs，4 GB 内存）有 250 K 行。
- 在运行之前预热 Amazon Keyspaces 表。 CQLReplicator例如，八个 CQLReplicator 图块（AWS Glue 作业）WCUs 每秒最多可以写入 22 K，因此应将目标预热到每秒 25-30 K WCUs 。
- 要启用 AWS Glue 组件之间的通信，请对安全组中的所有 TCP 端口使用自引用入站规则。
- 使用增量流量策略随着时间的推移分配迁移工作负载。

操作说明

部署 CQLReplicator

Task	描述	所需技能
创建目标键空间和表。	1. 在 Amazon Keyspaces 中创建键空间和表。 有关写入容量的更多信息，请参阅此模式的 “其他信息” 部分中的写入单位计算。 您也可以使用 Cassandra Query Language (CQL) 创建键空间。有关更多信息，请参阅此模式的 “其他信息” 部分中的 “使用 CQL 创建密钥空间”。 **Note** 创建表后，请考虑将表切换到按需容量模式，以避免不必要的费用。	应用程序所有者，AWS 管理员，数据库管理员，应用程序开发人员

Task	描述	所需技能
	2. 要更新为吞吐量模式，请运行以下脚本： `ALTER TABLE target_keyspace.target_table WITH CUSTOM_PROPERTIES = { 'capacity_mode': { 'throughput_mode': 'PAY_PER_REQUEST'} }`	

Task	描述	所需技能
将 Cassandra 驱动程序配置为连接至 Cassandra。	使用以下配置脚本： `Datastax-java-driver { basic.request.consistency = “LOCAL_QUORUM” basic.contact-points = [“127.0.0.1:9042”] advanced.reconnect-on-init = true basic.load-balancing-policy { local-datacenter = “datacenter1” } advanced.auth-provider = { class = PlainTextAuthProvider username = “user-at-sample” password = “S@MPLE=PASSWORD=” } }` **Note** 前面的脚本使用 Spark Cassandra 连接器。有关更多信息，请参阅 Cassandra 的参考配置。	数据库管理员

Task	描述	所需技能
将 Cassandra 驱动程序配置为连接至 Amazon Keyspaces。	使用以下配置脚本： `datastax-java-driver { basic { load-balancing-policy { local-datacenter = us-west-2 } contact-points = ["cassandra.us-west-2.amazonaws.com:9142"] request { page-size = 2500 timeout = 360 seconds consistency = LOCAL_QUORUM } } advanced { control-connection { timeout = 360 seconds } session-leak.threshold = 6 connection { connect-timeout = 360 seconds init-query-timeout = 360 seconds warn-on-init-error = false } auth-provider = { class = software.aws.mcs.auth.SigV4AuthProvider aws-region = us-west-2`	数据库管理员

Task	描述	所需技能

	<pre>}
	
	ssl-engine-factory {
	class = DefaultSs
	lEngineFactory
	}
	}
	}</pre>
	Note
	前面的脚本使用 Spark Cassandra 连接器。有关更多信息，请参阅 Cassandra 的参考配置。	

Task	描述	所需技能
为 AWS Glue 作业创建 IAM 角色。	创建一个`glue-cassandra-migration` 以 AWS Glue 作为可信实体命名的新 AWS 服务角色。 **Note** `glue-cassandra-migration` 应提供对 S3 存储桶和 Amazon Keyspaces 的读写权限。S3 存储桶包含.jar 文件、Amazon Keyspaces 和 Cassandra 的配置文件以及中间 Parquet 文件。例如，它包含`AWSGlueServiceRole` `AmazonS3FullAccess` 、和`AmazonKeyspacesFullAccess` 托管策略。	AWS DevOps

Task	描述	所需技能

CQLReplicator 在 AWS 中下载 CloudShell。	通过运行以下命令将项目下载到您的主文件夹：
	`git clone https://github.com/aws-samples/cql-replicator.git`
	`cd cql-replicator/glue`
	`# Only for AWS CloudShell, the bc package includes bc and dc. Bc is an arbitrary precision numeric processing arithmetic language`
	`sudo yum install bc -y`	

修改参考配置文件。	将`CassandraConnector.conf` 和复制`Keyspaces Connector.conf` 到项目文件夹中的`../glue/conf` 目录中。	AWS DevOps

Task	描述	所需技能

启动迁移过程。	以下命令初始化 CQLReplicator 环境。初始化包括复制.jar 工件、创建 AWS Glue 连接器、S3 存储桶、AWS Glue 任务、`migration` 密钥空间和表：`ledger`
	
	`cd cql-replicator/glue/bin`
	`./cqlreplicator --state init --sg '"sg-1","sg-2"' \ --subnet "subnet-XXXXXXXXXXXX" \ --az us-west-2a --region us-west-2 \ --glue-iam-role glue-cassandra-migration \ --landing-zone s3://cql-replicator-1234567890-us-west-2`
	
	此脚本包含以下参数：
	
	• `--sg`— 允许从 AWS Glue 访问 Cassandra 集群并包含所有流量的自引用入站规则的安全组
	• `--subnet`— Cassandra 集群所属的子网
	• `--az`— 子网的可用区
		AWS DevOps

Task	描述	所需技能
	• `--region`— 部署 Cassandra 集群的 AWS 区域 • `--glue-iam-role` — AWS Glue 在代表你调用 Amazon Keyspaces 和 Amazon S3 时可以假设的 IAM 角色权限 • `--landing zone`— 用于重复使用 S3 存储桶的可选参数（如果您不为该`--landing zone`参数提供值，则该`init`过程将尝试创建一个新的存储桶来存储配置文件、.jar 工件和中间文件。）	
验证部署。	运行上一个命令后，AWS 账户应包含以下内容： • CQLReplicator AWS Glue 任务和 AWS Glue 中的 AWS Glue 连接器 • 存储工件的 S3 存储桶 • 目标密钥空间`migration`和 Amazon Keyspaces 中的`ledger`表	AWS DevOps

跑 CQLReplicator

Task	描述	所需技能
开始迁移过程。	要 CQLReplicator 在 AWS Glue 上操作，您需要使用`--`	AWS DevOps

Task	描述	所需技能
	`state run`命令和一系列参数。这些参数的精确配置主要取决于您的独特迁移需求。例如，如果您选择复制生存时间 (TTL) 值和更新，或者将超过 1 MB 的对象卸载到 Amazon S3，则这些设置可能会有所不同。 要将工作负载从 Cassandra 集群复制到 Amazon Keyspaces，请运行以下命令： `./cqlreplicator --state run --tiles 8 \ --landing-zone s3://cql-replicator-1234567890-us-west-2 \ --region us-west-2 \ --src-keyspace source_keyspace \ --src-table source_table \ --trg-keyspace taget_keyspace \ --writetime-column column_name \ --trg-table target_table --inc-traffic` 您的源密钥空间和表位于 Cassandra 集群`source_ke`	

Task	描述	所需技能
	yspace.source_tabl e 中。您的目标密钥空间和表位target_keyspace.ta rget_table 于 Amazon 密钥空间中。该参数--inc-tra ffic 有助于防止增量流量因大量请求而使 Cassandra 集群和 Amazon Keyspaces 过载。 要复制更新，请--writeti me-column regular_c olumn_name 添加到命令行。常规列将用作写入时间戳的来源。	

监控迁移过程

Task	描述	所需技能
在历史迁移阶段验证迁移的 Cassandra 行。	要获取回填阶段复制的行数，请运行以下命令： `./cqlreplicator --state stats \ -- landing-zone s3://cql-replicator-1234567890-us-west-2 \ --src-keyspace source_keyspace --src-table source_table --region us-west-2`	AWS DevOps

停止迁移过程

Task	描述	所需技能
使用cqlreplicator 命令或 AWS Glue 控制台。	要正常停止迁移过程，请运行以下命令： `./cqlreplicator --state request-stop --tiles 8 \ --landing-zone s3://cql-replicator-1234567890-us-west-2 \ --region us-west-2 \ --src-keyspace source_keyspace --src-table source_table` 要立即停止迁移过程，请使用 AWS Glue 控制台。	AWS DevOps

清理

Task	描述	所需技能
删除已部署的资源。	以下命令将删除 AWS Glue 任务、连接器、S3 存储桶和密钥空间表：ledger `./cqlreplicator --state cleanup --landing-zone s3://cql-replicato`	AWS DevOps

Task	描述	所需技能
	`r-1234567890-us-we st-2`	

故障排除

事务	解决方案
AWS Glue 任务失败并返回内存不足 (OOM) 错误。	1. 更改工作人员类型（向上扩展）。例如，更改`G0.25X`为`G.1X`或改`G.1X`为`G.2X`。或者，增加 DPUs 每个 AWS Glue 任务的数量（向外扩展）CQLReplicator。 2. 从迁移过程中断的地方开始迁移过程。要重新启动失败的 CQLReplicator 作业，请使用相同的参数重新运行该`--state run`命令。

相关资源

- CQLReplicator 使用 AWS Glue README.MD
- AWS Glue 文档
- Amazon Keyspaces 文档
- 阿帕奇·卡桑德拉

其他信息

迁移注意事项

您可使用 AWS Glue 将 Cassandra 工作负载迁移到 Amazon Keyspaces，同时在迁移过程中保持您的 Cassandra 源数据库完全正常运行。复制完成后，您可选择将应用程序割接到 Amazon Keyspaces，同时将 Cassandra 集群和 Amazon Keyspaces 之间的复制延迟降至最低（少于几分钟）。为了保持数据一致性，您还可以使用类似的管线将数据从 Amazon Keyspaces 复制回至 Cassandra 集群。

编写单位计算

举个例子，假定您打算在一小时内写入 500,000,000，行大小为 1 KiB。您需要的 Amazon Keyspaces 写入单元总数 (WCUs) 是基于以下计算得出的：

```
(number of rows/60 mins 60s) 1 WCU per row = (500,000,000/(60*60s) * 1 WCU)
= 69,444 WCUs required
```

WCUs 每秒 69,444 是 1 小时的速度，但你可以为开销增加一些缓冲。 例如，`69,444 * 1.10 = 76,388` WCUs 有 10% 的开销。

使用 CQL 创建密钥空间

要通过使用 CQL 创建键空间，请运行以下命令：

```
CREATE KEYSPACE target_keyspace WITH replication = {'class': 'SingleRegionStrategy'}
CREATE TABLE target_keyspace.target_table ( userid uuid, level text, gameid int,
 description text, nickname text, zip text, email text, updatetime text, PRIMARY KEY
 (userid, level, gameid) ) WITH default_time_to_live = 0 AND CUSTOM_PROPERTIES =
 {'capacity_mode':{ 'throughput_mode':'PROVISIONED', 'write_capacity_units':76388,
 'read_capacity_units':3612 }} AND CLUSTERING ORDER BY (level ASC, gameid ASC)
```

使用迁移器将 Hadoop 数据迁移到 Amazon S3 WANdisco LiveData

创建者：Tony Velcich

摘要

此模式描述了将 Apache Hadoop 数据从 Hadoop Distributed File System (HDFS) 迁移到 Amazon Simple Storage Service (Amazon S3) 的过程。它使用 M WANdisco LiveData igrator 来自动执行数据迁移过程。

先决条件和限制

先决条件

- 将在其中安装 M LiveData igrator 的 Hadoop 群集边缘节点。节点应满足以下要求：
  - 最低规格：4 CPUs、16 GB 内存、100 GB 存储空间。
  - 最低网速为 2 Gbps。
  - 边缘节点上可访问端口 8081，用于访问 WANdisco 用户界面。
  - Java 1.8 64 位。
  - 安装在边缘节点的 Hadoop 客户端库。
  - 能够以 HDFS 超级用户身份进行身份验证（例如，“hdfs”）。
  - 如果在 Hadoop 集群上启用了 Kerberos，则边缘节点上必须有一个包含适用于 HDFS 超级用户的主体的有效密钥表。
- 可访问 S3 存储桶的有效的 Amazon Web Services account。
- 在本地 Hadoop 集群（特别是边缘节点）和 AWS 之间建立的 AWS Direct Connect 链接。

产品版本

- LiveData Migrator 1.8.6
- WANdisco 用户界面 (OneUI) 5.8.0

架构

源技术堆栈

- 本地 Hadoop 集群

目标技术堆栈

- Amazon S3

架构

下图显示了 M LiveData igrator 解决方案的体系结构。

该工作流由四个主要组件组成，用于将数据从本地 HDFS 迁移到 Amazon S3。

- LiveData 迁移器 — 自动将数据从 HDFS 迁移到 Amazon S3，并驻留在 Hadoop 集群的边缘节点上。
- HDFS – 分布式文件系统，可提供对应用程序数据的高吞吐量访问。
- Amazon S3 – 一种对象存储服务，提供可扩展性、数据可用性、安全性和性能。
- AWS Direct Connect – 一种服务，建立从您的本地数据中心至 AWS 的专用网络连接。

自动化和扩缩

您通常会创建多个迁移，以便您可按路径或目录从源文件系统中选择特定内容。您还可以通过定义多个迁移资源，将数据同时迁移到多个独立的文件系统。

操作说明

在您的 Amazon Web Services account 中配置 Amazon S3 存储

Task	描述	所需技能
登录您的 Amazon Web Services account。	登录 AWS 管理控制台并在上打开 Amazon S3 控制台 https://console.aws.amazon.com/s3/。	AWS 体验
创建 S3 存储桶。	如果您还没有现有 S3 存储桶可用作目标存储，请在 Amazon S3 控制台上选择创建存储桶选项，然后指定	AWS 体验

Task	描述	所需技能
	存储桶名称、AWS 区域和存储桶设置以阻止公有访问。AWS 并 WANdisco 建议您为 S3 存储桶启用阻止公有访问选项，并设置存储桶访问和用户权限策略以满足您组织的要求。AWS 示例可在以下网址中找walkthroughs-managing-access-example到 https://docs.aws.amazon.com/AmazonS3/ latest/dev/example-1.html。	

安装 LiveData 迁移器

Task	描述	所需技能
下载 LiveData 迁移器安装程序。	下载 LiveData 迁移器安装程序并将其上传到 Hadoop 边缘节点。你可以在 /aws.amazon 上下载 M LiveData igrator 的免费试用版。 https://www2.wandisco.com/ldm-trial. You can also obtain access to LiveData Migrator from AWS Marketplace, at https:/ com/marketplace/pp/B07B8SZND9。	Hadoop 管理员、应用程序所有者
安装 LiveData 迁移器。	使用下载的安装程序，在 Hadoop 群集的边缘节点上以 HDFS 超级用户身份安装 M LiveData igrator。有关安装命令，请参阅“其他信息”部分。	Hadoop 管理员、应用程序所有者

Task	描述	所需技能
检查 M LiveData igrator 和其他服务的状态。	使用 “其他信息” 部分 LiveData 中提供的命令检查 Migrator、Hive 迁移器和 WANdisco 用户界面的状态。	Hadoop 管理员、应用程序所有者

通过 WANdisco UI 配置存储

Task	描述	所需技能
注册您的 LiveData 迁移者账户。	通过端口 8081（在 Hadoop 边缘节点上）上的 Web 浏览器登录 WANdisco 用户界面，并提供您的详细信息以进行注册。例如，如果你在名为 myldmhost.example.com 的主 LiveData 机上运行 Migrator，则网址将是：http://myldmhost.example.com:8081	应用程序所有者
配置您的源 HDFS 存储。	提供您的源 HDFS 存储所需配置详细信息。这将包括 fs.defaultFS 值和用户定义的存储名称。如果启用了 Kerberos，请提供主体和密钥表位置以供 M LiveData igrator 使用。如果在集群上启用了 NameNode HA，请提供边缘节点上的 core-site.xml 和 hdfs-site.xml 文件的路径。	Hadoop 管理员、应用程序所有者
配置您的目标 Amazon S3 存储。	将目标存储添加至 S3a 类型。提供用户定义的存储名称与 S3 存储桶名称。在凭证提供者选项中输入 “org.apache.hadoop.fs.s3a.Simple Provider”，然后	AWS、应用程序所有者

Task	描述	所需技能
	AWSCredentials提供 S3 存储桶的 AWS 访问权限和密钥。还需要其他 S3a 属性。有关详细信息，请参阅 M LiveData igrator 文档中的 “s3a 属性” 部分，网址为 docs/command-reference https://docs.wandisco.com/live-data-migrator/ /#3a。filesystem-add-s	

准备迁移

Task	描述	所需技能
添加排除项（如果需要）。	如果要从迁移中排除特定数据集，请为源 HDFS 存储添加排除项。这些排除可以基于文件大小、文件名（基于正则表达式模式）和修改日期。	Hadoop 管理员、应用程序所有者

创建和启动迁移

Task	描述	所需技能
创建并配置迁移。	在 WANdisco 用户界面的控制面板中创建迁移。选择您的源 (HDFS) 和目标（S3 存储桶）。添加您在上一步中定义的新排除项。选择覆盖或如果大小匹配则跳过选项。在所有字段都填写完毕后创建迁移。	Hadoop 管理员、应用程序所有者
启动迁移。	在控制面板上，选择您创建的迁移。单击以启动迁移。您还	应用程序所有者

Task	描述	所需技能
	可以通过在创建迁移时选择自动启动选项来自动启动迁移。	

管理带宽（可选）

Task	描述	所需技能
设置源和目标之间的网络带宽限制。	在控制面板的存储列表中，选择您的源存储，然后在分组列表中选择带宽管理。清除无限制选项，然后定义最大带宽限制和单位。选择“应用”。	应用程序所有者、联网

监控和管理迁移

Task	描述	所需技能
使用 WANdisco 用户界面查看迁移信息。	使用 WANdisco 用户界面查看许可证、带宽、存储和迁移信息。UI 还提供了通知系统，因此您可接收有关错误、警告或重要使用里程碑的通知。	Hadoop 管理员、应用程序所有者
停止、恢复和删除迁移。	您可通过将迁移置于 STOPPED 状态来阻止迁移向其目标传输内容。可以恢复已停止迁移。处于 STOPED 状态迁移也可能是已删除。	Hadoop 管理员、应用程序所有者

相关资源

- LiveData 迁移器文档
- LiveData AWS Marketplace 中的迁移者

- WANdisco LiveData 迁移器演示（视频）

其他信息

安装 LiveData 迁移器

假设安装程序位于您的工作目录中，则可以使用以下命令来安装 M LiveData igrator：

```
su - hdfs
chmod +x livedata-migrator.sh && sudo ./livedata-migrator.sh
```

安装后检查 M LiveData igrator 和其他服务的状态

使用以下命令检查 LiveData 迁移器、Hive 迁移器和用户界面的状态： WANdisco

```
service livedata-migrator status
service hivemigrator status
service livedata-ui status
```

将 Oracle 商业智能 12c 从本地服务器迁移到 Amazon Web Services Cloud

由 Lanre（Lan-Ray）showunmi（AWS）和 Patrick Huang（AWS）创作

摘要

此模式展示了如何使用 AWS 将 Oracle 商业智能企业版 12c 从本地服务器迁移到 AWS CloudFormation 云。它还介绍了如何使用其他 Amazon Web Services 来实施 Oracle BI 12c 组件，这些组件可提供高可用性、安全性、灵活性和动态扩展能力。

有关将 Oracle BI 12c 迁移到 Amazon Web Services Cloud 相关的最佳实践列表，请参阅此模式的其他信息部分。

> **Note**
>
> 在将现有 Oracle BI 12c 数据传输到云端之前，最好先运行多次测试迁移。这些测试可帮助您微调迁移方法，识别和修复潜在问题，并更准确地估计停机时间需求。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 通过 AWS 虚拟专用网络 (AWS VPN) 服务或 AWS Direct Connect 的本地服务器和 AWS 之间的安全网络连接
- 适用于您的 Oracle 操作系统、Oracle BI 12c、甲骨文数据库、甲骨文服务器和 Oracle HTTP WebLogic 服务器的软件许可证

限制

有关存储大小限制的信息，请参阅 Amazon Relational Database Service（Amazon RDS）for Oracle 文档。

产品版本

- Oracle 商业智能企业版 12c

- 甲骨文 WebLogic 服务器 12c
- Oracle HTTP 服务器 12c
- Oracle 数据库 12c（或更高版本）
- Oracle Java SE 8

架构

下图显示了在 Amazon Web Services Cloud 中运行 Oracle BI 12c 组件的示例架构：

此图显示以下架构：

1. Amazon Route 53 提供域名服务 (DNS) 配置。
2. 弹性负载均衡（ELB）可分发网络流量，以提高 Oracle BI 12c 组件在多个可用区中的可扩展性和可用性。
3. 亚马逊弹性计算云 (亚马逊 EC2) Auto Scaling 组在多个可用区托管 Oracle HTTP 服务器、Weblogic 管理服务器和托管 BI 服务器。
4. 适用于 Oracle 数据库的 Amazon Relational Database Service（Amazon RDS）跨多个可用区存储 BI 服务器元数据。
5. Amazon Elastic File System（Amazon EFS）安装在每个 Oracle BI 12c 组件上，用于共享文件存储。

技术堆栈

- Amazon Elastic Block Store（Amazon EBS）
- 亚马逊弹性计算云（亚马逊 EC2）
- Amazon Elastic File System (Amazon EFS)
- Amazon RDS for Oracle
- AWS Certificate Manager (ACM)
- 弹性负载均衡（ELB）
- Oracle BI 12c
- 甲骨文 WebLogic 服务器 12c
- Oracle HTTP Server (OHS)

工具

- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。
- AWS Certificate Manager（ACM）可帮助您创建、存储和续订公有及私有 SSL/TLS X.509 证书和密钥，这些证书和密钥可保护 AWS 网站和应用程序。
- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到 Amazon Web Services Cloud，或者在云和本地设置的组合之间迁移。
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- Amazon A EC2 uto Scaling 可帮助您保持应用程序的可用性，并允许您根据自己定义的条件自动添加或删除亚马逊 EC2 实例。
- Amazon Elastic File System（Amazon EFS）可帮助您在 Amazon Web Services Cloud 中创建和配置共享文件系统。
- 弹性负载均衡将传入的应用程序或网络流量分发到多个目标。例如，您可以跨亚马逊弹性计算云 (Amazon EC2) 实例、容器以及一个或多个可用区中的 IP 地址分配流量。
- Amazon Relational Database Service (Amazon RDS) 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展关系数据库。
- Amazon Simple Storage Service(Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。
- Oracle 数据泵可帮助您将数据和元数据从一个数据库高速移动至另一个数据库。
- Oracle 融合中间件是一套用于身份管理、协作和商业智能报告的应用程序开发工具和集成解决方案。
- Oracle GoldenGate 可帮助您在 Oracle 云基础设施中设计、运行、编排和监控数据复制和流数据处理解决方案。
- Oracle WebLogic 脚本工具 (WLST) 提供了一个命令行界面，可帮助您水平扩展集群。 WebLogic

操作说明

评测源环境

Task	描述	所需技能
收集软件库存信息。	识别源技术堆栈中每个软件组件的版本和补丁级别，包括以下内容： • Oracle 操作系统 • Oracle Database • Oracle BI 12c • 甲骨文 WebLogic 服务器 • Oracle HTTP 服务器 • Java	迁移架构师、解决方案架构师、应用程序所有者、Oracle BI 管理员
收集计算和存储库存信息。	在源环境中，查看以下各项的当前和历史使用率指标： • CPU 使用率 • 内存使用量 • 存储空间使用量 ⚠ Important 请务必考虑使用量的历史峰值。	迁移架构师、解决方案架构师、应用程序所有者、Oracle BI 管理员、系统管理员
收集有关源环境架构及其要求的信息。	全面了解源环境的架构及其要求，包括以下方面的知识： • Oracle WebLogic 服务器域配置 • 集群 • 负载均衡	迁移架构师、解决方案架构师、应用程序所有者、Oracle BI 管理员

Task	描述	所需技能
	• 连接 • 可用性 • 灾难恢复要求	
识别 Java 数据库连接 (JDBC) 数据来源。	收集有关源环境的 JDBC 数据来源及其使用的每个数据库引擎的驱动程序的信息。	迁移架构师、应用程序所有者、Oracle BI 管理员、数据库工程师或管理员
收集有关特定环境设置的信息。	收集有关特定于源环境的设置和配置的信息，包括： • 自定义启动和关闭脚本 • Java 和其他环境变量 • 证书	迁移架构师、解决方案架构师、应用程序所有者、Oracle BI 管理员
确定对其他应用程序的任何依赖项。	收集有关源环境中各项集成的信息，这些集成使用其他应用程序创建依赖项。 **Important** 请务必确定任何轻型目录访问协议 (LDAP) 集成和其他网络要求。	迁移架构师、解决方案架构师、应用程序所有者、Oracle BI 管理员

设计目标环境

Task	描述	所需技能
创建高级设计文档。	创建目标架构设计文档。请务必使用在评测源环境时收集的信息来为设计文档提供信息。	解决方案架构师、应用程序架构师、数据库工程师、迁移架构师

Task	描述	所需技能
获得设计文档的批准。	与利益相关者一起审查设计文件并获得所需批准。	应用程序或服务所有者、解决方案架构师、应用程序架构师

部署基础设施

Task	描述	所需技能
在中准备基础架构代码 CloudFormation。	创建 CloudFormation 模板以在 AWS 云中配置您的 Oracle BI 12c 基础设施。 有关更多信息，请参阅 AWS CloudFormation 用户指南中的使用 AWS CloudFormation 模板。 **Note** 最佳做法是为每个 Oracle BI 12c 层创建模块化 CloudFormation 模板，而不是为所有资源创建一个大型模板。有关 CloudFormation 最佳实践的更多信息，请参阅 AWS 博客 CloudFormation上的 AWS 自动部署时的 8 个最佳实践。	云基础设施架构师、解决方案架构师、应用程序架构师
下载所需软件。	DDownload 以下软件以及来自 Oracle 网站的所需版本和补丁：	迁移架构师、数据库工程师、应用程序架构师

Task	描述	所需技能
	• Java JDK8 • 甲骨文 WebLogic 服务器 12c • Oracle BI 12c	
准备安装脚本。	创建运行静默安装的软件安装脚本。这些脚本简化了部署自动化。 有关更多信息，请参阅 Oracle Support 网站上的 OBIEE 12c：如何执行静默安装？。您需要 Oracle Support 账户才能查看文档。	迁移架构师、数据库工程师、应用程序架构师
为 Web 和应用程序层创建由 Amazon EBS 提供支持的 Linux AMI。	1. 为您的 Web 和应用程序层部署和配置 Amazon EC2 实例。确保实例满足运行以下内容的先决条件： • Oracle 操作系统环境设置 • Oracle 操作系统用户账户设置 • Java 软件安装 2. 创建实例的 Amazon 系统映像 (AMIs) 并保存副本以备将来使用。有关说明，请参阅亚马逊 Linux 实例 EC2 用户指南中的创建亚马逊 EBS 支持的 Linux AMI。	迁移架构师、数据库工程师、应用程序架构师

Task	描述	所需技能
使用启动您的 AWS 基础设施 CloudFormation。	使用您创建的 CloudFormation 模板，在模块中部署 Oracle BI 12c Web 和应用程序层。 有关说明，请参阅 AWS CloudFormation 用户指南 CloudFormation中的 AWS 入门。	云基础设施架构师、解决方案架构师、应用程序架构师

使用全新安装将 Oracle BI 12c 迁移到 AWS

Task	描述	所需技能
准备所需软件。	将所需的软件暂存到可供 Amazon EC2 实例访问的位置。例如，您可以将软件暂存到 Amazon S3 或其他可供您的 Web 和应用程序服务器访问的 Amazon EC2 实例中。	迁移架构师、Oracle BI 架构师、云基础设施架构师、解决方案架构师、应用程序架构师
为安装 Oracle BI 12c 准备存储库数据库。	通过对新的 Amazon RDS for Oracle 数据库实例运行 Oracle 存储库创建实用程序 (RCU) 来创建 Oracle BI 12c 架构。	云基础设施架构师、解决方案架构师、应用程序架构师、迁移架构师、Oracle BI 架构师
安装 Oracle 融合中间件 12c 和 Oracle BI 12c。	1. 从一个亚马逊 EC2 实例开始，安装 Oracle Fusion Middleware 12c 基础设施和 OBIEE 12c。有关更多信息，请参阅适用于 Oracle 商业智能的 Oracle 融合中间件企业部署指南的以下部分： • 启动基础架构安装程序 BIHOST1	迁移架构师、Oracle BI 架构师

Task	描述	所需技能
	• 安装 Oracle 商业智能为企业部署做准备 **Note** 使用 Amazon EFS 托管将在 Oracle BI 12c 集群节点之间共享的目录。 2. 在安装中应用所有必需的补丁。 3. 创建 AMIs 实例并保存副本以备将来使用。	
为 Oracle BI 12c 配置你的 Oracle WebLogic 服务器域。	将 Oracle BI 12c 域配置为非集群部署。 有关更多信息，请参阅适用于 Oracle 商业智能的 Oracle 融合中间件企业部署指南中的配置 BI 域。	迁移架构师、Oracle BI 架构师
对 Oracle BI 12c 进行水平横向扩展。	将单个节点水平横向扩展到所需节点数。 有关更多信息，请参阅适用于 Oracle 商业智能的 Oracle 融合中间件企业部署指南中的横向扩展 Oracle 商业智能。	迁移架构师、Oracle BI 架构师

Task	描述	所需技能
安装 Oracle HTTP 服务器 12c。	1. 在 Oracle Web 层亚马逊 EC2 实例上安装 Oracle HTTP Server 12c。有关说明，请参阅为 Oracle Access Management 12c 安装和配置 Oracle HTTP 服务器中的安装 Oracle HTTP 服务器 12c。 2. 在安装中应用所有必需的补丁。 3. 创建 AMIs 实例并保存副本以备将来使用。	迁移架构师、Oracle BI 架构师
为 SSL 终止配置负载均衡器。	1. 创建或在 ACM 中导入证书。 2. 将 SSL 证书与 ELB 关联。	云基础设施架构师、迁移架构师
将商业智能元数据构件迁移到 AWS。	1. 从本地 Oracle BI 12c 安装中导出 Oracle 商业智能应用程序档案（BAR）文件。要导出 BAR 文件，请使用WebLogic 脚本工具 (WLST) 运行exportServiceInstance命令。 2. 将本地 BAR 文件导入 AWS Oracle BI 12c 安装中。要导入 BAR 文件，请运行 importServiceInstanceWLST 命令。	迁移架构师、Oracle BI 架构师

Task	描述	所需技能
执行迁移后的任务。	导入 BAR 文件后，请执行以下操作： • 配置任何其他 JDBC 数据来源。 • 为 PostgreSQL 或 Amazon Redshift 等其他数据来源安装驱动程序。 • 配置 Oracle LDAP、SSL、单点登录 (SSO) 和WebLogic 安全存储。 • 配置 AWS Identity and Access Management (IAM) 策略。 • 激活使用情况跟踪。 • 设置与其他系统的集成。 • 迁移所有自定义脚本。	迁移架构师、Oracle BI 架构师

测试新环境

Task	描述	所需技能
测试新的 Oracle BI 12c 环境。	在新的 Oracle BI 12c 环境上进行 end-to-end测试。尽可能多地使用自动化。 测试活动示例包括以下内容： • 验证仪表板、报告和 URLs • 用户验收测试（UAT） • 操作验收测试 (OAT)	迁移架构师、解决方案架构师、应用程序所有者、Oracle BI 管理员

Task	描述	所需技能
	Note 根据需要进行其他测试和验证。	

割接至新环境

Task	描述	所需技能
断开流向本地 Oracle BI 12c 环境的流量。	在指定的割接窗口中，停止所有流向本地 Oracle BI 12c 环境的流量。	迁移架构师、解决方案架构师、应用程序所有者、Oracle BI 管理员
将新的 Oracle BI 12c 存储库数据库与源数据库重新同步。	将 Amazon RDS Oracle Oracle BI 12c 存储库数据库与本地数据库重新同步。 要同步数据库，您可以使用 Oracle 数据泵刷新或 AWS DMS 更改数据捕获（CDC）。	Oracle BI 管理员、数据库工程师/管理员
将您的 Oracle BI 12c 切换 URLs 到新的 AWS 环境。	更新内部 DNS 服务器 URLs 上的 Oracle BI 12c，使其指向新安装的 AWS。	迁移架构师、解决方案架构师、应用程序所有者、Oracle BI 管理员
监控新环境。	使用以下任一工具监控新的 Oracle BI 12c 环境： • Amazon CloudWatch • Amazon RDS 性能详情 • Oracle Enterprise Manager	Oracle BI 管理员、数据库工程师/管理员、应用程序管理员
获得项目签核。	与利益相关者一起审查测试结果，并获得完成迁移所需批准。	应用程序所有者、服务所有者、云基础设施架构师、迁移架构师、Oracle BI 架构师

相关资源

- 在 RDS for Oracle 上使用 Oracle 存储库创建实用程序（Amazon RDS 用户指南）
- Amazon RDS 上的 Oracle（Amazon RDS 用户指南）
- AWS 上 WebLogic 的 Oracle Server 12c（A WS 白皮书）
- 部署 Oracle 商业智能以实现高可用性（Oracle 帮助中心）
- Oracle 商业智能应用程序档案（BAR）文件（Oracle 帮助中心）
- 如何在环境之间迁移 OBI 12c（Oracle Support）

其他信息

以下是与将 Oracle BI 12c 迁移到 Amazon Web Services Cloud 相关的最佳实践列表。

存储库数据库

在 Amazon RDS for Oracle 实例上托管 Oracle BI 12c 数据库架构是一种最佳实践。此实例类型提供经济实惠、且可调整的容量，同时自动执行管理任务，例如硬件预调配、数据库设置、修补和备份。

有关更多信息，请参阅 Amazon RDS 用户指南中的在 RDS for Oracle 上使用 Oracle 存储库创建实用程序。

Web 和应用程序层

内存优化的 Amazon EC2 实例通常非常适合 Oracle BI 12c 服务器。无论您选择哪种实例类型，务必确保您预调配的实例满足系统的内存使用要求。此外，请确保根据您的 Amazon EC2 实例的可用内存配置足够的 J WebLogic ava 虚拟机 (JVM) 堆大小。

本地存储

I/O 在 Oracle BI 12c 应用程序的整体性能中起着重要作用。Amazon Elastic Block Store（Amazon EBS）提供针对不同工作负载模式进行优化的不同存储类别。请务必选择适合用例的 Amazon EBS 卷类型。

有关 EBS 卷类型的更多信息，请参阅 Amazon EBS 文档中的 Amazon EBS 功能。

共享存储

集群化的 Oracle BI 12c 域需要共享存储空间来存储以下资源：

- 配置文件

- Oracle BI 12c 单例数据目录 (SDD)
- Oracle 全局缓存
- Oracle BI 计划程序脚本
- 甲骨文 WebLogic 服务器二进制文件

您可以使用 Amazon EFS 来满足这一共享存储需求，它提供了可扩展、完全托管的弹性网络文件系统 (NFS) 文件系统。

微调共享存储性能

Amazon EFS 有两种吞吐量模式：预调配和突增。该服务还有两种性能模式：通用模式和最大 I/O 模式。

要微调性能，请首先在通用性能模式和预调配吞吐量模式下测试工作负载。进行这些测试将帮助您确定这些基准模式是否足以满足所需服务级别。

有关更多信息，请参阅 Amazon EFS 用户指南中的 Amazon EFS 性能。

可用性和灾难恢复

最佳做法是跨多个可用区部署 Oracle BI 12c 组件，以便在可用区出现故障时保护这些资源。以下是 Amazon Web Services Cloud 中托管的特定 Oracle BI 12c 资源的可用性和灾难恢复最佳实践列表：

- Oracle BI 12c 存储库数据库：将多可用区 Amazon RDS 数据库实例部署到Oracle BI 12c 存储库数据库。在多可用区部署中，Amazon RDS 会自动在不同可用区中预调配和维护一个同步备用副本。在计划内的系统维护期间，跨可用区运行 Oracle BI 12c 存储库数据库实例可以提高可用性，并帮助保护数据库以防数据库实例发生故障和可用区中断。
- Oracle BI 12c 托管服务器：为了实现容错能力，最佳做法是在配置为跨多个可用区域的 Amazon A EC2 uto Scaling 组中的托管服务器上部署 Oracle BI 12c 系统组件。Auto Scaling 会根据亚马逊运行 EC2 状况检查替换故障实例。如果可用区出现故障，Oracle HTTP 服务器会继续将流量引导到正常运行的可用区内的托管服务器。然后，自动扩缩会启动实例以满足主机数量要求。建议激活 HTTP 会话状态复制，以帮助确保现有会话顺畅地失效转移到正常运行的托管服务器。
- Oracle BI 12c 管理服务器：为确保您的管理服务器具有高可用性，请将其托管在配置为跨多个可用区域的 Amazon A EC2 uto Scaling 组中。然后，将组的最小和最大大小设置为 1。如果可用区出现故障，Amazon A EC2 uto Scaling 会在备用可用区中启动一台替换的管理服务器。要恢复同一可用区内任何出现故障的底层主机，您可以激活 Amazon A EC2 uto Recovery。
- Oracle Web 层服务器：最佳做法是将您的 Oracle HTTP 服务器与 Oracle WebLogic 服务器域关联起来。为了获得高可用性，请在配置为跨越多个可用区域的 Amazon A EC2 uto Scaling 组中部署

Oracle HTTP 服务器。然后，将服务器放在 ELB 弹性负载均衡器后面。要提供针对主机故障的额外保护，您可以激活 Amazon EC2 自动恢复。

可扩展性

Amazon Web Services Cloud 的弹性可帮助您根据工作负载要求水平或垂直扩展应用程序。

垂直扩展

要垂直扩展您的应用程序，您可以更改运行 Oracle BI 12c 组件的 Amazon EC2 实例的大小和类型。您无需在部署开始时过度配置实例，也无需产生不必要的成本。

横向扩展

Amazon A EC2 uto Scaling 可根据工作负载要求自动添加或删除托管服务器，从而帮助您水平扩展应用程序。

> **Note**
>
> 使用 Amazon A EC2 uto Scaling 进行横向扩展需要脚本编写技能和全面的测试才能实施。

备份和恢复

以下是 Amazon Web Services Cloud 中托管的特定 Oracle BI 12c 资源的备用和恢复最佳实践列表：

- Oracle 商业智能元数据存储库：Amazon RDS 会自动创建并保存数据库实例的备份。这些备份会保留您指定的时间。请务必根据数据保护要求配置 Amazon RDS 备份持续时间和保留期设置。有关更多信息，请参阅 Amazon RDS 备份和恢复。
- 托管服务器、管理服务器和 Web 层服务器：确保根据数据保护和保留要求配置 Amazon EBS 快照。
- 共享存储：您可以使用 AWS Backup 管理存储在 Amazon EFS 中的文件的备份和恢复。还可以部署 AWS Backup 服务来集中管理其他服务的备份和恢复，包括亚马逊 EC2、亚马逊 EBS 和 Amazon RDS。有关更多信息，请参阅什么是 AWS Backup？ 在 AWS Backup 开发人员指南中。

安全与合规

以下是可以帮助您保护 Amazon Web Services Cloud 中的 Oracle BI 12c 应用程序的安全最佳实践和 Amazon Web Services 列表：

- 静态加密：Amazon RDS、Amazon EFS 和 Amazon EBS 都支持行业标准加密算法。您可以使用 AWS Key Management Service（AWS KMS）来创建和管理加密密钥，并控制其在 Amazon Web Services 和应用程序中的使用。您还可以在托管 Oracle BI 12c 存储库数据库的 Amazon RDS for Oracle 数据库实例上配置 Oracle 透明数据加密 (TDE)。
- 传输中的加密：最佳做法是激活 SSL 或 TLS 协议，以保护在 Oracle BI 12c 安装的各个层之间的传输中数据。您可以使用 AWS Certificate Manager（ACM）为 Oracle BI 12c 资源预调配、管理和部署公有和私有 SSL 和 TLS 证书。
- 网络安全：确保将 Oracle BI 12c 资源部署在针对用例配置了相应访问控制的 Amazon VPC 中。配置您的安全组，以筛选来自正在运行您的安装的 Amazon EC2 实例的入站和出站流量。此外，请确保配置网络访问控制列表 (NACLs)，以根据定义的规则允许或拒绝流量。
- 监控和记录：您可以使用 AWS CloudTrail 来跟踪对您的 AWS 基础设施（包括 Oracle BI 12c 资源）的 API 调用。在跟踪基础设施变更或进行安全分析时，此功能非常有用。您还可以使用 Amazon CloudWatch 查看操作数据，这些数据可以让您深入了解 Oracle BI 12c 应用程序的性能和运行状况。您也可以配置警报并根据这些警报采取自动操作。Amazon RDS 提供了其他监控工具，包括增强型监控和性能详情。

使用将本地 Apache Kafka 集群迁移到亚马逊 MSK MirrorMaker

由 Han Zhang (AWS) 和 Tanner Pratt (AWS) 创作

摘要

此模式为将本地、自行管理或托管的 Apache Kafka 集群迁移至 Amazon Managed Streaming for Apache Kafka(Amazon MSK)。您也可以使用这种模式从一个 Amazon MSK 集群迁移至另一个。

Apache Kafka 包含该 MirrorMaker 功能，该功能可在两个 Kafka 集群之间复制数据。 MirrorMaker 由消费者集合组成，这些消费者属于消费者群体。使用者从源集群中的主题中读取数据，然后将这些数据传递至生产者，后者将数据写入目标集群。

亚马逊 MSK 文档包含使用 MirrorMaker 版本 1.0 将本地 Kafka 集群迁移到 Amazon MSK 的过程的高级概述。此模式通过提供全面的 2.0 MirrorMaker 版本使用 step-by-step说明来补充这些信息。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 以下其中一类 Kafka 源集群：
  - 在本地数据中心
  - 在云中自行管理
  - 合作伙伴托管

限制

- 要使用 2.0 MirrorMaker 版，源集群必须运行 Apache Kafka 版本 2.4.0 或更高版本。对于早期版本，请参阅 Amazon MSK 文档中的说明以使用 MirrorMaker 版本 1.0。

产品版本

- MirrorMaker 版本 2.0
- Apache Kafka 版本 2.4.0 或更高版本。有关 Amazon MSK 支持的 Apache Kafka 版本的更多信息，请参阅支持的 Apache Kafka 版本。

架构

源技术堆栈

- 本地或自行管理 Kafka 集群

目标技术堆栈

- Amazon MSK 集群

目标架构

此图显示以下流程：

1. MirrorMaker 从源 Kafka 集群中的主题和消费者组中读取数据。
2. MirrorMaker 将数据和消费者信息复制到目标 Amazon MSK 集群。

工具

Amazon Web Services

- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- Amazon Managed Streaming for Apache Kafka (Amazon MSK) 是一项完全托管式服务，可帮助您构建并运行使用 Apache Kafka 来处理流数据的应用程序。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。

其他工具

- Apache Kafka 是开源事件流式传输平台。在这种模式中，您可以使用 Kafka 的MirrorMaker功能来执行跨集群迁移。

最佳实践

您可以在源环境或目标环境中运行，但建议您在尽可能靠近目标集群的地方运行。 MirrorMaker 有关更多信息，请参阅 Apache Kafka 文档中的最佳实践：远程使用和本地生产。

操作说明

创建 VPC，并以 Amazon MSK 集群为目标

Task	描述	所需技能
创建 VPC。	1. 在目标 Amazon Web Services account 中创建 VPC。有关说明，请参阅创建 VPC。 2. 在新 VPC 不同的可用区创建三个私有子网。有关说明，请参阅创建子网。使用不同的可用区可提供高可用性与容错能力。 **Note** 如果您使用公共互联网连接迁移 Kafka 集群，请创建公有子网并启用 Amazon MSK 集群的公共访问权限。	AWS 系统管理员、 DevOps 工程师、云管理员
创建 Amazon MSK 集群。	创建 Amazon MSK 集群。有关说明，请参阅使用 AWS 管理控制台 创建集群或使用 AWS CLI 创建集群。将集群配置为：使用您之前创建的 VPC 和子网。	AWS 系统管理员、 DevOps 工程师、云管理员

设置 MirrorMaker

Task	描述	所需技能
安装 MirrorMaker。	1. 启动实 EC2 例。 2. Connect 连接到您的 EC2 实例。 3. 在 EC2 实例上，下载并解压最新的 Kafka 版本。有关说明，请参阅快速入门（Kafka 文档）。 **Note** 在这种模式下，您将 MirrorMaker 2.0 作为专用 MirrorMaker 集群安装在 Amazon EC2 实例上。此选项在可用于开发环境，也是此模式中使用的方法。有关 MirrorMaker 2.0 其他部署选项的更多信息，请参阅此模式的 “其他信息” 部分。	AWS 系统管理员、云管理员、DevOps 工程师
指定 Kafka 集群信息。	在 Kafka 客户端安装 `bin` 文件夹内，创建 mm2.pro perties 文件并将其配置为源 Kafka 集群。有关说明，请参阅运行专用 MirrorMaker 集群（Kafka 文档）。	AWS 系统管理员、云管理员、DevOps 工程师
开始 MirrorMaker。	输入以下命令启动 MirrorMaker 并传递 mm2.propertie s 文件。	AWS 系统管理员、云管理员、DevOps 工程师

Task	描述	所需技能
	`$ ./bin/connect-mirror-maker.sh mm2.properties`	
监控进度。	通过检查每个主题的上次偏移量与该主题的当前偏移量之间的滞后时间来检查进度。MirrorMaker 有关说明，请参阅 Kafka 文档中的监控异地复制。	AWS 系统管理员、云管理员、DevOps 工程师

割接

Task	描述	所需技能
停止使用者应用程序。	停止所有使用源集群数据的使用者应用程序。	应用程序开发人员
启动使用者应用程序。	更改应用程序的引导配置，以指向目标集群。然后开始用于目标集群。	应用程序开发人员
停止源集群上的生产者。	当使用者应用程序在目标集群上成功使用时，请停止源集群上的生产者。	应用程序开发人员
在目标集群上启动生产程序。	更改生产者的配置引导服务器，然后指向其目标集群。等待 MirrorMaker 完成源集群的所有数据的镜像，然后再启动生产者。	应用程序开发人员
停下来 MirrorMaker。	在生产者移至目标集群后，停止 MirrorMaker。	AWS 系统管理员、云管理员、DevOps 工程师

相关资源

AWS 资源

- 使用 MirrorMaker（Amazon MSK 文档）迁移集群
- Amazon MSK 迁移实验室（AWS 研习会参与平台 Workshop Studio）

其他资源

- MirrorMaker 2.0（Apache Kafka 改进提案）
- 异地复制：跨集群数据镜像（Apache Kafka 文档）

其他信息

此模式作为专用 MirrorMaker 集群在 Amazon 上运行 MirrorMaker 2.0 EC2。此选项适用于开发环境。尽管此模式中没有对此进行讨论，但你也可以在 Kafka Connect 集群中运行 MirrorMaker 2.0。此部署选项使用 Kafka 生态系统中的框架，可改善扩展和维护。您可以将连接器部署至具有相关配置的 Kafka Connect 集群中，以运行应用程序。连接器可以在独立模式下运行以进行开发或测试，也可以在分布式模式下运行以用于生产。有关更多信息，请参阅在 Connect 集群 MirrorMaker 中运行（Apache Kafka 文档）。有关其他 MirrorMaker 2.0 部署选项的更多信息，请参阅演练：运行 MirrorMaker 2.0（Kafka 文档）。

将 ELK 堆栈迁移至 Elastic Cloud on AWS

由 Battulga Purevragchaa (AWS)、uday reddy 和 Antony Prasad Thevaraj (AWS) 创建

摘要

Elastic 多年来一直提供服务，其用户和客户通常在本地自行管理 Elastic。Elastic Cloud 是一种托管的 Elasticsearch 服务，提供了一种使用 Elastic Stack (ELK Stack) 的方式，以及企业搜索、可观测性和安全的解决方案。您可以使用日志、指标、APM（应用程序性能监控）和 SIEM（安全信息和事件管理）等应用程序访问 Elastic 解决方案。您可以使用机器学习、索引生命周期管理、Kibana Lens（用于拖放可视化）等集成功能。

当你从自我管理的 Elasticsearch 迁移至 Elasticsearch 时，Elasticsearch 服务会处理以下几点：

- 配置和管理基础架构
- 创建和管理 Elasticsearch 集群
- 向上和向下扩展集群
- 升级、修补以及拍摄快照

这使您有更多时间专注解决其他挑战。

此模式定义了如何在 Amazon Web Services (AWS) 上将本地 Elasticsearch 7.13 迁移至 Elastic Cloud on Amazon Web Services (AWS)。其他版本可能需要对此模式中描述的过程稍作修改。有关更多信息，请您联系 Elastic 代表。

先决条件和限制

先决条件

- 具有访问Amazon Simple Storage Service (Amazon S3)以获取快照的有效的 Amazon Web Services account
- 一个安全、带宽足够高的私有链接，用于将快照数据文件复制至 Amazon S3
- Amazon S3 Transfer Acceleration
- Elastic Snapshot 策略，确保定期将数据摄取存档到足够大的本地数据存储或远程存储 (Amazon S3)

开始迁移之前，您必须了解本地快照的大小以及随附索引的生命周期策略。有关更多信息，请联系 Elastic。

角色与技能

迁移过程还需要下表中描述的角色和专长。

角色	专业知识	责任
App support	熟悉本地部署的 Elastic Cloud 和 Elastic	所有 Elastic 相关任务
系统管理员或 数据库管理员	深入了解本地 Elastic 环境及配置	能够预配置存储空间、安装和使用 AWS 命令行界面（AWS CLI），并识别在本地提供 Elastic 的所有数据来源
网络管理员	了解本地与 AWS 的网络连接、安全以及性能	在了解连接带宽的前提下，建立从本地至 Amazon S3 的网络链接

限制

- Elastic Cloud 上的 Elasticsearch 仅在支持的 AWS 区域 可用。

产品版本

- Elasticsearch 7.13

架构

源技术堆栈

本地 Elasticsearch 7.13 或更高版本：

- 集群快照
- 索引快照
- Beats 配置

源技术架构

下图显示了具有不同摄取方法、节点类型以及 Kibana 的典型本地架构。不同的节点类型反映了 Elasticsearch 集群、身份验证以及可视化角色。

1. 从 Beats 摄取至 Logstash
2. 从 Beats 摄取至 Apache Kafka 消息队列
3. 从 Filebeat 摄取至 Logstash
4. 从 Apache Kafka 消息队列摄取至 Logstash
5. 从 Logstash 摄取至 Elasticsearch 集群
6. Elasticsearch 集群
7. 身份验证以及通知节点
8. Kibana 和 blob 节点

目标技术堆栈

Elastic Cloud 通过跨集群复制功能部署到您在多个 AWS 区域的软件即服务（SaaS）账户。

- 集群快照
- 索引快照
- Beats 配置
- Elastic Cloud
- 网络负载均衡器
- Amazon Route 53
- Amazon S3

目标架构

托管 Elastic Cloud 基础架构是：

- 高度可用，存在于多个可用区和多个 AWS 区域。
- 由于数据（索引和快照）是使用 Elastic Cloud 跨集群复制 (CCR) ，因此可以容忍区域故障

- 存档，因为快照存档至 Amazon S3
- 通过网络负载均衡器和 Route 53 的组合实现分区容错性
- 源自（但不限于）Elastic APM、Beats、Logstash 的数据摄取

高级迁移步骤

Elastic 开发了自己的规范性方法，用于将本地 Elastic Cluster 迁移至 Elastic Cloud。Elastic 方法与 AWS 迁移指南和最佳实践（架构完善的框架和 AWS 迁移加速计划 (MAP)）直接一致并互为补充。通常，三个 AWS 迁移阶段如下：

- 评测
- 动员
- 迁移与现代化

Elastic 遵循类似迁移阶段，术语互补：

- 启动
- 规划
- 实施
- 交付
- Close

Elastic 使用 Elastic 实施方法促进项目成果交付。这在设计上具有包容性，可确保 Elastic、咨询团队和客户团队能清晰地协同工作，共同实现预期成果。

Elastic 方法在实施阶段将传统瀑布项目分阶段与 Scrum 相结合。技术需求的配置以协作方式迭代交付，同时最大限度降低风险。

工具

Amazon Web Services

- Amazon Route 53 是一项高度可用且可扩展的域名系统（DNS）Web 服务。您可以使用 Route 53 以任意组合执行三个主要功能：域注册、DNS 路由和运行状况检查。

- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项对象存储服务。您可以通过 Amazon S3 随时在 Web 上的任何位置存储和检索的任意大小的数据。此模式使用 S3 存储桶和 Amazon S3 Transfer Acceleration。
- Elastic L oad Balancing — Elastic Load Balancing 会自动将您的传入流量分配到一个或多个可用区的多个目标，例如 EC2 实例、容器和 IP 地址。

其他工具

- Beats – Beats 发布来自 Logstash 或 Elasticsear
- Elastic Cloud – Elastic Cloud 是一项用于托管 Elasticsearch 的托管服务。
- Elasticsearch – Elasticsearch 是一个搜索和分析引擎，它使用 Elastic Stack 集中存储您的数据，以便进行大规模的搜索和分析。此模式还使用快照创建与跨集群复制。
- Logstash — Logstash 是一项服务器端数据处理管道，它从多个来源摄取数据，对其进行转换，然后将其发送到您的数据存储。

操作说明

准备迁移

Task	描述	所需技能
识别可运行本地 Elastic 解决方案的服务器。	确认可支持弹性迁移。	应用程序所有者
了解本地的服务器配置。	要了解成功驱动本地工作负载所需服务器配置，请查找当前使用的服务器硬件占用空间、网络配置和存储特性	App Support
收集用户和应用程序的账户信息。	识别本地 Elastic 环境使用的用户名以及应用程序名称。	系统管理员，App support
记录 Beats 以及数据采集器配置。	若要记录配置，请查看现有的数据来源和接收器。有关更多信息，请参阅 Elastic 文档。	App support
确定数据的速度与数量。	为集群处理数据量设定基准。	系统管理员，App support

Task	描述	所需技能
记录 RPO 与 RTO 方案。	记录中断和服务级别协议方面的恢复点目标 (RPO) 和恢复时间目标 (RTO) 情景 (SLAs)。	应用程序所有者、系统管理员、应用程序支持
确定最佳快照生命周期设置。	定义在迁移期间和之后，需要使用 Elastic 快照保护数据的频率。	应用程序所有者、系统管理员、应用程序支持
定义迁移后的性能预计。	生成有关当前和预期屏幕刷新、查询运行时以及用户界面行为的指标。	系统管理员，App support
记录互联网接入传输、带宽以及可用性要求。	确定将快照复制到 Amazon S3 的互联网连接的速度、延迟以及弹性。	网络管理员
记录 Elastic 本地运行时系统的当前成本。	确保 AWS 目标环境的规模设计高性能且具有成本效益。	数据库管理员、系统管理员、App support
确定身份验证以及授权需求。	Elastic Stack 安全功能提供了内置领域，例如 Lightweight Directory Access Protocol (LDAP)、Security Assertion Markup Language (SAML) 和 OpenID Connect (OIDC).	数据库管理员、系统管理员、App support
根据地理位置了解具体监管要求。	确保根据您的要求和任何相关的国家要求，对数据进行导出和加密。	数据库管理员、系统管理员、App support

实施迁移

Task	描述	所需技能
在 Amazon S3 准备暂存区。	若要在 Amazon S3 上接收快照，请创建一个 S3 存储	AWS 管理员

Task	描述	所需技能
	桶和一个具有对新创建存储桶具有完全访问权限的临时 AWS Identity and Access Management (IAM) 角色。有关更多信息，请参阅创建向 IAM 用户委派权限的角色。使用 AWS Security Token Service 请求临时安全凭证。确保访问密钥 ID、秘密访问密钥和会话令牌安全。 在存储桶上启用 Amazon S3 Transfer Acceleration。	
在本地安装 AWS CLI 和 Amazon S3 插件。	在每个 Elasticsearch 节点上运行以下命令。 `sudo bin/elasticsearch-plugin install repository-s3` 然后重新启动该节点。	AWS 管理员

Task	描述	所需技能

配置 Amazon S3 客户端的访问权限。	通过运行以下命令，添加之前创建的密钥。
	`elasticsearch-keystore add s3.client.default.access_key`
	`elasticsearch-keystore add s3.client.default.secret_key`
	`elasticsearch-keystore add s3.client.default.session_token`
		AWS 管理员

为 Elastic 数据注册快照存储库	使用 Kibana Dev Tools 告诉本地集群要写入哪个远程 S3 存储桶。	AWS 管理员
配置快照策略。	要配置快照生命周期管理，请在 Kibana 策略选项卡选择 SLM 策略，然后定义应包含哪些时间、数据流或索引以及要使用的名称。 配置频繁拍摄快照策略。快照是增量，可有效利用存储空间。与准备情况评测决定相匹配。策略还可指定保留策略，并在不再需要快照时自动删除它们。	App support

Task	描述	所需技能
验证快照是否有效。	在 Kibana 开发人员工具，运行以下命令。 `GET _snapshot/<your_repo_name>/_all`	AWS 管理员，App support
在 Elastic Cloud 上部署新集群。	登录 Elastic，根据准备情况评测中的业务调查发现，选择“可观测性、搜索性或安全性”集群。	AWS 管理员，App support
设置集群密钥存储访问权限。	新集群需访问用于存储快照的 S3 存储桶。在 Elasticsearch Service Console 上，选择安全，然后输入您之前创建的访问和私有 IAM 密钥。	AWS 管理员
将 Elastic Cloud 托管集群配置为访问 Amazon S3。	设置对先前在 Amazon S3 中创建的快照存储库新集群的访问权限。使用 Kibana 执行以下操作： 1. 选择堆栈管理、快照设置、RegisterRepo。 2. 在 别名字段中，输入存储库的名称。 3. 对于 S3 客户端名称，选择辅助客户端。 4. 将之前创建的 S3 存储桶添加至存储库。 5. 选择压缩快照。 6. 对于加密设置，将值保留设置为默认值。	AWS 管理员，App support

Task	描述	所需技能
验证新 Amazon S3 存储库。	确保您可访问托管在 Elastic Cloud 集群中的新存储库。	AWS 管理员

初始化 Elasticsearch 服务集群。	在 Elasticsearch Service Console，从 S3 快照初始化 Elasticsearch 服务集群。
	
	运行以下“发布”命令。
	
	`*/_close?expand_wildcards=all`
	
	`/_snapshot/<your-repo-name>/ <your-snapshot-name>/_restore`
	
	`*/_open?expand_wildcards=all`
		App Support

完成迁移

Task	描述	所需技能
验证快照恢复是否成功。	使用 Kibana 开发人员工具运行以下命令。 `GET _cat/indices`	App support
重新部署摄取服务。	将 Beats 和 Logstash 的端点连接至新的 Elasticsearch 服务端点。	App support

测试集群环境并清理

Task	描述	所需技能
验证集群环境。	将本地 Elastic 集群环境迁移至 AWS 后，您可以连接到该环境并使用自己的用户验收测试 (UAT) 工具验证新环境。	App support
清除资源。	验证集群成功迁移后，移除用于迁移的 S3 存储桶以及 IAM 角色。	AWS 管理员

相关资源

Elastic references

- Elastic Cloud
- 在 AWS 上托管 Elasticsearch 和 Kibana
- 弹性企业搜索
- 弹性集成
- 弹性可观测性
- 弹性安全
- Beats
- Elastic APM
- 迁移至索引生命周期管理
- 弹性订阅
- 接触弹性

弹性博文

- 如何在 AWS 上从自我管理的 Elasticsearch 迁移至 Elastic Cloud（博客文章）
- 迁移至 Elastic Cloud（博客文章）

Elastic 文档

- 教程：使用 SLM 自动备份
- ILM：管理索引生命周期
- Logstash
- 跨集群复制 (CCR)
- 采集管道
- 运行 Elasticsearch API 请求
- 快照保留

Elastic 视频和网络研讨会

- Elastic 云迁移
- Elastic Cloud：客户为什么要迁移（网络研讨会）

AWS 参考

- Amazon Web Services Marketplace 上的弹性云
- AWS Command Line Interface
- AWS Direct Connect
- AWS 迁移加速计划
- 网络负载均衡器
- 区域和可用区
- Amazon Route 53
- Amazon Simple Storage Service
- Amazon S3 Transfer Acceleration
- VPN 连接
- 架构完善的框架

其他信息

如果您计划迁移复杂的工作负载，请使用 Elastic Consulting Services。如果您有与配置和服务相关的基本问题，请联系 Elastic Support 团队。

使用 Starburs AWS Cloud t 将数据迁移到

由 Antony Prasad Thevaraj (AWS)、Shaun Van Staden 和 Suresh Veeragoni (AWS) 创作

摘要

Starburst 通过提供企业查询引擎，将现有数据源整合到一个接入点中，从而帮助您加快向 Amazon Web Services (AWS) 的数据迁移之旅。在最终确定任何迁移计划之前，您可对多个数据来源进行分析，以获得有价值的见解。在不中断 business-as-usual分析的情况下，您可以使用 Starburst 引擎或专用的提取、转换和加载 (ETL) 应用程序迁移数据。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 虚拟私有云（VPC）
- Amazon Elastic Kubernetes Service (Amazon EKS) 集群
- 亚马逊弹性计算云 (亚马逊 EC2) Auto Scaling 小组
- 需要迁移的当前系统工作负载列表
- 从您的本地环境 AWS 到您的本地环境的网络连接

架构

参考架构

以下高级架构图显示了 Starburst Enterprise 的典型部署： AWS Cloud

1. Starburst Enterprise 集群在你的。 AWS 账户
2. 用户使用轻型目录访问协议 (LDAP) 或开放授权 (OAuth) 进行身份验证，并直接与 Starburst 集群进行交互。
3. Starburst 可以连接到多个 AWS 数据源，例如亚马逊简单存储服务 (Amazon S3) AWS Glue、亚马逊关系数据库服务 (Amazon RDS) 和亚马逊 Redshift。Starburst 提供跨云环境 AWS Cloud、本地或其他云环境中的数据源的联合查询功能。
4. 您可使用 Helm 图表在 Amazon EKS 集群中启动 Starburst Enterprise。

5. Starburst Enterprise 使用 Amazon A EC2 uto Scaling 组和 Amazon EC2 Spot 实例来优化基础设施。
6. Starburst Enterprise 直接连接到至您现有的本地数据来源以实时读取数据。此外，如果您在此环境中部署了现有的 Starburst Enterprise，则可以将中的新 Starburst 集群直接 AWS Cloud 连接到该现有集群。

请注意以下几点：

- Starburst 不是数据虚拟化平台。它是基于 SQL 的大规模并行处理 (MPP) 查询引擎，构成了整体数据网格分析策略的基础。
- 在迁移过程中部署 Starburst 时，它可以直接连接至现有的本地基础设施。
- Starburst 提供了多种内置的企业和开源连接器，便于连接到各种遗留系统。有关连接器及其功能的完整列表，请参阅 Starburst Enterprise 用户指南中的连接器。
- Starburst 可从本地数据来源实时查询数据。这样可防止在迁移数据时中断常规业务运营。
- 如果您要从现有的本地 Starburst Enterprise 部署迁移，则可以使用特殊连接器 Starburst Stargate 将您的 Starbur st Enterprise 集群 AWS 直接连接到本地集群。当业务用户和数据分析师将查询从联合 AWS Cloud 到您的本地环境时，这会带来额外的性能优势。

高级流程概述

您可使用 Starburst 加速数据迁移项目，因为 Starburst 可以在迁移数据之前对所有数据进行深入分析。下图显示了使用 Starburst 迁移数据的典型进程。

角色

使用 Starburst 完成迁移通常需要使用以下角色：

- 云管理员 – 负责提供云资源以运行 Starburst Enterprise 应用程序
- Starburst 管理员 – 负责安装、配置、管理和支持 Starburst 应用程序
- 数据工程师 – 负责：
  - 将遗留数据迁移到云
  - 构建语义视图以支持分析

- 解决方案或系统所有者 – 负责整体解决方案的实施

工具

AWS 服务

- 亚马逊弹性计算云 (Amazon EC2) 在中提供可扩展的计算容量 AWS Cloud。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- 亚马逊 Elastic Kubernetes Service（亚马逊 EKS）可帮助你在上面运行 AWS Kubernetes，而无需安装或维护自己的 Kubernetes 控制平面或节点。

其他工具

- Helm – Helm 是 Kubernetes 的软件包管理器，可帮助您在 Kubernetes 集群上安装和管理应用程序。
- Starburst Enterprise – Starburst Enterprise 是一款基于 SQL 的大规模并行处理 (MPP) 查询引擎，构成了总体数据网格分析策略的基础。
- Starburst Stargate — Starburst Stargate 将一个 Starburst Enterprise 环境（例如本地数据中心中的集群）中的目录和数据源链接到另一个 Starburst Enterprise 环境中的目录和数据源，例如中的集群。 AWS Cloud

操作说明

评测数据

Task	描述	所需技能
识别您的数据并确定优先级。	确定您要移动的数据。大型本地遗留系统可能包含您想要迁移的核心数据，以及您不想移动或由于合规性原因而无法移动的数据。从数据清单开始，帮助您优先考虑应首先定位的数据。有关更多信息，请参阅自动产品组合发现入门。	数据工程师、数据库管理员

Task	描述	所需技能
浏览、清点和备份数据。	验证数据与您的用例的质量、数量和相关性。根据需要备份或创建数据快照，并最终确定数据的目标环境。	数据工程师、数据库管理员

设置 Starburst Enterprise 环境

Task	描述	所需技能
在中配置 Starburst 企业版。AWS Cloud	在对数据进行编目时，在托管 Amazon EKS 集群中设置 Starburst Enterprise。有关更多信息，请参阅 Starburst Enterprise 参考文档中的使用 Kubernetes 部署。这允许在数据迁移过程中进行 business-as-usual分析。	AWS 管理员、应用程序开发人员
将 Starburst 连接至数据来源。	识别数据并设置 Starburst Enterprise 后，将 Starburst 连接到数据来源。Starburst 以 SQL 查询的形式直接从数据来源读取数据。有关更多信息，请参阅 Starburst Enprise 参考文档。	AWS 管理员、应用程序开发人员

迁移数据

Task	描述	所需技能
构建并运行 ETL 管线。	开始数据迁移进程。此活动可以与 business-as-usual分析同时发生。要进行迁移，您可使用第三方产品或 Starburst	数据工程师

Task	描述	所需技能
	。Starburst 能跨不同来源读取和写入数据。有关更多信息，请参阅 Starburst Enprise 参考文档。	
验证数据。	迁移数据后，验证数据以，确保所有必需的数据均已移动且完好无损。	数据工程师、 DevOps 工程师

割接

Task	描述	所需技能
割接数据。	数据迁移和验证完成后，您可割接数据。这涉及更改 Starburst 中的数据链接。与其指向本地资源，不如指向新云源并更新语义视图。有关更多信息，请参阅 Starburst Enterprise 参考文档中的连接器。	数据工程师，割接负责人
向用户推出。	数据使用者开始处理迁移数据来源。此过程对于分析最终用户来说是不可见的。	割接负责人，数据工程师

相关资源

AWS Marketplace

- Starburst Galaxy
- Starburst Enterprise
- Starburst 数据 JumpStart
- Starburst Enterprise with Graviton

Starburst 文档

- Starburst Enterprise 用户指南
- Starburst Enterprise 参考文档

其他 AWS 文档

- 开始自动发现投资组合（AWS 规范性指导）
- 开启 Starburst，优化云基础设施的成本和性能 AWS（AWS 博客文章）

优化 AWS 输入文件大小的 ETL 摄取

由 Apoorva Patrikar (AWS) 创建

摘要

本指南向您展示如何通过在处理数据之前优化文件大小，以优化 AWS Glue 上的大数据和 Apache Spark 工作负载的提取、转换、加载（ETL）过程的摄取步骤。使用此模式可防止或解决小文件问题。也就是说，当大量小文件由于文件总大小而减慢数据处理速度时。例如，数百个只有几百 KB 的文件可能会显着降低 AWS Glue 作业的数据处理速度。这是因为 AWS Glue 必须在 Amazon Simple Storage Service (Amazon S3) 上执行内部列表功能，并且 YARN (Yet Another Resource Negotiator) 必须存储大量元数据。为了提高数据处理速度，您可以使用分组，以使 ETL 任务将一组输入文件读取至单个内存分区。该分区会自动将较小文件组合在一起。或者，您可以使用自定义代码将批处理逻辑添加至现有文件。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 一个或多个 AWS glue 作业
- 一个或多个大数据或 Apache Spark 工作负载
- 一个 S3 存储桶

架构

以下指南展示了 AWS Glue 作业如何处理不同格式的数据，然后将其存储在 S3 存储桶中以了解性能。

图表显示了以下工作流：

1.

> **Note**
>
> AWS Glue 任务将 CSV、JSON 和 Parquet 格式的小文件转换至动态框架。：输入文件的大小对 AWS Glue 任务的性能影响最大。

2. AWS Glue 任务在 S3 存储桶执行内部列表功能。

工具

- AWS Glue 是一项完全托管的 ETL 服务。它可以帮助您在数据存储和数据流之间对数据进行可靠地分类、清理、扩充和移动。
- Amazon Simple Storage Service(Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

操作说明

使用分组在读取过程中优化 ETL 摄取

Task	描述	所需技能
指定群组大小。	如果您的文件数量超过 50,000 个，预设情况下将进行分组。但是，您也可以通过在 `connectionOptions` 参数中指定组大小来对少于 50,000 个文件使用分组。`connectionOptions` 参数使用 `create_dynamic_frame.from_options` 方法。	数据工程师
编写分组代码。	使用 `create_dynamic_frame` 方法创建动态框架。例如： `S3bucket_node1 = glueContext.create_dynamic_frame.from_options( format_options={"multiline": False}, connection_type="s3", format="json",`	数据工程师

Task	描述	所需技能
	connection_options ={ "paths": ["s3:// bucket/prefix/file.j son"], "recurse": True, "groupFiles": 'inPartition', "groupSize": 1048576 }, transformation_ctx ="S3bucket_node1",) **Note** groupFiles 用于对 Amazon S3 分区组中的文件进行分组。使用 groupSize 设置要在内存中读取的组的目标大小。以字节为指定 groupSize (1048576 = 1 MB)。	
将代码添加到工作流。	在 AWS Glue 中将分组代码添加至您的任务工作流。	数据工程师

通过自定义逻辑优化 ETL 摄取

Task	描述	所需技能
选择语言与处理平台。	选择针对您的用例量身定制的脚本语言与处理平台。	云架构师

Task	描述	所需技能
编写代码。	编写自定义逻辑，将文件一起批处理。	云架构师
将代码添加到工作流。	在 AWS Glue 中添加代码到您的工作流。这样，您就可在每次运行作业时应用自定义逻辑。	数据工程师

转换后写入数据时重新分区

Task	描述	所需技能
分析消费模式。	了解下游应用程序将如何使用您写入的数据。例如，如果他们每天查询数据，而您只按区域对数据进行分区，或者输出文件非常小，例如每个文件 2.5 KB，那么这并不是最佳的使用方式。	数据库管理员
在写入之前对数据进行重新分区。	在处理过程中（基于处理逻辑）和处理后（基于消耗量），根据联接或查询进行重新分区。例如，根据字节大小进行重新分区（例如）`.repartition(100000)` ，或基于列的重新分区，例如。`.repartition("column_name")`	数据工程师

相关资源

- 读取较大群组中的输入文件
- 监控 AWS Glue

- 使用亚马逊 CloudWatch 指标监控 AWS Glue
- 作业监控和调试
- 在 AWS Glue 上开启无服务器 ETL 入门

其他信息

确定文件大小

没有直接的方法可确定文件大小。文件大小对处理性能的影响，取决于集群的配置。在核心 Hadoop，我们建议您使用 128 MB 或 256 MB 文件，以充分利用数据块。

对于 AWS Glue 上的大多数文本文件工作负载，我们建议 5-10 DPU 个集群的文件大小在 100 MB 到 1 GB 之间。要找出输入文件的最佳大小，请监控 AWS Glue 任务预处理部分，然后检查该作业的 CPU 利用率和内存利用率。

其它注意事项

如果早期 ETL 阶段的性能存在瓶颈，请考虑在处理之前对数据文件进行分组或合并。如果您可完全控制文件生成过程，则在将原始数据发送至 AWS 之前，在源系统本身上聚合数据点会更加高效。

使用 AWS Step Functions 编排 ETL 管道，包含验证、转换和分区

由 Sandip Gangapadhyay (AWS) 创建

摘要

此示例介绍了如何构建无服务器提取、转换、加载（ETL）管道，以验证、转换、压缩和分区大型 CSV 数据集，从而实现性能和成本优化。该管道由 AWS Step Functions 编排，包含错误处理、自动重试和用户通知功能。

将 CSV 文件上传至 Amazon Simple Storage Service (Amazon S3) 存储桶文件夹，ETL 管道开始运行。该管道验证源 CSV 文件的内容和架构，将 CSV 文件转换为压缩 Apache Parquet 格式，按年、月和日对数据集进行分区，并将其存储在单独的文件夹中供分析工具处理。

自动执行此模式的代码可在带有 AWS Step Functi GitHub ons 存储库的 ETL Pipelin e 中找到。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- AWS 命令行接口 (AWS CLI) Line Interface 已使用您的 AWS 账户进行安装和配置，因此您可以通过部署 AWS 堆栈来创建 AW CloudFormation S 资源。建议使用 AWS CLI 第 2 版。有关安装说明，请参阅 AWS CLI 文档中的安装、更新和卸载 AWS CLI 版本 2。有关 AWS CLI 配置说明，请参阅 AWS CLI 文档中的配置和凭证文件设置。
- Amazon S3 存储桶。
- 具有正确架构的 CSV 数据集。(此模式中包含的代码存储库提供了示例 CSV 文件，其中包含您可以使用的正确架构和数据类型。)
- 支持与 AWS 管理控制台 配合使用的 Web 浏览器。（请参阅支持的浏览器列表。）
- AWS Glue 控制台访问权限。
- AWS Step Functions 控制台访问权限。

限制

- 在 AWS Step Functions 中，保存历史日志最大限制为 90 天。有关更多信息，请参阅 AWS Step Functions 文档中的配额和标准工作流配额。

产品版本

- 适用于 AWS Lambda 的 Python 3.11
- AWS Glue 版本 2.0

架构

图中所示的工作流包括以下高级步骤：

1. 用户将 CSV 文件上传至 Amazon S3 中的源文件夹。
2. Amazon S3 通知事件会启动 AWS Lambda 函数，该函数启动 Step Functions 状态机。
3. Lambda 函数验证原始 CSV 文件架构和数据类型。
4. 根据验证结果：
   a. 如源文件验证成功，则文件将移至舞台文件夹进行进一步处理。
   b. 如果验证失败，文件将移至错误文件夹，并由 Amazon Simple Notification Service (Amazon SNS) 发送错误通知。
5. AWS Glue 爬网程序从 Amazon S3 的阶段文件夹中创建原始文件架构。
6. AWS Glue 作业将原始文件转换、压缩并分区为 Parquet 格式。
7. AWS Glue 任务还会将文件移动至 Amazon S3 中的转换文件夹。
8. AWS Glue 爬网程序会根据转换后的文件创建架构。生成的架构用于任何分析作业。您还可以使用 Amazon Athena 中运行临时查询。
9. 如果管道在无错误的情况下完成，则架构文件将移至存档文件夹。如遇到任何错误，则会将文件移至错误文件夹。
10. Amazon SNS 会根据管道完成状态发送通知，指示成功或者失败。

此模式中使用的所有 AWS 资源均为无服务器。没有需要管理的服务器。

工具

Amazon Web Services

- AWS Glue – AWS Glue 是一项完全托管的 ETL 服务，可让客户轻松准备和加载数据以进行分析。

- AWS Step Functions - AWS Step Functions 是一项无服务器编排服务，可让您搭配使用 AWS Lambda 函数和其他 Amazon Web Services 来构建业务关键型应用程序。通过 AWS Step Functions 图形控制台，您可将应用程序的工作流视为一系列事件驱动的步骤。
- Amazon S3 – Amazon Simple Storage Service（Amazon S3）是一种对象存储服务，提供行业领先的可扩展性、数据可用性、安全性和性能。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 是一项高度可用、耐用、安全、完全托管的 pub/sub 消息服务，可让您分离微服务、分布式系统和无服务器应用程序。
- AWS Lambda - AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。只有在需要时 AWS Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。

代码

此模式的代码可在带有 AWS Step Fun GitHub ctions 存储库的 ETL Pipelin e 中找到。代码存储库包含以下文件和文件夹：

- `template.yml`— 用于使用 AWS Step Functions 创建 ETL 管道的 AWS CloudFormation 模板。
- `parameter.json` — 包含所有参数和参数值。您可更新此文件以更改参数值，如操作说明部分所述。
- `myLayer/python` 文件夹 — 包含为此项目创建所需 AWS Lambda 层所需 Python 包。
- `lambda` 文件夹 - 包含以下 Lambda 函数：
  - `move_file.py` — 将源数据集移动到存档、转换或错误文件夹。
  - `check_crawler.py` — 在 AWS Glue 爬网程序发送失败消息之前，根据 `RETRYLIMIT` 环境变量的配置多次检查其状态。
  - `start_crawler.py` — 启动 AWS Glue 爬网程序。
  - `start_step_function.py` — 启动 AWS Step Functions。
  - `start_codebuild.py`— 启动 AWS CodeBuild 项目。
  - `validation.py` — 验证输入的原始数据集。
  - `s3object.py` — 在 S3 存储桶内创建所需目录结构。
  - `notification.py` — 在管道结束时发送成功或错误通知。

若要使用示例代码，请按照操作部分的说明执行。

操作说明

准备源文件

Task	描述	所需技能
克隆示例代码存储库。	1. 打开ETL Pipeline with AWS Step Functions 存储库。 2. 在主存储库页面的文件列表上方选择代码，然后复制使用 HTTPS 克隆 下列出的 URL。 3. 将工作目录更改为要存储示例文件的位置。 4. 在终端或命令提示符处，输入命令： `git clone <repoURL>` 其中，`<repoURL>` 是指您在步骤 2 中复制的 URL。	开发人员
更新参数值。	在存储库本地副本中，编辑 `parameter.json` 文件并更新默认参数值，如下所示： • `pS3BucketName` – 用于存储数据集的 S3 存储桶的名称。此模板将为您创建此存储桶。存储桶名称必须全局唯一。 • `pSourceFolder` – S3 存储桶内用于上传源 CSV 文件的文件夹的名称。	开发人员

Task	描述	所需技能
	• pStageFolder – S3 存储桶内将在该过程中用作暂存区的文件夹的名称。 • pTransformFolder – S3 存储桶内用于存储已转换和分区数据集的文件夹的名称。 • pErrorFolder – S3 存储桶内的文件夹，如果无法验证源 CSV 文件，则该文件将被移至该文件夹。 • pArchiveFolder – S3 存储桶内用于存档源 CSV 文件的文件夹的名称。 • pEmailforNotification – 用于接收成功/错误通知的有效电子邮件地址。 • pPrefix– AWS Glue 抓取工具名称中将使用的前缀字符串。 • pDatasetSchema – 将对源文件进行验证的数据集架构。Cerberus Python 数据包用于源数据集验证。有关更多信息，请参阅 Cerberus 网站。	

Task	描述	所需技能
上传源代码到 S3 存储桶。	在部署自动执行 ETL 管道的 CloudFormation 模板之前，必须打包 CloudFormation 模板的源文件并将其上传到 S3 存储桶。为此，通过您预配置的配置文件运行以下 AWS CLI 命令： `aws cloudformation package --template-file template.yml --s3-bucket <bucket_name> --output-template-file packaged.template --profile <profile_name>` 其中： • `<bucket_name>` 是要部署堆栈的 AWS 区域中现有 S3 存储桶的名称。此存储桶用于存储 CloudFormation 模板的源代码包。 • `<profile_name>` 是您在设置 AWS CLI 时预先配置的有效 AWS CLI 配置文件。	开发人员

创建堆栈

Task	描述	所需技能
部署 CloudFormation 模板。	要部署 CloudFormation 模板，请运行以下 AWS CLI 命令：	开发人员

Task	描述	所需技能
	`aws cloudformation deploy --stack-name <stack_name> --template-file packaged.template --parameter-overrides file://parameter.json --capabilities CAPABILITY_IAM --profile <profile_name>` 其中： • `<stack_name>` 是 CloudFormation 堆栈的唯一标识符。 • `<profile-name>` 是您预先配置的 AWS CLI 配置文件。	
查看进度。	在 AWS CloudFormation 控制台上，查看堆栈开发进度。当状态为时 `CREATE_COMPLETE` ，表示堆栈已成功部署。	开发人员
记下 AWS Glue 数据库名称。	堆栈的输出 选项卡显示 AWS Glue 数据库名称。键名称为 `GlueDBOutput` 。	开发人员

测试管道

Task	描述	所需技能
启动 ETL 管道。	1. 导航到 S3 存储桶内的源文件夹（`source` 或您在	开发人员

Task	描述	所需技能
	`parameter.json` 文件中设置的文件夹名称）。 2. 将示例 CSV 文件上传至此文件夹。（代码存储库提供了一个名为 `Sample_Bank_Transaction_Raw_Dataset.csv` 的示例文件以供您使用。）上传文件后，将通过 Step Functions 启动 ETL 管道。 3. 在 Step Functions 控制台，检查 ETL 管道状态。	
查看分区数据集。	ETL 管道完成后，确认分区数据集在 Amazon S3 转换文件夹（`transform` 或您在 `parameter.json` 文件中设置的文件夹名称）中可用。	开发人员
检查已分区 AWS Glue 数据库。	1. 在 AWS Glue 控制台，选择堆栈创建的 AWS Glue 数据库（这是上文中标注的数据库）。 2. 验证 AWS Glue Data Catalog 中的分区表是否可用。	开发人员
运行查询。	（可选）使用 Amazon Athena 对已分区和转换的数据库运行临时查询。有关说明，请参阅 AWS 文档中的 使用 Amazon Athena 运行 SQL 查询。	数据库分析师

故障排除

事务	解决方案
AWS Glue 任务和爬虫的 AWS 身份和访问管理 (IAM) 权限	如果您进一步自定义 AWS Glue 任务或爬虫，请务必在 AWS Glue 任务使用的 IAM 角色中授予相应的 IAM 权限，或者向 AWS Lake Formation 提供数据权限。有关更多信息，请参阅 AWS 文档。

相关资源

Amazon Web Services 文档

- AWS Step Functions
- AWS Glue
- AWS Lambda
- Amazon S3
- Amazon SNS

其他信息

下图显示了 Step Functions Inspector 面板 面板中成功建立 ETL 管道的 AWS Step Functions 工作流。

下图所示为不合格的 ETL 管道 AWS Step Functions 工作流，原因是 Step Functions 检查器面板显示其输入验证错误。

使用 Amazon Redshift ML 执行高级分析

由 Po Hong (AWS) 和 Chyanna Antonio (AWS) 创作

摘要

在 Amazon Web Services (AWS) Cloud，您可使用 Amazon Redshift machine learning (Amazon Redshift ML) 对存储在 Amazon Redshift cluster 或 Amazon Simple Storage Service (Amazon S3) 的数据执行机器学习分析。Amazon Redshift ML 支持有监督学习，这种学习常用于高级分析。Amazon Redshift ML 用例包括收入预测、信用卡欺诈检测以及客户生命周期价值 (CLV) 或客户流失预测。

Amazon Redshift ML 使数据库用户可以轻松地使用标准的 SQL 命令创建、训练和部署 ML 模型。Amazon Redshift ML 使用 Amazon A SageMaker utopilot 自动训练和调整最佳机器学习模型，以便根据您的数据进行分类或回归，同时保持控制和可见性。

亚马逊 Redshift、Amazon S3 和亚马逊之间的所有交互 SageMaker 都是抽象和自动化的。ML 模型经过训练和部署后，它将作为用户定义函数 (UDF) 在 Amazon Redshift 中使用，并可用于 SQL 查询。

此模式补充了 AWS 博客中的 “使用 SQL 在 Amazon Redshift 中创建、训练和部署机器学习模型” 和 “入门资源中心” 中的 “使用 SageMaker亚马逊构建、训练和部署机器学习模型” 教程。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- Amazon Redshift 表内现有数据

技能

- 熟悉 Amazon Redshift ML 使用的术语和概念，包括机器学习、培训和预测。有关这方面的更多信息，请参阅 Amazon Machine Learning (Amazon ML) 文档中的ML 模型培训。
- 体验 Amazon Redshift 用户设置、访问管理以及标准 SQL 语法。有关的更多信息，请参阅 Amazon Redshift 文档中的 Amazon Redshift 入门。
- Amazon S3 和 AWS Identity and Acess Management (IAM) 方面的专长和经验。
- 在 AWS 命令行界面（AWS CLI）中运行命令也是有益的，但不是必需的。

限制

- Amazon Redshift 集群与 S3 存储桶必须位于同一 AWS 区域。
- 此模式方法仅支持有监督学习模型，例如回归、二进制分类以及多类分类。

架构

以下步骤说明了 Amazon Redshift 机器学习如何使用 SageMaker 来构建、训练和部署机器学习模型：

1. Amazon Redshift 将训练数据导出到 S3 存储桶中。
2. SageMaker Autopilot 会自动预处理训练数据。
3. 调用该`CREATE MODEL`语句后，Amazon Redshift ML 将 SageMaker 用于训练。
4. SageMaker Autopilot 搜索并推荐用于优化评估指标的机器学习算法和最佳超参数。
5. Amazon Redshift ML 会在 Amazon Redshift 集群中将输出 ML 模型注册为 SQL 函数。
6. ML 模型的函数可用于 SQL 语句。

技术堆栈

- Amazon Redshift
- SageMaker
- Amazon S3

工具

- Amazon Redshift – Amazon Redshift 是一种完全托管的企业 PB 级数据仓库服务。
- Amazon Redshift ML – Amazon Redshift 机器学习 (Amazon Redshift ML) 是一种基于云的稳健服务，能够让所有技能水平的分析人员和数据科学家都能轻松使用 ML 技术。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项面向互联网的存储服务。
- Amazon SageMaker — SageMaker 是一项完全托管的机器学习服务。
- Amazon A SageMaker uto SageMaker pil ot — Autopilot 是一款功能集，可自动执行自动机器学习 (AutoML) 过程中的关键任务。

代码

您可使用以下代码，在 Amazon Redshift 中创建受监管的 ML 模型：

```
“CREATE MODEL customer_churn_auto_model
FROM (SELECT state,
             account_length,
             area_code,
             total_charge/account_length AS average_daily_spend,
             cust_serv_calls/account_length AS average_daily_cases,
             churn
      FROM customer_activity
      WHERE  record_date < '2020-01-01'
     )
TARGET churn
FUNCTION ml_fn_customer_churn_auto
IAM_ROLE 'arn:aws:iam::XXXXXXXXXXXX:role/Redshift-ML'
SETTINGS (
  S3_BUCKET 'your-bucket'
);”)
```

> **Note**
>
> 该SELECT州可以参考亚马逊 Redshift 常规表、亚马逊 Redshift Spectrum 外部表或两者兼而有之。

操作说明

准备训练与测试数据集

Task	描述	所需技能
准备训练与测试数据集。	登录 AWS 管理控制台并打开亚马逊 SageMaker 控制台。按照构建、训练和部署机器学习模型教程创建包含标签列（监督训练）且没有标题的 .csv 或 Apache Parquet 文件。	数据科学家

Task	描述	所需技能
	Note 我们建议您将原始数据集进行洗牌并拆分为用于模型训练的训练集（70%）和用于模型性能评估的测试集（30%）。	

准备与配置技术堆栈

Task	描述	所需技能
创建和配置 Amazon Redshift 集群。	在 Amazon Redshift 控制台，根据您的要求创建集群。有关更多信息，请参阅 Amazon Redshift 文档中的创建集群。 **Important** 必须使用`SQL_PREVIEW` 维护轨道创建 Amazon Redshift 集群。有关预览轨道的更多信息，请参阅 Amazon Redshift 文档中的选择集群维护轨道。	数据库管理员、云架构师
创建 S3 存储桶以存储训练数据和模型构件。	在 Amazon S3 控制台，创建 S3 存储桶以训练和测试数据。有关创建 S3 存储桶的更多信息，请参阅 AWS 快速入门中的创建 S3 存储桶。	数据库管理员、云架构师

Task	描述	所需技能
	Important 确保您的 Amazon Redshift 集群和 S3 存储桶位于同一区域。	
创建 IAM policy，并将其附加至 Amazon Redshift 集群。	创建 IAM 策略以允许 Amazon Redshift 集群访问 SageMaker 和亚马逊 S3。有关说明和步骤，请参阅 Amazon Redshift 文档中的使用 Amazon Redshift ML 的集群设置。	数据库管理员、云架构师
允许 Amazon Redshift 用户和群组访问架构和表格。	授予权限，允许 Amazon Redshift 中的用户和群组访问内部和外部架构和表格。有关步骤和说明，请参阅 Amazon Redshift 文档的管理权限和所有权。	数据库管理员

在 Amazon Redshift 中创建和训练 ML 模型

Task	描述	所需技能
在 Amazon Redshift 中创建和训练 ML 模型。	在 Amazon Redshift ML 中创建和训练您的 ML 模型。有关更多信息，请参阅 Amazon Redshift 文档中的 `CREATE MODEL` 语句。	开发人员、数据科学家

在 Amazon Redshift 中执行批量推理与预测

Task	描述	所需技能
使用生成的 ML 模型函数执行推理。	有关使用生成 ML 模型函数执行推理的更多信息，请参阅 Amazon Redshift 文档中的预测。	数据科学家、商业智能用户

相关资源

准备训练与测试数据集

- 使用 Amazon 构建、训练和部署机器学习模型 SageMaker

准备与配置技术堆栈

- 创建 Amazon Redshift 集群。
- 选择 Amazon Redshift 集群维护轨道
- 创建 S3 存储桶
- 为使用 Amazon Redshift ML 设置 Amazon Redshift 集群
- 在 Amazon Redshift 中管理权限与所有权

在 Amazon Redshift 中创建和训练 ML 模型

- 在 Amazon Redshift 中创建模型语句

在 Amazon Redshift 中执行批量推理与预测

- Amazon Redshift 中的预测

其他资源

- Amazon Redshift ML 入门
- 使用 SQL 和 Amazon Redshift ML 在 Amazon Redshift 中构建、训练和部署 ML 模型
- Amazon Redshift 合作伙伴
- AWS 机器学习能力合作伙伴

使用 Amazon Athena 使用 SQL 查询亚马逊 DynamoDB 表

由 Gavin Perrie (AWS)、Ajit Ambike (AWS) 和 Brad Yates (AWS) 创作

摘要

如果您的数据包括亚马逊简单存储服务 (Amazon S3) Simple Service 以外的来源，则可以使用联合查询来访问这些关系、非关系、对象或自定义数据源。此模式显示如何使用 SQL 数据源连接器配置通过 Amazon Athena 到 Amazon DynamoDB 的联合查询访问权限。

使用此模式，您可以执行以下操作：

- 使用 SQL 查询 DynamoDB 表。
- 在 Athena 中运行联合 SQL 查询，并将 DynamoDB 表与其他支持的数据源联接。

先决条件和限制

先决条件

- 一个 DynamoDB 表。
- 将使用 Athena 引擎版本 2 的 Athena 工作组。有关说明，请参阅 Athena 文档。
- 一个 S3 存储桶，`AthenaDynamoDBConnector` AWS Lambda 函数可以在其中泄露数据。S3 存储桶和 Lambda 函数必须位于同一 AWS 区域。

如果这是您首次访问 Athena，则需要额外的 S3 存储桶用作查询结果位置。有关说明，请参阅 Athena 文档。

限制

- 不支持诸如 “插入到” 之类的写入操作。

产品版本

- Athena Query Federation 发布于 GitHub

架构

目标架构

下图显示了模式建立后的连接流程。用户连接到 Amazon Athena 以提供查询。Athena 将查询和目标传递给 DynamoDB 数据源连接器 Lambda 函数，该函数检索数据并将其返回给 Athena。如果返回大量数据，Athena 会在打包并返回完整数据集之前将临时结果存储在泄漏桶中。

工具

AWS 服务

- Amazon Athena 是一种交互式查询服务，可帮助您通过使用标准 SQL 直接分析 Amazon Simple Storage Service（Amazon S3）中的数据。此模式使用 Amazon Athena DynamoDB 连接器，该工具使用亚马逊 Athena Query Federation SDK 构建，并通过作为应用程序安装。 AWS Lambda AWS Serverless Application Repository
- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Storage Service (Amazon S3)是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

代码存储库

此模式的代码可在 GitHub Athena Query Federation 存储库中找到。

操作说明

设置和测试 DynamoDB 数据源连接器

Task	描述	所需技能
部署 AthenaDynamoDBConnector 应用程序。	要部署 AthenaDynamoDBConnecter，请执行以下操作： 1. 登录 AWS Management Console，然后选择你要用	AWS DevOps

Task	描述	所需技能
	于 DynamoDB 表和溢出桶的。 AWS 区域 2. 在以下位置打开 Serverless Applicat ion Re https://console.aws.amazon.com/serverlessrepo/ pository。 3. 在导航窗格中，选择 “可用应用程序”。 4. 对于 AWS Identity and Access Management (IAM) 访问权限，请在搜索栏下方选中显示创建自定义 IAM 角色或资源策略的应用程序复选框。 5. 搜索并选择 AthenaDynamoDBConnector，并确保列出的作者是亚马逊 Athena Federation。 6. 在应用程序设置中，输入以下值： • SpillBucket– 函数可以将数据泄漏到的位置。 • AthenaCatalogName– 将要创建的 Lambda 函数的名称。该名称还将用作 Athena 中的数据源名称。 7. 选中该复选框以确认 IAM 角色和策略的创建。 8. 选择部署。	

Task	描述	所需技能
为 Athena 创建数据源。	要创建数据源，请执行以下操作： 1. 在以下位置打开 Athena 主机 https://console.aws.amazon.com/athena/. 2. 展开导航窗格，然后选择“数据源”。 3. 选择创建数据来源。 4. 选择亚马逊 DynamoDB。 5. 输入数据源名称。 6. 选择您创建的 Lambda 函数。 7. 查看详细信息，然后选择创建数据源。	AWS DevOps
使用 Athena 查询 DynamoDB 表。	要查询 DynamoDB 表，请执行以下操作： 1. 在 Athena 控制台上，展开导航窗格，然后选择查询编辑器。 2. 在数据源下拉列表中，选择您创建的数据源。 3. 确认 DynamoDB 表已在“表” 下方列出。 4. 运行查询。	应用程序开发人员

故障排除

事务	解决方案
查询失败`GENERIC_INTERNAL_ERROR: The bucket is in this region: <region>`。	确保 Athena 溢出桶和 Lambda 函数是在同一个空间中创建的。 AWS 区域
新创建的数据源在 Athena 控制台上不可见。	Athena 的数据目录是区域性的。确保`AthenaDynamoDBConnector` 已在您尝试使用 Athena 的区域中部署。
您无法对新创建的数据源运行查询。	检查是否已设置查询结果位置。

相关资源

- 亚马逊 Athena DynamoDB 连接器
- 亚马逊 Athena 联合查询

使用 Athena 访问、查询和联接 Amazon DynamoDB 表

由 Moinul Al-Mamun (AWS) 创建

摘要

此模式说明如何使用 Amazon Athena DynamoDB 连接器在 Amazon Athena 和 Amazon DynamoDB 之间建立连接。连接器使用 AWS Lambda 函数查询 DynamoDB 中的数据。您无需编写任何代码即可设置连接。建立连接后，您可以使用 Athena 联合查询从 Athena 运行 SQL 命令，从而快速访问和分析 DynamoDB 表。您还可以将一个或多个 DynamoDB 表相互联接，或联接到其他数据来源，例如 Amazon Redshift 或 Amazon Aurora。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account 有权管理 DynamoDB 表、Athena 数据来源、Lambda 和 AWS Identity and Access Management (IAM) 角色
- 一个 Amazon Simple Storage Service（Amazon S3）存储桶，Athena 可以在其中存储查询结果
- 一个 S3 存储桶，Athena DynamoDB 连接器可以在其中保存数据
- 支持 Athena 引擎版本 2 的 AWS 区域
- 访问 Athena 和所需 S3 存储桶的 IAM 权限
- Amazon Athena DynamoDB 连接器，已安装

限制

查询 DynamoDB 表需要付费。表大小超过几千兆字节 (GBs) 可能会产生高昂的成本。我们建议您在执行任何全表 SCAN 操作之前考虑成本。有关更多信息，请参阅 Amazon DynamoDB 定价。为了降低成本并实现高性能，我们建议您始终在查询中使用 LIMIT（例如，`SELECT * FROM table1 LIMIT 10`）。此外，在生产环境中执行 JOIN 或 GROUP BY 查询之前，请考虑表的大小。如果您的表太大，请考虑其他选项，例如将表迁移到 Amazon S3。

架构

下图显示了用户如何从 Athena 对 DynamoDB 表运行 SQL 查询。

图表显示了以下工作流：

1. 要查询 DynamoDB 表，用户需要从 Athena 运行 SQL 查询。
2. Athena 启动 Lambda 函数。
3. Lambda 函数查询 DynamoDB 表中请求的数据。
4. DynamoDB 将请求的数据返回给 Lambda 函数。然后，该函数通过 Athena 将查询结果传输给用户。
5. Lambda 函数将数据存储在 S3 存储桶中。

技术堆栈

- Amazon Athena
- Amazon DynamoDB
- Amazon S3
- AWS Lambda

工具

- Amazon Athena 是一种交互式查询服务，可帮助您使用标准 SQL 直接在 Amazon S3 中分析数据。
- Amazon Athena DynamoDB Connector 是一种 AWS 工具，它使 Athena 能够与 DynamoDB 连接并使用 SQL 查询访问您的表。
- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。

操作说明

创建示例 DynamoDB 表

Task	描述	所需技能
创建第一个示例表。	1. 登录 AWS 管理控制台，打开 DynamoDB 控制台。 2. 选择创建表。	开发人员

Task	描述	所需技能
	3. 对于表名称，输入 dydbtable1。 4. 对于分区密钥，请输入PK1。 5. 对于排序键，输入 SK1。 6. 在表设置部分中，选择自定义设置。 7. 在表类部分中，选择 DynamoDB 标准。 8. 在读/写容量设置部分中，对于容量模式，选择按需。 9. 在静态加密部分中，选择由 Amazon DynamoDB 拥有。 10选择创建表。	

Task	描述	所需技能

将示例数据插入到第一个表中。	1. 打开 DynamoDB 控制台。
	2. 在导航窗格中，选择表，然后在名称列中选择您的表。
	3. 选择操作，然后选择创建项目。
	4. 选择 JSON 视图。
	5. 在属性编辑器的标题栏中，关闭查看 DynamoDB JSON。
	6. 在属性编辑器中，逐个输入以下示例数据：
	
	`{ "PK1": "1234", "SK1": "info", "Salary": "5000" }`
	
	`{ "PK1": "1235", "SK1": "info", "Salary": "5200" }`
		开发人员

Task	描述	所需技能
创建第二个示例表。	1. 打开 DynamoDB 控制台。 2. 选择 Create Table。 3. 对于表名称，输入 dydbtable2。 4. 对于分区键，输入 PK2。 5. 对于排序键，输入 SK2。 6. 在表设置部分中，选择自定义设置。 7. 在表类部分中，选择 DynamoDB 标准。 8. 在读/写容量设置部分中，对于容量模式，选择按需。 9. 在静态加密部分中，选择由 Amazon DynamoDB 拥有。 10选择创建表。	开发人员

Task	描述	所需技能

将示例数据插入到第二个表中。	1. 打开 DynamoDB 控制台。
	2. 在导航窗格中，选择表，然后在名称列中选择您的表。
	3. 选择操作，然后选择创建项目。
	4. 在属性编辑器的标题栏中，关闭查看 DynamoDB JSON。
	5. 在属性编辑器中，逐个输入以下示例数据：
	
	`{ "PK2": "1234", "SK2": "bonus", "Bonus": "500" }`
	
	`{ "PK2": "1235", "SK2": "bonus", "Bonus": "1000" }`
		开发人员

在 Athena 中为 DynamoDB 创建数据来源

Task	描述	所需技能
设置数据来源连接器。	为 DynamoDB 创建数据来源，然后创建 Lambda 函数以连接到该数据来源。 1. 登录 AWS 管理控制台 并打开 Athena 控制台。	开发人员

Task	描述	所需技能
	2. 在导航窗格中，选择数据来源，然后选择创建数据来源。 3. 选择 Amazon DynamoDB 数据来源，然后选择下一步。 4. 在 数据来源详细信息 部分中，对于 数据来源名称，输入 testDynamoDB。 5. 在 连接详细信息 部分中，选择已部署的 Lambda 函数，或者如果您没有要用于此模式的 Lambda 函数，请选择 创建 Lambda 函数。注意：有关创建 Lambda 函数的更多信息，请参阅《Lambda 开发人员指南》中的 Lambda 入门。 6. （可选）如果您选择创建 Lambda 函数，则必须先配置 Java 应用程序包含的 AWS CloudFormation 模板，然后再部署该堆栈。该模板包括 ApplicationName SpillBucket、 AthenaCatalogName、和其他应用程序设置。注意：部署此基于 Java 的应用程序后，堆栈将创建一个 Lambda 函数，使 Athena 能够与 DynamoDB 进行通信。这样，您的表就可以通过 SQL 命令访问。 7. 部署 Lambda 函数。	

Task	描述	所需技能
	8. 选择下一步。	
验证 Lambda 函数是否可以访问 S3 溢出存储桶。	1. 打开 Lambda 控制台。 2. 在导航窗格中，选择函数，然后选择您之前创建的函数。 3. 选择配置选项卡。 4. 在左窗格中，选择环境变量，然后确认键的值为 `spill_bucket` 。 5. 在左侧窗格中，选择权限，然后在执行角色部分中，选择附加的 IAM 角色。注意：您将被定向到 IAM 控制台中附加到 Lambda 函数的 IAM 角色。 6. 确认您对 `spill_bucket` 存储桶具有写入权限。 如果遇到错误，请参阅此模式中的其他信息部分以获取指导。	开发人员

从 Athena 访问 DynamoDB 表

Task	描述	所需技能
查询 DynamoDB 表。	1. 登录 AWS 管理控制台 并打开 Athena 控制台。 2. 在导航窗格中，选择数据来源，然后选择创建数据来源。	开发人员

Task	描述	所需技能
	3. 在导航窗格中，选择 Query editor (查询编辑器)。 4. 在编辑器选项卡的数据部分中，对于数据来源，为数据来源选择数据源。 5. 对于数据库，选择您的数据库。 6. 对于查询 1，输入以下查询：`SELECT * FROM dydbtable1 t1;` 7. 选择运行，然后验证表中的输出。 8. 对于查询 2，输入以下查询：`SELECT * FROM dydbtable2 t2;` 9. 选择运行，然后验证表中的输出。	
连接两个 DynamoDB 表。	DynamoDB 是 NoSQL 数据存储，不支持 SQL 联接操作。因此，您必须对两个 DynamoDB 表执行联接操作： 1. 选择加号图标以创建另一个查询。 2. 对于查询 3，输入以下查询： `SELECT pk1, salary, bonus FROM dydbtable1 t1 JOIN dydbtable2 t2 ON t1.pk1 = t2.pk2;`	开发人员

相关资源

- Amazon Athena DynamoDB 连接器（AWS Labs）
- 使用 Amazon Athena 的新联合查询查询任何数据来源（AWS 大数据博客）
- Athena 引擎版本参考（Athena 用户指南）
- 使用 AWS Glue 和 Amazon Athena 简化 Amazon DynamoDB 数据提取和分析（AWS 数据库博客）

其他信息

如果您在 Athena 中使用 `spill_bucket` 的 `{bucket_name}/folder_name/` 格式运行查询，则可能会收到以下错误消息：

```
"GENERIC_USER_ERROR: Encountered an exception[java.lang.RuntimeException] from your
 LambdaFunction[arn:aws:lambda:us-east-1:xxxxxx:function:testdynamodb] executed in
 context[retrieving meta-data] with message[You do NOT own the spill bucket with the
 name: s3://amzn-s3-demo-bucket/athena_dynamodb_spill_data/]
This query ran against the "default" database, unless qualified by the query. Please
 post the error message on our forum  or contact customer support with Query Id:
 [query-id]"
```

要解决此错误，请将 Lambda 函数的环境变量 `spill_bucket` 更新为 `{bucket_name_only}`，然后更新存储桶写入访问权限的以下 Lambda IAM policy：

```
{
             "Action": [
                 "s3:GetObject",
                 "s3:ListBucket",
                 "s3:GetBucketLocation",
                 "s3:GetObjectVersion",
                 "s3:PutObject",
                 "s3:PutObjectAcl",
                 "s3:GetLifecycleConfiguration",
                 "s3:PutLifecycleConfiguration",
                 "s3:DeleteObject"
             ],
             "Resource": [
                 "arn:aws:s3:::spill_bucket",
                 "arn:aws:s3:::spill_bucket/*"
```

```
            ],
             "Effect": "Allow"
        }
```

或者，您可以删除之前创建的 Athena 数据来源连接器，然后仅对 `spill_bucket` 使用 `{bucket_name}` 重新创建它。

设置最小可行数据空间，以便在组织之间共享数据

由 Ramy Hcini（Think-it）、Ismail Abdellaoui（Think-it）、Malte Gasseling（Think-it）、豪尔赫·埃尔南德斯·苏亚雷斯（AWS）和迈克尔·米勒（AWS）创作

摘要

数据空间是用于数据交换的联合网络，其核心原则是信任和控制自己的数据。它们通过提供经济实惠且与技术无关的解决方案，使组织能够大规模共享、交换和协作处理数据。

通过使用数据驱动的问题解决 end-to-end方法，让所有相关利益相关者都参与其中，数据空间有可能显著推动可持续未来的努力。

这种模式将引导您了解两家公司如何在 Amazon Web Services (AWS) 上使用数据空间技术来推动其碳减排战略向前发展的示例。在这种情况下，X公司提供碳排放数据，Y公司使用这些数据。有关以下数据空间规范的详细信息，请参阅 “其他信息” 部分：

- 参与者
- 商业案例
- 数据空间管理局
- 数据空间组件
- 数据空间服务
- 要交换的数据
- 数据模型
- Tractus-X EDC 连接器

该模式包括以下步骤：

- 部署由两个参与者运行的基本数据空间所需的基础架构 AWS。
- 以安全的方式使用连接器交换碳排放强度数据。

这种模式部署了一个 Kubernetes 集群，该集群将通过亚马逊 Elastic Kubernetes Service（亚马逊 EKS）托管数据空间连接器及其服务。

Eclipse 数据空间组件 (EDC) 控制平面和数据平面都部署在 A mazon EKS 上。官方的 Tractus-X Helm 图表将 PostgreSQL 和 Vault 服务部署为依赖项。 HashiCorp

此外，身份服务部署在亚马逊弹性计算云 (Amazon EC2) 上，以复制最小可行数据空间 (MVDS) 的现实生活场景。

先决条件和限制

先决条件

- 在您选择 AWS 账户 的环境中部署基础架构 AWS 区域
- 有权访问 Amazon S3 并暂时用作技术用户的 AWS Identity and Access Management (IAM) 用户（EDC 连接器目前不支持使用角色。 我们建议您专门为此演示创建一个 IAM 用户，该用户将拥有与之关联的有限权限。 ）
- AWS Command Line Interface (AWS CLI) 已在您选择的设备中安装和配置 AWS 区域
- AWS 安全凭证
- 在你的工作站上@@ 使用 eksctl
- 在你的工作站上@@ 使用 Git
- kubectl
- Helm
- 邮差
- AWS Certificate Manager (ACM) SSL/TLS 证书
- 指向 Application Load Balancer 的 DNS 名称（ACM 证书必须覆盖 DNS 名称）
- HashiCorp Vault（有关使用 AWS Secrets Manager 管理密钥的信息，请参阅 “其他信息” 部分。 ）

产品版本

- AWS CLI 版本 2+
- 邮差合集 v2.1

限制

- 连接器选择 – 此部署使用基于 EDC 的连接器。但是，请务必考虑 EDC 和 FIWARE True 连接器的优势和功能，以便根据部署的具体需求做出明智的决定。
- EDC 连接器构建 – 所选的部署解决方案依赖于 Tractus-X EDC Connect or Helm 图表，这是一个成熟且经过广泛测试的部署选项。之所以决定使用此图表，是因为它的常见用法以及在提供的版本中包含了基本的扩展。虽然 PostgreSQL HashiCorp 和 Vault 是默认组件，但您可以根据需要灵活地自定义自己的连接器构建。

- 私有集群访问权限 – 仅限私有渠道访问已部署的 EKS 集群。与集群的交互只能通过使用kubectl和 IAM 来执行。可以通过使用负载均衡器和域名来实现对群集资源的公开访问，必须有选择地实现这些功能，才能将特定服务暴露给更广泛的网络。但是，我们不建议提供公共访问权限。
- 安全重点 – 重点是将安全配置抽象为默认规范，这样您就可以集中精力 EDC 连接器数据交换所涉及的步骤。尽管保留了默认的安全设置，但在将集群暴露给公共网络之前，必须启用安全通信。这种预防措施确保了安全数据处理的坚实基础。
- 基础设施成本 ¿基础设施成本的估算值可通过以下方式找到AWS 定价计算器。一个简单的计算表明，部署的基础架构每月的成本可能高达162.92美元。

架构

MVDS 架构包括两个虚拟私有云 (VPCs)，一个用于动态属性配置系统 (DAPS) 身份服务，另一个用于 Amazon EKS。

DAPS 架构

下图显示了在由 Auto Scaling 组控制的 EC2 实例上运行的 DAPS。Application Load Balancer 和路由表公开 DAPS 服务器。Amazon Elastic File System（亚马逊 EFS）在 DAPS 实例之间同步数据。

亚马逊 EKS 架构

数据空间被设计为与技术无关的解决方案，并且存在多种实现方式。此模式使用 Amazon EKS 集群来部署数据空间技术组件。下图显示了 EKS 集群的部署情况。工作节点安装在私有子网中。Kubernetes 容器可以访问同样位于私有子网中的 PostgreSQL 版亚马逊关系数据库服务 (Amazon RDS) 实例。Kubernetes 容器在 Amazon S3 中存储共享数据。

工具

AWS 服务

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在资源的整个生命周期中跨地区对其 AWS 账户 进行管理。
- 亚马逊弹性计算云 (Amazon EC2) 在中提供可扩展的计算容量 AWS Cloud。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。

- Amazon Elastic File System（Amazon EFS）可帮助您在 AWS Cloud中创建和配置共享文件系统。
- 亚马逊 Elastic Kubernetes Service（亚马逊 EKS）可帮助你在上面运行 AWS Kubernetes，而无需安装或维护自己的 Kubernetes 控制平面或节点。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- 弹性负载均衡（ELB）将传入的应用程序或网络流量分配到多个目标。例如，您可以在一个或多个可用区域中的 EC2 实例、容器和 IP 地址之间分配流量。

其他工具

- eksctl 是一种用于在 Amazon EKS 上创建和管理 Kubernetes 集群的命令行实用程序。
- Git 是一个开源的分布式版本控制系统。
- HashiCorp Vault 提供安全存储，可控制凭据和其他敏感信息的访问权限。
- Helm 是一款适用于 Kubernetes 的开源软件包管理器，可帮助你在 Kubernetes 集群上安装和管理应用程序。
- kubectl：针对 Kubernetes 集群运行命令的命令行界面。
- Postman 是一个 API 平台。

代码存储库

此模式的 Kubernetes 配置 YAML 文件和 Python 脚本可在存储库中找到。 GitHub aws-patterns-edc该模式还使用 Tractus-X E DC 存储库。

最佳实践

Amazon EKS 和参与者基础设施的隔离

在这种模式下，Kubernetes 中的命名空间会将 X 公司提供商的基础架构与 Y 公司的消费者的基础设施分开。有关更多信息，请参阅 EKS 最佳实践指南。

在更现实的情况下，每个参与者都将在自己的集群中运行一个单独的 Kubernetes 集群。 AWS 账户共享基础设施（这种模式下的DAPS）将可供数据空间参与者访问，同时与参与者的基础设施完全分开。

操作说明

设置环境并配置 EKS 集群和 EC2 实例

Task	描述	所需技能
克隆存储库。	要将存储库克隆到您的工作站，请运行以下命令： `git clone https://github.com/Think-iT-Labs/aws-patterns-edc` 工作站必须有权访问您的 AWS 账户。	DevOps 工程师

配置 Kubernetes 集群并设置命名空间。	要在您的账户中部署简化的默认 EKS 集群，请在克隆存储库的工作站上运行以下`eksctl`命令：
	`eksctl create cluster`
	该命令创建跨越三个不同可用区的 VPC 以及私有和公有子网。创建网络层后，该命令将在一个 Auto Scaling 组中创建两个`m5.large` EC2 实例。
	有关更多信息和输出示例，请参阅 eksctl 指南。
	配置私有集群后，通过运行以下命令将新的 EKS 集群添加到本地 Kubernetes 配置中：
	`aws eks update-kubeconfig --name <EKS`
		DevOps 工程师

Task	描述	所需技能
	CLUSTER NAME> --region <AWS REGION> 此模式使用eu-west-1 AWS 区域 来运行所有命令。但是，您可以根据自己的喜好运行相同的命令 AWS 区域。 要确认您的 EKS 节点正在运行且处于就绪状态，请运行以下命令： `kubectl get nodes`	

设置命名空间。	要为提供者和使用者创建命名空间，请运行以下命令：
	
	`kubectl create ns provider`
	`kubectl create ns consumer`
	
	在这种模式中，使用provider和consumer作为命名空间以适应下一步的配置非常重要。
		DevOps 工程师

部署身份服务

Task	描述	所需技能
使用部署 DAPS AWS CloudFormation。	为了便于管理 DAPS 操作，DAPS 服务器安装在实例上 EC2 。	DevOps 工程师

Task	描述	所需技能
	要安装 DAPS，请使用AWS CloudFormation 模板。您需要先决条件部分中的 ACM 证书和 DNS 名称。该模板部署和配置以下内容： • 应用程序负载均衡器 • 自动扩缩组 • EC2 实例配置了用户数据以安装所有必需的软件包 • IAM 角色 • DAPS 您可以登录 AWS Management Console 并使用AWS CloudFormation 控制台来部署 AWS CloudFormation 模板。您也可以使用如下 AWS CLI 命令来部署模板： `aws cloudformation create-stack --stack-name daps \ --template-body file://aws-patterns-edc/cloudformation.yml --parameters \ ParameterKey=CertificateARN,ParameterValue=<ACM Certificate ARN> \ ParameterKey=DNSName,ParameterValue=<DNS name> \ ParameterKey=InstanceType,Paramete`	

Task	描述	所需技能
	rValue=<EC2 instance type> \ ParameterKey=EnvironmentName,ParameterValue=<Environment Name> --capabilities CAPABILITY_NAMED_IAM 环境名称由您自己选择。我们建议使用有意义的术语`DapsInfrastructure`，例如，因为它会反映在 AWS 资源标签中。 对于这种模式，`t3.small`它足够大，可以运行 DAPS 工作流程，该工作流程有三个 Docker 容器。 该模板将 EC2 实例部署在私有子网中。这意味着无法通过 SSH（安全外壳）从互联网直接访问这些实例。这些实例已配置必要的 IAM 角色和 AWS Systems Manager 代理，以便能够通过会话管理器访问正在运行的 AWS Systems Manager实例，该功能为。 我们建议使用会话管理器进行访问。或者，您可以配置堡垒主机以允许从互联网进行 SSH 访问。使用堡垒主机方法时，EC2 实例可能需要几分钟才能开始运行。	

Task	描述	所需技能

	成功部署 AWS CloudFormation 模板后，将 DNS 名称指向您的 Application Load Balancer DNS 名称。要确认这一点，请运行以下命令：
	
	`dig <DNS NAME>`
	
	该输出应该类似于以下内容：
	
	`; <<>> DiG 9.16.1-Ubuntu <<>> edc-pattern.think-it.io`
	`;; global options: +cmd`
	`;; Got answer:`
	`;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 42344`
	`;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 1`
	
	`;; OPT PSEUDOSECTION:`
	`; EDNS: version: 0, flags:; udp: 65494`
	`;; QUESTION SECTION:`
	`;edc-pattern.think-it.io. IN A`
	
	`;; ANSWER SECTION:`
	`edc-pattern.think-it.io. 276 IN CNAME daps-alb-iap9zmwy3kn8-1328773120.eu-west-1.elb.amazonaws.com.`
	`daps-alb-iap9zmwy3kn8-1328773120.eu-w`	

Task	描述	所需技能
	est-1.elb.amazonaw s.com. 36 IN A 52.208.240.129 daps-alb-iap9zmwy3kn8 -1328773120.eu-wes t-1.elb.amazonaws. com. 36 IN A 52.210.15 5.124	

Task	描述	所需技能

将参与者的连接器注册到 DAPS 服务。	在为 DAPS 预置的任何 EC2 实例中，注册参与者：
	1. 使用 root 用户在 EC2 实例上运行可用的脚本：
	`cd /srv/mvds/omejdn-daps`
	2. 注册提供商：
	`bash scripts/register_connector.sh <provider_name>`
	3. 注册消费者：
	`bash scripts/register_connector.sh <consumer_name>`
	名称的选择不会影响后续步骤。我们建议使用`provider`和`consumer`或`com`
	注册命令还将使用从创建的证书和密钥中获取的所需信息自动配置 DAPS 服务。
	登录到 DAPS 服务器时，请收集后续安装步骤所需的信息：
	1. `omejdn-daps/config/clients.yml` 从`client id`为提供者和消费者获取。这
		DevOps 工程师

Task	描述	所需技能
	些client id值是十六进制数字的长字符串。 2. 从omejdn-daps/keys 目录中复制、consumer.cert consumer.key provider.cert 、和provider.key 文件的内容。 我们建议将文本复制并粘贴到工作站daps-上以相似名称为前缀的文件中。 您应该拥有提供者和 IDs 使用者的客户端，并且在工作站的工作目录中应有四个文件： • 源文件名consumer.cert 变为工作站文件名daps-consumer.cert 。 • 源文件名consumer.key 变为工作站文件名daps-consumer.key 。 • 源文件名provider.cert 变为工作站文件名daps-provider.cert 。 • 源文件名provider.key 变为工作站文件名daps-provider.key 。	

部署参与者的连接器

Task	描述	所需技能

克隆 Tractus-X EDC 存储库并使用 0.4.1 版本。	Tractus-X EDC 连接器的版本需要部署和使用 PostgreSQL（资产数据库）和 HashiCorp 保管库（机密管理）服务。
	
	Tractus-X EDC Helm 图表有许多不同的版本。此模式指定版本 0.4.1，因为它使用 DAPS 服务器。
	
	最新版本使用托管身份钱包 (MIW) 和身份服务的分布式实现。
	
	在你创建两个 Kubernetes 命名空间的工作站上，克隆 tractusx-edc 存储库，然后查看分支。`release/0.4.1`
	
	`git clone https://github.com/eclipse-tractusx/tractusx-edc`
	
	`cd tractusx-edc`
	
	`git checkout release/0.4.1`
		DevOps 工程师

配置 Tractus-X EDC Helm 控制图。	修改 Tractus-X Helm 图表模板配置，使两个连接器能够一起交互。 为此，您需要将命名空间添加到服务的 DNS 名称中，以便集群中的其他服务可以对	DevOps 工程师

Task	描述	所需技能

	其进行解析。应对charts/tractusx-connector/templates/_helpers.tpl 文件进行这些修改。此模式提供了此文件的最终修改版本供您使用。将其复制并放在文件的daps部分中charts/tractusx-connector/templates/_helpers.tpl 。
	
	请务必在以下位置注释所有 DAPS 依赖项：charts/tractusx-connector/Chart.yaml
	
	`dependencies:`
	`  # IDS Dynamic Attribute Provisioning Service (IAM)`
	`  #  - name: daps`
	`  # version: 0.0.1`
	`  # repository: "file://./subcharts/omejdn"`
	`  # alias: daps`
	`  # condition: install.daps`	

Task	描述	所需技能

将连接器配置为在亚马逊 RDS 上使用 PostgreSQL。	（可选）此模式中不需要亚马逊关系数据库服务 (Amazon RDS) 实例。但是，我们强烈建议使用 Amazon RDS 或 Amazon Aurora，因为它们提供高可用性以及备份和恢复等功能。
	
	要将 Kubernetes 上的 PostgreSQL 替换为 Amazon RDS，请执行以下操作：
	
	1. 预配置 Amazon RDS for PostgreSQL 实例。
	2. 在中`Chart.yaml` ，评论该`PostgreSQL` 部分。
	3. 在`provider_values.yml` 和中`consumer_values.yml` ，按如下方式配置该`postgresql` 部分：
	
	`postgresql:`
	`  auth:`
	`    database: edc`
	`    password: <RDS PASSWORD>`
	`    username: <RDS Username>`
	`  jdbcUrl: jdbc:postgresql://<RDS DNS NAME>:5432/edc`
	`  username: <RDS Username>`
	`  password: <RDS PASSWORD>`
		DevOps 工程师

Task	描述	所需技能

	<pre>primary:
	  persistence:
	  enabled: false
	readReplicas:
	  persistence:
	  enabled: false</pre>	

Task	描述	所需技能
配置和部署提供商连接器及其服务。	要配置提供商连接器及其服务，请执行以下操作： 1. 要将`provider_edc.yaml` 文件从`edc_helm_configs` 目录下载到当前 Helm chart 文件夹，请运行以下命令： `wget -q https://raw.githubusercontent.com/Think-iT-Labs/aws-patterns-edc/main/edc_helm_configs/provider_edc.yaml> -P charts/tractusx-connector/` 2. 将以下变量（也标记在文件中）替换为其值： • `CLIENT_ID` – 由 DAPS 生成的身份证。`CLIENT_ID` 应该在 DAPS 服务器`/srv/mvds/omejdn-daps/config/clients.yml/config/clients.yml` 上。它应该是一串十六进制字符。 • `DAPS_URL`– DAPS 服务器的网址。它应该`https://{DNS name}`使用您在运行 AWS CloudFormation 模板时设置的 DNS 名称。	DevOps 工程师

Task	描述	所需技能

	• VAULT_TOKEN – 用于保管库授权的令牌。选择任意值。
	• vault.fullnameOverride – vault-provider 。
	• vault.hashicorp.url – http://vault-provider:8200/ 。
	前面的值假设部署名称和命名空间名称是 provider。
	3. 要从您的工作站运行 Helm 图表，请使用以下命令：
	`cd charts/tractusx-connector`
	`helm dependency build`
	`helm upgrade --install provider ./ -f provider_edc.yaml -n provider`	

Task	描述	所需技能

将证书和密钥添加到提供商保管库。	为避免混淆，请在`tractusx-edc/charts` 目录之外生成以下证书。
	
	例如，运行以下命令切换到您的主目录：
	
	`cd ~`
	
	现在，您需要将提供商所需的密钥添加到保管库中。
	
	保管库中密钥的名称是`provider_edc.yml` 文件`secretNames:` 部分中密钥的值。默认情况下，它们的配置如下：
	
	`secretNames:`
	`transferProxyTokenSignerPrivateKey: transfer-proxy-token-signer-private-key`
	`transferProxyTokenSignerPublicKey: transfer-proxy-token-signer-public-key`
	`transferProxyTokenEncryptionAesKey: transfer-proxy-token-encryption-aes-key`
	`dapsPrivateKey: daps-private-key`
		DevOps 工程师

Task	描述	所需技能
	dapsPubli cKey: daps-public-key	

	首先生成高级加密标准 (AES) 密钥、私钥、公钥和自签名证书。这些信息随后会作为机密添加到保管库中。
	
	此外，此目录应包含您从 DAPS 服务器复制的`daps-provider.cert` 和`daps-provider.key` 文件。
	
	1. 运行以下 命令：
	
	`# generate a private key`
	`openssl ecparam -name prime256v1 -genkey -noout -out provider-private-key.pem`
	`# generate corresponding public key`
	`openssl ec -in provider-private-key.pem -pubout -out provider-public-key.pem`
	`# create a self-signed certificate`
	`openssl req -new -x509 -key provider-private-key.pem -out provider-cert.pem -days 360`
	`# generate aes key`
	`openssl rand -base64 32 > provider-aes.key`	

Task	描述	所需技能

	2. 在将机密添加到保管库之前，请将换行符替换\n为：
	
	`cat provider-private-key.pem | sed 's/$/\\\n/'|tr -d '\\n' > provider-private-key.pem.line`
	`cat provider-public-key.pem | sed 's/$/\\\n/'|tr -d '\\n' > provider-public-key.pem.line`
	`cat provider-cert.pem | sed 's/$/\\\\n/'|tr -d '\\n' > provider-cert.pem.line`
	`cat provider-aes.key | sed 's/$/\\\\n/'|tr -d '\\n' > provider-aes.key.line`
	
	`## The following block is for daps certificate and key`
	`openssl x509 -in daps-provider.cert -outform PEM | sed 's/$/\\\\n/'|tr -d '\\n' > daps-provider.cert.line`
	`cat daps-provider.key | sed 's/$/\\\\n/'|tr -d '\\n' > daps-provider.key.line`
	
	3. 要格式化将添加到 Vault 中的密钥，请运行以下命令：	

Task	描述	所需技能
	``` JSONFORMAT='{"cont ent": "%s"}' #create a single line in JSON format printf "${JSONFO RMAT}\\n" "`cat  provider-private- key.pem.line`" > provider-private-k ey.json printf "${JSONFO RMAT}\\n" "`cat  provider-public- key.pem.line`" > provider-public-ke y.json printf "${JSONFO RMAT}\\n" "`cat  provider-cert.pem. line`" > provider- cert.json printf "${JSONFO RMAT}\\n" "`cat  provider-aes.key.l ine`" > provider- aes.json printf "${JSONFO RMAT}\\n" "`cat daps- provider.key.line`" > daps-provider.key. json printf "${JSONFO RMAT}\\n" "`cat daps- provider.cert.line`" > daps-provider.cert .json ``` 密钥现在采用 JSON 格式，可以随时添加到保管库中。	

Task	描述	所需技能

	4. 要获取文件库的 pod 名称，请运行以下命令：
	
	`kubectl get pods -n provider|egrep "vault|NAME"`
	
	Pod 名称将类似于"vault-provider-0" 。此名称用于创建转发到存储库的端口。端口转发允许您访问保管库以添加密钥。您应该在配置了 AWS 凭证的工作站上运行它。
	5. 要访问存储库，请使用配置kubectl转发端口：
	
	`kubectl port-forward <VAULT_POD_NAME> 8200:8200 -n provider`
	
	现在，您应该能够通过浏览器或 CLI 访问保管库了。
	
	浏览器
	
	1. 使用浏览器导航到 http://127.0.0.1:8200，它将使用您配置的转发端口。
	2. 使用您之前在中配置的令牌登录provider_edc.yml 。在密钥引擎中，创建三个密钥。每个密钥都有一个Path for this secret值，即以下	

Task	描述	所需技能
	列表中显示的密钥名称。在该secret data部分中，密钥的名称将是content，值将是名为的相应文件中的单行文本.line。 3. 机密名称来自provider_edc.yml 文件中的secretNames 部分。 4. 创建以下密钥： • transfer-proxy-token-signer-private-key 带有文件名的密钥 provider-private-key.pem.line • transfer-proxy-token-signer-public-key 带有文件名的密钥 provider-cert.pem.line • transfer-proxy-token-encryption-aes-key 带有文件名的密钥 provider-aes.key.line • daps-private-key 带有文件名的密钥 daps-provider.key.line • daps-public-key 带有文件名的密钥 daps-provider.cert.line	

Task	描述	所需技能

	Vault CLI
	
	CLI 还将使用您配置的转发端口。
	
	1. 在您的工作站上，按照保险柜文档中的说明安装 HashiCorp Vault CLI。
	2. 要使用您在中设置的令牌登录文件库`provider_edc.yml` ，请运行以下命令：
	
	`vault login -address=http://127.0.0.1:8200`
	
	使用正确的令牌，您应该会看到消息 `"Success! You are now authenticated."`
	3. 要使用之前创建的 JSON 格式文件创建密钥，请运行以下代码：
	
	`vault kv put -address=http://127.0.0.1:8200 secret/transfer-proxy-token-signer-private-key @provider-private-key.json`
	`vault kv put -address=http://127.0.0.1:8200 secret/transfer-proxy-token-signer-public-key @provider-cert.json`	

Task	描述	所需技能
	vault kv put -address=http://127.0.0.1:8200 secret/transfer-proxy-token-encryption-aes-key @provider-aes.json vault kv put -address=http://127.0.0.1:8200 secret/daps-private-key @daps-provider.key.json vault kv put -address=http://127.0.0.1:8200 secret/daps-public-key @daps-provider.cert.json	

Task	描述	所需技能

配置和部署使用者连接器及其服务。	配置和部署使用者的步骤与您为提供商完成的步骤类似：
	
	1. 要consumer_edc.yaml 从 aws-patterns-edcrepo 中复制到 tractusx-edc/charts/tractusx-connector 文件夹，请运行以下命令：
	
	`cd tractusx-edc`
	
	`wget -q https://raw.githubusercontent.com/Think-iT-Labs/aws-patterns-edc/main/edc_helm_configs/consumer_edc.yaml -P charts/tractusx-connector/`
	
	2. 用其实际值更新以下变量：
	• CONSUMER_CLIENT_ID – 由 DAPS 生成的身份证。CONSUMER_CLIENT_ID 应该在 DAPS 服务器config/clients.yml 上。
	• DAPS_URL– 与您用于提供商的 DAPS 网址相同。
	• VAULT_TOKEN – 用于保管库授权的令牌。选择任意值。	

Task	描述	所需技能

	• vault.fullnameOverride – vault-consumer
	• vault.hashicorp.url – http://vault-provider:8200/
	
	前面的值假设部署名称和命名空间名称为consumer。
	3. 要运行 Helm 图表，请使用以下命令：
	
	`cd charts/tractusx-connector`
	
	`helm upgrade --install consumer ./ -f consumer_edc.yaml -n consumer`	

Task	描述	所需技能

将证书和密钥添加到使用者保管库。	从安全角度来看，我们建议为每个数据空间参与者重新生成证书和密钥。此模式为使用者重新生成证书和密钥。
	
	这些步骤与提供商的步骤非常相似。您可以验证consumer_edc.yml 文件中的密钥名称。
	
	保管库中密钥的名称是该secretNames: 部分中密钥的值consumer_edc.yml file。默认情况下，它们的配置如下：
	
	`secretNames:`
	`  transferProxyTokenSignerPrivateKey: transfer-proxy-token-signer-private-key`
	`  transferProxyTokenSignerPublicKey: transfer-proxy-token-signer-public-key`
	`  transferProxyTokenEncryptionAesKey: transfer-proxy-token-encryption-aes-key`
	`  dapsPrivateKey: daps-private-key`
	`  dapsPublicKey: daps-public-key`
		DevOps 工程师

Task	描述	所需技能
	您从 DAPS 服务器复制的daps-consumer.cert 和daps-consumer.key 文件应该已经存在于此目录中。 1. 运行以下 命令： ``` # generate a private key openssl ecparam -name prime256v1 -genkey -noout -out consumer-private-key.pem # generate corresponding public key openssl ec -in consumer-private-key.pem -pubout -out consumer-public-key.pem # create a self-signed certificate openssl req -new -x509 -key consumer-private-key.pem -out consumer-cert.pem -days 360 # generate aes key openssl rand -base64 32 > consumer-aes.key ``` 2. 手动编辑文件以替换换行符\n，或者使用三个类似于以下内容的命令： ``` cat consumer-private-key.pem | sed 's/$/\\\n/'|tr -d '\\n' >	

Task	描述	所需技能
	consumer-private-key.pem.line cat consumer-public-key.pem | sed 's/$/\\\n/'\|tr -d '\\n' > consumer-public-key.pem.line cat consumer-cert.pem \| sed 's/$/\\\\n/'|tr -d '\\n' > consumer-cert.pem.line cat consumer-aes.key | sed 's/$/\\\\n/'\|tr -d '\\n' > consumer-aes.key.line cat daps-consumer.cert \| sed 's/$/\\\\n/'|tr -d '\\n' > daps-consumer.cert.line cat daps-consumer.key | sed 's/$/\\\\n/'\|tr -d '\\n' > daps-consumer.key.line 3. 要格式化将添加到 Vault 中的密钥，请运行以下命令： JSONFORMAT='{"content": "%s"}' #create a single line in JSON format printf "${JSONFORMAT}\\n" "`cat consumer-private-key.pem.line`" >	

Task	描述	所需技能
	consumer-private-key.json printf "${JSONFORMAT}\\n" "`cat consumer-public-key.pem.line`" > consumer-public-key.json printf "${JSONFORMAT}\\n" "`cat consumer-cert.pem.line`" > consumer-cert.json printf "${JSONFORMAT}\\n" "`cat consumer-aes.key.line`" > consumer-aes.json printf "${JSONFORMAT}\\n" "`cat daps-consumer.key.line`" > daps-consumer.key.json printf "${JSONFORMAT}\\n" "`cat daps-consumer.cert.line`" > daps-consumer.cert.json 密钥现在采用 JSON 格式，可以随时添加到保管库中。 4. 要获取使用者保管库的 pod 名称，请运行以下命令： kubectl get pods -n consumer | egrep "vault|NAME"	

Task	描述	所需技能
	Pod 名称将类似于"vault-consumer-0" 。此名称用于创建转发到存储库的端口。端口转发允许您访问保管库以添加密钥。您应该在配置了 AWS 凭据的工作站上运行它。 5. 要访问存储库，请使用配置kubectl转发端口： `kubectl port-forward <VAULT_POD_NAME> 8201:8200 -n consumer` 这次的本地端口是 8201，因此您可以为生产者和消费者设置端口转发。 浏览器 您可以使用浏览器连接到 http://localhost:8201/ 以访问消费者保管库，并使用概述的名称和内容创建机密。 包含内容的密钥和文件如下： • transfer-proxy-token-signer-private-key 带有文件名的密钥 consumer-private-key.pem.line • transfer-proxy-token-signer-public-key 带有文件名的密钥	

Task	描述	所需技能

	consumer-cert.pem.line
	• transfer-proxy-token-encryption-aes-key 带有文件名的密钥 consumer-aes.key.line
	
	Vault CLI
	
	使用 Vault CLI，您可以运行以下命令登录文件库并创建密钥：
	
	1. 使用您在以下位置配置的令牌登录文件库consumer_edc.yml ：
	
	`vault login -address=http://127.0.0.1:8201`
	
	使用正确的令牌，您应该会看到消息 "Success! You are now authenticated."
	2. 要使用您之前创建的 JSON 格式文件创建密钥，请运行以下代码：
	
	`vault kv put -address=http://127.0.0.1:8201 secret/transfer-proxy-token-signer-private-key @consumer-private-key.json`	

Task	描述	所需技能

	`vault kv put -address=http://127.0.0.1:8201 secret/transfer-proxy-token-signer-public-key @consumer-cert.json`
	`vault kv put -address=http://127.0.0.1:8201 secret/transfer-proxy-token-encryption-aes-key @consumer-aes.json`
	`vault kv put -address=http://127.0.0.1:8201 secret/daps-private-key @daps-consumer.key.json`
	`vault kv put -address=http://127.0.0.1:8201 secret/daps-public-key @daps-consumer.cert.json`	

设置 HTTP 客户端以与连接器的管理 API 进行交互

Task	描述	所需技能

设置端口转发。	1. 要检查 Pod 的状态，请运行以下命令：
	`kubectl get pods -n provider`
	`kubectl get pods -n consumer`
	2. 要确保 Kubernetes 部署成功，请运行以下命令来
		DevOps 工程师

Task	描述	所需技能

	查看提供程序和使用者 Kubernetes 容器的日志：
	`kubectl logs -n provider <producer control plane pod name>`
	`kubectl logs -n consumer <consumer control plane pod name>`
	该集群是私有的，不能公开访问。要与连接器交互，请使用 Kubernetes 端口转发功能将计算机生成的流量转发到连接器控制平面。
	1. 在第一个终端上，通过端口 8300 将消费者的请求转发到管理 API：
	`kubectl port-forward deployment/consumer-tractusx-connector-controlplane 8300:8081 -n consumer`
	2. 在第二个终端上，通过端口 8400 将提供商的请求转发到管理 API：
	`kubectl port-forward deployment/provider-tractusx-connector-controlplane 8400:8081 -n provider`	

Task	描述	所需技能
为提供者和使用者创建 S3 存储桶。	EDC 连接器目前不使用临时的 AWS 证书，例如代入角色时提供的证书。EDC 仅支持使用 IAM 访问密钥 ID 和私有访问密钥组合。 后续步骤需要两个 S3 存储桶。一个 S3 存储桶用于存储提供商提供的数据。另一个 S3 存储桶用于存储消费者接收的数据。 IAM 用户应仅有权读取和写入两个已命名的存储桶中的对象。 需要创建访问密钥 ID 和私有访问密钥 pair 并妥善保管。在此 MVDS 停用后，应删除 IAM 用户。 以下代码是该用户的 IAM 策略示例： `{ "Version": "2012-10-17", "Statement": [ { "Sid": "Stmt1708699805237", "Action": [ "s3:GetObject", "s3:GetObjectVersion", "s3:ListAllMyBuckets",`	DevOps 工程师

Task	描述	所需技能
	"s3:ListB ucket", "s3:ListB ucketMultipartUplo ads", "s3:ListB ucketVersions", "s3:PutObject"], "Effect": "Allow", "Resource": ["arn:aws: s3:::<S3 Provider Bucket>", "arn:aws: s3:::<S3 Consumer Bucket>", "arn:aws: s3:::<S3 Provider Bucket>/*", "arn:aws: s3:::<S3 Consumer Bucket>/*"] }] }	
将 Postman 设置为与连接器进行交互。	现在，您可以通过您的 EC2 实例与连接器进行交互。使用 Postman 作为 HTTP 客户端，为提供者和消费者连接器提供 Postman 集合。 将集合从`aws-pattern-edc`存储库导入到你的 Postman 实例。 此模式使用 Postman 集合变量为您的请求提供输入。	应用程序开发人员、数据工程师

通过连接器提供 X 公司的碳排放足迹数据

Task	描述	所需技能
准备要共享的碳排放强度数据。	首先，您需要决定要共享的数据资产。X公司的数据代表了其车队的碳排放足迹。根据（WTW）测量，重量是以吨为单位的车辆总重（GVW），排放量以克/千米（g e CO2 /t-km）为单位： CO2 Wheel-to-Well • 车辆类型：面包车；重量：< 3.5；排放：800 • 车辆类型：城市卡车；重量：3.5−7.5；排放：315 • 车辆类型：中型货车 (MGV)；重量：7.5−20；排放：195 • 车辆类型：重型货车 (HGV)；重量：> 20；排放：115 示例数据位于`aws-patterns-edc` 存储库`carbon_emissions_data.json` 的文件中。 X 公司使用 Amazon S3 来存储对象。 创建 S3 存储桶并将示例数据对象存储在那里。以下命令使用默认安全设置创建 S3 存储桶。我们强烈建议您查阅 Amazon S3 的安全最佳实践。	数据工程师、应用程序开发人员

Task	描述	所需技能

	`aws s3api create-bucket <BUCKET_NAME> --region <AWS_REGION>`
	`# You need to add '--create-bucket-configuration`
	`# LocationConstraint=<AWS_REGION>' if you want to create # the bucket outside of us-east-1 region`
	
	`aws s3api put-object --bucket <BUCKET_NAME> \`
	`--key <S3 OBJECT NAME> \`
	`--body <PATH OF THE FILE TO UPLOAD>`
	
	S3 存储桶名称应具有全球唯一性。有关命名规则的更多信息，请参阅 AWS 文档。	

Task	描述	所需技能
使用 Postman 将数据资产注册到提供商的连接器。	EDC 连接器数据资产保存数据的名称及其位置。在这种情况下，EDC 连接器数据资产将指向 S3 存储桶中创建的对象： • 连接器：提供商 • 请求：创建资产 • 集合变量：更新ASSET_NAME 。选择一个代表资产的有意义的名称。 • 请求正文：使用您为提供商创建的 S3 存储桶更新请求正文。 `"dataAddress": { "edc:type": "AmazonS3", "name": "Vehicle Carbon Footprint", "bucketName": "<REPLACE WITH THE SOURCE BUCKET NAME>", "keyName": "<REPLACE WITH YOUR OBJECT NAME>", "region": "<REPLACE WITH THE BUCKET REGION>", "accessKeyId": "<REPLACE WITH YOUR ACCESS KEY ID>", "secretAccessKey": "<REPLACE WITH SECRET ACCESS KEY>" }`	应用程序开发人员、数据工程师

Task	描述	所需技能
	• 响应：成功请求将返回创建时间和新创建资产的资产 ID。 `{ "@id": "c89aa31c-ec4c-44ed-9e8c-1647f19d7583" }` • 集合变量 ASSET_ID：ASSET_ID使用 EDC 连接器在创建后自动生成的 ID 更新 Postman 集合变量。	

定义资产的使用政策。	EDC 数据资产必须与明确的使用政策相关联。首先，在提供商连接器中创建策略定义。
	
	X公司的政策是允许数据空间的参与者使用碳排放足迹数据。
	
	• 请求正文：
	  • 连接器：提供商
	  • 请求：创建策略
	  • 集合变量：使用策略名称更新Policy Name变量。
	• 响应：成功请求将返回创建时间和新创建策略的策略 ID。使用 EDC 连接POLICY_ID 器在创建后生成的策略的 ID 更新集合变量。
		应用程序开发人员、数据工程师

Task	描述	所需技能
为资产及其使用政策定义 EDC 合同报价。	要允许其他参与者请求访问您的数据，请在合同中提供该数据，该合同规定了使用条件和权限： • 连接器：提供商 • 请求：创建合同定义 • 集合变量：使用合约报价的名称或定义更新`Contract Name`变量。	应用程序开发人员、数据工程师

发现资产并就已定义的合同达成协议

Task	描述	所需技能
索取 X 公司共享的数据目录。	作为数据领域的数据消费者，Y 公司首先需要发现其他参与者共享的数据。 在此基本设置中，您可以通过要求消费者连接器直接向提供商连接器请求可用资产目录来实现此目的。 • 连接器：消费类 • 请求：索取产品目录 • 响应：提供商提供的所有可用数据资产及其附带的使用政策。作为数据使用者，请查找您感兴趣的合同，并相应地更新以下集合变量。 • `CONTRACT_OFFER_ID` - 消费者想要协商的合同报价的ID	应用程序开发人员、数据工程师

Task	描述	所需技能
	• ASSET_ID-消费者想要协商的资产的ID • PROVIDER_CLIENT_ID – 要与之协商的提供商连接器的 ID	
就X公司的碳排放强度数据启动合同谈判。	既然您已经确定了要消耗的资产，那么在消费者和提供商连接器之间启动合同谈判流程。 • 连接器：消费类 • 请求：合同谈判 • 集合变量：使用要与之协商的使用者连接器的 ID 更新CONSUMER_CLIENT_ID 变量。 该过程可能需要一些时间才能达到“已验证”状态。 您可以使用Get Negotiation 请求来检查合同谈判的状态和相应的协议 ID。	应用程序开发人员、数据工程师

使用合同协议使用数据

Task	描述	所需技能
使用来自 HTTP 端点的数据。	（选项 1）要使用 HTTP 数据平面来使用数据空间中的数据，可以使用 webhook.site 来模拟 HTTP 服务器，然后在使用者连接器中启动传输过程：	应用程序开发人员、数据工程师

Task	描述	所需技能

	• 连接器：消费类
	• 请求：合同谈判
	• 集合变量：使用 EDC 连接器生成的合约协议的 ID 更新`Contract Agreement ID`变量。
	• 请求正文：更新请求正文，将其指定HTTP为 webhook 网址`dataDestination` 旁边：
	
	`{ "dataDestination": { "type": "HttpProxy" }, "privateProperties": { "receiverHttpEndpoint": "<WEBHOOK URL>" } }`
	
	连接器会将下载文件所需的信息直接发送到 webhook 网址。
	
	收到的有效载荷类似于以下内容：
	
	`{ "id": "dcc90391-3819-4b54-b401-1a005a029b78",`	

Task	描述	所需技能
	<pre>"endpoint": "http://consumer-tractusx-connector-dataplane.consumer:8081/api/public", "authKey": "Authorization", "authCode": "<AUTHCODE YOU RECEIVE IN THE ENDPOINT>", "properties": { "https://w3id.org/edc/v0.0.1/ns/cid": "vehicle-carbon-footprint-contract:4563abf7-5dc7-4c28-bc3d-97f45e32edac:b073669b-db20-4c83-82df-46b583c4c062" } }</pre> 使用收到的凭证获取提供商共享的 S3 资产。 在最后一步中，您必须将请求发送到消费者数据平面（正确转发端口），如有效载荷 (endpoint) 中所述。	

Task	描述	所需技能
直接使用 S3 存储桶中的数据。	（选项 2）使用 Amazon S3 与 EDC 连接器的集成，并直接指向使用者基础设施中的 S3 存储桶作为目标： • 请求正文：更新请求正文以将 S3 存储桶指定为 DataDestination。 这应该是您之前为存储使用者收到的数据而创建的 S3 存储桶。 `{ "dataDestination": { "type": "AmazonS3", "bucketName": "{{ REPLACE WITH THE DESTINATION BUCKET NAME }}", "keyName": "{{ REPLACE WITH YOUR OBJECT NAME }}", "region": "{{ REPLACE WITH THE BUCKET REGION }}", "accessKeyId": "{{ REPLACE WITH YOUR ACCESS KEY ID }}", "secretAccessKey": "{{ REPLACE WITH SECRET ACCESS KEY }}" } }`	数据工程师、应用程序开发人员

故障排除

事务	解决方案
连接器可能会引发有关证书 PEM 格式的问题。	通过添加将每个文件的内容连接成一行。\n

相关资源

- DSSC
- 为可持续发展用例构建数据空间（Think-it 的 AWS Prescriptive Guiden c e 策略）
- 适用于数据空间的 AWS
- Tractus-X 文档
- DAPS
- 通过数据空间和 AWS 实现数据共享（博客文章）

其他信息

数据空间规格

参与者

参与者	公司简介	公司的重点
X 公司	运营一支横跨欧洲和南美的车队，用于运输各种货物。	旨在做出以数据为导向的决策，以降低其碳排放足迹强度。
Y 公司	环境监管机构	执行环境法规和政策，旨在监测和减轻企业和行业对环境的影响，包括碳排放强度。

商业案例

X 公司使用数据空间技术与合规审计机构 Y 公司共享碳足迹数据，以评估和解决 X 公司物流运营对环境的影响。

数据空间管理局

数据空间管理局是由管理数据空间的组织组成的联盟。在这种模式下，X 公司和 Y 公司组成治理机构，代表联邦数据空间管理机构。

数据空间组件

组件	选择的实现	其他信息
数据集交换协议	数据空间协议版本 0.8	• JSON-LD • 数据目录词汇 (DCAT)
数据空间连接器	Tractus-X EDC 连接器版本 0.4.1	• EDC 扩展
数据交换政策	默认使用政策	• 开放数字版权语言 (ODRL)

数据空间服务

服务	实施	其他信息
身份服务	动态属性配置系统 (DAPS)	“动态属性配置系统 (DAPS) 旨在确定组织和连接器的某些属性。因此，第三方无需信任后者，前提是他们信任DAPS的说法。” — DAPS 为了专注于连接器的逻辑，数据空间是使用 Docker Compose 在亚马逊 EC2 计算机上部署的。
发现服务	Gaia-X 联邦目录	“联邦目录构成了Gaia-X自我描述的索引存储库，用于发现和选择提供商及其服务产品。自我描述是参与者以财产和索赔的形式提供的有关自己及其服

务的信息。” — Gaia-X 生态系统 Kickstarter

要交换的数据

数据资产	描述	格式
碳排放数据	整个车队中指定区域（欧洲和南美）不同车辆类型的强度值	JSON 文件

数据模型

```
{
  "region": "string",
  "vehicles": [
// Each vehicle type has its Gross Vehicle Weight (GVW) category and its emission
 intensity in grams of CO2 per Tonne-Kilometer (g CO2 e/t-km) according to the "Well-
to-Wheel" (WTW) measurement.
    {
      "type": "string",
      "gross_vehicle_weight": "string",
      "emission_intensity": {
        "CO2": "number",
        "unit": "string"
      }
    }
  ]
}
```

Tractus-X EDC 连接器

有关每个 Tractus-X EDC 参数的文档，请参阅原始值文件。

下表列出了所有服务及其相应的暴露端口和端点以供参考。

服务名称	端口和路径
控制面板	• 管理：– 端口：8081 路径：/management

	• 控制 – 端口：8083 路径：/control • 协议端口：8084 路径：/api/v1/dsp • m etrics – 端口：9090 路径：/metrics • 可观察性 – 端口：8085 路径：/observability
数据层面	默认 – 端口：8080 路径：/api p@@ ubli c – 端口：8081 路径：/api/dataplane/control 代理 – 端口：8186 路径：/proxy metrics – 端口：9090 路径：/metrics 可观察性 – 端口：8085 路径：/observability
文件库	端口：8200
PostgreSQL	端口：5432

使用 AWS Secrets Manager 管理器

可以使用 Secrets Manager 代替 HashiCorp Vault 作为密钥管理器。为此，您必须使用或构建 AWS Secrets Manager EDC 扩展。

你将负责创建和维护自己的镜像，因为 Tractus-X 不支持 Secrets Manager。

为此，你需要通过引入你的 AWS Secrets Manager EDC 扩展来修改连接器的控制平面和数据平面的构建 Gradle 文件（有关示例，请参阅此 maven 工件），然后构建、维护和引用 Docker 镜像。

有关重构 Tractus-X 连接器 Docker 镜像的更多见解，请参阅 Refactus-X EDC Helm 图表。

为简单起见，我们避免以这种模式重建连接器映像，而是使用 HashiCorp Vault。

使用标量 Python UDF 为 Amazon Redshift 查询结果设置特定于语言的排序

由 Ethan Stark (AWS) 编写

摘要

此模式提供了使用标量 Python UDF（用户定义的函数）为 Amazon Redshift 查询结果设置不区分大小写的语言排序的步骤和示例代码。有必要使用标量 Python UDF，因为 Amazon Redshift 返回基于二进制 UTF-8 排序的结果，并且不支持特定于语言的排序。Python UDF 是基于 Python 2.7 程序并在数据仓库运行的非 SQL 处理代码。您可在单个查询中使用 SQL 语句运行 Python UDF 代码。有关更多信息，请参阅 A mazon Redshift UDFs 中的 Python 简介 AW S 大数据博客文章。

此模式中的示例数据基于土耳其字母表，用于演示目的。此模式中的标量 Python UDF 旨在使 Amazon Redshift 的默认查询结果符合土耳其语字符的语言顺序。有关更多信息，请参阅此模式的其他信息部分中的土耳其语示例。您可以用这种模式修改其他语言标量 Python UDF。

先决条件和限制

先决条件

- 带有数据库、架构和表的 Amazon Redshift 集群
- 具有创建表和创建函数权限的 Amazon Redshift 用户
- Python 2.7 或更高版本

限制

此模式中的查询使用的语言排序不区分大小写。

架构

技术堆栈

- Amazon Redshift
- Python UDF

工具

Amazon Web Services

- Amazon Redshift 是一项在 Amazon Web Services Cloud 中托管的 PB 级数据仓库服务。Amazon Redshift 与数据湖集成，让您可以使用数据获得对您的业务和客户的新见解。

其他工具

- Python (UDFs) 用户定义的函数是可以用 Python 编写然后在 SQL 语句中调用的函数。

操作说明

开发代码以按语言顺序对查询结果进行排序

Task	描述	所需技能

为您的示例数据创建一个表。	要在 Amazon Redshift 中创建表并将示例数据插入到该表中，请使用以下 SQL 语句：
	
	`CREATE TABLE my_table (first_name varchar(30));`
	
	`INSERT INTO my_table (first_name) VALUES ('ali'), ('Ali'), ('ırmak'), ('IRMAK'), ('irem'), ('İREM'), ('oğuz'), ('OĞUZ'), ('ömer'), ('ÖMER'), ('sedat'), ('SEDAT'),`
		数据工程师

Task	描述	所需技能
	('şule'),	
	Note 样本数据中的名字包括土耳其语字母表中的特殊字符。有关此示例中土耳其语注意事项的更多信息，请参阅此模式的其他信息部分中的土耳其语示例。	

Task	描述	所需技能

检查样本数据默认排序。	若要在 Amazon Redshift 中查看示例数据的默认排序，请运行以下查询：
	
	`SELECT first_name FROM my_table ORDER BY first_name;`
	
	该查询返回您之前创建表中的名字列表：
	
	`first_name`
	`---------------`
	`Ali`
	`IRMAK`
	`OĞUZ`
	`SEDAT`
	`ali`
	`ırem`
	`oğuz`
	`sedat`
	`ÖMER`
	`ömer`
	`İREM`
	`ırmak`
	`ŞULE`
	`şule`
	
	查询结果的顺序不正确，因为默认二进制 UTF-8 排序不适应土耳其语特殊字符的语言顺序。
		数据工程师

Task	描述	所需技能
创建标量 Python UDF。	若要创建标量 Python UDF，请使用以下 SQL 代码： `CREATE OR REPLACE FUNCTION collate_sort (value varchar) RETURNS varchar IMMUTABLE AS $$ def sort_str(val): import string dictionary = { 'I': 'ı', 'ı': 'h~', 'İ': 'i', 'Ş': 's~', 'ş': 's~', 'Ğ': 'g~', 'ğ': 'g~', 'Ü': 'u~', 'ü': 'u~', 'Ö': 'o~', 'ö': 'o~', 'Ç': 'c~', 'ç': 'c~' } for key, value in dictionary.items(): val = val.replace(key, value) return val.lower() return sort_str(value)`	数据工程师

Task	描述	所需技能
	`$$ LANGUAGE plpythonu;`	

查询示例数据。	若要使用 Python UDF 查询示例数据，请运行以下 SQL 查询：
	`SELECT first_name FROM my_table ORDER BY collate_order(first_name);`
	现在，查询以土耳其语顺序返回示例数据：
	`first_name`
	`---------------`
	`ali`
	`Ali`
	`ırmak`
	`IRMAK`
	`irem`
	`İREM`
	`oğuz`
	`OĞUZ`
	`ömer`
	`Ömer`
	`sedat`
	`SEDAT`
	`şule`
	`ŞULE`
		数据工程师

相关资源

- ORDER BY 子句（Amazon Redshift 文档）
- 创建标量 Python UDF（Amazon Redshift 文档）

其他信息

土耳其语示例

Amazon Redshift 根据二进制 UTF-8 排序顺序（而不是特定于语言的排序顺序）返回查询结果。这意味着，如果您查询包含土耳其语字符的 Amazon Redshift 表，则查询结果不会根据土耳其语的语言顺序进行排序。土耳其语包含六个不出现在拉丁字母表中的特殊字符 (ç、ı、ğ、 ö、ş, 以及 ü)。这些特殊字符放置在基于二进制 UTF-8 排序的排序结果集的末尾，如下表所示。

二进制 UTF-8 排序	土耳其语语言排序
a	a
b	b
c	c
d	ç (*)
e	d
f	e
g	f
h	g
i	ğ (*)
j	h
k	ı (*)
l	i
m	j
n	k
o	l
p	m

r	n
s	o
t	ö (*)
u	p
v	r
y	s
z	ş (*)
ç (*)	t
ğ (*)	u
ı (*)	ü (*)
ö (*)	v
ş (*)	y
ü (*)	z

> **Note**
>
> 星号 (*) 表示土耳其语中的特殊字符。

如上表所示，特殊字符 q 在土耳其语语言排序中介于 c 和 d 之间，但在二进制 UTF-8 排序中在 z 之后。此模式中的标量 Python UDF 使用以下字符替换目录将土耳其语特殊字符替换为相应的拉丁语等效字符。

土耳其语特殊字符	拉丁语等效字符
ç	c~
ı	h~

ğ	g~
ö	o~
ş	s~
ü	u~

> **Note**
>
> 在替换相应土耳其语特殊字符的拉丁字符的末尾附加波浪号 (~) 字符。

修改标量 Python UDF 函数

要从此模式修改标量 Python UDF 函数，以便该函数接受定位参数并支持多事务字典，请使用以下 SQL 代码：

```
CREATE OR REPLACE FUNCTION collate_sort (value varchar, locale varchar)
RETURNS varchar
IMMUTABLE
AS
$$
    def sort_str(val):
       import string
       # Turkish Dictionary
       if locale == 'tr-TR':
            dictionary = {
               'I': 'ı',
               'ı': 'h~',
               'İ': 'i',
               'Ş': 's~',
               'ş': 's~',
               'Ğ': 'g~',
               'ğ': 'g~',
               'Ü': 'u~',
               'ü': 'u~',
               'Ö': 'o~',
               'ö': 'o~',
               'Ç': 'c~',
               'ç': 'c~'
```

```
        }
        # German Dictionary
        if locale == 'de-DE':
            dictionary = {
               ....
               ....
        }

        for key, value in dictionary.items():
            val = val.replace(key, value)

        return val.lower()

    return sort_str(value)

$$ LANGUAGE plpythonu;
```

以下示例代码展示了如何查询修改后的 Python UDF：

```
SELECT first_name FROM my_table ORDER BY collate_order(first_name, 'tr-TR');
```

将 Lambda 函数，以接收来自不同 AWS 区域中的 S3 存储桶的事件通知

由 Suresh Konathala、Andrew Preston 和 Arindom Sarkar 创作

摘要

Amazon Simple Storage Service (Amazon S3) 事件通知会针对您的 S3 存储桶中的某些事件（例如，对象创建事件、对象移除事件或恢复对象事件）发布通知。您可使用 AWS Lambda 函数根据应用程序的要求处理这些通知。但是，Lambda 函数无法直接订阅来自不同 AWS 区域托管的 S3 存储桶的通知。

这种模式的方法通过为每个区域使用 Amazon Simple Notification Service (Amazon SNS)主题，部署扇出场景来处理来自跨区域 S3 存储桶的 Amazon S3 通知。这些区域 SNS 主题将 Amazon S3 事件通知发送到也包含您的 Lambda 函数的中央区域中的 Amazon Simple Queue Service (Amazon SQS) 队列。Lambda 函数订阅此 SQS 队列，并根据您组织的要求处理事件通知。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 多个区域中的现有 S3 存储桶，包括用于托管 Amazon SQS 队列和 Lambda 函数的中央区域。
- 已安装和配置 AWS 命令行界面（AWS CLI）。有关这方面的更多信息，请参阅 AWS CLI 文档中的安装、更新和卸载 AWS CLI。
- 熟悉 Amazon SNS 的扇出场景。有关这方面的更多信息，请参阅 Amazon SNS 文档中的常见 Amazon SNS 场景。

架构

下图显示了此模式方法的架构。

图表显示了以下工作流：

1. Amazon S3 向同一区域的 SNS 主题发送有关 S3 存储桶（例如，已创建对象、删除对象或恢复对象）的事件通知。

2. SNS 主题将事件发布到中央区域的 SQS 队列。

3. SQS 队列配置为 Lambda 函数的事件源，并缓冲 Lambda 函数的事件消息。

4. Lambda 函数轮询 SQS 队列中的消息，并根据您的应用程序的要求处理 Amazon S3 事件通知。

技术堆栈

- Lambda
- Amazon SNS
- Amazon SQS
- Amazon S3

工具

- AWS CLI – AWS 命令行界面（AWS CLI）是一种开源工具，用于通过命令行 Shell 中的命令与 Amazon Web Services 交互。仅需最少的配置，即可使用 AWS CLI 开始运行命令，以便从终端程序中的命令提示符实现与基于浏览器的 AWS 管理控制台所提供的功能等同的功能。
- AWS CloudFormation — AWS CloudFormation 可帮助您建模和设置 AWS 资源，快速一致地配置这些资源，并在资源的整个生命周期中对其进行管理。您可以使用模板来描述资源及其依赖关系，然后将它们作为堆栈一起启动和配置，而不必单独管理资源。您可以跨多个 Amazon Web Services account 和 AWS 区域管理和预置堆栈。
- AWS Lambda – AWS Lambda 是一项计算服务，支持无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。您只需为消耗的计算时间付费 - 代码未运行时不产生费用。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 可协调和管理发布者和客户端之间消息的传送或发送，包括 Web 服务器和电子邮件地址。订阅用户接收所有发布至他们所订阅主题的消息，并且一个主题的所有订阅用户收到的消息都相同。
- Amazon SQS – Amazon Simple Queue Service (Amazon SQS) 提供了一个安全、持久且可用的托管队列，以允许您集成和分离分布式软件系统与组件。Amazon SQS 支持标准队列和 FIFO 队列。

操作说明

在您的中央区域创建 SQS 队列和 Lambda 函数

Task	描述	所需技能
使用 Lambda 触发器创建 SQS 队列。	登录 AWS 管理控制台，按照 AWS Lambda 文档中的将 Lambda 与 Amazon SQS 结合使用教程中的说明在您的中央区域创建以下资源： • Lambda 执行角色 • 用于处理 Amazon S3 事件的 Lambda 函数 • SQS 队列 **Note** 确保将 SQS 队列配置为 Lambda 函数的事件源。	AWS DevOps，云架构师

创建 SNS 主题，并为每个所需区域中的 S3 存储桶设置事件通知

Task	描述	所需技能
创建 SNS 主题以接收 Amazon S3 事件通知。	在您想要接收 Amazon S3 事件通知的区域中创建 SNS 主题。有关这方面的更多信息，请参阅 Amazon SNS 文档中的创建 SNS 主题。	AWS DevOps，云架构师

Task	描述	所需技能
	Important 请务必记录您的 SNS 主题的亚马逊资源名称 (ARN)。	
为中央 SQS 队列订阅 SNS 主题。	为您的中央区域托管的 SQS 队列订阅您的 SNS 主题。有关这方面的更多信息，请参阅 Amazon SNS 文档中的订阅 SNS 主题。	AWS DevOps，云架构师

Task	描述	所需技能
更新 SNS 主题访问策略。	1. 打开 Amazon SNS 控制台上，选择主题，然后选择您之前创建的 SNS 主题。 2. 选择编辑，然后展开访问策略 - 可选部分。 3. 将以下访问策略附加至您的 SNS 主题，以允许对 Amazon S3 使用 `sns:publish` 权限，然后选择保存： `{ "Version": "2012-10-17", "Statement": [ { "Sid": "0", "Effect": "Allow", "Principal": { "Service": "s3.amazonaws.com" }, "Action": "sns:Publish", "Resource": "arn:aws:sns:us-west-2::s3Events-SNSTopic-us-west-2" } ] }`	AWS DevOps，云架构师

Task	描述	所需技能
为区域中的每个 S3 存储桶设置通知。	为区域中的每个 S3 存储桶设置事件通知。有关这方面的更多信息，请参阅 Amazon S3 文档中的使用 Amazon S3 控制台启用和配置事件通知。 **Note** 在目标部分，选择 SNS 主题并指定您之前创建的 SNS 主题的 ARN。	AWS DevOps，云架构师
在所有必需的区域重复此操作说明。	**Important** 针对您希望从中接收 Amazon S3 事件通知的每个区域（包括您的中心区域）重复此长篇故事中的任务。	AWS DevOps，云架构师

相关资源

- 配置访问策略（Amazon SQS 文档）
- 将 SQS 队列配置为事件源（AWS Lambda 文档）
- 配置 SQS 队列以启动 Lambda 函数（Amazon SQS 文档）
- AWS::Lambda::Function 资源（AWS CloudFormation 文档）

三种用于将数据转换为 Apache Parquet 的 AWS Glue ETL 作业类型

由 Adnan Alvee (AWS)、Karthikeyan Ramachandran (AWS) 和 Nith Govindasivan (AWS) 创作

摘要

在 Amazon Web Services（AWS）Cloud 上，AWS Glue 是一项完全托管的提取、转换、加载（ETL）服务。AWS Glue 使您能够经济高效地对数据进行分类、清理和扩充，并在各种数据存储和数据流之间可靠地移动数据。

此模式在 AWS Glue 中提供了不同的作业类型，并使用三种不同的脚本来演示 ETL 作业的创作。

您可以使用 AWS Glue 在 Python Shell 环境中编写 ETL 作业。您还可以在托管 Apache Spark 环境中使用 Python (PySpark) 或 Scala 创建批处理和流式处理 ETL 作业。为了开始创作 ETL 作业，此模式侧重于使用 Python shell、和 Scala 的批处理 ETL 作业。 PySparkPython Shell 作业适用于需要较低计算能力的工作负载。托管 Apache Spark 环境适用于需要高计算能力的工作负载。

Apache Parquet 旨在支持高效的压缩和编码方案。它可以加快分析工作负载的速度，因为它以列式方式存储数据。从长远来看，将数据转换为 Parquet 可以为您节省存储空间、成本和时间。要了解有关 Parquet 的更多信息，请参阅博客文章 Apache Parquet：如何使用开源列式数据格式成为英雄。

先决条件和限制

先决条件

- AWS Identity and Access Management (IAM) 角色（如果您没有角色，请参阅其他信息部分。）

架构

目标技术堆栈

- AWS Glue
- Amazon Simple Storage Service（Amazon S3）
- Apache Parquet

自动化和扩缩

- AWS Glue 工作流程支持 ETL 管线的完全自动化。
- 您可以将数据处理单元的数量 (DPUs) 或工作器类型更改为水平和垂直扩展。

工具

Amazon Web Services

- Amazon Simple Storage Service(Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Glue 是一项完全托管的 ETL 服务，用于在各种数据存储和数据流之间对数据进行分类、清理、扩充和移动。

其他工具

- Apache Parquet 是一种专为存储和检索而设计的开源列式数据文件格式。

配置

使用以下设置来配置 AWS Glue ETL 的计算能力。要降低成本，请在运行此模式中提供的工作负载时使用最低设置。

- Python Shell – 您可以使用 1 个 DPU 来利用 16 GB 的内存，也可以使用 0.0625 DPU 来使用 1 GB 的内存。此模式使用 0.0625 DPU，这是 AWS Glue 控制台中的默认设置。
- Python 或 Scala for Spark – 如果您在控制台中选择与 Spark 相关的作业类型，AWS Glue 默认使用 10 个 Worker 和 G.1X Worker 类型。这种模式使用两个 Worker（这是允许的最小数量），标准 Worker 类型足够且具有成本效益。

下表显示了 Apache Spark 环境的不同 AWS Glue Worker 类型。由于 Python Shell 作业不使用 Apache Spark 环境来运行 Python，因此它未包含在表中。

	Standard	G.1X	G.2X
vCPU	4	4	8
内存	16 GB	16 GB	32 GB
磁盘空间	50 GB	64 GB	128 GB

每个 Worker 的执行程序	2	1	1

代码

有关此模式中使用的代码，包括 IAM 角色和参数配置，请参阅其他信息部分。

操作说明

上传数据

Task	描述	所需技能
将数据上传到新的或现有 S3 存储桶。	在账户中创建 S3 存储桶或使用现有 S3 存储桶。从附件部分上传 sample_data.csv 文件，并记下 S3 存储桶和前缀位置。	常规 AWS

创建并运行 AWS Glue 作业

Task	描述	所需技能
创建 AWS Glue 作业。	在 AWS Glue 控制台的 ETL 部分下方，添加一个 AWS Glue 作业。选择相应的作业类型、AWS Glue 版本以及相应的 DPU/Worker 类型和 Worker 数量。有关详细信息，请参阅配置部分。	开发人员、云或数据
更改输入和输出位置。	复制与 AWS Glue 作业对应的代码，然后更改您在上传数据操作说明中记下的输入和输出位置。	开发人员、云或数据

Task	描述	所需技能
配置参数。	您可以使用其他信息部分中提供的片段为 ETL 作业设置参数。AWS Glue 在内部使用四个参数名称： • `--conf` • `--debug` • `--mode` • `--JOB_NAME` 必须在 AWS Glue 控制台上明确输入该 `--JOB_NAME` 参数。选择作业、编辑作业、安全配置、脚本库和作业参数（可选）。输入 `--JOB_NAME` 作为密钥并提供一个值。您也可以使用 AWS 命令行界面（AWS CLI）或 AWS Glue API 来设置此参数。Spark 使用该 `--JOB_NAME` 参数，在 Python Shell 环境作业中不需要该参数。 必须在每个参数名称之前添加 `--`；否则，代码将无法运行。例如，对于代码片段，位置参数必须由 `--input_loc` 和 `--output_loc` 调用。	开发人员、云或数据
运行 ETL 作业。	运行作业并检查输出。注意与原始文件相比减少了多少空间。	开发人员、云或数据

相关资源

参考

- Apache Spark
- AWS Glue：如何运作
- AWS Glue 定价

教程和视频

- 什么是 AWS Glue？

其他信息

IAM 角色

创建 AWS Glue 作业时，您可以使用具有以下代码片段所示权限的现有 IAM 角色或新角色。

要创建新角色，请使用以下 YAML 代码。

```
# (c) 2022 Amazon Web Services, Inc. or its affiliates. All Rights Reserved. This AWS
 Content is provided subject to the terms of the AWS Customer
# Agreement available at https://aws.amazon.com/agreement/ or other written agreement
 between Customer and Amazon Web Services, Inc.

AWSTemplateFormatVersion: "2010-09-09"

Description: This template will setup IAM role for AWS Glue service.

Resources:
  rGlueRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          - Effect: "Allow"
            Principal:
              Service:
                - "glue.amazonaws.com"
            Action:
```

```
              - "sts:AssumeRole"
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AWSGlueServiceRole
      Policies:
        - PolicyName: !Sub "${AWS::StackName}-s3-limited-read-write-inline-policy"
          PolicyDocument:
            Version: "2012-10-17"
            Statement:
              - Effect: Allow
                Action:
                  - "s3:PutObject"
                  - "s3:GetObject"
                Resource: "arn:aws:s3:::*/*"
      Tags:
        - Key  : "Name"
          Value : !Sub "${AWS::StackName}"

Outputs:
  oGlueRoleName:
    Description: AWS Glue IAM role
    Value:
      Ref: rGlueRole
    Export:
      Name: !Join [ ":", [ !Ref "AWS::StackName", rGlueRole ] ]
```

AWS Glue (Python Shell)

Python 代码使用 Pandas 和 PyArrow 库将数据转换为 Parquet。Pandas 库已经可用。该 PyArrow 库是在你运行模式时下载的，因为这是一次性运行。您可以使用 wheel 文件 PyArrow 转换为库并将该文件作为库包提供。有关打包 Wheel 文件的更多信息，请参阅提供您自己的 Python 库。

AWS Glue Python Shell 参数

```
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["input_loc", "output_loc"])
```

AWS Glue（Python Shell）代码

```
from io import BytesIO
import pandas as pd
import boto3
import os
```

```
import io
import site
from importlib import reload
from setuptools.command import easy_install
install_path = os.environ['GLUE_INSTALLATION']
easy_install.main( ["--install-dir", install_path, "pyarrow"] )
reload(site)
import pyarrow


input_loc = "s3://bucket-name/prefix/sample_data.csv"
output_loc = "s3://bucket-name/prefix/"


input_bucket = input_loc.split('/', 1)[0]
object_key = input_loc.split('/', 1)[1]


output_loc_bucket = output_loc.split('/', 1)[0]
output_loc_prefix = output_loc.split('/', 1)[1]


s3 = boto3.client('s3')
obj = s3.get_object(Bucket=input_bucket, Key=object_key)
df = pd.read_csv(io.BytesIO(obj['Body'].read()))


parquet_buffer = BytesIO()
s3_resource = boto3.resource('s3')
df.to_parquet(parquet_buffer, index=False)
s3_resource.Object(output_loc_bucket, output_loc_prefix +  'data' +
 '.parquet').put(Body=parquet_buffer.getvalue())
```

使用 Python 的 AWS Glue Spark 作业

要在 Python 中使用 AWS Glue Spark 作业类型，请选择 Spark 作为作业类型。选择作业启动时间缩短的 Spark 3.1、Python 3（Glue 版本 3.0）作为 AWS Glue 版本。

AWS Glue Python 参数

```
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME", "input_loc", "output_loc"])
```

使用 Python 代码的 AWS Glue Spark 作业

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.transforms import *
from awsglue.dynamicframe import DynamicFrame
from awsglue.utils import getResolvedOptions
from awsglue.job import Job


sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)

input_loc = "s3://bucket-name/prefix/sample_data.csv"
output_loc = "s3://bucket-name/prefix/"

inputDyf = glueContext.create_dynamic_frame_from_options(\
    connection_type = "s3", \
    connection_options = {
        "paths": [input_loc]}, \
    format = "csv",
    format_options={
        "withHeader": True,
        "separator": ","
    })


outputDF = glueContext.write_dynamic_frame.from_options(\
    frame = inputDyf, \
    connection_type = "s3", \
    connection_options = {"path": output_loc \
        }, format = "parquet")
```

对于大量压缩的大文件（例如，1,000 个文件，每个文件约为 3 MB），请使用带有 `recurse` 参数的 `compressionType` 参数读取前缀内的所有可用文件，如以下代码所示。

```
input_loc = "bucket-name/prefix/"
output_loc = "bucket-name/prefix/"

inputDyf = glueContext.create_dynamic_frame_from_options(
```

```
                    connection_type = "s3",
                    connection_options = {"paths": [input_loc],
                                          "compressionType":"gzip","recurse" :"True",
                                          },
                    format = "csv",
                    format_options={"withHeader": True,"separator": ","}
                    )
```

对于大量压缩的小文件（例如，1,000 个文件，每个文件大小约为 133 KB），请使用 `groupFiles` 参数以及 `compressionType` 和 `recurse` 参数。`groupFiles` 参数将小文件分组为多个大文件，`groupSize` 参数将分组控制为以字节为单位的指定大小（例如，1 MB）。以下代码片段提供了在代码中使用这些参数的示例。

```
input_loc = "bucket-name/prefix/"
output_loc = "bucket-name/prefix/"

inputDyf = glueContext.create_dynamic_frame_from_options(
                    connection_type = "s3",
                    connection_options = {"paths": [input_loc],
                                          "compressionType":"gzip","recurse" :"True",
                                           "groupFiles" :"inPartition",
 "groupSize" :"1048576",
                                          },
                    format = "csv",
                    format_options={"withHeader": True,"separator": ","}
                    )
```

无需对 Worker 节点进行任何更改，这些设置就使 AWS Glue 作业可以读取多个文件（大或小，有或没有压缩），然后以 Parquet 格式将它们写入目标。

使用 Scala 的 AWS Glue Spark 作业

要在 Scala 中使用 AWS Glue Spark 作业类型，请选择 Spark 作为作业类型，选择语言作为 Scala。选择作业启动时间缩短的 Spark 3.1、Scala 2（Glue 版本 3.0）作为 AWS Glue 版本。为了节省存储空间，以下 Scala 示例中的 AWS Glue 还使用该 `applyMapping` 功能来转换数据类型。

AWS Glue Scala 参数

```
import com.amazonaws.services.glue.util.GlueArgParser val args =
 GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME", "inputLoc",
 "outputLoc").toArray)
```

使用 Scala 代码的 AWS Glue Spark 作业

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.DynamicFrame
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._


object GlueScalaApp {
  def main(sysArgs: Array[String]) {

    @transient val spark: SparkContext = SparkContext.getOrCreate()
    val glueContext: GlueContext = new GlueContext(spark)

    val inputLoc = "s3://bucket-name/prefix/sample_data.csv"
    val outputLoc = "s3://bucket-name/prefix/"

    val readCSV = glueContext.getSource("csv", JsonOptions(Map("paths" ->
 Set(inputLoc)))).getDynamicFrame()

    val applyMapping = readCSV.applyMapping(mappings = Seq(("_c0", "string", "date",
 "string"), ("_c1", "string", "sales", "long"),
    ("_c2", "string", "profit", "double")), caseSensitive = false)

    val formatPartition = applyMapping.toDF().coalesce(1)

    val dynamicFrame = DynamicFrame(formatPartition, glueContext)

    val dataSink = glueContext.getSinkWithFormat(
        connectionType = "s3",
        options = JsonOptions(Map("path" -> outputLoc )),
        transformationContext = "dataSink", format =
 "parquet").writeDynamicFrame(dynamicFrame)
  }
}
```

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用亚马逊 Athena 和亚马逊可视化 Amazon Redshift 审计日志 QuickSight

由 Sanket Sirsikar (AWS) 和 Gopal Krishna Bhatia (AWS) 创建

摘要

安全性是 Amazon Web Services（AWS）云上数据库操作不可或缺的一部分。您的组织应确保监视数据库用户活动和连接，以检测潜在的安全事件和风险。此模式有助于您监控数据库以确保安全并进行故障排除，该流程通常称为数据库审计。

此模式提供了一个 SQL 脚本，可以自动创建 Amazon Athena 表和亚马逊报告控制面板的视图，从而帮助您审计 A QuickSight mazon Redshift 日志。这可确保负责监控数据库活动的用户能够方便地访问数据安全功能。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 现有 Amazon Redshift 集群。有关更多信息，请参阅 Amazon Redshift 文档中的创建 Amazon Redshift 集群。
- 访问现有 Athena 工作组。有关更多信息，请参阅 Amazon Athena 文档中的工作组的工作原理。
- 具有所需 AWS Identity and Access Management（IAM）权限的现有 Amazon Simple Storage Service（Amazon S3）源存储桶。有关更多信息，请参阅 Amazon Redshift 文档中的数据库审计日志记录中的 Amazon Redshift 审计日志记录的存储桶权限。

架构

技术堆栈

- Athena
- Amazon Redshift
- Amazon S3
- QuickSight

工具

- Amazon Athena – Athena 是一种交互式查询服务，方便使用标准 SQL 分析 Amazon S3 的数据。
- Amazon QuickSight — QuickSight 是一项可扩展、无服务器、可嵌入、由机器学习提供支持的商业智能 (BI) 服务。
- Amazon Redshift – Amazon Redshift 是一种完全托管的企业 PB 级数据仓库服务。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项面向互联网的存储服务。

操作说明

配置 Amazon Redshift 集群

Task	描述	所需技能
为 Amazon Redshift 集群启用审计日志记录。	1. 登录 AWS 管理控制台，打开 Amazon Redshift 控制台，选择集群，然后选择要为其启用日志记录的集群。 2. 选择属性选项卡，然后按照 Amazon Redshift 文档中的使用控制台配置审核中的说明启用审核。	数据库管理员、数据工程师
在 Amazon Redshift 集群参数组中启用日志记录。	您可以使用 AWS 管理控制台、Amazon Redshift API 参考或 AWS 命令行界面（AWS CLI）同时启用对连接日志、用户日志和用户活动日志的审计。 要审计用户活动日志，您还必须启用 `enable_user_activity_logging` 数据库参数。如果您仅启用审计日志记录功能，但不启用相关参数，则数据库审计日	数据库管理员、数据工程师

Task	描述	所需技能
	志将仅为连接日志和用户日志记录信息，而不为用户活动日志记录信息。默认情况下不启用 `enable_user_activity_logging` 参数，但可以通过将其从 `false` 更改为 `true` 来启用它。 **Important** 您需要在启用参数的情况下创建一个新的集群参数组，并将其附加到您的 Amazon Red `user_activity_logging` shift 集群。有关更多信息，请参阅 Amazon Redshift 文档中的修改集群。 有关此任务的更多信息，请参阅 Amazon Redshift 文档中的 Amazon Redshift 参数组和使用控制台配置审计。	

Task	描述	所需技能
为 Amazon Redshift 集群日志记录配置 S3 存储桶权限。	当您启用日志记录时，Amazon Redshift 会收集日志记录信息并将其上载到 S3 存储桶中存储的日志文件。您可以使用现有的 S3 存储桶或创建新存储桶。 **Important** 确保 Amazon Redshift 拥有访问 S3 存储桶所需的 IAM 权限。有关此内容的更多信息，请参阅 Amazon Redshift 文档中的数据库审计日志记录中的 Amazon Redshift 审计日志记录的存储桶权限。	数据库管理员、数据工程师

创建 Athena 表和视图

Task	描述	所需技能
创建 Athena 表和视图以从 S3 存储桶查询 Amazon Redshift 审计日志数据。	打开 Amazon Athena 控制台，并使用 `AuditLogging.sql` SQL 脚本（附加）中的数据定义语言（DDL）查询为用户活动日志、用户日志和连接日志创建表和视图。 有关更多信息和说明，请参阅 Amazon Athena 研讨会中的创建表和运行查询教程。	数据工程师

在 QuickSight 控制面板中设置日志监控

Task	描述	所需技能
使用 Athena 作为数据源创建 QuickSight 控制面板。	打开亚马逊 QuickSight 控制台，按照亚马逊 Athena 研讨会的 “ QuickSight 使用 Athena 进行可视化” 教程中的说明创建控制 QuickSight 面板。	数据库管理员、数据工程师

相关资源

- 在 Athena 中创建表并运行查询
- QuickSight 使用 Athena 进行可视化

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 Amazon 可视化所有 AWS 账户的 IAM 凭证报告 QuickSight

由 Parag Nagwekar (AWS) 和 Arun Chandapillai (AWS) 编写

摘要

> **Warning**
>
> IAM 用户具有长期凭证，这会带来安全风险。为帮助减轻这种风险，我们建议仅向这些用户提供执行任务所需的权限，并在不再需要这些用户时将其移除。

您可以使用 AWS Identity and Access Management (IAM) 凭证报告帮助您满足组织的安全性、审计和合规性要求。凭证报告提供您 Amazon Web Services account 中所有用户的列表，并显示他们的凭证状态，如密码、访问密钥和多重身份验证（MFA）设备。您可以为由 AWS Organizations 管理的多个 Amazon Web Services account 使用凭证报告。

此模式包括步骤和代码，可帮助您使用 Amazon QuickSight 控制面板为组织中的所有 AWS 账户创建和共享 IAM 证书报告。您可以与组织中的利益相关者共享控制面板。这些报告可帮助您的组织实现以下目标业务成果：

- 识别与 IAM 用户相关的安全事件
- 追踪 IAM 用户向单点登录（SSO）身份验证的实时迁移
- 追踪 IAM 用户访问的 AWS 区域
- 保持合规
- 与其他利益相关者共享信息

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 具有成员账户的组织。
- 有权访问 Organizations 中账户的 IAM 角色
- 已安装和配置 AWS 命令行界面（AWS CLI）版本 2。

- 订阅 Amazon QuickSight 企业版

架构

技术堆栈

- Amazon Athena
- Amazon EventBridge
- Amazon QuickSight
- Amazon Simple Storage Service（Amazon S3）
- AWS Glue
- AWS Identity and Access Management（IAM）
- AWS Lambda
- AWS Organizations

目标架构

下图显示了用于设置从多个 Amazon Web Services account 捕获 IAM 凭证报告数据的工作流架构。

1. EventBridge 每天调用 Lambda 函数。
2. Lambda 函数在整个组织的每个 Amazon Web Services account 中扮演 IAM 角色。然后，该函数创建 IAM 凭证报告，并将报告数据存储在集中式 S3 存储桶中。必须在 S3 存储桶上启用加密，并停用公共访问。
3. AWS Glue 爬网程序每天都会爬取 S3 存储桶，并相应地更新 Athena 表。
4. QuickSight 导入和分析证书报告中的数据，并构建一个可由利益相关者可视化并与其共享的仪表板。

工具

Amazon Web Services

- Amazon Athena 是一种交互式查询服务，方便使用标准 SQL 分析 Amazon S3 的数据。

- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如，Lambda 函数、使用 API 目标的 HTTP 调用端点或其他 Amazon Web Services account 中的事件总线。
- Amazon QuickSight 是一项云规模的商业智能 (BI) 服务，可帮助您在单个控制面板中可视化、分析和报告数据。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。

代码

此模式的代码可在 GitHub getiamcredsreport-allaccounts-org存储库中找到。您可以使用此存储库中的代码在 Organizations 的 Amazon Web Services account 中创建 IAM 凭证报告，并将其存储至中心位置。

操作说明

设置基础设施

Task	描述	所需技能
设置亚马逊 QuickSight 企业版。	1. 在您的 AWS 账户中激活亚马逊 QuickSight 企业版。有关更多信息，请参阅 QuickSight 文档 QuickSight 中的在 Amazon 内部管理用户访问权限。 2. 要授予控制面板权限，请获取用户的亚马逊资源名称 (ARN)。 QuickSight	AWS 管理员、AWS DevOps、云管理员、云架构师
将亚马逊 QuickSight 与亚马逊 S3 和 Athena 集成。	在部署 AWS 堆栈之前，您必须授权QuickSight 才能使用 Amazon S3 和 Athena。 CloudFormation	AWS 管理员、AWS DevOps、云管理员、云架构师

部署基础设施

Task	描述	所需技能
克隆 GitHub 存储库。	1. 通过运行以下命令将 GitHub getiamcredsreport-allaccounts-org存储库克隆到本地计算机：`git clone https://github.com/aws-samples/getiamcredsreport-allaccounts-org`	AWS 管理员
部署基础设施。	1. 登录 AWS 管理控制台并打开CloudFormation 控制台。 2. 在导航窗格中，选择创建堆栈，然后选择使用新资源（标准）。 3. 在标识资源页面上，选择下一步。 4. 在指定模板页面，对于模板来源，选择上传模板文件。 5. 选择 “选择文件”，从克隆的 GitHub 存储库中选择`Cloudformation-createcredrepo.yaml` 文件，然后选择 “下一步”。 6. 在参数中，使用您的 IAM 角色更新 `IAMRoleName` 。这应该就是您希望 Lambda 在组织的每个账户中代入的 IAM 角色。此角色创建凭证报告。注意：在创建堆栈时，该角色不必出现在所有账户中。	AWS 管理员

Task	描述	所需技能
	7. 在参数中，将 S3BucketName 更新为 Lambda 存储所有账户凭证的 S3 存储桶名称。 8. 在堆栈名称中，输入堆栈名称。 9. 选择提交。 10.记下 Lambda 函数角色名称。	
创建 IAM 权限策略。	使用以下权限，为组织中的每个 Amazon Web Services account 创建 IAM policy： `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "iam:GenerateCredentialReport", "iam:GetCredentialReport" ], "Resource": "*" } ] }`	AWS DevOps、云管理员、云架构师、数据工程师

Task	描述	所需技能
创建带有信任策略的 IAM 角色。	1. 为 Amazon Web Services account 创建 IAM 角色，并附加在上一步中创建的权限策略。 2. 将以下信任策略附加到 IAM 角色： `{ "Version":"2012-10-17", "Statement":[ { "Effect": "Allow", "Principal":{ "AWS":[ "arn:aws:iam::<MasterAccountID>:role/<LambdaRole>" ] }, "Action": "sts:AssumeRole" } ] }` **Important** `arn:aws:iam::<MasterAccountID>:role/<LambdaRole>` 替换为您之前记下	云管理员、云架构师、AWS 管理员

Task	描述	所需技能
	的 Lambda 角色的 ARN。 **Note** 组织通常使用自动化为其 AWS 账户创建 IAM 角色。建议您使用此自动化技术（如有）。或者，您可以使用代码存储库中的 `CreateRoleforOrg.py` 脚本。该脚本需要现有的管理角色，或任何其他有权在每个 Amazon Web Services account 中创建 IAM policy 和角色的 IAM 角色。	
将 Amazon 配置 QuickSight 为可视化数据。	1. 使用您的 QuickSight凭据登录。 2. 使用 Athena 创建数据集（使用 `iamcredreportdb` 数据库和 “`cfn_iamcredreport`” 表），然后自动刷新数据集。 3. 在中创建分析 QuickSight。 4. 创建 QuickSight 仪表板。	AWS DevOps、云管理员、云架构师、数据工程师

其他信息

其他注意事项

请考虑以下事项：

- 使用 CloudFormation 部署基础设施后，您可以等待在 Amazon S3 中创建并由 Athena 分析的报告，直到 Lambda 和 AWS Glue 按计划运行。或者，您可手动运行 Lambda 在 Amazon S3 中获取报告，然后运行 AWS Glue 爬网程序，以获取根据数据创建的 Athena 表。
- QuickSight 是一款根据您的业务需求分析和可视化数据的强大工具。您可以使用中的参数根据您选择的数据字段 QuickSight 来控制小组件数据。此外，您还可以使用 QuickSight 分析从数据集中创建参数（例如，账户、日期和用户字段，`user`分别为`partition_0`、`partition_1`和），以添加帐户、日期和用户的参数控件。
- 要创建自己的 QuickSight 控制面板，请参阅 AWS Wor QuickSight k shop Studio 网站上的研讨会。
- 要查看示例 QuickSight 仪表板，请参阅 GitHub getiamcredsreport-allaccounts-org代码存储库。

目标业务成果

您可以使用此模式来实现以下目标业务成果：

- 识别与 IAM 用户相关的安全事件 — 使用单一控制面板调查组织每个 Amazon Web Services account 的每个用户。您可跟踪 IAM 用户最近访问的各个 AWS 区域及其使用的服务的趋势。
- 跟踪 IAM 用户向 SSO 身份验证的实时迁移 — 通过使用 SSO，用户可使用单个凭证登录一次并访问多个 Amazon Web Services account 和应用程序。如果您计划将 IAM 用户迁移至 SSO，此模式可以帮助您过渡到 SSO，并追踪所有 Amazon Web Services account 中的所有 IAM 用户凭证使用情况(例如访问 AWS 管理控制台或访问密钥的使用情况)。
- 追踪 IAM 用户访问的 AWS 区域 — 您可出于各种目的控制 IAM 用户对区域的访问权限，例如数据主权和成本控制。您还可追踪任何 IAM 用户对区域的使用情况。
- 保持合规 — 遵循最低权限原则，您只能授予执行特定任务所需 IAM 权限。此外，您还可以跟踪对 Amazon Web Services 的访问权限、AWS 管理控制台以及长期凭证的使用情况。
- 与其他利益相关者共享信息 — 您可以与其他利益相关者共享精心策划的控制面板，无需授予他们访问 IAM 凭证报告或 Amazon Web Services account 的权限。

更多模式

- 使用亚马逊 Cognito 和 AWS Amplify 用户界面对现有 React 应用程序用户进行身份验证
- 使用 Amazon Textract 从 PDF 文件中自动提取内容
- 在 Amazon A SageMaker I Studio Lab 中使用 Deepar 计算时间序列构建冷启动预测模型
- 使用 AWS Cost Explorer 成本管理服务为 Amazon EMR 集群创建详细的成本和使用情况报告
- 为 Amazon RDS 和 Amazon Aurora 创建详细的成本和使用情况报告
- 使用 AWS Cost Explorer 成本管理服务为 AWS Glue 作业创建详细的成本和使用情况报告
- 使用 AWS CodePipeline CI/CD 管道部署 AWS Glue 作业
- 在本地 Angular 应用程序中嵌入亚马逊 QuickSight 控制面板
- 确保 Amazon Redshift 集群在创建时已加密
- 确保在发布时启用 Amazon EMR 静态数据加密
- 估算 Amazon DynamoDB 表的存储成本
- 使用 Terraform 执行 Amazon Redshift SQL 查询
- 提取和查询 AWS IoT SiteWise 数据湖中的元数据属性
- 使用 AWS 大型机现代化和 Amazon Q 生成数据见解 QuickSight
- 使用和 AWS Mainframe Modernization Amazon Q 生成 Db2 z/OS 数据见解 QuickSight
- 为 SageMaker 笔记本实例提供对另一个 AWS 账户中 CodeCommit 存储库的临时访问权限
- 在 Amazon Data Firehose 资源未使用密钥加密时进行识别并发出警报 AWS KMS
- 将 psycopg2 库导入到以与你的 PostgreS AWS Lambda QL 数据库进行交互
- 在 Microsoft Sentinel 中提取和分析 AWS 安全日志
- 使用 AWS DMS 在 SSL 模式下将 Amazon RDS for Oracle 迁移到 Amazon RDS for PostgreSQL
- 使用 Oracle GoldenGate 平面文件适配器将 Oracle 数据库迁移到 Amazon RDS for Oracle
- 使用 AWS DMS 和 AWS SCT 将 Oracle 数据库迁移至 Amazon Redshift
- 使用 PrivateLink 适用于 Amazon S3 的 DistCp AWS 将数据从本地 Hadoop 环境迁移到 Amazon S3
- 从 Couchbase Server 迁移至 AWS 上的 Couchbase Capella
- 将本地 Cloudera 工作负载迁移到 Cloudera Data Platform on AWS
- 在启动时监控 Amazon EMR 集群的传输中加密
- 使用应用程序恢复控制器管理 EMR 集群的多可用区故障转移
- 使用 IaC 原则自动部署 Amazon Aurora 全球数据库
- 使用 GitHub 操作基于 AWS CloudFormation 模板配置 AWS Service Catalog 产品

- AWS Glue 使用 pytest 框架对 Python ETL 作业运行单元测试
- 为 AWS 设置一个 Grafana 监控控制面板 ParallelCluster
- 跨账户将数据从 Amazon Redshift 集群卸载到亚马逊 S3
- 验证新的 Amazon Redshift 集群是否有所需 SSL 端点
- 验证新 Amazon Redshift 集群是否在 VPC 中启动
- 使用 Flask 和 AWS Elastic Beanstalk 查看人工智能/机器学习 (AI/ML) 模型结果

计算

主题

- 容器和微服务
- 无服务器
- 网络连接
- 内容分发

容器和微服务

主题

- 从亚马逊 EKS 容器访问亚马逊 Neptune 数据库
- 使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon ECS 上私密访问容器应用程序
- 使用 AWS Fargate PrivateLink、AWS 和网络负载均衡器在 Amazon ECS 上私下访问容器应用程序
- 使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon EKS 上私密访问容器应用程序
- 在 Amazon EKS 上使用 AWS 私有 CA 在 AWS App Mesh 中激活 mTLS
- 使用 AWS Batch 自动备份 Amazon RDS for PostgreSQL 数据库实例
- 使用 CI/CD 管道在 Amazon EKS 中自动部署 Node Termination Handler
- 使用 CI/CD 管道自动构建 Java 应用程序并将其部署到 Amazon EKS
- 使用 Amazon EFS 创建 Amazon ECS 任务定义并在 EC2 实例上挂载文件系统
- 使用容器映像部署 Lambda 函数
- 使用 AWS Fargate 在 Amazon ECS 上部署 Java 微服务
- 使用 Amazon EKS 和 Amazon S3 中的 Helm 图表存储库部署 Kubernetes 资源和软件包
- 在 Amazon EKS 上部署示例 Java 微服务并使用应用程序负载均衡器公开该微服务
- 使用 AWS Copilot 将集群应用程序部署至 Amazon ECS
- 在 Amazon EKS 集群上部署基于 gRPC 的应用程序并使用应用程序负载均衡器访问它
- 部署和调试 Amazon EKS 集群
- 使用 Elastic Beanstalk 部署容器
- 使用 Lambda 函数、Amazon VPC 和无服务器架构生成静态出站 IP 地址
- 迁移到 Amazon ECR 存储库时自动识别重复的容器映像
- 使用 Kubernetes 在亚马逊 EKS 工作节点上安装 SSM 代理 DaemonSet
- 使用在 Amazon EKS 工作节点上安装 SSM CloudWatch 代理和代理 preBootstrapCommands
- 启用 Amazon EKS 自动模式时迁移 NGINX 入口控制器
- 将你的容器工作负载从 Azure 红帽 OpenShift (ARO) 迁移到 AWS 云端 Red Hat OpenShift 服务 (ROSA)
- 优化 AWS App2Container 生成的 Docker 映像
- 使用节点关联性、污点和容忍度将 Kubernetes 容器组（pod）置于 Amazon EKS 上
- 跨账户或区域复制已筛选的 Amazon ECR 容器映像
- 在不重启容器的情况下轮换数据库凭证

- 使用 Amazon ECS Anywhere 在亚马逊 WorkSpaces 上运行亚马逊 ECS 任务
- 在亚马 EC2 逊 Linux 实例上运行 ASP.NET Core Web API Docker 容器
- 使用 AWS Fargate 大规模运行消息驱动型工作负载
- 使用带 AWS Fargate 的 Amazon EFS on Amazon EKS，运行带持久数据存储的有状态工作负载
- 使用 Amazon EKS Pod Identity 和 KEDA 在 Amazon EKS 中设置事件驱动的自动缩放
- 使用 PGO 简化 PostgreSQL 在 Amazon EKS 上的部署
- 使用应用程序负载均衡器在 Amazon ECS 中使用双向 TLS 简化应用程序身份验证
- 更多模式

从亚马逊 EKS 容器访问亚马逊 Neptune 数据库

由 Ramakrishnan Palaninathan (AWS) 创作

摘要

这种模式在完全托管的图形数据库 Amazon Neptune 和容器编排服务 Amazon Elastic Kubernetes Service（Amazon EKS）之间建立了连接，用于访问海王星数据库。Neptune 数据库集群仅限于上的虚拟私有云 (VPC) 中。 AWS因此，访问 Neptune 需要仔细配置 VPC 以启用连接。

与适用于 PostgreSQL 的亚马逊关系数据库服务（Amazon RDS）不同，Neptune 不依赖典型的数据库访问证书。相反，它使用 AWS Identity and Access Management (IAM) 角色进行身份验证。因此，从 Amazon EKS 连接到 Neptune 需要设置一个具有访问 Neptune 所需权限的 IAM 角色。

此外，Neptune 终端节点只能在集群所在的 VPC 内访问。这意味着您必须配置网络设置以促进亚马逊 EKS 和 Neptune 之间的通信。根据您的具体要求和网络偏好，有多种方法可以配置 VPC，以实现 Neptune 和 Amazon EKS 之间的无缝连接。每种方法都有不同的优势和注意事项，这为设计数据库架构提供了灵活性，以满足应用程序的需求。

先决条件和限制

先决条件

- 安装最新版本的 kubectl（参见说明书）。要检查您的版本，请运行：

```
kubectl version --short
```

- 安装最新版本的 eksctl（参见说明书）。要检查您的版本，请运行：

```
eksctl info
```

- 安装最新版本的 AWS Command Line Interface (AWS CLI) 版本 2（参见说明）。要检查您的版本，请运行：

```
aws --version
```

- 创建 Neptune 数据库集群（参见说明）。确保通过 VPC 对等或其他方法在集群的 VPC 和 Amazon E KS 之间建立通信。AWS Transit Gateway此外，请确保集群的状态为 “可用”，并且在端口 8182 上有针对安全组的入站规则。
- 在现有的 Amazon EKS 集群上配置 IAM OpenID Connect (OIDC) 提供商（参见说明）。

产品版本

- 亚马逊 EKS 1.27
- 亚马逊 Neptune 引擎版本 1.3.0.0 (2023-11-15)

架构

下图显示了 Amazon EKS 集群中的 Kubernetes Pod 与 Neptune 之间的连接，以提供对 Neptune 数据库的访问权限。

自动化和扩缩

您可以使用 Amazon EKS 水平吊舱自动扩缩器来扩展此解决方案。

工具

服务

- 亚马逊 Elastic Kubernetes Service（亚马逊 EKS）可帮助你在上面运行 AWS Kubernetes，而无需安装或维护自己的 Kubernetes 控制平面或节点。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- Amazon Neptune 是一项图形数据库服务，可帮助您构建和运行可处理高度互联数据集的应用程序。

最佳实践

有关最佳实践，请参阅 Amazon EKS 最佳实践指南中的 Identity and A ccess 管理。

操作说明

设置环境变量

Task	描述	所需技能
验证集群上下文。	在使用 Helm 或其他命令行工具与 Amazon EKS 集群进行交互之前，必须定义用于封装集	AWS 管理员、云管理员

Task	描述	所需技能
	群详细信息的环境变量。这些变量将在后续命令中使用，以确保它们针对正确的群集和资源。 首先，确认您在正确的群集环境中运行。这样可以确保将任何后续命令发送到预期的 Kubernetes 集群。要验证当前上下文，请运行以下命令。 `kubectl config current-context`	
定义CLUSTER_NAME 变量。	为您的 Amazon EKS 集群定义CLUSTER_NAME 环境变量。在以下命令中，将示例值us-west-2 替换 AWS 区域 为适用于您的集群的正确值。将示例值eks-workshop 替换为您现有的集群名称。 `export CLUSTER_NAME=$(aws eks describe-cluster --region us-west-2 --name eks-workshop --query "cluster.name" --output text)`	AWS 管理员、云管理员

Task	描述	所需技能
验证输出。	要验证变量设置是否正确，请运行以下命令。 `echo $CLUSTER_NAME` 验证此命令的输出是否与您在上一步中指定的输入相匹配。	AWS 管理员、云管理员

创建 IAM 角色并将其与 Kubernetes 关联

Task	描述	所需技能
创建 服务账户。	您可以为服务账户使用 IAM 角色将您的 Kubernetes 服务账户映射到 IAM 角色，从而为在 Amazon EKS 上运行的应用程序启用精细的权限管理。您可以使用 eksctl 创建 IAM 角色并将其与您的 Amazon EKS 集群中的特定 Kubernetes 服务账户相关联。 AWS 托管策略`NeptuneFullAccess` 允许对您指定的 Neptune 集群进行写入和读取访问。 **Important** 在运行这些命令之前，您必须拥有与集群关联的 OIDC 终端节点。	AWS 管理员、云管理员

Task	描述	所需技能

	创建要与名为的 AWS 托管策略关联的服务帐号NeptuneFullAccess 。
	
	`eksctl create iamserviceaccount --name eks-neptune-sa --namespace default --cluster $CLUSTER_NAME --attach-policy-arn arn:aws:iam::aws:policy/NeptuneFullAccess --approve --override-existing-serviceaccounts`
	
	其中eks-neptune-sa 是您要创建的服务帐号的名称。
	
	完成后，此命令将显示以下响应：
	
	`2024-02-07 01:12:39 [#] created serviceaccount "default/eks-neptune-sa"`	

Task	描述	所需技能

验证账户设置是否正确。	确保在集群的默认命名空间中正确设置了eks-neptune-sa服务帐户。
	
	`kubectl get sa eks-neptune-sa -o yaml`
	
	输出应该如下所示：
	
	`apiVersion: v1`
	`kind: ServiceAccount`
	`metadata:`
	`  annotations:`
	`    eks.amazonaws.com/role-arn: arn:aws:iam::123456789123:role/eksctl-eks-workshop-addon-iamserviceaccount-d-Role1-Q35yKgdQOlmM`
	`  creationTimestamp: "2024-02-07T01:12:39Z"`
	`  labels:`
	`    app.kubernetes.io/managed-by: eksctl`
	`  name: eks-neptune-sa`
	`  namespace: default`
	`  resourceVersion: "5174750"`
	`  uid: cd6ba2f7-a0f5-40e1-a6f4-4081e0042316`
		AWS 管理员、云管理员

Task	描述	所需技能

检查连通性。	部署一个名为 Neptune 的示例 pod pod-util 并检查与 Neptune 的连接。
	
	`apiVersion: v1`
	`kind: Pod`
	`metadata:`
	`  name: pod-util`
	`  namespace: default`
	`spec:`
	`  serviceAccountName: eks-neptune-sa`
	`  containers:`
	`  - name: pod-util`
	`  image: public.ecr.aws/patrickc/troubleshoot-util`
	`  command:`
	`  - sleep`
	`  - "3600"`
	`  imagePullPolicy: IfNotPresent`
	
	`kubectl apply -f pod-util.yaml`
	
	`kubectl exec --stdin --tty pod-util -- /bin/bash`
	`bash-5.1# curl -X POST -d '{"gremlin":"g.V().limit(1)"}' https://db-neptune-1.cluster-xxxxxxxxxxxx.us-west-2.neptune.amazonaws.com:8182/gremlin`
	`{"requestId":"a4964f2d-12b1-4ed3-8a14-e`
		AWS 管理员、云管理员

Task	描述	所需技能

	<pre>ff511431a0e","stat
	us":{"message":"",
	"code":200,"attrib
	utes":{"@type":"g:
	Map","@value":[]}}
	,"result":{"data":
	{"@type":"g:List",
	"@value":[]},"meta
	":{"@type":"g:Map"
	,"@value":[]}}}
	bash-5.1# exit
	exit</pre>	

验证连接活动

Task	描述	所需技能
启用 IAM 数据库身份验证。	默认情况下，创建 Neptune 数据库集群时禁用 IAM 数据库身份验证。您可以使用启用或禁用 IAM 数据库身份验证 AWS Management Console。 按照 AWS 文档中的步骤在 Nept une 中启用 IAM 数据库身份验证。	AWS 管理员、云管理员
验证连接。	在此步骤中，您将与已处于运行状态的`pod-util`容器进行交互，以安装 awscurl 并验证连接。 1. 运行以下命令查找 pod。 <pre>kubectl get pods</pre> 输出应该如下所示：	AWS 管理员、云管理员

Task	描述	所需技能
	``` NAME READY STATUS RESTARTS AGE pod-util 1/1 Running 0 50m ``` 2. 运行以下命令安装 aws curl。 ``` kubectl exec --stdin --tty pod-util -- / bin/bash bash-5.1# pip3 install awscurl Installing collected packages: idna, configparser, configargparse, charset-normalizer , certifi, requests, awscurl Successfully installed awscurl-0 .32 certifi-2024.2.2 charset-normalizer -3.3.2 configarg parse-1.7 configpar ser-6.0.0 idna-3.6 requests-2.31.0 bash-5.1# awscurl https://db-neptune -1.cluster-xxxxxxx xxxxx.us-west-2.ne ptune.amazonaws.co m:8182/status -- region us-west-2 -- service neptune-db {"status":"healthy ","startTime":"Thu Feb 08 01:22:14 UTC 2024","dbEngineVer sion":"1.3.0.0.R1" ,"role":"writer"," ```	

Task	描述	所需技能
	dfeQueryEngine":"viaQueryHint","gremlin":{"version":"tinkerpop-3.6.4"},"sparql":{"version":"sparql-1.1"},"opencypher":{"version":"Neptune-9.0.20190305-1.0"},"labMode":{"ObjectIndex":"disabled","ReadWriteConflictDetection":"enabled"},"features":{"SlowQueryLogs":"disabled","ResultCache":{"status":"disabled"},"IAMAuthentication":"enabled","Streams":"disabled","AuditLog":"disabled"},"settings":{"clusterQueryTimeoutInMs":"120000","SlowQueryLogsThreshold":"5000"}}	

故障排除

事务	解决方案
无法访问 Neptune 数据库。	查看附加到服务账号的 IAM 政策。确保它允许对要运行的操作执行必要的操作（例如`neptune:Connec`,`neptune:DescribeDBInstances`）。

相关资源

- AWS 使用 Kubernetes 服务账户授予 Kubernetes 工作负载访问权限（亚马逊 EKS 文档）
- 服务账户的 IAM 角色（亚马逊 EKS 文档）
- 创建新的 Neptune 数据库集群（亚马逊 Neptune 文档）

使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon ECS 上私密访问容器应用程序

由 Kirankumar Chandrashekar (AWS) 创建

摘要

此模式描述了如何在网络负载均衡器后面的亚马逊弹性容器服务 (Amazon ECS) 上私下托管 Docker 容器应用程序，以及如何使用 AWS 访问该应用程序。 PrivateLink然后，您便可以使用专用网络安全地访问 Amazon Web Services (AWS) Cloud 上的服务。Amazon Relational Database Service (Amazon RDS) 为在具有高可用性 (HA) 的 Amazon ECS 上运行的应用程序关系数据库提供托管。如果应用程序需要持久性存储，请使用 Amazon Elastic File System (Amazon EFS)。

运行 Docker 应用程序的 Amazon ECS 服务在前端装有 Network Load Balancer，可以与虚拟私有云 (VPC) 终端节点相关联，以便通过 AWS PrivateLink 进行访问。然后，可以使用 VPC 终端节点与其他 VPCs 人共享此 VPC 终端节点服务。

你也可以使用 AWS Fargate 代替 Amazon A EC2 uto Scaling 群组。有关更多信息，请参阅使用 AWS Fargate、AWS 和网络负载均衡器在 Amazon ECS PrivateLink 上私下访问容器应用程序。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 已在 Linux、macOS 或 Windows 上安装并配置 AWS 命令行界面（AWS CLI）版本 2
- Docker，已在 Linux、macOS 或 Windows 上安装并配置
- 在 Docker 上运行的应用程序

架构

技术堆栈

- Amazon CloudWatch
- 亚马逊弹性计算云（亚马逊 EC2）
- Amazon A EC2 uto Scaling

- Amazon Elastic Container Registry(Amazon ECR)
- Amazon ECS
- Amazon RDS
- Amazon Simple Storage Service（Amazon S3）
- AWS Lambda
- AWS PrivateLink
- AWS Secrets Manager
- 应用程序负载均衡器
- 网络负载均衡器
- VPC

自动化和扩缩

- 您可以使用 AWS 通过使用基础设施即代码 CloudFormation来创建此模式。

工具

- 亚马逊 EC2 — 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。
- Amazon A EC2 ut EC2 o Scaling — Amazon Auto Scaling 可帮助您确保有正确数量的亚马逊 EC2 实例可用来处理应用程序的负载。
- Amazon ECS - Amazon Elastic Container Service (Amazon ECS)是一项高度可扩展的快速容器管理服务，可帮助轻松运行、停止和管理集群上的容器。
- Amazon ECR - Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可靠且可扩展的 AWS 托管容器映像注册表服务。
- Amazon EFS - Amazon Elastic File System (Amazon EFS)可提供简单、可扩展、完全托管的弹性 NFS 文件系统，以便与 Amazon Web Services Cloud 服务和本地资源配合使用。
- AWS Lambda - Lambda 是一项计算服务，使您无需预调配或管理服务器即可运行代码。
- Amazon RDS – Amazon Relational Database Service (Amazon RDS) 是一项 Web 服务，使用户能够在 Amazon Web Services Cloud 中轻松设置、操作和扩展关系数据库。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项面向互联网的存储服务。该服务旨在降低开发人员进行网络规模级计算的难度。
- AWS Secrets Manager - Secrets Manager 允许您将代码中的硬编码凭证(包括密码)替换为对 Secrets Manager 的 API 调用，并以编程方式检索密钥。

- Amazon VPC - Amazon Virtual Private Cloud (Amazon VPC)可助您将 AWS 资源启动到您已定义的虚拟网络中。
- Elastic L oad Balancing — Elastic Load Balancing 将传入的应用程序或网络流量分配到多个可用区的多个目标，例如亚马逊 EC2 实例、容器和 IP 地址。
- Docker - Docker 允许开发人员打包、交付和运行任何应用程序，并将其作为轻量、便携且自给自足的容器。

操作说明

创建联网组件

Task	描述	所需技能
创建 VPC。	1. 登录 AWS 管理控制台，打开 Amazon VPC 控制台。选择创建 VPC，然后选择 VPC 及其他。 2. 输入 VPC 的名称，然后选择适当的 CIDR 块范围。 3. 指定两个可用区、两个公有子网、四个私有子网。两个私有子网用于 Amazon ECS 任务，两个私有子网用于 Amazon RDS 数据库。 4. 为每个可用区指定一个 NAT 网关。 5. 选择创建 VPC。	云管理员

创建负载均衡器

Task	描述	所需技能
创建网络负载均衡器。	1. 打开亚马逊 EC2 控制台，选择包含您的 VPC 的 AWS 区域。	云管理员

Task	描述	所需技能
	2. 在负载均衡下方选择负载均衡器，然后选择创建负载均衡器。 3. 选择网络负载均衡器，然后选择创建。 4. 在配置负载均衡器页面上，配置您的网络负载均衡器和侦听器。重要提示：请务必将网络负载均衡器的模式选择为内部。 5. 选择适用的安全设置，配置安全组和目标组。在配置路由部分中选择实例或 IP 作为目标类型。确保您没有注册目标。 6. 在配置完所有设置后，选择下一步：查看，然后选择创建。	

Task	描述	所需技能
创建应用程序负载均衡器。	1. 在 Amazon EC2 控制台上，选择包含您的 VPC 的相同区域。 2. 在负载均衡下方选择负载均衡器，然后选择创建负载均衡器。 3. 选择应用程序负载均衡器，然后选择创建。 4. **Important** 配置应用程序负载均衡器及其侦听器 确保将应用程序负载均衡器的方案选择为“内部”。 5. 选择适用的安全设置，配置安全组和目标组。在配置路由部分中选择实例或 IP 作为目标类型。确保您没有注册目标。 6. 在配置完所有设置后，选择下一步：查看，然后选择创建。	云管理员

创建 Amazon EFS 文件系统

Task	描述	所需技能
创建 Amazon EFS 文件系统。	1. 打开 Amazon EFS 控制台，然后选择创建文件系统。	云管理员

Task	描述	所需技能
	2. 在创建文件系统对话框中，输入文件系统名称，然后选择您的 VPC。 3. 选择创建以创建文件系统。 4. 设置并配置您的 Amazon EFS 文件系统。	
子网的挂载目标。	1. 返回 Amazon EFS 控制台，然后选择文件系统。文件系统页面会显示您账户中的 Amazon EFS 文件系统。 2. 选择您创建的文件系统，然后选择管理以显示可用区。要添加挂载目标，请选择添加挂载目标，然后添加您创建的四个私有子网。	云管理员
验证子网是否已挂载为目标。	1. 在 Amazon EFS 控制台上，选择文件系统。 2. 选择网络以显示现有挂载目标的列表。确保这些子网包括您创建的四个子网。	云管理员

创建 S3 存储桶

Task	描述	所需技能
创建 S3 存储桶。	打开 Amazon S3 控制台并创建一个 S3 存储桶以存储应用程序的静态资产(如有需要)。	云管理员

创建 Secrets Manager 密钥

Task	描述	所需技能
创建 AWS KMS 密钥以加密 Secrets Manager 密钥。	打开 AWS Key Management Service (AWS KMS)控制台并创建 KMS 密钥。	云管理员
创建 Secrets Manager 密钥以存储 Amazon RDS 密码。	1. 打开 AWS Secrets Manager 控制台，选择存储新密钥以创建一个新密钥。 2. 选择您创建的 KMS 密钥，并存储您的新密钥。	云管理员

创建 Amazon RDS 实例

Task	描述	所需技能
创建数据库子网组。	1. 打开 Amazon RDS 控制台，然后选择子网组。 2. 选择创建数据库子网组，然后输入数据库子网组的名称和描述。 3. 选择您之前创建的 VPC，然后选择可用区和子网。然后选择创建。	云管理员
创建 Amazon RDS 实例。	在私有子网中创建和配置 Amazon RDS 实例。确保已启用多可用区以实现高可用性(HA)。	云管理员
将数据载入 Amazon RDS 实例。	将应用程序所需关系数据加载到 Amazon RDS 实例中。此流程将根据应用程序的需求以及	云管理员、数据库管理员

Task	描述	所需技能
	数据库架构的定义和设计方式而有所不同。	

创建 Amazon ECS 组件

Task	描述	所需技能
创建 ECS 集群。	1. 打开 Amazon ECS 控制台并选择集群。 2. 选择创建集群，然后根据所需规范设置 ECS 集群。	云管理员
创建 Docker 映像	按照相关资源部分中的说明创建 Docker 映像。	云管理员
创建 Amazon ECR 存储库。	1. 在 Amazon ECR 控制台中选择存储库。 2. 选择创建存储库，然后输入存储库的唯一名称。 3. 根据您的规范配置存储库，包括 AWS KMS 加密（如有需要）。	云管理员、 DevOps 工程师
对您的 Amazon ECR 注册表进行 Docker CLI 身份验证。	要对 Amazon ECR 存储库的 Docker 客户端进行身份验证，请在 AWS CLI 中运行 “`aws ecr get-login-password` 命令。	云管理员
推送 Docker 映像至 Amazon ECR 存储库	1. 确定要推送的 Docker 映像，然后在 AWS CLI 中运行 `docker images` 命令。	云管理员

Task	描述	所需技能
	2. 使用 Amazon ECR 注册表、存储库和可选映像标签名称组合标记您的映像。 3. 运行 `docker push` 命令推送 Docker 映像。 4. 对所有需要的映像重复上述步骤。	
创建 Amazon ECS 任务定义。	需要任务定义才能在 Amazon ECS 中运行 Docker 容器。 1. 返回 Amazon ECS 控制台，选择任务定义，然后选择创建新任务定义。 2. 在选择兼容性页面上，选择您的任务应使用的启动类型，然后选择下一步。 **⚠ Important** 有关设置任务定义的帮助，请参阅相关资源部分中的“创建任务定义”。请务必提供推送到亚马逊 ECR 的 Docker 镜像。	云管理员
创建 Amazon ECS 服务	使用您之前创建的 ECS 集群创建 Amazon ECS 服务。确保选择 Amazon EC2 作为启动类型，然后选择在上一步中创建的任务定义以及 Application Load Balancer 的目标组。	云管理员

创建 Amazon A EC2 uto Scaling 群组

Task	描述	所需技能
创建启动配置。	打开 Amazon EC2 控制台，然后创建启动配置。确保用户数据包含允许 EC2 实例加入所需的 ECS 集群的代码。有关所需代码的示例，请参阅相关资源部分。	云管理员
创建 Amazon A EC2 uto Scaling 群组。	返回亚马逊 EC2 控制台，在 A uto S caling 下，选择 A uto Scaling 群组。设置一个 Amazon A EC2 uto Scaling 群组。请确保您选择了之前创建的私有子网和启动配置。	云管理员

设置 AWS PrivateLink

Task	描述	所需技能
设置 AWS 终 PrivateLink 端节点。	1. 在 Amazon VPC 控制台上，创建 AWS PrivateLink 终端节点。 2. 将此端点与网络负载均衡器相关联，使客户可私密使用 Amazon ECS 上托管的应用程序。 有关更多信息，请参阅相关资源部分。	云管理员

创建 VPC 端点

Task	描述	所需技能
创建 VPC 端点。	为您之前创建的 AWS 终端 PrivateLink 节点创建 VPC 终端节点。VPC 终端节点完全限定域名 (FQDN) 将指向 AWS PrivateLink 终端节点 FQDN。这将创建一个 DNS 端点可以访问的 VPC 端点服务的弹性网络接口。	云管理员

创建 Lambda 函数

Task	描述	所需技能
创建 Lambda 函数。	在 AWS Lambda 控制台上，创建 Lambda 函数以将应用程序负载均衡器 IP 地址更新为网络负载均衡器的目标。有关这方面的更多信息，请参阅使用 AWS Lambda 为应用程序负载均衡器启用静态 IP 地址博客文章。	应用程序开发人员

相关资源

创建负载均衡器：

- 为 Amazon ECS 使用网络负载均衡器
- 创建网络负载均衡器
- 为 Amazon ECS 使用应用程序负载均衡器
- 创建应用程序负载均衡器

创建 Amazon EFS 文件系统：

- 创建 Amazon EFS 文件系统
- 在 Amazon EFS 中创建挂载目标

创建 S3 存储桶：

- 创建 S3 存储桶

创建 Secrets Manager 密钥：

- 在 AWS KMS 中创建密钥
- 在 AWS Secrets Manager 中创建密钥

创建 Amazon RDS 实例：

- 创建 Amazon RDS 数据库实例

创建 Amazon ECS 组件：

- 创建 Amazon ECS 集群
- 创建 Docker 映像
- 创建 Amazon ECR 存储库
- 使用 Amazon ECR 存储库对 Docker 进行身份验证
- 将映像推送至 Amazon ECR 存储库
- 创建 Amazon ECS 任务定义
- 创建 Amazon ECS 服务

创建 Amazon A EC2 uto Scaling 群组：

- 创建启动配置
- 使用启动配置创建自动扩缩组
- 使用亚马逊 EC2 用户数据引导容器实例

设置 AWS PrivateLink：

- VPC 终端节点服务 (AWS PrivateLink)

创建 VPC 端点：

- 接口 VPC 终端节点 (AWS PrivateLink)

创建 Lambda 函数：

- 创建 Lambda 函数

其他资源：

- 为应用程序负载均衡器使用静态 IP 地址
- 通过 AWS 安全访问服务 PrivateLink

使用 AWS Fargate PrivateLink、AWS 和网络负载均衡器在 Amazon ECS 上私下访问容器应用程序

由 Kirankumar Chandrashekar (AWS) 创建

摘要

此模式描述了如何使用带有 AWS Fargate 启动类型的亚马逊弹性容器服务 (Amazon ECS)，在网络负载均衡器后面，在亚马逊网络服务 (AWS) 云上私下托管 Docker 容器应用程序，并使用 AWS 访问该应用程序。 PrivateLinkAmazon Relational Database Service (Amazon RDS) 为在具有高可用性 (HA) 的 Amazon ECS 上运行的应用程序关系数据库提供托管。如果应用程序需要持久性存储，则您可以使用 Amazon Elastic File System (Amazon EFS)。

此模式对运行 Docker 应用程序的 Amazon ECS 服务使用 Fargate 启动类型，并在前端使用网络负载均衡器。然后可以将其与虚拟私有云 (VPC) 终端节点关联，以便通过 AWS 进行访问 PrivateLink。然后，可以使用 VPC 终端节点与其他 VPCs 人共享此 VPC 终端节点服务。

您可以将 Fargate 与 Amazon ECS 配合使用来运行容器，而无需管理服务器或亚马逊弹性计算云 (Amazon EC2) 实例的集群。你也可以使用 Amazon A EC2 uto Scaling 群组来代替 Fargate。有关更多信息，请参阅使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon ECS 上私下访问容器应用程序。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 已在 Linux、macOS 或 Windows 上安装并配置 AWS 命令行界面（AWS CLI）版本 2
- Docker，已在 Linux、macOS 或 Windows 上安装并配置
- 在 Docker 上运行的应用程序

架构

技术堆栈

- Amazon CloudWatch

- Amazon Elastic Container Registry(Amazon ECR)
- Amazon ECS
- Amazon EFS
- Amazon RDS
- Amazon Simple Storage Service (Amazon S3)
- AWS Fargate
- AWS PrivateLink
- AWS Secrets Manager
- 应用程序负载均衡器
- 网络负载均衡器
- VPC

自动化和扩缩

- 您可以使用 AWS 通过使用基础设施即代码 CloudFormation来创建此模式。

工具

AWS 服务

- Amazon Elastic Container Registry (Amazon ECR) Amazon ECRA 是一项托管的 AWS 容器镜像注册服务，安全、可扩展且可靠。
- Amazon Elastic Container Service (Amazon ECS) 是一项高度可扩展、快速的容器管理服务，可以轻松地在集群上运行、停止和管理容器。
- Amazon Elastic File System (Amazon EFS) 提供了一个简单、可扩展、完全托管的弹性 NFS 文件系统，可用于 AWS 云服务和本地资源。
- AWS Fargate是一种可以与 Amazon ECS 配合使用的技术，无需管理服务器或亚马逊 EC2 实例集群即可运行容器。
- Amazon Relational Database Service (Amazon RDS) 是一项网络服务，可以更轻松地在中设置、操作和扩展关系数据库 AWS Cloud。
- 亚马逊简单存储服务 (Amazon S3) S ervice 是互联网存储。该服务旨在降低开发人员进行网络规模级计算的难度。

- AWS Secrets Manager 有助于您通过对 Secrets Manager 的 API 调用来替换代码中的硬编码凭证（包括密码），以编程方式检索密钥。
- Amazon Virtual Private Cloud（亚马逊 VPC）可帮助您将AWS资源启动到您定义的虚拟网络中。
- El@@ astic Load Balancing (ELB) 将传入的应用程序或网络流量分发到多个可用区域中的多个目标，例如 EC2 实例、容器和 IP 地址。

其他工具

- Docker 可帮助开发人员轻松打包、交付和运行任何应用程序，将其作为轻量级、便携且自给自足的容器。

操作说明

创建联网组件

Task	描述	所需技能
创建 VPC。	1. 登录 AWS 管理控制台，打开 Amazon VPC 控制台。选择创建 VPC，然后选择 VPC 及其他。 2. 输入 VPC 的名称，然后选择适当的 CIDR 块范围。 3. 指定两个可用区、两个公有子网、四个私有子网。两个私有子网用于 Amazon ECS 任务，两个私有子网用于 Amazon RDS 数据库。 4. 为每个可用区指定一个 NAT 网关。 5. 选择创建 VPC。	云管理员

创建负载均衡器

Task	描述	所需技能
创建网络负载均衡器。	1. 打开亚马逊 EC2 控制台，然后选择包含您的 VPC 的 AWS 区域。 2. 在负载均衡下方选择负载均衡器，然后选择创建负载均衡器。 3. 选择网络负载均衡器，然后选择创建。 4. 在配置负载均衡器页面上，配置您的网络负载均衡器和侦听器。重要提示：请务必将网络负载均衡器的模式选择为内部。 5. 选择适用的安全设置，配置安全组和目标组。在配置路由部分中选择 IP 作为目标类型。确保您没有注册目标。 6. 在配置完所有设置后，选择下一步：查看，然后选择创建。 要获取有关此操作和其他操作的帮助，请参阅相关资源部分。	云管理员
创建应用程序负载均衡器。	1. 在 Amazon EC2 控制台上，选择包含您的 VPC 的相同区域。 2. 在负载均衡下方选择负载均衡器，然后选择创建负载均衡器。	云管理员

Task	描述	所需技能
	3. 选择应用程序负载均衡器，然后选择创建。 4. **Important** 配置应用程序负载均衡器及其侦听器 确保将应用程序负载均衡器的方案选择为“内部”。 5. 选择适用的安全设置，配置安全组和目标组。在配置路由部分中选择 IP 作为目标类型。确保您没有注册目标。 6. 在配置完所有设置后，选择下一步：查看，然后选择创建。	

创建 Amazon EFS 文件系统

Task	描述	所需技能
创建 Amazon EFS 文件系统。	1. 打开 Amazon EFS 控制台，然后选择创建文件系统。 2. 在创建文件系统对话框中，输入文件系统名称，然后选择您的 VPC。 3. 选择创建以创建文件系统。 4. 设置并配置您的 Amazon EFS 文件系统。	云管理员
子网的挂载目标。	1. 返回 Amazon EFS 控制台，然后选择文件系统。文件系	云管理员

Task	描述	所需技能
	统页面会显示您账户中的 Amazon EFS 文件系统。 2. 选择您创建的文件系统，然后选择管理以显示可用区。 3. 要添加挂载目标，请选择添加挂载目标，然后添加您创建的四个私有子网。	
验证子网是否已挂载为目标。	1. 在 Amazon EFS 控制台上，选择文件系统。 2. 选择网络以显示现有挂载目标的列表。确保这些子网包括您创建的四个子网。	云管理员

创建 S3 存储桶

Task	描述	所需技能
创建 S3 存储桶。	如果需要，打开 Amazon S3 控制台并创建一个 S3 存储桶来存储应用程序的静态资产。	云管理员

创建 Secrets Manager 密钥

Task	描述	所需技能
创建 AWS KMS 密钥以加密 Secrets Manager 密钥。	打开 AWS Key Management Service (AWS KMS)控制台并创建 KMS 密钥。	云管理员
创建 Secrets Manager 密钥以存储 Amazon RDS 密码。	1. 打开 AWS Secrets Manager 控制台，选择存储新密钥以创建一个新密钥。	云管理员

Task	描述	所需技能
	2. 选择您创建的 KMS 密钥，并存储您的新密钥。	

创建 Amazon RDS 实例

Task	描述	所需技能
创建数据库子网组。	1. 打开 Amazon RDS 控制台，然后选择子网组。 2. 选择创建数据库子网组，然后输入数据库子网组的名称和描述。 3. 选择您之前创建的 VPC，然后选择可用区和子网。然后选择创建。	云管理员
创建 Amazon RDS 实例。	在私有子网中创建和配置 Amazon RDS 实例。确保已启用多可用区以实现高可用性 (HA)。	云管理员
将数据载入 Amazon RDS 实例。	将应用程序所需关系数据加载到 Amazon RDS 实例中。此流程将根据应用程序的需求以及数据库架构的定义和设计方式而有所不同。	数据库管理员

创建 Amazon ECS 组件

Task	描述	所需技能
创建 ECS 集群。	1. 打开 Amazon ECS 控制台并选择集群。	云管理员

Task	描述	所需技能
	2. 选择创建集群，然后根据所需规范设置 ECS 集群。	
创建 Docker 映像	按照文档中的说明创建 Docker 镜像。AWS	云管理员
创建 Amazon ECR 存储库。	1. 打开 Amazon ECR 控制台，然后选择存储库。 2. 选择创建存储库，然后输入存储库的唯一名称。 3. 根据您的规范配置存储库，包括 AWS KMS 加密（如有需要）。	云管理员、 DevOps 工程师
推送 Docker 映像至 Amazon ECR 存储库	1. 确定要推送的 Docker 映像，然后在 AWS CLI 中运行 `docker images` 命令。 2. 使用 Amazon ECR 注册表、存储库和可选映像标签名称组合标记您的映像。 3. 运行 `docker push` 命令推送 Docker 映像。 4. 对所有需要的映像重复上述步骤。	云管理员

Task	描述	所需技能
创建 Amazon ECS 任务定义。	需要任务定义才能在 Amazon ECS 中运行 Docker 容器。 1. 返回 Amazon ECS 控制台，选择任务定义，然后选择创建新任务定义。 2. 在选择兼容性页面上，选择您的任务应使用的启动类型，然后选择下一步。 **Important** 有关设置任务定义的帮助，请参阅相关资源部分中的“创建任务定义”。请务必提供推送到亚马逊 ECR 的 Docker 镜像。	云管理员
创建 ECS 服务，然后选择 Fargate 作为启动类型。	1. 使用您之前@@ 创建的 ECS 集群创建 Amazon ECS 服务。确保选择 Fargate 作为启动类型。 2. 选择在上一步中创建的任务定义，并选择应用程序负载均衡器的目标组。	云管理员

设置 AWS PrivateLink

Task	描述	所需技能
设置 AWS 终 PrivateLink 端节点。	1. 打开 Amazon VPC 控制台，然后创建 AWS PrivateLink 终端节点。 2. 将此端点与网络负载均衡器相关联，使客户可私密使用 Amazon ECS 上托管的应用程序。	云管理员

创建 VPC 端点

Task	描述	所需技能
创建 VPC 端点。	为您之前@@ 创建的 AWS 终端 PrivateLink 节点创建 VPC 终端节点。VPC 终端节点完全限定域名 (FQDN) 将指向 AWS PrivateLink 终端节点 FQDN。这会为 VPC 端点服务创建一个可供域名服务端点访问的弹性网络接口。	云管理员

设定目标

Task	描述	所需技能
将 Application Load Balancer 添加为目标。	要将应用程序负载均衡器添加为网络负载均衡器的目标，请按照AWS 文档中的说明进行操作。	应用程序开发人员

相关资源

创建负载均衡器：

- 为 Amazon ECS 使用网络负载均衡器
- 创建网络负载均衡器
- 为 Amazon ECS 使用应用程序负载均衡器
- 创建应用程序负载均衡器

创建 Amazon EFS 文件系统：

- 创建 Amazon EFS 文件系统
- 在 Amazon EFS 中创建挂载目标

创建 Secrets Manager 密钥：

- 在 AWS KMS 中创建密钥
- 在 AWS Secrets Manager 中创建密钥

创建 Amazon RDS 实例：

- 创建 Amazon RDS 数据库实例

创建 Amazon ECS 组件

- 创建 Amazon ECR 存储库
- 使用 Amazon ECR 存储库对 Docker 进行身份验证
- 将映像推送至 Amazon ECR 存储库
- 创建 Amazon ECS 任务定义
- 创建 Amazon ECS 服务

其他资源：

- 通过 AWS 安全访问服务 PrivateLink

使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon EKS 上私密访问容器应用程序

由 Kirankumar Chandrashekar (AWS) 创建

摘要

此模式描述了如何在网络负载均衡器后面的亚马逊 Elastic Kubernetes Service（Amazon EKS）上私下托管 Docker 容器应用程序，以及如何使用 AWS 访问该应用程序。 PrivateLink然后，您便可以使用专用网络安全地访问 Amazon Web Services (AWS) Cloud 上的服务。

运行 Docker 应用程序的 Amazon EKS 集群在前端装有 Network Load Balancer，可以与虚拟私有云 (VPC) 终端节点相关联，以便通过 AWS PrivateLink 进行访问。然后，可以使用 VPC 终端节点与其他 VPCs 人共享此 VPC 终端节点服务。

此模式描述的设置是在 VPCs 和 AWS 账户之间共享应用程序访问权限的安全方式。它不需要特殊的连接或路由配置，因为使用者和提供商账户之间的连接位于全球 AWS 主干网上，不会遍历公共互联网。

先决条件和限制

先决条件

- Docker，已在 Linux、macOS 或 Windows 上安装并配置。
- 在 Docker 上运行的应用程序。
- 一个有效的 Amazon Web Services account。
- AWS 命令行界面（AWS CLI）版本 2，已在 Linux、macOS 或 Windows 上安装并配置。
- 具有标记的私有子网并配置为托管应用程序的现有 Amazon EKS 集群。有关更多信息，请参阅 Amazon EKS 文档中的子网标记。
- Kubectl，已安装并配置为访问 Amazon EKS 集群上的资源。有关更多信息，请参阅 Amazon EKS 文档中的安装 kubectl 。

架构

技术堆栈

- Amazon EKS
- AWS PrivateLink

- 网络负载均衡器

自动化和扩缩

- Kubernetes 清单可以在基于 Git 的存储库上进行跟踪和管理，也可以使用 AWS 中的持续集成和持续交付 (CI/CD) 进行部署。 CodePipeline
- 您可以使用 AWS CloudFormation 通过使用基础设施即代码 (IaC) 来创建此模式。

工具

- AWS CLI - AWS 命令行界面（AWS CLI）是一种开源工具，它使您能够使用命令行 Shell 中的命令与 Amazon Web Services 交互。
- Elastic L oad Balancing — Elastic Load Balancing 将传入的应用程序或网络流量分配到一个或多个可用区的多个目标，例如亚马逊弹性计算云 (Amazon EC2) 实例、容器和 IP 地址。

- Amazon EKS – Amazon Elastic Kubernetes Service (Amazon EKS) 是一项托管服务，可让您在 AWS 上轻松运行 Kubernetes，而无需安装、操作和维护您自己的 Kubernetes 控制面板或节点。
- Amazon VPC - Amazon Virtual Private Cloud (Amazon VPC)可助您将 AWS 资源启动到您已定义的虚拟网络中。
- Kubectl - Kubectl 是一个命令行实用程序，用于对 Kubernetes 集群运行命令。

操作说明

部署 Kubernetes 部署和服务清单文件

Task	描述	所需技能

创建 Kubernetes 部署清单文件。	根据您的要求修改以下示例文件，以创建部署清单文件。
	`apiVersion: apps/v1`
	`kind: Deployment`
	`metadata:`
	`  name: sample-app`
	`spec:`
	`  replicas: 3`
		DevOps 工程师

Task	描述	所需技能

	<pre> selector:
	    matchLabels:
	      app: nginx
	  template:
	    metadata:
	      labels:
	        app: nginx
	    spec:
	      containers:
	        - name: nginx
	          image:
	 public.ecr.aws/z9d
	2n7e1/nginx:1.19.5
	          ports:
	            - name: http
	              container
	Port: 80</pre>
	Note
	这是使用 NGINX Docker 镜像部署的 NGINX 示例配置文件。有关更多信息，请参阅 Docker 文档中的如何使用官方 NGINX Docker 映像。	

部署 Kubernetes 部署清单文件。	运行以下命令，将部署清单文件应用于 Amazon EKS 集群： `kubectl apply -f <your_deployment_file_name>`	DevOps 工程师

Task	描述	所需技能

创建 Kubernetes 服务清单文件。	根据您的要求修改以下示例文件，创建服务清单文件。
	
	`apiVersion: v1`
	`kind: Service`
	`metadata:`
	`  name: sample-service`
	`  annotations:`
	`  service.beta.kubernetes.io/aws-load-balancer-type: nlb`
	`  service.beta.kubernetes.io/aws-load-balancer-internal: "true"`
	`spec:`
	`  ports:`
	`  - port: 80`
	`  targetPort: 80`
	`  protocol: TCP`
	`  type: LoadBalancer`
	`  selector:`
	`  app: nginx`
	
	Important
	在定义内部 Network Load Balancer 时，请务必包括以下内容`annotations` ：
	
	`service.beta.kubernetes.io/aws-load-balancer-type: nlb`
	`service.beta.kubernetes.io/aws-l`
		DevOps 工程师

Task	描述	所需技能
	`oad-balancer-inter nal: "true"`	
部署 Kubernetes 服务清单文件。	运行以下命令以将服务清单文件应用于 Amazon EKS 集群： `kubectl apply -f <your_service_file _name>`	DevOps 工程师

创建端点

Task	描述	所需技能
记录网络负载均衡器的名称。	运行以下命令来检索网络负载均衡器的名称： `kubectl get svc sample-service -o wide` 记录网络负载均衡器的名称，这是创建 AWS PrivateLink 终端节点所必需的。	DevOps 工程师
创建 AWS PrivateLink 终端节点。	登录 AWS 管理控制台，打开 Amazon VPC 控制台，然后创建 AWS PrivateLink 终端节点。将此端点与网络负载均衡器相关联，这使得应用程序可供客户私下使用。有关更多信息，请参阅 Amazon VPC 文档中的 VPC 终端节点服务 (AWS PrivateLink)。	云管理员

Task	描述	所需技能
	Important 如果使用者账户需要访问应用程序，则必须将使用者账户的 AWS 账户 ID 添加到 AWS PrivateLink 终端节点配置的允许委托人列表中。有关更多信息，请参阅 Amazon VPC 文档中的添加和删除端点服务的权限。	
创建 VPC 端点。	在 Amazon VPC 控制台中，选择端点服务，然后选择您的端点服务。为 AWS 终端节点创建 VPC PrivateLink 终端节点。 VPC 终端节点的完全限定域名 (FQDN) 指向 AWS PrivateLink 终端节点的 FQDN。这将创建一个 DNS 端点可以访问的 VPC 端点服务的弹性网络接口。	云管理员

相关资源

- 使用官方 NGINX Docker 映像
- Amazon EKS 上的网络负载均衡
- 创建 VPC 终端节点服务 (AWS PrivateLink)
- 为您的端点服务添加和删除权限

在 Amazon EKS 上使用 AWS 私有 CA 在 AWS App Mesh 中激活 mTLS

由奥马尔·卡希尔（AWS）、伊曼纽尔·萨利乌（AWS）、穆罕默德·沙扎德（AWS）和安迪·王（AWS）创作

摘要

此模式展示了如何使用 AWS App Mesh 中的 AWS 私有证书颁发机构 (AWS Private CA) 的证书在 Amazon Web Services (AWS) 上实施相互传输层安全性 (mTLS)。它通过面向所有人的安全生产标识框架 (SPIFFE) 使用 Envoy 机密发现服务 (SDS) API。SPIFFE 是一个云原生计算基金会 (CNCF) 开源项目，具有广泛的社区支持，可提供细粒度和动态的工作负载身份管理。要实现 SPIFFE 标准，请使用 SPIRE SPIFFE 运行时环境。

在 App Mesh 中使用 mTLS 可提供双向对等身份验证，因为它在 TLS 上增加了一层安全性，并允许网格中的服务验证正在建立连接的客户端。客户端-服务器关系中的客户端还会在会话协商过程中提供 X.509 证书。服务器使用此证书来识别和验证客户端。这有助于验证证书是否由受信任的证书颁发机构 (CA) 颁发，以及证书是否有效。

先决条件和限制

先决条件

- 具有自行管理或托管节点组的 Amazon Elastic Kubernetes Service（Amazon EKS）集群
- 在已激活 SDS 的集群上部署的 App Mesh 控制器
- 由 AWS Private CA 颁发的来自 AWS Certificate Manager（ACM）的私有证书

限制

- 无法在 AWS Fargate 上安装 SPIRE，因为 SPIRE 代理必须作为 Kubernetes 运行。 DaemonSet

产品版本

- AWS App Mesh Controller 图表 1.3.0 或更高版本

架构

下图显示了 VPC 中具有 App Mesh 的 EKS 集群。一个 Worker 节点中的 SPIRE 服务器与其他工作线程节点中的 SPIRE 代理以及 AWS 私有 CA 进行通信。Envoy 用于 SPIRE Agent Worker 节点之间的 mTLS 通信。

下图说明了以下步骤：

1. 颁发证书。
2. 请求证书签名和证书。

工具

Amazon Web Services

- AWS 私有 CA — AWS 私有证书颁发机构 (AWS Private CA) 支持创建私有证书颁发机构 (CA) 层次结构 CAs，包括根和下级结构，而无需支付运营本地 CA 的投资和维护成本。
- AWS App Mesh – AWS App Mesh 是一种服务网格，可轻松监控和控制服务。App Mesh 将服务通信方式标准化，为应用程序中的每个服务提供一致的可见性和网络流量控制。
- Amazon EKS – Amazon Elastic Kubernetes Service (Amazon EKS) 是一项托管服务，可让您在 AWS 上轻松运行 Kubernetes，而无需安装、操作和维护您自己的 Kubernetes 控制面板或节点。

其他工具

- Helm – Helm 是 Kubernetes 的软件包管理器，可帮助您在 Kubernetes 集群上安装和管理应用程序。此模式使用 Helm 部署 AWS App Mesh Controller。
- AWS App Mesh 控制器图表 – 此模式使用 AWS App Mesh 控制器图表在 Amazon EKS 上启用 AWS App Mesh。

操作说明

设置环境

Task	描述	所需技能
使用 Amazon EKS 设置 App Mesh。	按照存储库中提供的基本部署步骤进行操作。	DevOps 工程师
安装 SPIRE。	使用 spire_setup.yaml 在 EKS 集群上安装 SPIRE。	DevOps 工程师

Task	描述	所需技能
安装 AWS 私有 CA 证书。	按照 AWS 文档中的说明为您的私有根 CA 创建并安装证书。	DevOps 工程师
向集群节点实例角色授予权限。	若要将策略附加到集群节点实例角色，请使用其他信息部分中的代码。	DevOps 工程师
添加适用于 AWS Private CA 的 SPIRE 插件。	若要将插件添加到 SPIRE 服务器配置中，请使用其他信息部分中的代码。将 `certificate_authority_arn` Amazon 资源名称（ARN）替换为您的私有 CA ARN。使用的签名算法必须与私有 CA 上的签名算法相同。将 `your_region` 替换为您的 AWS 区域。 有关该插件的更多信息，请参阅服务器插件： UpstreamAuthority “aws_pca”。	DevOps 工程师
更新 bundle.cert。	创建 SPIRE 服务器后，将创建一个 `spire-bundle.yaml` 文件。将 `spire-bundle.yaml` 文件中的 `bundle.crt` 值从私有 CA 更改为公共证书。	DevOps 工程师

部署和注册工作负载

Task	描述	所需技能
向 SPIRE 注册节点和工作负载条目。	要向 SPIRE Server 注册节点和工作负载（服务），请使用存储库中的代码。	DevOps 工程师
在激活 mTLS 的情况下在 App Mesh 中创建网格。	在 App Mesh 中创建一个新网格，其中包含微服务应用程序的所有组件（例如，虚拟服务、虚拟路由器和虚拟节点）。	DevOps 工程师
检查已注册的条目。	您可以通过运行以下命令来检查节点和工作负载的已注册条目。 `kubectl exec -n spire spire-server-0 -- /opt/spire/bin/spire-server entry show` 这将显示 SPIRE 代理的条目。	DevOps 工程师

验证 mTLS 流量

Task	描述	所需技能
验证 mTLS 流量。	1. 从前端服务向后端服务发送 HTTP 标头，并使用在 SPIRE 中注册的服务验证响应是否成功。 2. 对于双向 TLS 身份验证，可以通过运行以下命令来检查 `ssl.handshake` 统计信息。	DevOps 工程师

Task	描述	所需技能

	`kubectl exec -it $POD -n $NAMESPACE -c envoy -- curl http://localhost:9901/stats | grep ssl.handshake`
	
	运行上一个命令后，应会看到侦听器 `ssl.handshake` 计数，该计数将类似于以下示例：
	
	`listener.0.0.0.0_15000.ssl.handshake: 2`	

验证证书是否是从 AWS 私有 CA 颁发的。	您可以通过查看 SPIRE 服务器中的日志来检查插件是否已正确配置，以及证书是否从上游私有 CA 颁发。运行以下命令。 `kubectl logs spire-server-0 -n spire` 然后查看生成的日志。此代码假定您的服务器名为 `spire-server-0` ，并托管在您的 spire 命名空间中。您应该看到插件已成功加载，并且正在与上游私有 CA 建立连接。	DevOps 工程师

相关资源

- 在 Amazon EKS 上的 AWS App Mesh 中将 mTLS 与 SPIFFE/SPIRE 结合使用

- 在多账户 Amazon EKS 环境中使用 SPIFFE/SPIRE 在 AWS App Mesh 中启用 mTLS
- 此模式中使用的演练
- 服务器插件： UpstreamAuthority “aws_pca”
- Kubernetes 快速入门

其他信息

将权限附加到集群节点实例角色

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ACMPCASigning",
            "Effect": "Allow",
            "Action": [
                "acm-pca:DescribeCertificateAuthority",
                "acm-pca:IssueCertificate",
                "acm-pca:GetCertificate",
                "acm:ExportCertificate"
            ],
            "Resource": "*"
        }
    ]
}
AWS Managed Policy: "AWSAppMeshEnvoyAccess"
```

添加适用于 ACM 的 SPIRE 插件

```
Add the SPIRE plugin for ACM
Change certificate_authority_arn to your PCA ARN. The signing algorithm used must be
 the same as the signing algorithm on the PCA. Change your_region to the appropriate
 AWS Region.
UpstreamAuthority "aws_pca" {
    plugin_data {
        region = "your_region"
        certificate_authority_arn = "arn:aws:acm-pca:...."
        signing_algorithm = "your_signing_algorithm"
    }
}
```

使用 AWS Batch 自动备份 Amazon RDS for PostgreSQL 数据库实例

由 Kirankumar Chandrashekar (AWS) 创建

摘要

备份 PostgreSQL 数据库是一项重要的任务，通常可以使用 pg_dump utility 完成，该实用程序默认使用 COPY 命令创建 PostgreSQL 数据库的架构和数据转储。但是，如果您需要定期备份多个 PostgreSQL 数据库，则此进程可能会变得重复。如果您的 PostgreSQL 数据库托管在云端，您也可以利用Amazon Relational Database Service (Amazon RDS) 为 PostgreSQL 提供的自动备份功能。此示例介绍了如何使用 pg_dump 实用程序自动执行适用于 Amazon RDS for PostgreSQL 数据库实例的定期备份。

注意：这些说明假定您使用的是 Amazon RDS。但是，您也可以对托管在 Amazon RDS 外部的 PostgreSQL 数据库采用这种方法。若要进行备份，AWS Lambda 函数必须能访问您的数据库。

基于时间的亚马逊 CloudWatch 事件会启动 Lambda 函数，该函数搜索应用于 Amazon RDS 上 PostgreSQL 数据库实例元数据的特定备份标签。如果 DBDump PostgreSQL 数据库实例具有 bkp: Automated = Active 标签和其他必需的备份标签，则 Lambda 函数将每个数据库备份的单个任务提交到 AWS Batch。

AWS Batch 会处理这些任务，并将备份数据上至到 Amazon Simple Storage Service（Amazon S3）存储桶。此模式使用 Dockerfile 和 entrypoint.sh 文件构建 Docker 容器映像，该映像用于在 AWS Batch 作业中进行备份。备份过程完成后，AWS Batch 会将备份详细信息记录至 Amazon DynamoDB 的库存表。作为一项额外的保护措施，如果任务在 AWS Batch 中失败， CloudWatch 事件事件会启动亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 通知。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 现有托管或非托管计算环境。有关更多信息，请参阅 AWS Batch 文档中的托管和非托管计算环境。
- AWS 命令行界面（CLI）版本 2 Docker 映像，已安装并配置。
- 现有的 Amazon RDS for PostgreSQL 数据库实例。
- 现有的 S3 存储桶。
- Docker，已在 Linux、macOS 或 Windows 上安装并配置。

- 熟悉 Lambda 编码。

架构

技术堆栈

- 亚马逊 CloudWatch 活动
- Amazon DynamoDB
- Amazon Elastic Container Registry (Amazon ECR)
- Amazon RDS
- Amazon SNS
- Amazon S3
- Amazon Batch
- AWS Key Management Service（AWS KMS）
- AWS Lambda
- AWS Secrets Manager
- Docker

工具

- Amazon CloudWatch Ev CloudWatch ents — Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。
- Amazon DynamoDB - DynamoDB 是一种全托管 NoSQL 数据库服务，提供快速而可预测的性能，能够实现无缝扩展。
- Amazon ECR - Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可靠且可扩展的 AWS 托管容器映像注册表服务。
- Amazon RDS - Amazon Relational Database Service (Amazon RDS) 是一项 Web 服务，让用户能够在 Amazon Web Services Cloud 中轻松设置、操作和扩展关系数据库。
- Amazon SNS - Amazon Simple Notification Service（Amazon SNS）是一项托管服务，提供从发布者至订阅用户的消息传输。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项面向互联网的存储服务。
- AWS Batch – AWS Batch 可帮助您在 AWS Cloud上运行批量计算工作负载。

- AWS KMS - AWS Key Management Service（AWS KMS）是一项托管服务，可让您轻松创建和控制加密您的数据所用的加密密钥。
- AWS Lambda – AWS Lambda 是一项计算服务，可使您无需预置或管理服务器即可运行代码。
- AWS Secrets Manager - Secrets Manager 帮助您将代码中的硬编码凭证（包括密码）替换为对 Secrets Manager 的 API 调用，以便以编程方式检索密钥。
- Docker - Docker 有助于开发人员轻松打包、交付和运行任何应用程序，将其作为轻量级、便携且自给自足的容器。

您在 Amazon RDS 上的 PostgreSQL 数据库实例必须为其元数据应用标签。Lambda 函数搜索标签以识别应备份的数据库实例，这通常使用以下标签。

Tag	描述
BKP: 自动 = 激活 DBDump	将 Amazon RDS 数据库实例标识为备份候选实例。
bkp: = AutomatedBackupSecret <secret_name>	标识 Secrets Manager 密钥，其中包含 Amazon RDS 登录凭证。
bkp: 自动 S3Bucket = DBDump <s3_bucket_name>	标识要向其发送备份的目标 S3 存储桶。
bkpDBDump: 自动频率	确定数据库备份的频率和时间。
bkpDBDump: 自动时间	
bkp:pgdumpcommand = <pgdump_command>	标识需要对其执行备份的数据库。

操作说明

在 DynamoDB 中创建清单表

Task	描述	所需技能
在 DynamoDB 中创建表。	登录 AWS 管理控制台，打开 Amazon DynamoDB 控制台，然后创建表。要获取有关此	云管理员、数据库管理员

Task	描述	所需技能
	操作和其他操作的帮助，请参阅相关资源部分。	
确认已创建表格。	运行 `aws dynamodb describe-table --table-name <table-name> | grep TableStatus` 命令。如果该表存在，则该命令将返回 `"TableStatus": "ACTIVE",` 结果。	云管理员、数据库管理员

在 AWS Batch 中为失败作业事件创建 SNS 主题

Task	描述	所需技能
创建 SNS 主题。	打开 Amazon SNS 控制台，选择主题，然后创建名为 `JobFailedAlert` 的 SNS 主题。以有效电子邮箱地址订阅此主题，并查看您的电子邮件收件箱以确认来自 AWS Notifications 的 SNS 订阅电子邮件。	云管理员
为 AWS Batch 创建失败作业事件规则。	打开 Amazon CloudWatch 控制台，选择事件，然后选择创建规则。选择显示高级选项，并选择编辑。对于构建模式，选择按您的目标处理的事件，将现有文本替换为其他信息部分的“失败作业事件”代码。此代码定义了在 A CloudWatch WS Batch 有`Failed`事件时启动的事件规则。	云管理员

Task	描述	所需技能
添加事件规则目标。	在目标中，选择添加目标，然后选择 `JobFailedAlert` SNS 主题。配置其余详细信息，并创建 Cloudwatch Events 规则。	云管理员

推送 Docker 映像到 Amazon ECR 存储库

Task	描述	所需技能
创建 Amazon ECR 存储库。	打开 Amazon ECR 控制台，选择要在其中创建存储库的 AWS 区域。选择存储库，然后选择创建存储库。根据要求配置存储库。	云管理员
撰写 Dockerfile。	登录 Docker，使用其他信息部分中的“示例 Dockerfile”和“样本 entrypoint.sh 文件”构建 Dockerfile。	DevOps 工程师
创建 Docker 映像并将其推送到 Amazon ECR 存储库。	将 Dockerfile 构建至 Docker 映像，并将其推送至 Amazon ECR 存储库。有关此步骤的帮助，请参阅相关资源部分。	DevOps 工程师

创建 AWS Batch 组件

Task	描述	所需技能
创建 AWS Batch 作业定义。	打开 AWS Batch 控制台，创建作业定义，其将 Amazon ECR 存储库的 Uniform Resource	云管理员

Task	描述	所需技能
	Identifier (URI) 包含为 `Image` 属性。	
配置 AWS Batch 作业队列。	在 AWS Batch 控制台，选择作业队列，然后选择创建队列。创建作业存储队列，直至 AWS Batch 在您的计算环境资源中运行这些任务。重要提示：请务必为 AWS Batch 编写逻辑，以将备份详细信息记录至 DynamoDB 清单表。	云管理员

创建并计划 Lambda 函数

Task	描述	所需技能
创建 Lambda 函数以搜索标签。	创建 Lambda 函数，用于在您的 PostgreSQL 数据库实例上搜索标签并识别备用备份。确保您的 Lambda 函数可以识别 `bkp:AutomatedDBDump = Active` 标签和所有其他必需的标签。重要提示：Lambda 函数还必须能够将任务添加至 AWS Batch 作业队列中。	DevOps 工程师
创建基于时间 CloudWatch 的事件事件。	打开 Amazon CloudWatch 控制台并创建一个 CloudWatch 事件事件，该事件使用 cron 表达式定期运行您的 Lambda 函数。重要提示：所有计划的事件都使用 UTC 时区。	云管理员

测试备份自动化

Task	描述	所需技能
创建 Amazon KMS 密钥。	打开 Amazon KMS 控制台并创建 KMS 密钥，该密钥可用于加密存储在 AWS Secrets Manager 中的 Amazon RDS 证书。	云管理员
创建 AWS Secrets Manager 密钥。	打开 AWS Secrets Manager 控制台，将您的 Amazon RDS for PostgreSQL 数据库证书存储为机密。	云管理员
向 PostgreSQL 数据库实例添加所需标签。	**Important** 打开 Amazon RDS 控制台，为要自动备份的 PostgreSQL 数据库实例添加标签。您可以使用工具部分表格中的标签。如果您需要从同一 Amazon RDS 实例中的多个 PostgreSQL 数据库进行备份，请`-d test:-d test1`使用作为标签的值。`bkp:pgdumpcommand` `test`和`test1`是数据库名称。确保冒号 (:) 后没有空格。	云管理员
验证备份自动化。	若要验证备份自动化，您可以调用 Lambda 函数或等待备份计划开始。备份过程完成后，	云管理员

Task	描述	所需技能
	请检查 DynamoDB 清单表中是否包含适用于 PostgreSQL 数据库实例的有效备份条目。如果其匹配，则表示备份自动化过程成功。	

相关资源

在 DynamoDB 中创建清单表

- 创建 Amazon DynamoDB 表

在 AWS Batch 中为失败作业事件创建 SNS 主题

- 创建 Amazon SNS 主题
- 在 AWS Batch 中针对失败作业事件发送 SNS 提醒

推送 Docker 映像到 Amazon ECR 存储库

- 创建 Amazon ECR 存储库
- 编写 Dockerfile，创建 Docker 映像，然后将其推送至 Amazon ECR

创建 AWS Batch 组件

- 创建 AWS Batch 作业定义
- 配置您的计算环境和 AWS Batch 作业队列
- 在 AWS Batch 中创建作业队列

创建 Lambda 函数

- 创建 Lambda 函数和编写代码
- 将 Lambda 与 DynamoDB 共同使用

创建 CloudWatch 活动事件

- 创建基于时间 CloudWatch 的事件事件
- 在 Cloudwatch 事件中采用 cron 表达式

测试备份自动化

- 创建 Amazon KMS 密钥
- 创建 Secrets Manager 密钥
- 为 Amazon RDS 实例添加标签

其他信息

失败作业事件：

```
{
  "detail-type": [
    "Batch Job State Change"
  ],
  "source": [
    "aws.batch"
  ],
  "detail": {
    "status": [
      "FAILED"
    ]
  }
}
```

示例 Dockerfile：

```
FROM alpine:latest
RUN apk --update add py-pip postgresql-client jq bash && \
pip install awscli && \
rm -rf /var/cache/apk/*
ADD entrypoint.sh /usr/bin/
RUN chmod +x /usr/bin/entrypoint.sh
ENTRYPOINT ["entrypoint.sh"]
```

entrypoint.sh 文件示例：

```
 #!/bin/bash
set -e
DATETIME=`date +"%Y-%m-%d_%H_%M"`
FILENAME=RDS_PostGres_dump_${RDS_INSTANCE_NAME}
FILE=${FILENAME}_${DATETIME}

aws configure --profile new-profile set role_arn arn:aws:iam::${TargetAccountId}:role/
${TargetAccountRoleName}
aws configure --profile new-profile set credential_source EcsContainer

echo "Central Account access provider IAM role is: "
aws sts get-caller-identity

echo "Target Customer Account access provider IAM role is: "
aws sts get-caller-identity --profile new-profile

securestring=$(aws secretsmanager get-secret-value --secret-id $SECRETID --output json
 --query 'SecretString' --region=$REGION --profile new-profile)

if [[ ${securestring} ]]; then
    echo "successfully accessed secrets manager and got the credentials"
    export PGPASSWORD=$(echo $securestring | jq --raw-output | jq -r '.DB_PASSWORD')
    PGSQL_USER=$(echo $securestring | jq --raw-output | jq -r '.DB_USERNAME')
    echo "Executing pg_dump for the PostGres endpoint ${PGSQL_HOST}"
    # pg_dump -h $PGSQL_HOST -U $PGSQL_USER -n dms_sample | gzip -9 -c  | aws s3 cp -
 --region=$REGION  --profile new-profile s3://$BUCKET/$FILE
    # in="-n public:-n private"
    IFS=':' list=($EXECUTE_COMMAND);
    for command in "${list[@]}";
      do
        echo $command;
        pg_dump -h $PGSQL_HOST -U $PGSQL_USER ${command} | gzip -9 -c  | aws s3 cp - --
region=$REGION --profile new-profile s3://${BUCKET}/${FILE}-${command}".sql.gz"
```

```
        echo $?;
        if  [[ $? -ne 0 ]]; then
            echo "Error occurred in database backup process. Exiting now....."
            exit 1
        else
            echo "Postgresql dump was successfully taken for the RDS endpoint
 ${PGSQL_HOST} and is uploaded to the following S3 location s3://${BUCKET}/${FILE}-
${command}.sql.gz"
            #write the details into the inventory table in central account
            echo "Writing to DynamoDB inventory table"
            aws dynamodb put-item --table-name ${RDS_POSTGRES_DUMP_INVENTORY_TABLE} --
region=$REGION --item '{ "accountId": { "S": "'"${TargetAccountId}"'" }, "dumpFileUrl":
 {"S": "'"s3://${BUCKET}/${FILE}-${command}.sql.gz"'" }, "DumpAvailableTime": {"S":
 "'"`date +"%Y-%m-%d::%H::%M::%S"` UTC"'"}}'
            echo $?
            if  [[ $? -ne 0 ]]; then
                echo "Error occurred while putting item to DynamoDb Inventory Table.
 Exiting now....."
                exit 1
            else
                echo "Successfully written to DynamoDb Inventory Table
 ${RDS_POSTGRES_DUMP_INVENTORY_TABLE}"
            fi
        fi
      done;
else
    echo "Something went wrong {$?}"
    exit 1
fi

exec "$@"
```

使用 CI/CD 管道在 Amazon EKS 中自动部署 Node Termination Handler

由 Sandip Gangapadhyay (AWS)、John Vargas (AWS)、Pragtideep Singh (AWS)、Sandeep Gawande (AWS) 和 Viyoma Sachdeva (AWS) 编写

摘要

注意： CodeCommit AWS 不再向新客户开放。AWS 的现有客户 CodeCommit 可以继续照常使用该服务。了解更多

在亚马逊网络服务 (AWS) 云上，您可以使用开源项目 AWS 节点终止处理程序来优雅地处理 Kubernetes 中的亚马逊弹性计算云 (Amazon EC2) 实例关闭。AWS 节点终止处理程序有助于确保 Kubernetes 控制平面对可能导致您的 EC2 实例不可用的事件做出适当的响应。这些事件包括：

- EC2 实例定期维护
- Amazon EC2 竞价型实例中断
- 自动扩缩组横向缩减
- 自动扩缩组在可用区域间再平衡
- EC2 通过 API 或 AWS 管理控制台终止实例

如果未处理事件，您的应用程序代码可能无法正常停止。恢复完全可用性也可能需要更长的时间，或者可能会意外地将工作安排到正在关闭的节点上。`aws-node-termination-handler`（NTH）可以在两种不同的模式下运行：实例元数据服务（IMDS）或队列处理器。有关这两种模式的更多信息，请参阅自述文件。

这种模式使用 AWS CodeCommit，它通过持续集成和持续交付 (CI/CD) 管道使用队列处理器自动部署 NTH。

> **Note**
>
> 如果您使用的是 EKS 托管节点组，则不需要`aws-node-termination-handler`。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 支持与 AWS 管理控制台 配合使用的 Web 浏览器。参阅支持的浏览器列表。

- AWS Cloud Development Kit (AWS CDK) 已安装。
- `kubectl`，Kubernetes 命令行工具，已安装。
- `eksctl`，安装了适用于 Amazon Elastic Kubernetes Service (Amazon EKS) 的 AWS 命令行界面（AWS CLI）。
- 运行的 EKS 集群，版本 1.20 或以上。
- 连接至 EKS 集群的自托管式节点组。运行以下命令创建具有自托管式节点组的Amazon EKS集群。

```
eksctl create cluster --managed=false --region <region> --name <cluster_name>
```

  有关 `eksctl` 的更多信息，请参阅 eksctl 文档。
- 适用于您的集群的 AWS Identity and Access Management (IAM) OpenID Connect (OIDC) 提供程序。有关更多信息，请参阅为您的集群创建 IAM OIDC 提供程序。

限制

- 您必须使用支持 Amazon EKS 服务的 AWS 区域。

产品版本

- Kubernetes 版本 1.20 或更高版本
- `eksctl` 版本 0.107.0 或更高版本
- AWS CDK 版本 2.27.0 或更高版本

架构

目标技术堆栈

- 虚拟私有云（VPC）
- EKS 集群
- Amazon Simple Queue Service(Amazon SQS)
- IAM
- Kubernetes

目标架构

下图显示了节点终止开始时 end-to-end步骤的高级视图。

图中显示的工作流包含以下概要步骤：

1. 自动扩展 EC2 实例终止事件被发送到 SQS 队列。
2. NTH 容器组（pod）监控 SQS 队列中的新消息。
3. NTH 容器组（pod）收到新消息并执行以下操作：
   - 封锁节点，这样新的容器组（pod）就不会在该节点上运行。
   - 排空节点，以便撤出现有容器组（pod）
   - 向自动扩缩组发送生命周期挂钩信号，以便可以终止该节点。

自动化和扩缩

- 代码由 AWS CDK 管理和部署，由 AWS CloudFormation 嵌套堆栈提供支持。
- Amazon EKS 控制面板跨多个可用区运行以确保高可用性。
- 为了实现自动扩缩，Amazon EKS 支持 Kubernetes 集群自动扩缩程序和 Karpenter。

工具

Amazon Web Services

- AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，可帮助您在代码中定义和预调配 Amazon Web Services Cloud 基础设施。
- AWS CodeBuild 是一项完全托管的构建服务，可帮助您编译源代码、运行单元测试和生成可随时部署的项目。
- AWS CodeCommit 是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。
- AWS CodePipeline 可帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件变更所需的步骤。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可帮助您在 AWS 上运行 Kubernetes，而无需安装或维护您自己的 Kubernetes 控制面板或节点。
- Amazon A EC2 uto Scaling 可帮助您保持应用程序的可用性，并允许您根据自己定义的条件自动添加或删除亚马逊 EC2 实例。

- Amazon Simple Queue Service (Amazon SQS) 提供了一个安全、持久且可用的托管队列，它可帮助您集成和分离分布式软件系统与组件。

其他工具

- kubectl 是针对 Kubernetes 集群运行命令的Kubernetes命令行工具。您可使用 kubectl 部署应用程序、检查和管理集群资源以及查看日志。

代码

此模式的代码可在 GitHub .com 的deploy-nth-to-eks存储库中找到。代码库包含以下文件和文件夹。

- `nth folder`— Helm 图表、值文件以及用于扫描和部署节点终止处理程序的 AWS CloudFormation 模板的脚本。
- `config/config.json` — 应用程序的配置参数文件。此文件包含部署 CDK 所需所有参数。
- `cdk` — AWS CDK 源代码。
- `setup.sh` — 用于部署 AWS CDK 应用程序以创建所需 CI/CD 管道和其他所需资源的脚本。
- `uninstall.sh` — 用于清理资源的脚本。

要使用示例代码，请按照操作说明部分中的说明执行操作。

最佳实践

有关自动化 AWS Node Termination Handler 的最佳实践，请参见以下内容：

- EKS 最佳实践指南
- Node Termination Handler - 配置

操作说明

设置您的环境

Task	描述	所需技能
克隆存储库。	要使用 SSH（Secure Shell）克隆存储库，请运行以下命令。	应用程序开发人员、AWS DevOps、 DevOps 工程师

Task	描述	所需技能

	`git clone git@github.com:aws-samples/deploy-nth-to-eks.git`
	
	要使用HTTPS克隆存储库，请运行以下命令。
	
	`git clone https://github.com/aws-samples/deploy-nth-to-eks.git`
	
	克隆存储库会创建一个名为 `deploy-nth-to-eks` 的文件夹。
	
	切换到该目录。
	
	`cd deploy-nth-to-eks`	

设置 kubeconfig 文件。	在您的终端设置您的 AWS 凭证，并确认您有权担任集群角色。您可使用以下示例代码。 `aws eks update-kubeconfig --name <Cluster_Name> --region <region>--role-arn <Role_ARN>`	AWS DevOps， DevOps 工程师，应用程序开发人员

部署 CI/CD 管道

Task	描述	所需技能
设置参数。	在 `config/config.json` 文件中，设置以下必需参数。 • `pipelineName` ：要由 AWS CDK 创建的 CI/CD 管道的名称（例如 `deploy-nth-to-eks-pipeline` ）。AWS CodePipeline 将创建一个具有此名称的管道。 • `repositoryName` ：要创建的 CodeCommit AWS 存储库（例如，`deploy-nth-to-eks-repo` ）。AWS CDK 将创建此存储库并将其设置为 CI/CD 管道来源。 **Note** 此解决方案将创建此 CodeCommit 存储库和分支（在以下 branch 参数中提供）。 • `branch`：存储库中的分支名称（例如 `main`）。对该分支的提交将启动 CI/CD 管道。 • `cfn_scan_script` ：将用于扫描 AWS CloudFormation 模板以获取 NTH (`scan.sh`) 的脚本路径。	应用程序开发人员、AWS DevOps、 DevOps 工程师

Task	描述	所需技能
	此脚本存在于将成为 AWS CodeCommit 存储库一部分的nth文件夹中。 • cfn_deploy_scrip t: 将用于部署 NTH 的 AWS CloudFormation 模板的脚本路径 (installApp.sh)。 • stackName ：要部署的 CloudFormation 堆栈的名称。 • eksClusterName –现有 EKS 集群的名称。 • eksClusterRole ：用于访问所有 Kubernetes API 调用的 EKS 集群的 IAM 角色（例如 clusterad min ）。通常，此角色在 aws-auth ConfigMap 中添加。 • create_cluster_rol e ：若要创建 eksCluste rRole IAM 角色，请输入是。如果要在 eksCluste rRole 参数中提供现有的集群角色，请输入否。 • create_iam_oidc_pr ovider ：要为集群创建 IAM OIDC 提供程序，请输入是。如果 IAM OIDC 提供程序已经存在，请输入否。有关更多信息，请参阅为您的集群创建 IAM OIDC 提供程序。	

Task	描述	所需技能
	• `AsgGroupName` ：属于 EKS 集群的自动扩缩组名称的逗号分隔列表（例如 `ASG_Group_1,ASG_Group_2` ）。 • `region`：集群所在的 AWS 区域的名称（例如 `us-east-2` ）。 • `install_cdk` ：如果计算机上当前未安装 AWS CDK，请输入是。运行 `cdk --version` 命令检查安装的 AWS CDK 版本是否为 2.27.0 或更高版本。在这种情况下，请输入否。 如果您输入是，setup.sh 脚本将运行在计算机上安装 AWS CDK `sudo npm install -g cdk@2.27.0` 命令。该脚本需要 sudo 权限，因此请在出现提示时提供账户密码。	

Task	描述	所需技能
创建 CI/CD 管道，以部署 NTH。	运行setup.sh脚本。 `./setup.sh` 该脚本将部署 AWS CDK 应用程序，该应用程序将根据文件中的`config/config.json` 用户输入参数使用示例代码、管道和 CodeBuild 项目创建 CodeCommit 存储库。 该脚本在使用 sudo 命令安装 npm 软件包时将要求输入密码。	应用程序开发人员、AWS DevOps、 DevOps 工程师

Task	描述	所需技能
查看 CI/CD 管道。	打开 AWS 管理控制台，查看在堆栈中创建的以下资源。 • CodeCommit 包含文件夹内容的 repo `nth` • AWS CodeBuild 项目`cfn-scan`，它将扫描 CloudFormation 模板中是否存在漏洞。 • CodeBuild 项目`Nth-Deploy` ，它将通过 AWS CodePipeline 管道部署 AWS CloudFormation 模板和相应的 NTH Helm 图表。 • 用于部署 NTH 的 CodePipeline 管道。 管道成功运行后，Helm 版本 `aws-node-termination-handler` 将安装在 EKS 集群中。此外，名为 `aws-node-termination-handler` 的容器组（pod）正在集群的 `kube-system` 命名空间中运行。	应用程序开发人员、AWS DevOps、 DevOps 工程师

测试NTH部署

Task	描述	所需技能
模拟自动扩缩组横向缩减事件。	要模拟自动扩缩横向缩减事件，请执行以下操作：	

Task	描述	所需技能
	1. 在 AWS 控制台上，打开控制EC2 台，然后选择 Auto Scaling Groups。 2. 选择与 `config/config.json` 中提供的组同名的自动扩缩组，然后选择编辑。 3. 将所需容量和最小容量减少 1。 4. 选择更新。	
查看日志。	在缩容事件期间，第 N 个 Pod 将封锁并耗尽相应的工作节点（作为缩容事件的一部分终止的 EC2 实例）。要查看日志，请使用其他信息部分中的代码。	应用程序开发人员、AWS DevOps、 DevOps 工程师

清理

Task	描述	所需技能
清理全部 AWS 资源。	锐欧要清理此模式创建的资源，请运行以下命令。 `./uninstall.sh` 这将通过删除 CloudFormation 堆栈来清理在此模式中创建的所有资源。	DevOps 工程师

故障排除

事务	解决方案

npm 注册表设置不正确。	在此解决方案的安装过程中，脚本会安装 npm install 以下载所有必需的软件包。如果在安装过程中看到找不到模块的消息，则可能无法正确设置 npm 注册表。要查看当前的注册表设置，请运行以下命令。
	`npm config get registry`
	运行以下命令以设置 `https://registry.npmjs.org/` 注册表。
	`npm config set registry https://registry.npmjs.org`

延迟 SQS 消息传送。	作为故障排除的一部分，如果您想延迟向 NTH 容器组（pod）传送 SQS 消息，可以调整 SQS 传送延迟参数。有关更多信息，请参阅 Amazon SQS 延迟队列。

相关资源

- AWS Node Termination Handler 源代码
- EC2 工作坊
- AWS CodePipeline
- Amazon Elastic Kubernetes Service(Amazon EKS)
- AWS Cloud Development Kit
- AWS CloudFormation

其他信息

1. 找到 NTH 容器组（pod）的名字。

```
kubectl get pods -n kube-system |grep aws-node-termination-handler
aws-node-termination-handler-65445555-kbqc7   1/1     Running   0          26m
kubectl get pods -n kube-system |grep aws-node-termination-handler
aws-node-termination-handler-65445555-kbqc7   1/1     Running   0          26m
```

2. 检查日志。日志示例如下所示。它表明在发送自动扩缩组生命周期挂钩完成信号之前，该节点已被封锁并耗尽。

```
kubectl -n kube-system logs aws-node-termination-handler-65445555-kbqc7
022/07/17 20:20:43 INF Adding new event to the event store
 event={"AutoScalingGroupName":"eksctl-my-cluster-target-nodegroup-
ng-10d99c89-NodeGroup-ZME36IGAP7O1","Description":"ASG Lifecycle Termination
 event received. Instance will be interrupted at 2022-07-17 20:20:42.702
 +0000 UTC \n","EndTime":"0001-01-01T00:00:00Z","EventID":"asg-lifecycle-
term-33383831316538382d353564362d343332362d613931352d383430666165636334333564","InProgress":fal
east-2.compute.internal","NodeProcessed":false,"Pods":null,"ProviderID":"aws:///us-
east-2c/i-0409f2a9d3085b80e","StartTime":"2022-07-17T20:20:42.702Z","State":""}
2022/07/17 20:20:44 INF Requesting instance drain event-id=asg-lifecycle-
term-33383831316538382d353564362d343332362d613931352d383430666165636334333564
 instance-id=i-0409f2a9d3085b80e kind=SQS_TERMINATE node-name=ip-192-168-75-60.us-
east-2.compute.internal provider-id=aws:///us-east-2c/i-0409f2a9d3085b80e
2022/07/17 20:20:44 INF Pods on node node_name=ip-192-168-75-60.us-
east-2.compute.internal pod_names=["aws-node-qchsw","aws-node-termination-
handler-65445555-kbqc7","kube-proxy-mz5x5"]
2022/07/17 20:20:44 INF Draining the node
2022/07/17 20:20:44 ??? WARNING: ignoring DaemonSet-managed Pods: kube-system/aws-node-
qchsw, kube-system/kube-proxy-mz5x5
2022/07/17 20:20:44 INF Node successfully cordoned and drained
 node_name=ip-192-168-75-60.us-east-2.compute.internal reason="ASG Lifecycle
 Termination event received. Instance will be interrupted at 2022-07-17 20:20:42.702
 +0000 UTC \n"
2022/07/17 20:20:44 INF Completed ASG Lifecycle Hook (NTH-K8S-TERM-HOOK) for instance
 i-0409f2a9d3085b80e
```

使用 CI/CD 管道自动构建 Java 应用程序并将其部署到 Amazon EKS

创建者：MAHESH RAGHUNANDANAN（AWS）、James Radtke（AWS）和 Jomcy Pappachen（AWS）

摘要

此模式描述了如何创建持续集成和持续交付 (CI/CD) 管道，该管道可根据推荐的 DevSecOps 做法自动构建并部署到上的 Amazon Elastic Kubernetes Service (Amazon EKS) 集群。 AWS Cloud此模式使用采用 Spring Boot Java 框架开发的问候应用程序，该应用程序使用 Apache Maven。

您可以使用这种模式的方法为 Java 应用程序构建代码，将应用程序构件打包为 Docker 映像，对映像进行安全扫描，然后将该映像作为工作负载容器上传到 Amazon EKS 上。如果您想从紧密耦合的单片架构迁移到微服务架构，则此模式的方法非常有用。该方法还可以帮助您监控和管理 Java 应用程序的整个生命周期，从而确保更高的自动化水平并有助于避免错误或程序错误。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- AWS Command Line Interface (AWS CLI) 版本 2，已安装并配置。有关这方面的更多信息，请参阅 AWS CLI 文档 AWS CLI中的安装或更新到最新版本的。

  AWS CLI 版本 2 必须使用与创建 Amazon EKS 集群的相同 AWS Identity and Access Management (IAM) 角色进行配置，因为只有该角色才有权向中添加其他 IAM 角色`aws-authConfigMap`。有关配置的信息和步骤 AWS CLI，请参阅 AWS CLI 文档中的配置设置。
- 具有完全访问权限的 IAM 角色和权限 AWS CloudFormation。有关这方面的更多信息，请参阅 AWS CloudFormation 文档中的使用 IAM 控制访问权限。
- 现有的 Amazon EKS 集群，包含 EKS 集群中工作节点的 IAM 角色名称和 IAM 角色的亚马逊资源名称 (ARN) 的详细信息。
- Kubernetes 集群自动扩缩器，已在 Amazon EKS 集群中安装和配置。有关更多信息，请参阅 Amazon EKS 文档中的使用 Karpenter 扩展集群计算和集群自动扩缩器。
- 访问 GitHub 存储库中的代码。

> **Important**
>
> AWS Security Hub 已作为此模式代码中包含的 AWS CloudFormation 模板的一部分启用。默认情况下，启用 Security Hub 后，它会提供 30 天的免费试用期。试用结束后，会产生与此相关的费用 AWS 服务。有关定价的更多信息，请参阅 AWS Security Hub 定价。

产品版本

- Helm 版本 3.4.2 或更高版本
- Apache Maven 版本 3.6.3 或更高版本
- BridgeCrew Checkov 版本 2.2 或更高版本
- Aqua Security Trivy 版本 0.37 或更高版本

架构

技术堆栈

- AWS CodeBuild
- AWS CodeCommit

  注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多。但是，此解决方案适用于任何版本控制系统 (VCS) Git 提供程序，例如 GitHub 或只需进行最少 GitLab 的更改。
- 亚马逊 CodeGuru
- AWS CodePipeline
- Amazon Elastic Container Registry (Amazon ECR)
- Amazon EKS
- 亚马逊 EventBridge
- AWS Security Hub
- Amazon Simple Notification Service (Amazon SNS)

目标架构

图表显示了以下工作流：

1. 开发人员更新 CodeCommit 存储库基础分支中的 Java 应用程序代码，从而创建拉取请求 (PR)。
2. 提交 PR 后，Amazon CodeGuru Reviewer 会自动审查代码，根据 Java 的最佳实践对其进行分析，并向开发者提供建议。
3. PR 合并到基础分支后，将创建一个 Amazon EventBridge 事件。
4. 该 EventBridge 事件启动 CodePipeline 管道，管道启动。
5. CodePipeline 运行 CodeSecurity 扫描阶段（持续安全）。
6. AWS CodeBuild 启动安全扫描流程，在该流程中，使用 Checkov 扫描 Dockerfile 和 Kubernetes 部署 Helm 文件，并根据增量代码更改扫描应用程序源代码。应用程序源代码扫描由 CodeGuru Reviewer 命令行界面 (CLI) 包装器执行。
7. 如果安全扫描阶段成功，则启动构建阶段（持续集成）。
8. 在构建阶段， CodeBuild 构建工件，将构件打包到 Docker 镜像，使用 Aqua Security Trivy 扫描映像中是否存在安全漏洞，然后将映像存储在 Amazon ECR 中。
9. 步骤 8 中检测到的漏洞将上传到 Security Hub，供开发人员或工程师进一步分析。Security Hub 提供了修复漏洞的概述和建议。
10. CodePipeline 管道中连续阶段的电子邮件通知通过 Amazon SNS 发送。
11. 持续集成阶段完成后， CodePipeline 进入部署阶段（持续交付）。
12. 使用 Helm 图表将 Docker 映像作为容器工作负载（容器组（pod））部署到 Amazon EKS。
13. 应用程序容器使用 Amazon P CodeGuru rofiler 代理进行配置，该代理会将应用程序的分析数据（CPU、堆使用情况和延迟）发送到 P CodeGuru rofiler，从而帮助开发人员了解应用程序的行为。

工具

AWS 服务

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在资源的整个生命周期中跨地区对其 AWS 账户 进行管理。
- AWS CodeBuild是一项完全托管的生成服务，可帮助您编译源代码、运行单元测试和生成可随时部署的工件。
- AWS CodeCommit是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。

  注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

- Amazon CodeGuru Profiler 会从您的实时应用程序收集运行时性能数据，并提供建议，以帮助您微调应用程序性能。
- Amazon CodeGuru Reviewer 使用程序分析和机器学习来检测开发人员难以发现的潜在缺陷，并提供改进您的 Java 和 Python 代码的建议。
- AWS CodePipeline帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件更改所需的步骤。
- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- 亚马逊 Elastic Kubernetes Service（亚马逊 EKS）可帮助你在上面运行 AWS Kubernetes，而无需安装或维护自己的 Kubernetes 控制平面或节点。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来，包括 AWS Lambda 函数、使用 API 目标的 HTTP 调用终端节点或其他来源的事件总线。 AWS 账户
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Security Hub提供了您的安全状态的全面视图 AWS。它还可以帮助您根据安全行业标准和最佳实践检查您的 AWS 环境。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

其他服务

- Helm 是 Kubernetes 的开源软件包管理器。
- Apache Maven 是一款软件项目管理及理解工具。
- BridgeCrew Checkov 是一种静态代码分析工具，用于扫描基础设施即代码 (IaC) 文件，以查找可能导致安全性或合规性问题的错误配置。
- Aqua Security Trivy 是一款全面的扫描工具，可检测容器映像、文件系统和 Git 存储库中的漏洞以及配置问题。

代码

此模式的代码可在 GitHub aws-codepipeline-devsecops-amazoneks存储库中找到。

最佳实践

- 此模式遵循 IAM 安全最佳实践，在解决方案的所有阶段都对 IAM 实体适用最低权限原则。如果您想使用其他工具 AWS 服务 或第三方工具扩展解决方案，我们建议您查看 IAM 文档中有关应用最低权限的部分。
- 如果您有多个 Java 应用程序，我们建议您为每个应用程序创建单独的 CI/CD 管道。
- 如果您有单体应用程序，我们建议您尽可能将应用程序分解为微服务。微服务更加灵活，可以更轻松地将应用程序部署为容器，并且可以更好地了解应用程序的整体构建和部署。

操作说明

设置环境

Task	描述	所需技能
克隆 GitHub 存储库。	要克隆存储库，请运行以下命令。 `git clone https://github.com/aws-samples/aws-codepipeline-devsecops-amazoneks`	应用程序开发者、 DevOps 工程师
创建 S3 存储桶并上传代码。	1. 登录 AWS Management Console，打开 Amazon S3 控制台，然后在您计划部署此解决方案的 AWS 区域 位置创建 S3 存储桶。有关更多信息，请参阅 Amazon S3 文档中的创建存储桶。 2. 在 S3 存储桶中，创建一个名为 `code` 的文件夹。 3. 导航到您克隆存储库的位置。要创建扩展名为.zip (`cicdstack.zip`) 的整个代码的压缩版本并验证.zip	AWS DevOps，云管理员，DevOps 工程师

Task	描述	所需技能

	文件，请按顺序运行以下命令。
	
	`cd aws-codepipeline-devsecops-amazoneks`
	`python -m zipfile -c cicdstack.zip *`
	`python -m zipfile -t cicdstack.zip`
	
	Note
	如果python命令失败并显示未找到 Python，请python3改用。
	
	4. 将`cicdstack.zip` 文件上传到您在上一步的 S3 存储桶中创建的代码文件夹。	

Task	描述	所需技能
创建 AWS CloudFormation 堆栈。	1. 打开 AWS CloudFormation 控制台，然后选择 Create stack (创建堆栈)。 2. 在指定模板中，选择上传模板文件，上传 `cf_templates/codecommit_ecr.yaml` 文件，然后选择下一步。 3. 在指定堆栈详细信息中，输入堆栈名称，然后提供以下输入参数值： • `CodeCommitRepositoryBranchName` ：您的代码所在分支的名称（默认为`main`） • `CodeCommitRepositoryName` ：CodeCommitrepository 要创建的名称 • `CodeCommitRepositoryS3Bucket` ：您在其中创建代码文件夹的 S3 存储桶的名称 • `CodeCommitRepositoryS3BucketObjKey` ：`code/cicdstack.zip` • `ECRRepositoryName` ：要创建的 Amazon ECR 存储库的名称 4. 选择下一步，使用配置堆栈选项的默认设置，然后选择下一步。	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	5. 在查看部分中，验证模板和堆栈的详细信息，然后选择创建堆栈。然后创建堆栈，包括 CodeCommit 和 Amazon ECR 存储库。 6. 记下 CodeCommit 和 Amazon ECR 存储库的名称，它们是设置 Java CI/CD 管道所必需的。	
验证 CloudFormation 堆栈部署。	1. 在 CloudFormation 控制台的 “堆栈” 下，验证您部署的 CloudFormation 堆栈的状态。堆栈的状态应为创建完成。 2. 在控制台中，验证 CloudFormation 和 Amazon ECR 存储库是否已配置并准备就绪。	AWS DevOps， DevOps 工程师
删除 S3 存储桶。	清空并删除您之前创建的 S3 存储桶。有关更多信息，请参阅 Amazon S3 文档中的删除存储桶。	AWS DevOps， DevOps 工程师

配置 Helm 图表

Task	描述	所需技能
配置 Java 应用程序的 Helm 图表。	1. 在您克隆 GitHub 存储库的位置，导航到该文件夹`helm_charts/aws-proserve-java-greeting` 。在此文件夹中，该	DevOps 工程师

Task	描述	所需技能

	`values.dev.yaml` 文件包含有关 Kubernetes 资源配置的信息，您可以针对将容器部署到 Amazon EKS 来修改这些配置。通过提供您的 AWS 账户 ID 和 Amazon ECR 存储库名称来更新 Docker 存储库参数。AWS 区域
	`image:`
	`  repository:`
	`  <account-id>.dkr.ecr.<region>.amazonaws.com/<app-ecr-repo-name>`
	2. Java 容器组（pod）的服务类型设置为 `LoadBalancer` 。
	`service:`
	`  type: LoadBalancer`
	`  port: 80`
	`  targetPort: 8080`
	`  path: /hello`
	`  initialDelaySeconds: 60`
	`  periodSeconds: 30`
	要使用其他服务（例如NodePort），可以更改此参数。有关更多信息，请参阅 Kubernetes 文档。
	3. 您可以通过将参数 `autoscaling` 更改为 `enabled: true` 来激活	

Task	描述	所需技能

	Kubernetes 水平容器组（pod）自动扩缩器。
	
	`autoscaling:`
	`  enabled: true`
	`  minReplicas: 1`
	`  maxReplicas: 100`
	`  targetCPUUtilizationPercentage: 80`
	`  # targetMemoryUtilizationPercentage: 80`
	
	4. 您可以通过更改`values.<ENV>.yaml` 文件中的值为 Kubernetes 工作负载启用不同的功能，您的开发、生产、UAT 或 QA 环境`<ENV>`位于何处。	

Task	描述	所需技能

验证 Helm 图表是否存在语法错误。	1. 在终端上，运行以下命令，验证 Helm v3 是否已安装在本地工作站中。
	`helm --version`
	如果未安装 Helm v3，请安装。
	2. 在终端中，导航到 Helm 图表目录（`helm_charts/aws-proserve-java-greeting` ），然后运行以下命令。
	`helm lint . -f values.dev.yaml`
	这将检查 Helm 图表中是否存在任何语法错误。
		DevOps 工程师

设置 Java CI/CD 管线

Task	描述	所需技能
创建 CI/CD 管线。	1. 打开 AWS CloudFormation 控制台，然后选择 Create stack (创建堆栈)。 2. 在指定模板中，选择上传模板文件，上传 `cf_templates/build_deployment.yaml` 模板，然后选择下一步。	AWS DevOps

Task	描述	所需技能
	3. 在指定堆栈详细信息中，指定堆栈名称，然后提供以下输入参数值： • `CodeBranchName` ：您的代码所在 CodeCommit 存储库的分支名称 • `EKSClusterName` ：您的 EKS 集群的名称（不是 `EKSCluster` ID） • `EKSCodeBuildAppName` ：应用程序名称 Helm 图表 (`aws-proserve-java-greeting`) • `EKSWorkerNodeRoleARN` ：分配给 Amazon EKS 工作节点的 IAM 角色的 ARN • `EKSWorkerNodeRoleName` ：分配给 Amazon EKS 工作节点的 IAM 角色的名称 • `EcrDockerRepository` ：用于存储代码的 Docker 镜像的 Amazon ECR 存储库的名称 • `EmailRecipient` ：应向其发送构建通知的电子邮件地址 • `EnvType`: 环境（例如，开发、测试或生产） • `SourceRepoName` ：您的代码所在 CodeCommit 存储库的名称	

Task	描述	所需技能
	4. 选择下一步。使用配置堆栈选项的默认设置，然后选择下一步。 5. 在 “查看” 部分中，验证 CloudFormation 模板和堆栈的详细信息，然后选择 “下一步”。 6. 选择创建堆栈。 7. 在 CloudFormation 堆栈部署期间，您在参数中提供的电子邮件地址的所有者将收到一条订阅 SNS 主题的消息。要订阅 Amazon SNS，所有者必须选择消息中的链接。 8. 创建堆栈后，打开堆栈的输出选项卡，然后记录 `EksCodeBuildkubeRoleARN` 输出密钥的 ARN 值。稍后，当您为 IAM 角色提供在 Amazon EKS 集群中部署工作负载的权限时，将需要此 CodeBuild IAM ARN 值。	

激活 Security Hub 和 Aqua Security 之间的集成

Task	描述	所需技能
打开 Aqua Security 集成。	要将 Trivy 报告的 Docker 映像漏洞调查发现上传到 Security Hub，必须执行此步骤。由于 AWS CloudFormation 不支持	AWS 管理员、 DevOps 工程师

Task	描述	所需技能
	Security Hub 集成，因此必须手动完成此过程。 1. 打开AWS Security Hub 控制台，然后导航到 “集成”。 2. 搜索 Aqua Security，然后选择 Aqua Security：Aqua Security。 3. 选择接受调查发现。	

配置 CodeBuild 为运行 Helm 或 kubectl 命令

Task	描述	所需技能
CodeBuild 允许在 Amazon EKS 集群中运行 Helm 或 kubectl 命令。	CodeBuild 要通过身份验证才能在 Amazon EKS 集群中使用 Helm 或`kubectl`命令，您必须将 IAM 角色添加到`aws-authConfigMap` 。在本例中，添加 IAM 角色的 ARN`EksCodeBuildkubeRoleARN` ，即为 CodeBuild 服务创建的 IAM 角色，用于访问 Amazon EKS 集群并在其上部署工作负载。这是一次性活动。 **Important** 在部署批准阶段之前，必须完成以下程序 CodePipeline。	DevOps

Task	描述	所需技能

	1. 在 Amazon Linux 或 macOS 环境中打开 `cf_templates/kube_aws_auth_configmap_patch.sh` Shell 脚本。
	2. 通过运行以下命令对 Amazon EKS 集群进行验证。
	`aws eks --region <aws-region> update-kubeconfig --name <eks-cluster-name>`
	3. 使用以下命令运行 Shell 脚本，将 `<rolearn-eks-codebuild-kubectl>` 替换为之前记录的 `EksCodeBuildkubeRoleARN` ARN 值。
	`bash cf_templates/kube_aws_auth_configmap_patch.sh <rolearn-eks-codebuild-kubectl>`
	已配置 `aws_auth ConfigMap` ，并授予访问权限。	

验证 CI/CD 管线

Task	描述	所需技能

验证 CI/CD 管线是否自动启动。	1. 如果 Checkov 在 Dockerfile 或 Helm 图表中检测到漏洞，则管道中的 CodeSecurity 扫描阶段通常会失败。但是，此示例的目的是建立一个识别潜在安全漏洞的过程，而不是通过 CI/CD 管道（通常是一个 DevSecOps 过程）修复漏洞。在文件 `buildspec/buildspec_secscan.yaml` 中，该 `checkov` 命令使用 `--soft-fail` 标志来避免管线故障。
	
	`- echo -e "\n Running Dockerfile Scan"`
	`- checkov -f code/app/Dockerfile --framework dockerfile --soft-fail --summary-position bottom`
	`- echo -e "\n Running Scan of Helm Chart files"`
	`- cp -pv helm_charts/$EKS_CODEBUILD_APP_NAME/values.dev.yaml helm_charts/$EKS_CODEBUILD_APP_NAME/values.yaml`
	`- checkov -d helm_charts/$EKS_CODEBUILD_APP_NAME --framework helm --soft-fail`
		DevOps

Task	描述	所需技能
	--summary-position bottom - rm -rfv helm_charts/$EKS_CODEBUILD_APP_NAME/values.yaml	
	要使管线在报告 Dockerfile 和 Helm 图表的漏洞时失败，必须从 checkov 命令中删除该 --soft-fail 选项。然后，开发人员或工程师可以修复漏洞并将更改提交到 CodeCommit 源代码存储库。	
	2. 与 CodeSecurity 扫描类似，在将应用程序推送到 Amazon ECR 之前，构建阶段使用 Aqua S HIGH ecurity Trivy 来识别 CRITICAL Docker 镜像漏洞。 - AWS_REGION=$AWS_DEFAULT_REGION AWS_ACCOUNT_ID=$AWS_ACCOUNT_ID trivy -d image --no-progress --ignore-unfixed --exit-code 0 --severity HIGH,CRITICAL --format template --template "@securityhub/asff.tpl" -o securityhub/report.asff $AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/$IMAG	

Task	描述	所需技能
	E_REPO_NAME:$CODEB UILD_RESOLVED_SOUR CE_VERSION 在此示例中，当报告 Docker 镜像漏洞时，管道不会失败，因为buildspec/buildspec.yml 文件中的trivy命令包含值为**--exit-code** 的0标志。要使管道在报告CRTICAL漏洞时HIGH出现故障，请将的值更改--exit-code 为1。然后，开发人员或工程师可以修复漏洞并将更改提交到 CodeCommit 源代码存储库。 3. Aqua Security Trivy 报告的 Docker 映像漏洞已上传到 Security Hub。在 Security Hub 控制台上，导航到调查结果。使用记录状态 = 活跃和产品 = Aqua Security来筛选调查发现。这列出了 Security Hub 中的 Docker 镜像漏洞。漏洞可能需要 15 分钟到一个小时才会出现在 Security Hub 中。 有关使用启动管道的更多信息 CodePipeline，请参阅 CodePipeline 文档中的启动管道CodePipeline、手动启动管道和按计划启动管道。	

Task	描述	所需技能
批准部署。	1. 构建阶段完成后，将有一个部署批准门。审阅者或发布经理应检查该版本，如果满足所有要求，则应批准该版本。对于使用持续交付进行应用程序部署的团队，推荐使用这种方法。 2. 批准后，管线将启动部署阶段。 3. 部署阶段成功后，此阶段的 CodeBuild 日志将提供应用程序的 URL。使用 URL 验证应用程序是否准备就绪。	DevOps
验证应用程序分析。	部署完成并将应用程序容器部署在 Amazon EKS 中后，在应用程序中配置的 Amazon P CodeGuru rofiler 代理将尝试将应用程序的分析数据（CPU、堆摘要、延迟和瓶颈）发送到 CodeGuru Profiler。 对于应用程序的初始部署，CodeGuru Profiler 大约需要 15 分钟来可视化分析数据。	AWS DevOps

相关资源

- AWS CodePipeline 文档
- 在 AWS CodePipeline（AWS 博客文章）中使用 Trivy 扫描图像
- 使用 Amazon P CodeGuru rofiler 改进您的 Java 应用程序（AWS 博客文章）
- AWS 安全调查结果格式 (ASFF) 语法
- 亚马逊 EventBridge 事件模式

- Helm 升级

其他信息

- AWS CodeCommit 不再向新客户提供。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多。此解决方案还适用于任何版本控制系统 (VCS) Git 提供程序，例如 GitHub 或只需进行最少 GitLab 的更改。
- CodeGuru 在功能方面，不应将 Profiler 与 AWS X-Ray 服务混淆。我们建议您使用 CodeGuru Profiler 来识别可能导致瓶颈或安全问题的最昂贵的代码行，并在它们成为潜在风险之前对其进行修复。X-Ray 服务用于监控应用程序性能。
- 在此模式中，事件规则与默认事件总线相关联。如果需要，您可以扩展模式以使用自定义事件总线。
- 此模式使用 CodeGuru Reviewer 作为应用程序代码的静态应用程序安全测试 (SAST) 工具。您也可以将此管道用于其他工具，例如 SonarQube 或 Checkmarx。您可以添加其中任何工具的扫描设置说明`buildspec/buildspec_secscan.yaml`来替换 CodeGuru 扫描指令。

使用 Amazon EFS 创建 Amazon ECS 任务定义并在 EC2 实例上挂载文件系统

由 Durga Prasad Cheepuri (AWS) 创建

摘要

此模式提供了创建亚马逊弹性容器服务 (Amazon ECS) 任务定义的代码示例和步骤，该任务定义在亚马逊网络服务 (AWS) 云的 EC2亚马逊弹性计算云 (Amazon) 实例上运行，同时使用亚马逊弹性文件系统 (Amazon EFS) 在这些实例上 EC2 挂载文件系统。使用 Amazon EFS 的 Amazon ECS 任务会自动挂载您在任务定义中指定的文件系统，并使这些文件系统可供 AWS 区域中所有可用区中的任务容器使用。

为了满足您的持久性存储和共享存储要求，您可以结合使用 Amazon ECS 和 Amazon EFS。例如，您可以使用 Amazon EFS 存储持久性用户数据和应用程序数据，并在不同的可用区中运行活动和备用 ECS 容器对以实现高可用性。您还可以使用 Amazon EFS 存储可由 ECS 容器和分布式作业工作负载并行访问的共享数据。

要将 Amazon EFS 与 Amazon ECS 结合使用，您可以将一个或多个卷定义添加到任务定义中。卷定义包括 Amazon EFS 文件系统 ID、接入点 ID 以及 AWS Identity and Access Management（AWS IAM）授权或传输层安全性协议(TLS)传输中加密的配置。您可以在任务定义中使用容器定义来指定容器运行时挂载的任务定义卷。当使用 Amazon EFS 文件系统的任务运行时，Amazon ECS 会确保文件系统已挂载并可供需要访问该文件系统的容器使用。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 具有虚拟专用网络(VPN) 端点或路由器的虚拟私有云(VPC)
- （推荐）Amazon ECS 容器代理 1.38.0 或更高版本，以与 Amazon EFS 接入点和 IAM 授权功能兼容（有关更多信息，请参阅 AWS Blog 文章 Amazon EFS 的新增功能 - IAM 授权和接入点）。

限制

- 1.35.0 之前的 Amazon ECS 容器代理版本不支持 Amazon EFS 文件系统执行使用 EC2 启动类型的任务。

架构

下图显示了一个使用 Amazon ECS 创建任务定义并在 ECS 容器中的 EC2 实例上挂载 Amazon EFS 文件系统的应用程序示例。

图表显示了以下工作流：

1. 创建 Amazon EFS 文件系统。
2. 创建带有容器的任务定义。
3. 配置容器实例以挂载 Amazon EFS 文件系统。任务定义引用卷挂载，因此容器实例可使用 Amazon EFS 文件系统。ECS 任务可以访问相同的 Amazon EFS 文件系统，无论这些任务是在哪个容器实例上创建的。
4. 创建具有三个任务定义实例的 Amazon ECS 服务。

技术堆栈

- Amazon EC2
- Amazon ECS
- Amazon EFS

工具

- 亚马逊 EC2 — 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要使用 Amazon EC2 启动任意数量或数量的虚拟服务器，也可以进行横向扩展或扩展。
- Amazon ECS - Amazon Elastic Container Service (Amazon ECS) 是一项高度可扩展的快速容器管理服务，可助您轻松运行、停止和管理集群上的容器。您可以在由 AWS Fargate 托管的无服务器基础设施上运行任务和服务。或者，为了更好地控制您的基础架构，您可以在自己管理的 EC2 实例集群上运行任务和服务。
- Amazon EFS - Amazon Elastic File System (Amazon EFS) 可提供简单、可扩展、完全托管的弹性 NFS 文件系统，以便与 Amazon Web Services Cloud 服务和本地资源配合使用。
- AWS CLI - AWS 命令行界面（AWS CLI）是一种开源工具，它使您能够使用命令行 Shell 中的命令与 Amazon Web Services 交互。仅需最少的配置，即可使用 AWS CLI 开始运行命令，以便从终端程序中的命令提示符实现与基于浏览器的 AWS 管理控制台所提供的功能等同的功能。

操作说明

创建 Amazon EFS 文件系统

Task	描述	所需技能
使用 AWS 管理控制台 创建 Amazon EFS 文件系统。	1. **Note** 创建 Amazon EFS 文件系统，然后选择包含容器的 VPC。：如果您使用其他 VPC，请设置 VPC 对等连接。 2. 请记下文件系统 ID 值。	AWS DevOps

使用 Amazon EFS 文件系统或 AWS CLI 创建 Amazon ECS 任务定义

Task	描述	所需技能
使用 Amazon EFS 文件系统创建任务定义。	使用具有以下配置的新 Amazon ECS 控制台或经典 Amazon ECS 控制台创建任务定义： • 如果您使用新的控制台，请为应用程序环境选择 Amazon EC2 实例。如果您使用经典控制台，请选择EC2作为启动类型。 • 添加卷。输入卷的名称，选择 EFS 作为卷类型，然后选择您之前记下的文件系统 ID。对于根目录，选择要在 Amazon ECS 容器主机上托	AWS DevOps

Task	描述	所需技能
	管的 Amazon EFS 文件系统路径。	

使用 AWS CLI 创建任务定义。	1. 若要为任务定义创建带有输入参数占位符的 JSON 模板，请运行以下命令：
	`aws ecs register-task-definition --generate-cli-skeleton`
	2. 若要使用 JSON 模板创建任务定义，请运行以下命令：
	`aws ecs register-task-definition --cli-input-json file://<path_to_your_json_file>`
	3. **Note**
	根据 `task_definition_parameters.json` 文件(附件)在您的 JSON 模板中输入输入参数。：有关输入参数的更多信息，请参阅任务定义参数（Amazon ECS 文档）和 register-task-definition（AWS CLI 命令参考）。
		AWS DevOps

相关资源

- Amazon ECS 任务定义
- Amazon EFS 卷

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用容器映像部署 Lambda 函数

创建者：Ram Kandaswamy (AWS)

摘要

AWS Lambda 支持将容器镜像作为部署模型。此模式介绍了如何通过容器映像部署 Lambda 函数。

Lambda 是一项无服务器、事件驱动计算服务，让您能够为几乎任何类型的应用程序或后端服务运行代码，而无需预调配或管理服务器。借助对 Lambda 函数的容器映像支持，您可以为应用程序构件提供高达 10 GB 的存储空间，并能够使用熟悉的容器映像开发工具。

此模式中的示例使用 Python 作为基础编程语言，但您也可以使用其他语言，例如 Java、Node.js 或 Go。要获取来源，可以考虑基于 Git 的系统，例如 GitHub、或 Bitbucket GitLab，或者使用亚马逊简单存储服务 (Amazon S3) Service。

先决条件和限制

先决条件

- Amazon Elastic Container Registry(Amazon ECR) 已激活
- 应用程序代码
- 带运行时系统接口客户端以及最新版本的 Python 的 Docker 映像
- Git 的工作知识

限制

- 支持的最大映像大小为 10 GB。
- 基于容器部署的 Lambda 的最长运行时间为 15 分钟。

架构

目标架构

1. 您创建 Git 存储库并将应用程序代码提交到存储库。
2. 该 AWS CodeBuild 项目由提交更改触发。

3. 该 CodeBuild 项目创建 Docker 映像并将构建的映像发布到 Amazon ECR。
4. 您可以使用 Amazon ECR 中的图像创建 Lambda 函数。

自动化和扩缩

可以通过使用 SDK 中的 AWS CloudFormation AWS Cloud Development Kit (AWS CDK)、或 API 操作来自动执行此模式。Lambda 可根据请求数量自动扩缩，您可以使用并发参数对其进行优化。有关更多信息，请参阅 Lambda 文档。

工具

Amazon Web Services

- AWS CloudFormationAWS CloudFormationhelps 您可以设置 AWS 资源，快速一致地配置资源，并在和的整个生命周期中 AWS 账户 对其进行管理 AWS 区域。此模式使用 App AWS CloudFormation lication Composer，它可以帮助您直观地查看和编辑 AWS CloudFormation 模板。
- AWS CodeBuild是一项完全托管的生成服务，可帮助您编译源代码、运行单元测试和生成可随时部署的工件。
- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。

其他工具

- Docker 是一组平台即服务（PaaS）产品，它们使用操作系统级别的虚拟化技术在容器中交付软件。
- GitHubGitLab、和 Bitbucket 是一些常用的基于 Git 的源代码控制系统，用于跟踪源代码更改。

最佳实践

- 使函数足够高效足够小，以避免加载不必要的文件。
- 努力将静态层置于 Docker 文件列表的高处，并将变化频率更高的层置于低处。这将改善缓存，从而提高性能。
- 映像拥有者负责更新和修补映像。将此更新周期添加至您的操作流程。有关更多信息，请参阅 AWS Lambda 文档。

操作说明

在中创建项目 CodeBuild

Task	描述	所需技能
创建一个 Git 存储库。	创建一个 Git 存储库，其中包含应用程序源代码、Dockerfile 和文件。`buildspec.yaml`	开发人员
创建 CodeBuild 项目。	要使用 CodeBuild 项目创建自定义 Lambda 映像，请执行以下操作： 1. 登录 AWS Management Console，然后在 https://console.aws.amazon.com/codesuite/ codebuild/ 上打开 CodeBuild 控制台。 2. 创建新 项目 对于源代码，请选择您创建的 Git 存储库。有关不同类型的 Git 仓库集成的信息，请参阅使用连接文档。 3. 确认特权模式已启用。若要构建 Docker 映像，此为必要条件。否则，将无法成功构建映像。 4. 提供项目名称和描述的值。	开发人员
编辑 Dockerfile。	Dockerfile 应位于您开发应用程序的顶级目录中。Python 代码应该在 `src` 文件夹中。 创建映像时，使用 Lambda 支持的官方映像。否则，将发生引导错误，从而导致打包过程更加困难。	开发人员

Task	描述	所需技能
	有关详细信息，请参阅其他信息部分。	
在 Amazon ECR 中创建存储库。	在 Amazon ECR 中创建容器存储库。在以下示例命令中，创建的存储库的名称为`cf-demo`： `aws ecr create-repository --cf-demo` `buildspec.yaml` 文件中将引用存储库。	AWS 管理员、开发人员
将映像推送到 Amazon ECR	您可以使用 CodeBuild 来执行映像构建过程。 CodeBuild 需要获得权限才能与 Amazon ECR 交互并使用 S3。作为该过程的一部分，将构建 Docker 映像并将其推送至 Amazon ECR 注册表。有关模板和代码的详细信息，请参阅 其他信息部分。	开发人员
验证映像是否在存储库中。	若要验证映像是否在存储库中，请在 Amazon ECR 控制台上选择存储库。如果 Amazon ECR 设置中启用了漏洞扫描功能，则应列出带有标签和漏洞扫描报告结果的映像。 有关更多信息，请参阅 AWS 文档。	开发人员

创建运行映像的 Lambda 函数

Task	描述	所需技能
创建 Lambda 函数。	在 Lambda 控制台上，选择创建函数，然后选择容器映像。输入函数名称和 Amazon ECR 存储库中映像的 URI，然后选择创建函数。有关更多信息，请参阅 AWS Lambda 文档。	应用程序开发人员
测试 Lambda 函数。	若要调用并测试该函数，请选择测试。有关更多信息，请参阅 AWS Lambda 文档。	应用程序开发人员

故障排除

事务	解决方案
构建未成功。	1. 检查 CodeBuild 项目是否已开启特权模式。 2. 确保与 Docker 相关的命令有必要权限。正在尝试将 sudo 添加至命令。 3. 验证与之关联的 IAM 角色是否 CodeBuild 具有相应操作的策略，用于与 Amazon ECR、Amazon S3 和 CloudWatch 日志进行交互。

相关资源

- Lambda 的基本映像
- 的 Docker 示例 CodeBuild
- 传递临时凭证

其他信息

编辑 Dockerfile

以下代码显示了您在 Dockerfile 中编辑的命令：

```
FROM public.ecr.aws/lambda/python:3.xx

# Copy function code
COPY app.py ${LAMBDA_TASK_ROOT}
COPY requirements.txt  ${LAMBDA_TASK_ROOT}

# install dependencies
RUN pip3 install --user -r requirements.txt

# Set the CMD to your handler (could also be done as a parameter override outside of
 the Dockerfile)
CMD [ "app.lambda_handler" ]
```

在FROM命令中，为 Lambda 支持的 Python 版本使用适当的值（例如，3.12）。这将是公共 Amazon ECR 映像存储库中可用的基础映像。

COPY app.py ${LAMBDA_TASK_ROOT} 命令将代码复制至 Lambda 函数将使用的任务根目录。此命令使用环境变量，因此我们无要担心实际路径。将要运行的函数作为参数传递至 CMD ["app.lambda_handler"] 命令。

COPY requirements.txt 命令捕获代码所需依赖项。

RUN pip install --user -r requirements.txt 命令将依赖项安装至本地用户目录。

若要构建映像，请运行以下命令。

```
docker build -t <image name> .
```

在 Amazon ECR 中添加映像

在以下代码中，将 aws_account_id 替换为账号，如果正在使用其他区域，则替换为 us-east-1。该buildspec文件使用 CodeBuild 内部版本号将图像版本唯一标识为标签值。您可以更改此设置，以满足您的要求。

buildspec 自定义代码

```
phases:
  install:
    runtime-versions:
       python: 3.xx
  pre_build:
    commands:
      - python3 --version
      - pip3 install --upgrade pip
      - pip3 install --upgrade awscli
      - sudo docker info
  build:
    commands:
      - echo Build started on `date`
      - echo Building the Docker image...
      - ls
      - cd app
      - docker build -t cf-demo:$CODEBUILD_BUILD_NUMBER .
      - docker container ls
  post_build:
    commands:
      - echo Build completed on `date`
      - echo Pushing the Docker image...
      - aws ecr get-login-password --region us-east-1 | docker login --username AWS --
password-stdin aws_account_id.dkr.ecr.us-east-1.amazonaws.com
      - docker tag cf-demo:$CODEBUILD_BUILD_NUMBER aws_account_id.dkr.ecr.us-
east-1.amazonaws.com/cf-demo:$CODEBUILD_BUILD_NUMBER
      - docker push aws_account_id.dkr.ecr.us-east-1.amazonaws.com/cf-demo:
$CODEBUILD_BUILD_NUMBER
```

使用 AWS Fargate 在 Amazon ECS 上部署 Java 微服务

创建者：Vijay Thompson (AWS) 和 Sandeep Bondugula (AWS)

摘要

此模式为使用 AWS Fargate 在 Amazon Elastic Container Service (Amazon ECS) 上部署容器化 Java 微服务提供了指导。该模式不使用 Amazon Elastic Container Registry (Amazon ECR) 进行容器管理；相反，Docker 映像是从某 Docker 中心提取的。

先决条件和限制

先决条件

- 在 Docker 中心上的现有 Java 微服务应用程序
- 公共 Docker 存储库
- 一个有效的 Amazon Web Services account
- 熟悉 Amazon Web Services，包括 Amazon ECS 和 Fargate
- Docker、Java 和 Spring Boot 框架
- Amazon Relational Database Service（Amazon RDS）已启动并运行（可选）
- 如果应用程序需要 Amazon RDS（可选），则为虚拟私有云（VPC）

架构

源技术堆栈

- Java 微服务（例如，在 Spring Boot 中实施）并部署在 Docker 上

源架构

目标技术堆栈

- 使用 Fargate 托管每项微服务的 Amazon ECS 集群
- 用于托管 Amazon ECS 集群和相关安全组的 VPC 网络

- 使用 Fargate 启动容器的每个微服务的集群/任务定义

目标架构

工具

工具

- Amazon ECS 使您无需安装和操作自己的容器编排软件、管理和扩展虚拟机集群，也不需要在这些虚拟机上调度容器。
- AWS Fargate 可帮助您运行容器，而无需管理服务器或亚马逊弹性计算云 (Amazon EC2) 实例。它与 Amazon Elastic Container Service（Amazon ECS）配合使用。
- Docker 软件平台可以快速构建、测试和部署应用程序。Docker 将软件打包成称为容器的标准化单元，容器拥有软件运行所需一切，包括库、系统工具、代码和运行时系统。

Docker 代码

以下 Dockerfile 指定了所使用的 Java 开发套件（JDK）版本、Java 存档（JAR）文件所在的版本、公开的端口号以及应用程序的入口点。

```
FROM openjdk:11
ADD target/Spring-docker.jar Spring-docker.jar
EXPOSE 8080
ENTRYPOINT ["java","-jar","Spring-docker.jar"]
```

操作说明

创建新的任务定义

Task	描述	所需技能
创建任务定义。	在 Amazon ECS 中运行 Docker 容器需要任务定义。打开 Amazon ECS 控制台 https://console.aws.amazon	AWS 系统管理员、应用程序开发人员

Task	描述	所需技能
	.com/ecs/，选择任务定义，然后创建新的任务定义。有关更多信息，请参阅 Amazon ECS 文档。	
选择启动类型。	选择 Fargate 作为启动类型。	AWS 系统管理员、应用程序开发人员
配置任务。	定义任务名称并使用适当数量的任务内存和 CPU 配置应用程序。	AWS 系统管理员、应用程序开发人员
定义容器。	指定容器名称。对于映像，输入 Docker 站点名称、存储库名称和 Docker 映像的标签名称（`docker.io/sample-repo/sample-application:sample-tag-name`）。为应用程序设置内存限制，为允许的端口设置端口映射（`8080, 80`）。	AWS 系统管理员、应用程序开发人员
创建任务。	任务和容器配置到位后，创建任务。有关详细说明，请查看相关资源部分的链接。	AWS 系统管理员、应用程序开发人员

配置集群

Task	描述	所需技能
创建和配置集群。	选择仅限网络作为集群类型，配置名称，然后创建集群或使用现有集群（如果有）。有关更多信息，请参阅 Amazon ECS 文档。	AWS 系统管理员、应用程序开发人员

配置任务

Task	描述	所需技能
创建 任务。	在集群中，选择运行新任务。	AWS 系统管理员、应用程序开发人员
选择启动类型。	选择 Fargate 作为启动类型。	AWS 系统管理员、应用程序开发人员
选择任务定义、修订版和平台版本。	选择要运行的任务，然后选择任务定义的修订版和平台版本。	AWS 系统管理员、应用程序开发人员
选择 集群。	选择要在其中运行任务的集群。	AWS 系统管理员、应用程序开发人员
指定任务数量。	配置应运行的任务数。如果您启动时有两个或更多任务，则需要负载均衡器在这些任务中分发流量。	AWS 系统管理员、应用程序开发人员
指定任务组。	（可选）指定任务组名称以将一组相关任务标识为任务组。	AWS 系统管理员、应用程序开发人员
配置集群 VPC、子网和安全组。	配置集群 VPC 和要在其中部署应用程序的子网。创建或更新安全组（HTTP、HTTPS 和端口 8080）以提供对入站和出站连接的访问权限。	AWS 系统管理员、应用程序开发人员
配置公有 IP 设置。	启用或禁用公有 IP，具体取决于您是否要使用公有 IP 地址执行 Fargate 任务。推荐的默认选项为启用。	AWS 系统管理员、应用程序开发人员
查看设置并创建任务	查看设置，然后选择运行任务。	AWS 系统管理员、应用程序开发人员

割接

Task	描述	所需技能
复制应用程序 URL。	当任务状态更新为正在运行时，选择该任务。在“网络”部分中，复制公有 IP。	AWS 系统管理员、应用程序开发人员
测试您的应用程序。	在浏览器中，输入公有 IP 以测试应用程序。	AWS 系统管理员、应用程序开发人员

相关资源

- Amazon ECS 的 Docker 基础知识（Amazon ECS 文档）
- AWS Fargate 上的 Amazon ECS（Amazon ECS 文档）
- 创建任务定义（Amazon ECS 文档）
- 创建集群（Amazon ECS 文档）
- 配置基本服务参数（Amazon ECS 文档）
- 配置网络（Amazon ECS 文档）
- 在 Amazon ECS 上部署 Java 微服务（博客文章）

使用 Amazon EKS 和 Amazon S3 中的 Helm 图表存储库部署 Kubernetes 资源和软件包

由 Sagar Panigrahi (AWS) 编写

摘要

此模式可帮助您高效管理 Kubernetes 应用程序，无论复杂性如何。此模式将 Helm 集成至现有持续集成和持续交付 (CI/CD) 管道，以便将应用程序部署至 Kubernetes 集群。Helm 是 Kubernetes 软件包管理器，可以帮助您管理 Kubernetes 应用程序。Helm 图表可用于定义、安装和升级复杂的 Kubernetes 应用程序。图表可以版本控制，并存储在 Helm 存储库中，从而缩短停机期间的平均恢复时间 (MTTR)。

此模式对 Kubernetes 集群使用了 Amazon Elastic Kubernetes Service (Amazon EKS)。其使用 Amazon Simple Storage Service (Amazon S3) 作为 Helm 图表存储库，如此整个组织的开发人员即可集中管理和访问此类图表。

先决条件和限制

先决条件

- 具有虚拟私有云（VPC）的活跃 Amazon Web Services (AWS) Account
- Amazon EKS 集群
- 在 Amazon EKS 集群中设置并准备处理工作负载的工作节点
- 在客户端计算机中为目标集群配置 Amazon EKS kubeconfig 文件的 Kubectl
- 用于创建 S3 存储桶的 AWS Identity and Access Management (IAM) 访问权限
- 通过客户端计算机访问 Amazon S3 的 IAM（编程或角色）
- 源代码管理和 CI/CD 管道

限制

- 目前不支持升级、删除或管理自定义资源定义 (CRDs)。
- 如果使用的是引用 CRD 的资源，则必须单独安装 CRD（图表外）。

产品版本

- Helm v3.6.3

架构

目标技术堆栈

- Amazon EKS
- Amazon VPC
- Amazon S3
- 源代码管理
- Helm
- Kubectl

目标架构

自动化和扩缩

- AWS CloudFormation 可用于自动创建基础设施。有关更多信息，请参阅 A mazon EKS 文档 CloudFormation中的使用 AWS 创建亚马逊 EKS 资源。
- Helm 将整合至您现有的 CI/CD 自动化工具中，以自动执行 Helm 图表的打包和版本控制（超出了此模式范围）。
- GitVersion 或者 Jenkins 内部版本号可用于自动化图表的版本控制。

工具

工具

- Amazon EKS – Amazon Elastic Kubernetes Service (Amazon EKS) 是一项托管服务，可用来在 AWS 上运行 Kubernetes，而无需支持或维护您自己的 Kubernetes 控制面板。Kubernetes 是一个用于实现容器化应用程序的部署、扩缩和管理自动化的开源系统。
- Helm – Helm 是 Kubernetes 的软件包管理器，可帮助您在 Kubernetes 集群上安装和管理应用程序。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项面向互联网的存储服务。您可以通过 Amazon S3 随时在 Web 上的任何位置存储和检索的任意大小的数据。

- Kubectl – Kubectl 是针对 Kubernetes 集群运行命令的命令行实用程序。

代码

示例代码附后。

操作说明

配置并初始化 Helm

Task	描述	所需技能
安装 Helm 客户端。	若要在本地系统上下载并安装 Helm 客户端，请使用以下命令。 `sudo curl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 | bash`	DevOps 工程师
验证 Helm 安装	若要验证 Helm 是否与 Amazon EKS 集群中的 Kubernetes API 服务器通信，请运行 `helm version`。	DevOps 工程师

在 Amazon EKS 集群中创建并安装 Helm 图表

Task	描述	所需技能
为 NGINX 创建 Helm 图表。	若要在客户端计算机上创建名为 `my-nginx` 的 Helm 图表，请运行 `helm create my-nginx`。	DevOps 工程师
查看图表结构。	若要查看图表的结构，请运行树命令 `tree my-nginx/` 。	DevOps 工程师

Task	描述	所需技能
在图表中停用服务账户创建。	在 `values.yaml` 的 `serviceAccount` 部分下，将 `create` 密钥设置为 `false`。将其关闭，原因在于无需为此模式创建服务账户。	DevOps 工程师
验证 (lint) 修改后的图表是否存在语法错误。	若要验证将图表安装至目标集群之前图表是否存在任何语法错误，请运行 `helm lint my-nginx/` 。	DevOps 工程师
安装部署 Kubernetes 资源的图表。	若要运行 Helm 图表 安装程序，请使用以下命令。 `helm install --name my-nginx-release --debug my-nginx/ --namespace helm-space` 可选 `debug` 标志在安装过程中输出所有调试消息。`namespace` 标志指定将在其中创建此图表的资源部分的命名空间。	DevOps 工程师
查看 Amazon EKS 集群中的资源。	若要查看 `helm-space` 命名空间中的 Helm 图表部分创建的资源，请使用以下命令。 `kubectl get all -n helm-space`	DevOps 工程师

回滚至 Kubernetes 应用程序的先前版本

Task	描述	所需技能
修改并升级版本。	若要修改图表，请在 `values.yaml` 中将 `replicaCount` 值更改为 2。然后通过运行以下命令升级已安装发布： `helm upgrade my-nginx-release my-nginx/ --namespace helm-space`	DevOps 工程师
查看 Helm 版本的历史记录。	若要列出使用 Helm 安装的特定版本的所有修订版本，请运行以下命令。 `helm history my-nginx-release`	DevOps 工程师
查看特定修订版的详细信息。	在切换或回滚至工作版本前以及在安装修订版前进行额外验证前，请使用以下命令查看已传递至每个修订版本的值。 `helm get --revision=2 my-nginx-release`	DevOps 工程师
回滚至先前版本。	若要回滚至以前的修订版，请使用以下命令。 `helm rollback my-nginx-release 1` 此示例正在回滚至修订版号 1。	DevOps 工程师

将 S3 存储桶初始化为 Helm 存储库

Task	描述	所需技能
为 Helm 图表创建 S3 存储桶。	创建唯一 S3 存储桶。在存储桶中，创建一个名为 `charts` 的文件夹。此模式中的示例使用 `s3://my-helm-charts/charts` 作为目标图表存储库。	云管理员
安装适用于 Amazon S3 的 Helm 插件。	若要在客户端计算机上安装 helm-s3 插件，请使用以下命令。 `helm plugin install https://github.com/hypnoglow/helm-s3.git --version 0.10.0` 请注意：提供使用插件版本 0.9.0 及更高版本的 Helm V3 支持。	DevOps 工程师
初始化 Amazon S3 Helm 存储库。	若要将目标文件夹初始化为 Helm 存储库，请使用以下命令。 `helm S3 init s3://my-helm-charts/charts` 该命令在目标系统中创建 `index.yaml` 文件，用于跟踪存储在该位置的所有图表信息。	DevOps 工程师
将 Amazon S3 存储库添加至 Helm。	要向客户端计算机中添加存储库，请使用以下命令。	DevOps 工程师

Task	描述	所需技能
	`helm repo add my-helm-charts s3://my-helm-charts/charts` 此命令将别名添加至 Helm 客户端计算机中的目标存储库。	
查看存储库列表。	若要查看 Helm 客户端计算机中的存储库列表，请运行 `helm repo list`。	DevOps 工程师

在 Amazon S3 Helm 存储库中打包和存储图表

Task	描述	所需技能
打包图表。	若要打包您创建的 `my-nginx` 图表，请运行 `helm package ./my-nginx/` 。该命令将 `my-nginx` 图表文件夹的所有内容打包成存档文件，该文件使用 `Chart.yaml` 文件中提到的版本号命名。	DevOps 工程师
将包存储至 Amazon S3 Helm 存储库。	若要将数据包上传至 Amazon S3 中的 Helm 存储库，请使用正确的 `.tgz` 文件名运行以下命令。 `helm s3 push ./my-nginx-0.1.0.tgz my-helm-charts`	DevOps 工程师
搜索 Helm 图表。	若要确认图表同时出现在本地和 Amazon S3 的 Helm 存储库中，请运行以下命令。	DevOps 工程师

Task	描述	所需技能
	`helm search repo my-nginx`	

修改、版本化和打包图表

Task	描述	所需技能
修改并打包图表。	将 `values.yaml` 中，将 `replicaCount` 的值设为 1。然后通过运行 `helm package ./my-nginx/` 来打包图表，这次是将版本 `Chart.yaml` 更改为 `0.1.1`。 最好使用诸如 CI/CD 管道中的 Jenkins 内部版本号 GitVersion 之类的工具通过自动化来更新版本控制。自动生成的版本号超出了这种模式范围。	DevOps 工程师
将新版本推送至 Amazon S3 中的 Helm 存储库。	若要将版本为 0.1.1 的新软件包推送至 Amazon S3 中的 `my-helm-charts` Helm 存储库，请运行以下命令。 `helm s3 push ./my-nginx-0.1.1.tgz my-helm-charts`	DevOps 工程师

从 Amazon S3 Helm 存储库中搜索并安装图表

Task	描述	所需技能
搜索 my-nginx 图表的所有版本。	若要查看图表的所有可用版本，请使用 `--versions` 标志运行以下命令： `helm search repo my-nginx --versions` 如果没有标志，Helm 默认会显示图表的最新上传版本。	DevOps 工程师
从 Amazon S3 Helm 存储库安装图表。	上一个任务的搜索结果显示了 `my-nginx` 图表的多个版本。若要从 Amazon S3 Helm 存储库安装新版本 (0.1.1)，请使用以下命令。 `helm upgrade my-nginx-release my-helm-charts/my-nginx --version 0.1.1 --namespace helm-space`	DevOps 工程师

相关资源

- HELM 文档
- helm-s3 插件（MIT 许可证）
- HELM 客户端二进制
- Amazon EKS 文档

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

在 Amazon EKS 上部署示例 Java 微服务并使用应用程序负载均衡器公开该微服务

由 Vijay Thompson (AWS) 和 Akkamahadevi Hiremath (AWS) 编写

摘要

此模式描述如何使用 `eksctl` 命令行实用程序和 Amazon Elastic Container Registry (Amazon ECR) 将示例 Java 微服务部署为 Amazon Elastic Kubernetes Service (Amazon EKS) 上的容器化应用程序。您可以使用应用程序负载均衡器均衡应用程序流量。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- AWS 命令行界面（AWS CLI）版本 1.7 或更高版本，已在 macOS、Linux 或 Windows上安装和配置。
- 正在运行的 Docker 进程守护程序
- 在 macOS、Linux 或 Windows 上安装和配置的 `eksctl` 命令行实用程序（有关更多信息，请参阅 Amazon EKS 文档中的 Amazon EKS 入门 — eksctl。）
- 在 macOS、Linux 或 Windows 上安装和配置的`kubectl`命令行实用程序（有关更多信息，请参阅 Amazon EKS 文档中的安装或更新 kubectl。）

限制

- 此模式不包括为应用程序负载均衡器安装 SSL 证书。

架构

目标技术堆栈

- Amazon ECR
- Amazon EKS
- Elastic Load Balancing

目标架构

下图显示了在 Amazon EKS 上容器化 Java 微服务的架构。

工具

- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可帮助您在 AWS 上运行 Kubernetes，而无需安装或维护您自己的 Kubernetes 控制面板或节点。
- AWS 命令行界面（AWS CLI)是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- Elastic L oad Balancing 会自动将您的传入流量分配到一个或多个可用区域中的多个目标，例如亚马逊弹性计算云 (Amazon EC2) 实例、容器和 IP 地址。
- eksctl 可以帮助您在Amazon EKS 上创建集群。
- kubectl 使对 Kubernetes 集群运行命令成为可能。
- Docker 帮助您在名为容器的软件包中构建、测试和交付应用程序。

操作说明

使用 eksctl 创建 Amazon EKS 集群

Task	描述	所需技能
创建 Amazon EKS 集群。	要创建使用两个 t2.small Amazon EC2 实例作为节点的 Amazon EKS 集群，请运行以下命令： `eksctl create cluster --name <your-cluster-name> --version <version-number> --nodes=1 --node-type=t2.small`	开发人员、系统管理员

Task	描述	所需技能
	Note 该过程可能需要 15 到 20 分钟。创建集群后，相应的 Kubernetes 配置将添加至您的 kubeconfig 文件中。您可将该 kubeconfig 文件与 kubectl 一起使用，以便在后续步骤中部署应用程序。	
验证 Amazon EKS 集群。	要验证集群是否已创建并且您可连接到该集群，请运行 kubectl get nodes 命令。	开发人员、系统管理员

创建 Amazon ECR 存储库并推送 Docker 映像。

Task	描述	所需技能
创建 Amazon ECR 存储库。	请遵循 Amazon ECR 文档中的创建私有存储库的说明操作。	开发人员、系统管理员
创建 POM XML 文件。	根据此模式的其他信息部分中的示例 POM 文件代码创建 pom.xml 文件。	开发人员、系统管理员
创建源文件。	根据以下示例，按 src/main/java/eksExample 路径创建名为 HelloWorld.java 的源文件。 `package eksExample;`	

Task	描述	所需技能

	``
	import static spark.Spa
	rk.get;
	
	public class HelloWorl
	d {
	    public static void
	 main(String[] args) {
	        get("/", (req,
	 res) -> {
	            return
	 "Hello World!";
	        });
	    }
	}
	``
	
	请确保采用以下目录结构：
	
	``
	### Dockerfile
	### deployment.yaml
	### ingress.yaml
	### pom.xml
	### service.yaml
	### src
	    ### main
	        ### java
	            ###
	 eksExample
	                ###
	 HelloWorld.java
	``	

创建 Dockerfile。	根据此模式的其他信息部分中的示例 Dockerfile代码创建 `Dockerfile` 。	开发人员、系统管理员

Task	描述	所需技能

构建和推送 Docker 映像。	在您想要 Dockerfile 构建、标记映像并将其推送至 Amazon ECR 的目录中，运行以下命令：
	
	`aws ecr get-login -password --region <region>| docker login --username <username> --password-stdin <account_number>.dkr.ecr.<region>.amazonaws.com`
	`docker buildx build --platform linux/amd64 -t hello-world-java:v1 .`
	`docker tag hello-world-java:v1 <account_number>.dkr.ecr.<region>.amazonaws.com/<repository_name>:v1`
	`docker push <account_number>.dkr.ecr.<region>.amazonaws.com/<repository_name>:v1`
	
	Note
	在上述命令中修改 AWS 区域、账号和存储库详细信息。请务必记下图片网 URL，以备日后使用。	

Task	描述	所需技能
	Important 搭载 M1 芯片的 macOS 系统在构建与平台上运行的 Amazon EKS 兼容的映像时出现问题。 AMD64 要解决此问题，请使用 docker buildx构建可在 Amazon EKS 上运行的 Docker 映像。	

部署 Java 微服务

Task	描述	所需技能
创建部署文件。	根据此模式的其他信息部分中的示例部署文件代码创建名为 `deployment.yaml` 的 YAML 文件。 **Note** 使用您之前复制的图片 URL 作为 Amazon ECR 存储库的图像文件路径。	开发人员、系统管理员
在Amazon EKS 集群上部署 Java 微服务。	要在您的 Amazon EKS 集群中创建部署，请运行 `kubectl apply -f deployment.yaml` 命令。	开发人员、系统管理员

Task	描述	所需技能
验证容器组（pod）的状态。	1. 要验证容器组（pod）的状态，请运行 `kubectl get pods` 命令。 2. 等待状态更改为就绪。	开发人员、系统管理员
创建服务。	1. 根据此模式的其他信息部分中的示例服务文件代码创建名为 `service.yaml` 的文件。 2. 运行 `kubectl apply -f service.yaml` 命令。	开发人员、系统管理员
安装 AWS Load Balancer Controller 附加组件。	按照 Amazon EKS 文档中的安装 AWS Load Balancer Controller 附加组件进行操作。 **Note** 必须安装插件才能为 Kubernetes 服务创建应用程序负载均衡器或网络负载均衡器。	开发人员、系统管理员
创建入口资源。	根据此模式的其他信息部分中的示例入口资源文件代码创建名为 `ingress.yaml` 的 YAML 文件。	开发人员、系统管理员
创建应用程序负载均衡器。	要部署入口资源并创建应用程序负载均衡器，请运行 `kubectl apply -f ingress.yaml` 命令。	开发人员、系统管理员

测试应用程序

Task	描述	所需技能
测试和验证应用程序。	1. 要从 ADDRESS 字段获取负载均衡器的 DNS 名称，请运行 `kubectl get ingress.networking.k8s.io/java-microservice-ingress` 命令。 2. 在与您的 Amazon EKS 节点位于同一 VPC 中的 EC2 实例上，运行该`curl -v <DNS address from previous command>`命令。	开发人员、系统管理员

相关资源

- 创建私有存储库（Amazon ECR 文档）
- 推送 Docker 映像（Amazon ECR 文档）
- 入口控制器（Amazon EKS Workshop）
- Docker buildx（Docker 文档）

其他信息

POM 文件示例

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/
maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
```

```
  <groupId>helloWorld</groupId>
  <artifactId>helloWorld</artifactId>
  <version>1.0-SNAPSHOT</version>


  <dependencies>
    <dependency>
      <groupId>com.sparkjava</groupId><artifactId>spark-core</
artifactId><version>2.0.0</version>
    </dependency>
  </dependencies>
  <build>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId><artifactId>maven-jar-plugin</
artifactId><version>2.4</version>
        <configuration><finalName>eksExample</finalName><archive><manifest>
              <addClasspath>true</addClasspath><mainClass>eksExample.HelloWorld</
mainClass><classpathPrefix>dependency-jars/</classpathPrefix>
            </manifest></archive>
        </configuration>
      </plugin>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId><artifactId>maven-compiler-plugin</
artifactId><version>3.1</version>
        <configuration><source>1.8</source><target>1.8</target></configuration>
      </plugin>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId><artifactId>maven-assembly-plugin</
artifactId>
        <executions>
          <execution>
            <goals><goal>attached</goal></goals><phase>package</phase>
            <configuration>
              <finalName>eksExample</finalName>
              <descriptorRefs><descriptorRef>jar-with-dependencies</descriptorRef></
descriptorRefs>
              <archive><manifest><mainClass>eksExample.HelloWorld</mainClass></
manifest></archive>
            </configuration>
          </execution>
        </executions>
      </plugin>
    </plugins>
```

```
  </build>
</project>
```

Dockerfile 示例

```
FROM bellsoft/liberica-openjdk-alpine-musl:17

RUN apk add maven
WORKDIR /code

# Prepare by downloading dependencies
ADD pom.xml /code/pom.xml
RUN ["mvn", "dependency:resolve"]
RUN ["mvn", "verify"]

# Adding source, compile and package into a fat jar
ADD src /code/src
RUN ["mvn", "package"]

EXPOSE 4567
CMD ["java", "-jar", "target/eksExample-jar-with-dependencies.jar"]
```

部署文件示例

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: microservice-deployment
spec:
  replicas: 2
  selector:
    matchLabels:
      app.kubernetes.io/name: java-microservice
  template:
    metadata:
      labels:
        app.kubernetes.io/name: java-microservice
    spec:
      containers:
      - name: java-microservice-container
        image: .dkr.ecr.amazonaws.com/:
        ports:
```

```
        - containerPort: 4567
```

示例服务文件

```
apiVersion: v1
kind: Service
metadata:
  name: "service-java-microservice"
spec:
  ports:
    - port: 80
      targetPort: 4567
      protocol: TCP
  type: NodePort
  selector:
    app.kubernetes.io/name: java-microservice
```

入口资源文件示例

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: "java-microservice-ingress"
  annotations:
    kubernetes.io/ingress.class: alb
    alb.ingress.kubernetes.io/load-balancer-name: apg2
    alb.ingress.kubernetes.io/target-type: ip
  labels:
    app: java-microservice
spec:
  rules:
    - http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: "service-java-microservice"
                port:
                  number: 80
```

使用 AWS Copilot 将集群应用程序部署至 Amazon ECS

由 Jean-Baptiste Guillois (AWS)、Mathew George (AWS) 和 Thomas Scott (AWS) 编写

摘要

此模式介绍了如何通过两种方式在 Amazon Elastic Container Service (Amazon ECS) 集群中部署容器，即使用 Amazon Web Services (AWS) Management Console 和使用 AWS Copilot，以演示 AWS Copilot 如何简化部署任务。

Amazon ECS 是一项高度可扩展的快速容器管理服务，它可轻松运行、停止和管理集群上的容器。您的容器是在用于运行单个任务或服务内任务的任务定义中定义的。您可以在由 AWS Fargate 托管的无服务器基础设施上运行任务和服务。或者，为了更好地控制您的基础设施，您可以在您管理的亚马逊弹性计算云 (Amazon EC2) 实例集群上运行任务和服务。

AWS Copilot 命令行界面（CLI）命令简化了从本地开发环境在 Amazon ECS 上构建、发布和操作生产就绪的容器化应用程序。AWS Copilot CLI 与支持现代应用程序最佳实践的开发人员工作流保持一致：从将基础设施用作代码到创建代表用户配置的持续集成和持续交付（CI/CD）管道。您可以将 AWS Copilot CLI 用作日常开发和测试周期的一部分，替代 AWS 管理控制台。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- AWS 命令行界面（AWS CLI）在本地安装并配置为使用您的 Amazon Web Services account（请参阅 AWS CLI 文档中的安装说明和配置说明）
- AWS Copilot 已本地安装（参见 Amazon ECS 文档中的安装说明）
- 本地计算机上安装了 Docker（参见 Docker 文档）

限制

- 在免费计划中，Docker 强制每个 IP 地址每 6 小时拉取 100 容器映像。

架构

目标技术堆栈

- 设置了虚拟私有云（VPC）、公有子网和私有子网以及安全组的 AWS 环境

- Amazon ECS 集群
- Amazon ECS 服务和任务定义
- Amazon Elastic Container Registry (Amazon ECR)
- Amazon DynamoDB
- 应用程序负载均衡器
- AWS Fargate
- Amazon Identity and Access Management (IAM)
- Amazon CloudWatch
- AWS CloudTrail

目标架构

当您部署此模式的示例应用程序时，将在不同的可用区中创建和部署多个任务。每个任务将数据存储在 Amazon DynamoDB。当您访问某个任务的网页时，您可以查看所有其他任务的数据。

工具

Amazon Web Services

- Amazon ECR – Amazon Elastic Container Registry (Amazon ECR) 是一项 AWS 托管容器映像注册表服务，它安全、可扩展且可靠。Amazon ECR 支持私有存储库，其具有使用 IAM 的基于资源的权限。
- Amazon ECS – Amazon Elastic Container Service (Amazon ECS) 是一项高度可扩展的快速容器管理服务，可用于运行、停止和管理集群上的容器。您可以在由 AWS Fargate 托管的无服务器基础设施上运行任务和服务。或者，为了更好地控制您的基础设施，您可以在您管理的亚马逊弹性计算云 (Amazon EC2) 实例集群上运行任务和服务。
- AWS Copilot – AWS Copilot 提供了命令行界面，可帮助您在 AWS 上启动和管理容器化应用程序，包括推送到注册表、创建任务定义和创建集群。
- AWS Fargate — AWS Fargate 是一款无服务器 pay-as-you-go计算引擎，让您无需管理服务器即可专注于构建应用程序。AWS Fargate 与 Amazon ECS 和 Amazon Elastic Kubernetes Service(Amazon EKS) 兼容。运行具有 Fargate 启动类型或 Fargate 容量提供程序的 Amazon ECS 任务和服务时，将应用程序打包到容器中，指定 CPU 和内存要求，定义联网和 IAM policy，然后启动应用程序。每个 Fargate 任务都具有自己的隔离边界，不与其他任务共享底层内核、CPU 资源、内存资源或弹性网络接口。

- Amazon DynamoDB – Amazon DynamoDB 是一种全托管 NoSQL 数据库服务，提供快速而可预测的性能，能够实现无缝扩展。
- El@@ astic Load Balancing (ELB) — Elastic Load Balancing 会自动将您的传入流量分配到一个或多个可用区的多个目标，例如 EC2 实例、容器和 IP 地址。它会监控已注册目标的运行状况，并仅将流量传输到运行状况良好的目标。弹性负载均衡 根据传入流量随时间的变化对负载均衡器进行扩展。它可以自动扩缩来处理绝大部分工作负载。

工具

- Docker 命令行界面
- AWS 命令行界面（AWS CLI）
- AWS Copilot 命令行界面

代码

此模式中使用的示例应用程序的代码可在集群示例应用程序存储库中找到。 GitHub按照下一节中的说明使用示例文件。

操作说明

部署应用程序堆栈 - 选项 1（AWS 管理控制台）

Task	描述	所需技能
克隆 GitHub 存储库。	使用以下命令克隆代码存储库示例： `git clone https://github.com/aws-samples/cluster-sample-app cluster-sample-app && cd cluster-sample-app`	AWS 应用程序开发人员 DevOps
创建 Amazon ECR 存储库。	1. 登录 AWS 管理控制台并在https://console.aws.amazon.com/ecr/存储库中	AWS 应用程序开发人员 DevOps

Task	描述	所需技能
	打开 Amazon ECR 控制台。 2. 选择创建存储库。 3. 对于存储库名称，请输入cluster-sample-app。 4. 将所有其他设置保留为默认值。 5. 选择 Create repository(创建存储库)。 有关更多信息，请参阅 Amazon ECR 文档中的创建私有存储库。	

Task	描述	所需技能

构建、标记和推送 Docker 映像到您的 Amazon ECR 存储库。	1. 选择您刚刚创建的存储库，然后选择查看推送命令。
	2. 复制显示的命令并在本地运行它们以构建、标记和推送 Docker 映像。这些命令将与以下内容类似。
	
	将您的 Docker 客户端验证到注册表：
	
	`aws ecr get-login -password --region <YOUR_AWS_REGION> | docker login --username AWS --password-stdin <YOUR_AWS_ACCOUNT>.dkr.ecr.<YOUR_AWS_REGION>.amazonaws.com`
	
	构建 Docker 映像：
	
	`docker build -t cluster-sample-app .`
	
	要为 Docker 映像添加标记：
	
	`docker tag cluster-sample-app:latest <YOUR_AWS_ACCOUNT>.dkr.ecr.<YOUR_AWS_REGION>.amazonaws.com/cluster-sample-app:latest`
	
	推送 Docker 映像到存储库：
		AWS 应用程序开发人员 DevOps

Task	描述	所需技能
	`docker push <YOUR_AWS_ACCOUNT>.dkr.ecr.<YOUR_AWS_REGION>.amazonaws.com/cluster-sample-app:latest`	

Task	描述	所需技能
部署应用程序堆栈。	1. 打开 AWS CloudFormation 控制台，网址为https://console.aws.amazon.com/cloudformation/。 2. 选择创建堆栈。 3. 对于 Prepare template（准备模板），选择 Template is ready（模板就绪）。 4. 在指定模板部分，选择上传模板文件。 5. 选择您从 GitHub 存储库`cluster-sample-app-stack.yml` 中克隆的本地文件作为 CloudFormation 模板，然后选择 “下一步”。 6. 输入您堆栈的名称，然后选择下一步。 7. 保留所有默认选项，然后选择下一步。 8. 查看所有选项，确认 IAM 资源创建，然后选择创建堆栈。 9. 部署应用程序堆栈后，选择输出选项卡，复制 URL，然后在浏览器中将其打开以访问该应用程序。 有关部署 CloudFormation 模板的更多信息，请参阅 AWS CloudFormation 文档中的创建堆栈。	AWS DevOps，应用程序开发者

部署应用程序堆栈 — 选项 2 (AWS Copilot CLI)

Task	描述	所需技能
克隆 GitHub 存储库。	使用以下命令克隆代码存储库示例： `git clone https://github.com/aws-samples/cluster-sample-app cluster-sample-app && cd cluster-sample-app`	AWS 应用程序开发人员 DevOps
使用 AWS Copilot CLI 将您的容器映像部署到 AWS。	在项目根目录中使用以下命令一步部署应用程序： `copilot init --app cluster-sample-app --name demo --type "Load Balanced Web Service" --dockerfile ./Dockerfile --port 8080 --deploy` 然后，您应该能够使用作为输出提供的 DNS 名称来访问该应用程序。	AWS 应用程序开发人员 DevOps

删除已创建资源

Task	描述	所需技能
删除 AWS 管理控制台 创建的资源。	如果您使用选项 1（AWS 管理控制台）部署应用程序堆栈，请在准备好删除您创建的资源时按照以下步骤操作：	AWS 应用程序开发人员 DevOps

Task	描述	所需技能
	1. 打开 CloudFormation 控制台，网址为https://console.aws.amazon.com/cloudformation/。 2. 选择您创建的堆栈，然后选择删除。 3. 在https://console.aws.amazon.com/ecr/存储库中打开 Amazon ECR 控制台。 4. 选择您创建的存储库，然后选择删除。	
删除 AWS Copilot 创建的资源。	如果您使用选项 2 (AWS Copilot CLI) 部署应用程序堆栈，请在准备删除创建的资源时从项目的根目录运行以下命令： `copilot app delete`	AWS 应用程序开发人员 DevOps

相关资源

- 安装或更新最新版本的 AWS CLI（AWS CLI 文档）
- 使用 AWS Copilot 命令行界面（Amazon ECS 文档）
- AWS Fargate 上的 Amazon ECS（Amazon ECS 文档）
- Amazon ECS 文档
- Amazon ECR 文档
- 亚马逊 CloudFormation 文档
- Docker 桌面（Docker 文档）

在 Amazon EKS 集群上部署基于 gRPC 的应用程序并使用应用程序负载均衡器访问它

由 Kirankumar Chandrashekar (AWS) 和 Huy Nguyen (AWS) 创作

摘要

此模式描述如何在 Amazon Elastic Kubernetes Service (Amazon EKS) 集群上托管基于 gRPC 的应用程序，并通过应用程序负载均衡器安全地访问该应用程序。

gRPC 是一个开源远程过程调用 (RPC) 框架，可以在任何环境中运行。您可将其用于微服务集成和客户端-服务器通信。有关 gRPC 的更多信息，请参阅 AWS 博客文章 App lication Load Balancer 对 end-to-end HTTP/2 和 gRPC 的支持。

此模式向您展示如何托管在 Amazon EKS 上的 Kubernetes 容器上运行的基于 GRPC 的应用程序。gRPC 客户端通过 HTTP/2 协议通过 SSL/TLS 加密连接连接连接到 Application Load Balancer。应用程序负载均衡器将流量转发至 Amazon EKS 容器组（pod）上运行的 gRPC 应用程序。使用 Kubernetes Horizontal Pod Autoscaler，可以根据流量自动扩缩 gRPC 容器组（pod）的数量。应用程序负载均衡器的目标组对 Amazon EKS 节点执行运行状况检查，评估目标是否正常，并仅将流量转发到运行状况良好的节点。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- Docker，已在 Linux、macOS 或 Windows 上安装并配置。
- AWS 命令行界面（AWS CLI）版本 2，已在 Linux、macOS 或 Windows 上安装并配置。
- eksctl，在 Linux、macOS 或 Windows 上安装和配置。
- `kubectl`，已安装并配置为访问 Amazon EKS 集群资源。有关更多信息，请参阅 Amazon E KS 文档中的安装或更新 kubectl。
- g RPCurl，已安装并配置。
- 新的或现有的 Amazon EKS 集群。有关更多信息，请参阅 Amazon EKS 入门。
- 您的计算机终端已配置为访问 Amazon EKS 集群。有关更多信息，请参阅 Amazon EKS 文档中的配置您的计算机以与集群通信。
- AWS Load Balancer Controller，在 Amazon EKS 集群中配备。

- 带有有效 SSL 或 SSL/TLS 证书的现有 DNS 主机名。您可通过使用 AWS Certificate Manager (ACM) 或将现有证书上载到 ACM 获取域证书。有关这两个选项的更多信息，请参阅 ACM 文档中的请求公共证书和将证书导入 AWS Certificate Manager。

架构

下图显示了此模式实现的架构。

下图显示了一个工作流，其中从 gRPC 客户端接收 SSL/TLS 流量，然后将其卸载到应用程序负载均衡器。由于流量来自虚拟私有云（VPC），因此以明文形式转发到 gRPC 服务器。

工具

Amazon Web Services

- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- 弹性负载均衡在多个目标上分配传入的应用程序或网络流量。例如，您可以跨亚马逊弹性计算云 (Amazon EC2) 实例、容器以及一个或多个可用区中的 IP 地址分配流量。
- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可帮助您在 AWS 上运行 Kubernetes，而无需安装或维护您自己的 Kubernetes 控制面板或节点。

工具

- eksctl 是一款用于在 Amazon EKS 上创建集群的简单 CLI 工具。
- kubectl 是针对 Kubernetes 集群运行命令的命令行实用程序。
- AWS 负载均衡器控制器帮助管理 Kubernetes 集群的 AWS 弹性负载均衡器。
- g RPCurl 是一个命令行工具，可帮助您与 gRPC 服务进行交互。

代码存储库

此模式的代码可在 to-e GitHub grpc-traffic-on-albks 存储库中找到。

操作说明

构建 gRPC 服务器的 Docker 映像并将其推送到 Amazon ECR

Task	描述	所需技能
创建 Amazon ECR 存储库。	登录 AWS 管理控制台，打开 A mazon ECR 控制台，然后创建 Amazon ECR 存储库。有关更多信息，请参阅 Amazon ECR 文档中的创建存储库。请务必记录 Amazon ECR 存储库的 URL。 您也可以通过运行以下命令，使用 AWS CLI 创建 Amazon ECR 存储库： `aws ecr create-repository --repository-name helloworld-grpc`	云管理员

构建 Docker 映像。	1. 克隆 GitHub grpc-traffic-on-albto-eks 存储库。
	
	`git clone https://github.com/aws-samples/grpc-traffic-on-alb-to-eks.git`
	
	2. 在存储库的根目录中，确保 Dockerfile 存在，然后运行以下命令来构建 Docker 镜像：
	
	`docker build -t <amazon_ecr_repository_url>:<Tag> .`
		DevOps 工程师

Task	描述	所需技能
	Important 请务必<amazon_ecr_repository_url> 使用之前创建的 Amazon ECR 存储库的 URL 进行替换。	

Task	描述	所需技能

将 Docker 映像推送到 Amazon ECR。	1. 运行以下命令以登录 Amazon ECR 存储库：
	`aws ecr get-login -password --region us-east-1 --no-cli-auto-prompt | docker login --username AWS --password-stdin <your_aws_account_id>.dkr.ecr.us-east-1.amazonaws.com`
	2. 通过运行以下命令将 Docker 映像推送到 Amazon ECR 存储库。
	`docker push <your_aws_account_id>.dkr.ecr.us-east-1.amazonaws.com/helloworld-grpc:1.0`
	Important
	请务必使用您的 AWS 账户 ID `<your_aws_account_id>` 进行替换。
		DevOps 工程师

将 Kubernetes 清单部署到 Amazon EKS 集群

Task	描述	所需技能
修改 Kubernetes 清单文件值。	1. 根据您的要求修改存储`grpc-sample.yaml` 库 Kubernetes 文件夹中的 Kubernetes 清单文件。您必须修改入口资源的注释和主机名。有关示例入口资源，请参阅其他信息部分。有关入口注释的更多信息，请参阅 Kubernetes 文档中的入口注释。 2. 在 Kubernetes 部署资源中，将部署资源的 `image` 更改为：您将 Docker 映像推至的 Amazon ECR 存储库的统一资源标识符 (URI)。有关此部署资源的样本，请参阅其他信息部分。	DevOps 工程师
部署 Kubernetes 清单文件。	通过运行以下`kubectl`命令将`grpc-sample.yaml` 文件部署到 Amazon EKS 集群： `kubectl apply -f ./kubernetes/grpc-sample.yaml`	DevOps 工程师

为应用程序负载均衡器的 FQDN 创建 DNS 记录

Task	描述	所需技能
应用程序负载均衡器的 FQDN 记录。	1. 运行以下 `kubectl` 命令来描述管理应用程序负载均	DevOps 工程师

Task	描述	所需技能
	衡器的 Kubernetes 入口资源： `kubectl get ingress -n grpcserver` 输出示例在 “其他信息” 部分中提供。在输出中，HOSTS 字段显示为其创建 SSL 证书的 DNS 主机名。 2. 在输出Address字段中记录应用程序负载均衡器的完全限定域名 (FQDN)。 3. 创建指向应用程序负载均衡器的 FQDN 的 DNS 记录。如果您的 DNS 提供程序是 Amazon Route 53，请创建一条指向应用程序负载均衡器的 FQDN 的别名记录。有关此选项的更多信息，请参阅 Route 53 文档中的在别名和非别名记录之间进行选择。	

测试解决方案

Task	描述	所需技能
测试 gRPC 服务器。	使用 g 通过运行以下命令RPCurl 来测试端点： `grpcurl grpc.example.com:443 list`	DevOps 工程师

Task	描述	所需技能
	`grpc.reflection.v1alpha.ServerReflection helloworld.helloworld` **Note** `grpc.example.com` 替换为您的 DNS 名称。	

使用 gRPC 客户端测试 gRPC 服务器。	在`helloworld_client_ssl.py` 示例 gRPC 客户端中，将中的主机名替换为用于 gRPC 服务器的主机名。`grpc.example.com`
	
	下列代码示例显示了 gRPC 服务器对客户端请求的响应：
	
	`python ./app/helloworld_client_ssl.py`
	`message: "Hello to gRPC server from Client"`
	
	`message: "Thanks for talking to gRPC server!! Welcome to hello world. Received message is \"Hello to gRPC server from Client\""`
	`received: true`
	
	这表明客户端可与服务器通信，并且连接成功。
		DevOps 工程师

清理

Task	描述	所需技能
移除 DNS 记录。	删除您之前创建的指向应用程序负载均衡器的 FQDN 的 DNS 记录。	云管理员
移除负载均衡器。	在 Amazon EC2 控制台上，选择负载均衡器，然后移除 Kubernetes 控制器为您的入口资源创建的负载均衡器。	云管理员
删除 Amazon EKS 集群。	使用以下方法删除 Amazon EKS 集群`eksctl`： `eksctl delete cluster -f ./eks.yaml`	AWS DevOps

相关资源

- Amazon EKS 上的网络负载均衡器
- 应用程序负载均衡器的目标组

其他信息

入口资源示例：

```
---
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  annotations:
    alb.ingress.kubernetes.io/healthcheck-protocol: HTTP
    alb.ingress.kubernetes.io/ssl-redirect: "443"
    alb.ingress.kubernetes.io/backend-protocol-version: "GRPC"
    alb.ingress.kubernetes.io/listen-ports: '[{"HTTP": 80}, {"HTTPS":443}]'
    alb.ingress.kubernetes.io/scheme: internet-facing
```

```
    alb.ingress.kubernetes.io/target-type: ip
    alb.ingress.kubernetes.io/certificate-arn: arn:aws:acm:<AWS-
Region>:<AccountId>:certificate/<certificate_ID>
  labels:
    app: grpcserver
    environment: dev
  name: grpcserver
  namespace: grpcserver
spec:
  ingressClassName: alb
  rules:
  - host: grpc.example.com # <----- replace this as per your host name for which the
 SSL certtficate is available in ACM
    http:
      paths:
      - backend:
          service:
            name: grpcserver
            port:
              number: 9000
        path: /
        pathType: Prefix
```

部署资源示例：

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: grpcserver
  namespace: grpcserver
spec:
  selector:
    matchLabels:
      app: grpcserver
  replicas: 1
  template:
    metadata:
      labels:
        app: grpcserver
    spec:
      containers:
      - name: grpc-demo
```

```
        image: <your_aws_account_id>.dkr.ecr.us-east-1.amazonaws.com/helloworld-
grpc:1.0   #<------- Change to the URI that the Docker image is pushed to
        imagePullPolicy: Always
        ports:
        - name: grpc-api
          containerPort: 9000
        env:
        - name: POD_IP
          valueFrom:
            fieldRef:
              fieldPath: status.podIP
      restartPolicy: Always
```

示例输出：

```
NAME             CLASS           HOSTS                          Address
 PORTS          AGE
 grpcserver      <none>      <DNS-HostName>                 <ELB-address>
 80            27d
```

部署和调试 Amazon EKS 集群

由 Svenja Raether (AWS) 和 Mathew George (AWS) 编写

摘要

容器正在成为云原生应用程序开发中的重要组成部分。Kubernetes 提供了一种有效方法来管理和编排容器。Amazon Elastic Kubernetes Service(Amazon EKS) 是一项完全托管、经过认证的 Kubernetes 合规服务，用于在 Amazon Web Services(AWS) 上构建、保护、操作和维护 Kubernetes 集群。它支持在 AWS Fargate 上运行容器组（pod），以按需提供大小合适的计算容量。

对于开发人员和管理员来说，了解运行容器化工作负载时的调试选项非常重要。此模式将引导您使用 AWS Fargate 在 Amazon EKS 上部署和调试容器。它包括创建、部署、访问、调试以及清理 Amazon EKS 工作负载。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- AWS Identity and Access Management (IAM) 角色配置有足够的权限来创建 Amazon EKS、IAM 角色和服务相关角色并与之交互
- AWS 命令行界面（AWS CLI） 已在本地计算机上安装
- eksctl
- kubectl
- Helm

限制

- 此模式为开发人员提供了针对开发环境的有用调试实践。它没有说明生产环境的最佳实践标准。
- 如果您运行的是 Windows，请使用操作系统特定命令设置环境变量。

使用的产品版本

- AWS CLI 版本 2
- kubectl 版本与您正在使用的 Amazon EKS 控制面板只有一个小版本差异
- eksctl 最新版本

- Helm v3

架构

技术堆栈

- 应用程序负载均衡器
- Amazon EKS
- AWS Fargate

目标架构

图中显示的所有资源都是通过使用从本地计算机发出的 `eksctl` 和 `kubectl` 命令来调配的。私有集群必须在私有 VPC 内的实例上运行。

目标架构由使用 Fargate 启动类型的 EKS 集群组成。这可以提供按需、大小合适的计算能力，而无需指定服务器类型。EKS 集群有一个控制面板，用于管理集群节点和工作负载。容器组（pod）被配置到跨多个可用区的私有 VPC 子网中。引用 Amazon ECR 公开映像浏览馆来检索 NGINX Web 服务器映像并将其部署到集群的 容器组（pod）。

该图显示了如何使用 `kubectl` 命令访问 Amazon EKS 控制面板以及如何使用应用程序负载均衡器访问应用程序。

.

1. Amazon Web Services Cloud 外部的本地计算机将命令发送至 Amazon EKS 托管 VPC 内的 Kubernetes 控制面板。
2. Amazon EKS 根据 Fargate 配置文件中的选择器来调度容器组（pod）。
3. 本地计算机在浏览器打开应用程序负载均衡器 URL。
4. 应用程序负载均衡器在跨多个可用区的私有子网中部署的 Fargate 集群节点中的 Kubernetes 容器组（pod）之间分配流量。

工具

Amazon Web Services

- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可帮助您在 AWS 上运行 Kubernetes，而无需安装或维护您自己的 Kubernetes 控制面板或节点。这种模式还使用 eksctl 命令行工具在 Amazon EKS 上使用 Kubernetes 集群。
- AWS Fargate 可帮助您运行容器，而无需管理服务器或亚马逊弹性计算云 (Amazon EC2) 实例。它与 Amazon Elastic Container Service（Amazon ECS）配合使用。
- 弹性负载均衡（ELB）将传入的应用程序或网络流量分发到多个目标。例如，您可以跨亚马逊弹性计算云 (Amazon EC2) 实例、容器以及一个或多个可用区中的 IP 地址分配流量。在配置 Kubernetes 入口时，此模式使用 AWS Load Balancer Controller 组件创建应用程序负载均衡器。应用程序负载均衡器在多个目标之间分配传入流量。

其他工具

- Helm 是 Kubernetes 的开源软件包管理器。在这种模式中，Helm 用于安装 AWS Load Balancer Controller。
- Kubernetes 是一个用于实现容器化应用程序的部署、扩缩和管理自动化的开源系统。
- NGINX 是一款高性能 Web 和反向代理服务器。

操作说明

创建 EKS 集群。

Task	描述	所需技能
创建文件。	使用其他信息部分中的代码，创建以下文件： • `clusterconfig-fargate.yaml` • `nginx-deployment.yaml` • `nginx-service.yaml` • `nginx-ingress.yaml` • `index.html`	应用程序开发人员、AWS 管理员、AWS DevOps

Task	描述	所需技能

设置环境变量。	**Note**
	如果命令由于之前的任务未完成而失败，请等待几秒钟，然后再次运行该命令。
	
	此模式使用 `clusterconfig-fargate.yaml` 文件中定义的 AWS 区域 和集群名称。设置与环境变量相同的值以在进一步的命令中引用它们。
	
	`export AWS_REGION="us-east-1"`
	`export CLUSTER_NAME="my-fargate"`
		应用程序开发者、AWS DevOps、AWS 系统管理员

Task	描述	所需技能
创建 EKS 集群。	要创建使用 `clusterconfig-fargate.yaml` 文件中的规格的 EKS 集群，请运行以下命令。 `eksctl create cluster -f clusterconfig-fargate.yaml` 该文件包含 `ClusterConfig`，它配置一个在 `us-east-1` 区域中命为 `my-fargate-cluster` 的新 EKS 集群和一个默认 Fargate 配置文件 (`fp-default`)。 默认 Fargate 配置文件配置有两个选择器（`default` 和 `kube-system`）。	应用程序开发人员、AWS DevOps、AWS 管理员

Task	描述	所需技能

检查已创建集群。	使用以下命令查看已创建的集群。
	`eksctl get cluster --output yaml`
	输出应为以下内容。
	`- Name: my-fargate`
	`  Owned: "True"`
	`  Region: us-east-1`
	使用 CLUSTER_NAME 检查已创建的 Fargate 配置文件。
	`eksctl get fargatepr ofile --cluster $CLUSTER_NAME --output yaml`
	此命令显示有关资源的信息。您可使用这些信息来验证已创建的集群。输出应为以下内容。
	`- name: fp-default`
	`  podExecutionRoleAR N: arn:aws:iam::<YOUR -ACCOUNT-ID>:role/ eksctl-my-fargate- cluster-FargatePod ExecutionRole-xxx`
	`  selectors:`
	`  - namespace: default`
	`  - namespace: kube- system`
	`  status: ACTIVE`
	`  subnets:`
		应用程序开发者、AWS DevOps、AWS 系统管理员

Task	描述	所需技能

	<pre>- subnet-aaa
	- subnet-bbb
	- subnet-ccc</pre>	

部署容器

Task	描述	所需技能
部署 NGINX Web 服务器。	若要在集群上应用 NGINX Web 服务器部署，请运行以下命令。 <pre>kubectl apply -f ./ nginx-deployment.yaml</pre>输出应为以下内容。 <pre>deployment.apps/nginx- deployment created</pre>部署包括从Amazon ECR 公开映像浏览馆获取的 NGINX 映像的三个副本。映像部署到默认命名空间，并在正在运行的容器组（pod）的端口 80 上公开。	应用程序开发者、AWS DevOps、AWS 系统管理员
检查部署和容器组（pod）。	（可选）检查部署。可使用以下命令验证部署的状态。 <pre>kubectl get deployment</pre>输出应为以下内容。	应用程序开发人员、AWS DevOps、AWS 管理员

Task	描述	所需技能

	`NAME READY UP-TO-DATE AVAILABLE AGE nginx-deployment 3/3 3 3 7m14s`
	
	容器组（pod）是 Kubernetes 中的可部署对象，包含一个或多个容器。运行以下命令以获取全部容器组（pod）的列表。
	
	`kubectl get pods`
	
	输出应为以下内容。
	
	`NAME READY STATUS RESTARTS AGE nginx-deployment-xxxx-aaa 1/1 Running 0 94s nginx-deployment-xxxx-bbb 1/1 Running 0 94s nginx-deployment-xxxx-ccc 1/1 Running 0 94s`	

Task	描述	所需技能

扩展部署。	要将部署从 deployment.yaml 中指定的三个副本扩展到四个副本，请使用以下命令。
	`kubectl scale deployment nginx-deployment --replicas 4`
	输出应为以下内容。
	`deployment.apps/nginx-deployment scaled`
		应用程序开发者、AWS DevOps、AWS 系统管理员

部署 AWS Load Balancer Controller

Task	描述	所需技能

设置环境变量。	描述集群的 CloudFormation 堆栈以检索有关其 VPC 的信息。
	`aws cloudformation describe-stacks --stack-name eksctl-$CLUSTER_NAME-cluster --query "Stacks[0].Outputs[?OutputKey==\`VPC\`].OutputValue"`
	输出应为以下内容。
	`[ "vpc-<YOUR-VPC-ID>"`
		应用程序开发者、AWS DevOps、AWS 系统管理员

Task	描述	所需技能
	] 复制 VPC ID，并将其导出为环境变量。 `export VPC_ID="vpc-<YOUR-VPC-ID>"`	
配置集群服务账户 IAM	使用之前的操作说明中的 AWS_REGION 和 CLUSTER_NAME 为集群创建 IAM Open ID Connect 提供程序。 `eksctl utils associate-iam-oidc-provider \ --region $AWS_REGION \ --cluster $CLUSTER_NAME \ --approve`	应用程序开发者、AWS DevOps、AWS 系统管理员

Task	描述	所需技能

下载并创建 IAM policy。	下载 AWS Load Balancer Controller 的 IAM 策略，该策略允许它 APIs 代表您向 AWS 调用。
	`curl -o iam-policy.json https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/main/docs/install/iam_policy.json`
	使用 AWS CL 在您的 Amazon Web Services account 中创建此策略。
	`aws iam create-policy \ --policy-name AWSLoadBalancerControllerIAMPolicy \ --policy-document file://iam-policy.json`
	您应当看到如下输出。
	`{ "Policy": { "PolicyName": "AWSLoadBalancerControllerIAMPolicy", "PolicyId": "<YOUR_POLICY_ID>", "Arn": "arn:aws:iam::<YOUR-ACCOUNT-ID>:policy/AWSLoa`
		应用程序开发者、AWS DevOps、AWS 系统管理员

Task	描述	所需技能

	<pre>dBalancerControlle
	rIAMPolicy",
	"Path": "/",
	"DefaultV
	ersionId": "v1",
	"Attachme
	ntCount": 0,
	"Permissi
	onsBoundaryUsageCo
	unt": 0,
	"IsAttachable":
	true,
	"CreateDate":
	"<YOUR-DATE>",
	"UpdateDate":
	"<YOUR-DATE>"
	}
	}</pre> 保存策略的 Amazon 资源名称（ARN）为 $POLICY_ARN 。 <pre>export POLICY_AR
	N=”arn:aws:iam::<Y
	OUR-ACCOUNT-ID>:po
	licy/AWSLoadBalanc
	erControllerIAMPol
	icy”</pre>	

Task	描述	所需技能

创建 IAM 服务账户。	在 kube-system 命名空间中创建名为 aws-load-balancer-controller 的 IAM 服务账户。使用您之前配置的 CLUSTER_NAME 、AWS_REGION 和 POLICY_ARN 。
	`eksctl create iamserviceaccount \ --cluster=$CLUSTER_NAME \ --region=$AWS_REGION \ --attach-policy-arn=$POLICY_ARN \ --namespace=kube-system \ --name=aws-load-balancer-controller \ --override-existing-serviceaccounts \ --approve`
	验证创建。
	`eksctl get iamserviceaccount \ --cluster $CLUSTER_NAME \ --name aws-load-balancer-controller \ --namespace kube-system \ --output yaml`
	输出应为以下内容。
	`- metadata:`
		应用程序开发者、AWS DevOps、AWS 系统管理员

Task	描述	所需技能

	<pre> name: aws-load-
	balancer-controller
	    namespace: kube-
	system
	  status:
	    roleARN: arn:aws:i
	am::<YOUR-ACCOUNT-
	ID>:role/eksctl-my
	-fargate-addon-iam
	serviceaccount-ku-
	Role1-<YOUR-ROLE-ID>
	  wellKnownPolicies:
	    autoScaler: false
	    awsLoadBalancerCon
	troller: false
	    certManager: false
	    ebsCSIController:
	 false
	    efsCSIController:
	 false
	    externalDNS: false
	    imageBuilder: false</pre>	

Task	描述	所需技能

安装 AWS Load Balancer Controller。	更新 Helm 存储库。
	`helm repo update`
	将 Amazon EKS 图表存储库添加到 Helm 存储库中。
	`helm repo add eks https://aws.github.io/eks-charts`
	在后台应用 AWS Loa d Balancer Controller eks- chart 使用的 Kubernetes 自定义资源定义 (CRDs)。
	`kubectl apply -k "github.com/aws/eks-charts/stable/aws-load-balancer-controller//crds?ref=master"`
	输出应为以下内容。
	`customresourcedefinition.apiextensions.k8s.io/ingressclassparams.elbv2.k8s.aws created`
	`customresourcedefinition.apiextensions.k8s.io/targetgroupbindings.elbv2.k8s.aws created`
		应用程序开发者、AWS DevOps、AWS 系统管理员

Task	描述	所需技能

	使用之前设置的环境变量安装 Helm 图表。
	
	`helm install aws-load-balancer-controller eks/aws-load-balancer-controller \ --set clusterName=$CLUSTER_NAME \ --set serviceAccount.create=false \ --set region=$AWS_REGION \ --set vpcId=$VPC_ID \ --set serviceAccount.name=aws-load-balancer-controller \ -n kube-system`
	
	输出应为以下内容。
	
	`NAME: aws-load-balancer-controller LAST DEPLOYED: <YOUR-DATE> NAMESPACE: kube-system STATUS: deployed REVISION: 1 TEST SUITE: None NOTES: AWS Load Balancer controller installed!`	

Task	描述	所需技能

创建 NGINX 服务。	使用 `nginx-service.yaml` 文件创建用于公开 NGINX 容器组（pod）的服务。
	`kubectl apply -f nginx-service.yaml`
	输出应为以下内容。
	`service/nginx-service created`
		应用程序开发者、AWS DevOps、AWS 系统管理员

创建 Kubernetes 入口资源。	使用 `nginx-ingress.yaml` 文件创建一个用于公开 Kubernetes NGINX 入口的服务。
	`kubectl apply -f nginx-ingress.yaml`
	输出应为以下内容。
	`ingress.networking.k8s.io/nginx-ingress created`
		应用程序开发者、AWS DevOps、AWS 系统管理员

Task	描述	所需技能

获取负载均衡器 URL。	若要检索入口信息，请使用以下命令。
	`kubectl get ingress nginx-ingress`
	输出应为以下内容。
	`NAME CLASS HOSTS ADDRESS PORTS AGE`
	`nginx-ingress <none> * k8s-default-nginxing-xxx.us-east-1.elb.amazonaws.com 80 80s`
	从输出中复制 ADDRESS（例如 `k8s-default-nginxing-xxx.us-east-1.elb.amazonaws.com`），然后将其粘贴到浏览器中以访问该 `index.html` 文件。
		应用程序开发者、AWS DevOps、AWS 系统管理员

调试正在运行的容器

Task	描述	所需技能
选择容器组（pod）。	列出所有容器组（pod），然后复制所需容器组（pod）的名称。 `kubectl get pods`	应用程序开发者、AWS DevOps、AWS 系统管理员

Task	描述	所需技能

	输出应为以下内容。
	
	`NAME READY STATUS RESTARTS AGE nginx-deployment-xxxx-aaa 1/1 Running 0 55m nginx-deployment-xxxx-bbb 1/1 Running 0 55m nginx-deployment-xxxx-ccc 1/1 Running 0 55m nginx-deployment-xxxx-ddd 1/1 Running 0 42m`
	
	此命令列出了现有的容器组（pod）和其他信息。
	
	如果您对特定的容器组（pod）感兴趣，请为 POD_NAME 变量填写您感兴趣的容器组（pod）的名称或将其设置为环境变量。否则，请省略此参数，以查找所有资源。
	
	`export POD_NAME="nginx-deployment-<YOUR-POD-NAME>"`	

Task	描述	所需技能
访问日志。	从要调试的容器组（pod）获取日志。 `kubectl logs $POD_NAME`	应用程序开发者、AWS 系统管理员、AWS DevOps

转发 NGINX 端口。	使用端口转发将用于访问 NGINX Web 服务器的容器组（pod）端口映射到本地计算机上的端口。
	`kubectl port-forward deployment/nginx-deployment 8080:80`
	在浏览器中打开以下 URL。
	`http://localhost:8080`
	该 `port-forward` 命令提供对 `index.html` 文件的访问权限，而无需通过负载均衡器将其公开。这非常适用于在调试时访问正在运行的应用程序。您可以通过按键盘命令 Ctrl+C 来停止端口转发。
		应用程序开发者、AWS DevOps、AWS 系统管理员

Task	描述	所需技能
在容器组（pod）中运行命令。	要查看当前 `index.html` 文件，请使用以下命令。 `kubectl exec $POD_NAME -- cat /usr/share/nginx/html/index.html` 您可以使用 `exec` 命令直接在容器组（pod）中发出任何命令。这对于调试正在运行的应用程序非常有用。	应用程序开发者、AWS DevOps、AWS 系统管理员

将文件复制到容器组（pod）。	移除此容器组（pod）上的默认 `index.html` 文件。
	`kubectl exec $POD_NAME -- rm /usr/share/nginx/html/index.html`
	将自定义的本地文件 `index.html` 上传到容器组（pod）。
	`kubectl cp index.html $POD_NAME:/usr/share/nginx/html/`
	您可以使用 `cp` 命令将文件直接更改或添加至任何容器组（pod）。
		应用程序开发者、AWS DevOps、AWS 系统管理员

Task	描述	所需技能

使用端口转发显示更改。	使用端口转发验证您对此容器组（pod）所做的更改。
	`kubectl port-forward pod/$POD_NAME 8080:80`
	在浏览器中打开以下 URL。
	`http://localhost:8080`
	对 `index.html` 文件所做的更改应在浏览器中可见。
		应用程序开发者、AWS DevOps、AWS 系统管理员

删除资源

Task	描述	所需技能

删除负载均衡器。	删除入口。
	`kubectl delete ingress/nginx-ingress`
	输出应为以下内容。
	`ingress.networking.k8s.io "nginx-ingress" deleted`
	删除服务。
	`kubectl delete service/nginx-service`
	输出应为以下内容。
		应用程序开发者、AWS DevOps、AWS 系统管理员

Task	描述	所需技能

	`service "nginx-service" deleted`
	
	删除负载均衡器控制器。
	
	`helm delete aws-load-balancer-controller -n kube-system`
	
	输出应为以下内容。
	
	`release "aws-load-balancer-controller" uninstalled`
	
	删除服务账户。
	
	`eksctl delete iamserviceaccount --cluster $CLUSTER_NAME --namespace kube-system --name aws-load-balancer-controller`	

删除部署。	使用以下命令删除部署资源。
	
	`kubectl delete deploy/nginx-deployment`
	
	输出应为以下内容。
	
	`deployment.apps "nginx-deployment" deleted`
		应用程序开发者、AWS DevOps、AWS 系统管理员

Task	描述	所需技能
请删除集群。	使用以下命令删除 EKS 集群，其中 `my-fargate` 是集群名称。 `eksctl delete cluster --name $CLUSTER_NAME` 此命令删除整个集群，包括所有关联资源。	应用程序开发者、AWS DevOps、AWS 系统管理员
删除 IAM policy。	使用 AWS CL 删除之前创建的策略。 `aws iam delete-policy --policy-arn $POLICY_ARN`	应用程序开发人员、AWS 管理员、AWS DevOps

故障排除

事务	解决方案
创建集群时收到一条错误消息，指出您的目标可用区没有足够的容量来支持集群。您应该看到类似于以下内容的消息。 `Cannot create cluster 'my-fargate' because us-east-1e, the targeted availability zone, does not currently have sufficient capacity to support the cluster. Retry and choose from these availability zones: us-east-1a, us-east-1b, us-east-1c, us-east-1d, us-east-1f`	根据错误消息使用推荐的可用区重新创建集群。在 `clusterconfig-fargate.yaml` 文件的最后一行指定可用区列表（例如 `availabilityZones: ["us-east-1a", "us-east-1b", "us-east-1c"]` ）。

相关资源

- Amazon EKS 文档
- Amazon EKS 上的应用程序负载均衡
- EKS 最佳实践指南
- AWS Load Balancer Controller 文档
- eksctl 文档
- Amazon ECR 公开映像浏览馆 NGINX 映像
- Helm 文档
- 调试正在运行的容器组（pod）（Kubernetes 文档）
- Amazon EKS 研讨会
- EKS 集群创建错误

其他信息

clusterconfig-fargate.yaml

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-fargate
  region: us-east-1

fargateProfiles:
  - name: fp-default
    selectors:
      - namespace: default
      - namespace: kube-system
```

nginx-deplooment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: "nginx-deployment"
  namespace: "default"
spec:
```

```
  replicas: 3
  selector:
    matchLabels:
      app: "nginx"
  template:
    metadata:
      labels:
        app: "nginx"
    spec:
      containers:
      - name: nginx
        image: public.ecr.aws/nginx/nginx:latest
        ports:
        - containerPort: 80
```

nginx-service.yaml

```
apiVersion: v1
kind: Service
metadata:
  annotations:
    alb.ingress.kubernetes.io/target-type: ip
  name: "nginx-service"
  namespace: "default"
spec:
  ports:
  - port: 80
    targetPort: 80
    protocol: TCP
  type: NodePort
  selector:
    app: "nginx"
```

nginx-ingress.yaml

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  namespace: "default"
  name:  "nginx-ingress"
  annotations:
    kubernetes.io/ingress.class: alb
    alb.ingress.kubernetes.io/scheme: internet-facing
```

```
spec:
  rules:
    - http:
        paths:
        - path: /
          pathType: Prefix
          backend:
            service:
              name: "nginx-service"
              port:
                number: 80
```

index.html

```
<!DOCTYPE html>
<html>

<body>
    <h1>Welcome to your customized nginx!</h1>
    <p>You modified the file on this running pod</p>
</body>

</html>
```

使用 Elastic Beanstalk 部署容器

创建者：Thomas Scott (AWS) 和 Jean-Baptiste Guillois (AWS)

摘要

在 Amazon Web Services（AWS）Cloud 上，AWS Elastic Beanstalk 支持 Docker 作为可用平台，因此容器可以在创建的环境中运行。此模式展示了如何使用 Elastic Beanstalk 服务部署容器。此模式的部署将使用基于 Docker 平台的 Web 服务器环境。

要使用 Elastic Beanstalk 部署和扩展 Web 应用程序和服务，您需要上传代码，部署就会自动处理。还包括容量预调配、负载均衡、自动扩缩和应用程序运行状况监控。当您使用 Elastic Beanstalk 时，您可以完全控制它代表您创建的 AWS 资源。Elastic Beanstalk 不收取额外费用。您只需为用于存储和运行应用程序的 AWS 资源付费。

此模式包括使用 AWS Elastic Beanstalk 命令行界面（EB CLI）和 AWS 管理控制台 进行部署的说明。

使用案例

Elastic Beanstalk 的使用场景包括：

- 部署原型环境来演示前端应用程序。（此模式以 Dockerfile 为例。）
- 部署 API 来处理给定域名的 API 请求。
- 使用 Docker-Compose 部署编排解决方案（在此模式中 `docker-compose.yml` 未用作实际示例）。

先决条件和限制

先决条件

- Amazon Web Services account
- 本地已安装 AWS EB CLI
- Docker 已安装在本地机器上

限制

- 在免费套餐中，每个 IP 地址的 Docker 拉取限制为每 6 小时拉取 100 次。

架构

目标技术堆栈

- 亚马逊弹性计算云 (Amazon EC2) 实例
- 安全组
- 应用程序负载均衡器
- 自动扩缩组

目标架构

自动化和扩缩

AWS Elastic Beanstalk 可以根据发出的请求数自动扩缩。为环境创建的 AWS 资源包括一个 Application Load Balancer、一个 Auto Scaling 组和一个或多个 Amazon EC2 实例。

负载均衡器位于作为 Auto Scaling 组一部分的 Amazon EC2 实例的前面。Amazon A EC2 uto Scaling 会自动启动额外的亚马逊 EC2 实例，以适应应用程序不断增加的负载。如果您的应用程序的负载减少，Amazon A EC2 uto Scaling 会停止实例，但它会使至少一个实例保持运行。

自动扩缩触发器

您的 Elastic Beanstalk 环境中的 Auto Scaling 组使用 CloudWatch 两个亚马逊警报来启动扩展操作。当每个实例的平均出站网络流量在 5 分钟时间段内高于 6 MB 或低于 2 MB 时，默认触发器将扩展。要有效地使用 Amazon A EC2 uto Scaling，请配置适合您的应用程序、实例类型和服务要求的触发器。您可以基于若干个统计数据 (包括延迟、磁盘 I/O、CPU 使用率和请求计数) 来进行扩展。有关更多信息，请参阅自动扩缩触发器。

工具

Amazon Web Services

- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- AWS EB 命令行界面 (EB CLI) 是一个可用来创建、配置和管理 Elastic Beanstalk 环境的命令行客户端。
- 弹性负载均衡将传入的应用程序或网络流量分发到多个目标。例如，您可以跨亚马逊弹性计算云 (Amazon EC2) 实例、容器以及一个或多个可用区中的 IP 地址分配流量。

其他服务

- Docker 将软件打包成称为容器的标准化单元，其中包括库、系统工具、代码和运行时系统。

代码

此模式的代码可在 GitHub 集群示例应用程序存储库中找到。

操作说明

使用 Dockerfile 构建

Task	描述	所需技能
克隆远程存储库。	• 要克隆存储库，请运行 `git clone https://github.com/aws-samples/cluster-sample-app.git` 命令。< /p>	应用程序开发人员、AWS 管理员、AWS DevOps
初始化 Elastic Beanstalk Docker 项目。	1. 在根目录下创建一个名为 `aws.json` 的文件。 2. 在 `aws.json` 文件中，添加以下代码。 `{ "AWSEBDockerrunVersion":"1", "Image":{ "Name":"cluster-sample-app" }, "Ports":[ { "ContainerPort":80 , "HostPort":8080 }`	应用程序开发人员、AWS 管理员、AWS DevOps

Task	描述	所需技能

	``
	        ]
	    }
	``
	3. 在项目的根目录下运行 `eb init -p docker` 命令。	

在本地测试项目。	1. 在项目的根目录下运行 `eb local run` 命令。 2. 导航到 `http://localhost` ，测试应用程序。	应用程序开发人员、AWS 管理员、AWS DevOps

使用 EB CLI 进行部署

Task	描述	所需技能
运行部署命令	1. 在项目的根目录下运行 `eb create docker-sample-cluster-app` 命令。	应用程序开发人员、AWS 管理员、AWS DevOps
访问已部署的版本。	部署命令完成后，使用该 `eb open`命令访问项目。	应用程序开发人员、AWS 管理员、AWS DevOps

使用控制台进行部署

Task	描述	所需技能
使用浏览器部署应用程序。	1. 打开 控制台。 2. 导航到 Elastic Beanstalk 控制台。 3. 选择创建应用程序。 4. 在应用程序名称中，输入 Cluster-Sample-App。	应用程序开发人员、AWS 管理员、AWS DevOps

Task	描述	所需技能
	5. 选择 Docker 作为平台。 6. 选择上传代码。 7. 选择本地 .zip 文件（位于克隆项目的根目录中）或公共的 Amazon Simple Storage Service（Amazon S3）URL。	
访问已部署的版本。	部署后，访问已部署的应用程序，然后选择提供的 URL。	应用程序开发人员、AWS 管理员、AWS DevOps

相关资源

- Web 服务器环境
- 在 macOS 上安装 EB CLI
- 手动安装 EB CLI

其他信息

使用 Elastic Beanstalk 的好处

- 自动预调配基础设施
- 自动管理底层平台
- 自动修补和更新以支持应用程序
- 自动扩缩应用程序
- 能够自定义节点数
- 能够在需要时访问基础设施组件
- 与其他容器部署解决方案相比，易于部署

使用 Lambda 函数、Amazon VPC 和无服务器架构生成静态出站 IP 地址

创建者：Thomas Scott (AWS)

摘要

此模式描述如何使用无服务器架构在 Amazon Web Services (AWS) Cloud 中生成静态出站 IP 地址。如果您的组织想要使用安全文件传输协议 (SFTP) 将文件发送到单独的业务实体，则可以从此方法中受益。这意味着业务实体必须有权访问允许文件通过防火墙的 IP 地址。

该模式的方法可以帮助您创建使用弹性 IP 地址作为出站 IP 地址的 AWS Lambda 函数。通过遵循此模式中的步骤，您可以创建 Lambda 函数和虚拟私有云（VPC），通过具有静态 IP 地址的互联网网关路由出站流量。要使用静态 IP 地址，您可将 Lambda 函数附加到 VPC 及其子网。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- AWS Identity and Access Management (IAM) 权限，用于创建和部署 Lambda 函数，以创建 VPC 及其子网。有关这方面的更多信息，请参阅 AWS Lambda 文档中的执行角色和用户权限。
- 如果您计划使用基础设施即代码 (IaC) 来实现此模式的方法，则需要一个集成开发环境 (IDE)，例如 AWS Cloud9。有关这方面的更多信息，请参阅 AWS Cloud9 文档中的 AWS Cloud9 是什么？。

架构

下图显示此模式的无服务器架构。

图表显示了以下工作流：

1. 出站流量离开 `Public subnet 1` 中的 `NAT gateway 1`。
2. 出站流量离开 `Public subnet 2` 中的 `NAT gateway 2`。
3. Lambda 函数可在 `Private subnet 1` 或 `Private subnet 2` 中运行。
4. `Private subnet 1` 和 `Private subnet 2` 将流量路由到公有子网中的 NAT 网关。
5. NAT 网关将出站流量从公共子网发送至互联网网关。
6. 出站数据从互联网网关传输至外部服务器。

技术堆栈

- Lambda
- Amazon Virtual Private Cloud（Amazon VPC）

自动化和扩缩

您可以通过在不同的可用区使用两个公有子网和两个私有子网，确保高可用性 (HA)。即使一个可用区不可用，该模式的解决方案仍能继续发挥作用。

工具

- AWS Lambda – AWS Lambda 是一项计算服务，支持无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。您只需为消耗的计算时间付费 - 代码未运行时不产生费用。
- Amazon VPC – Amazon Virtual Private Cloud (Amazon VPC) 预调配 Amazon Web Services Cloud 的逻辑隔离部分，您可以在其中启动您定义的虚拟网络中的 AWS 资源。这个虚拟网络与您在数据中心中运行的传统网络极其相似，并会为您提供使用Amazon云科技可扩展基础设施的优势。

操作说明

创建新的 VPC

Task	描述	所需技能
创建新的 VPC。	登录 AWS 管理控制台，打开 Amazon VPC 控制台，然后创建一个名为 Lambda VPC IPv4 CIDR 范围的 VPC。`10.0.0.0/25` 有关创建 VPC 的更多信息，请参阅 Amazon VPC 文档中的 Amazon VPC 入门。	AWS 管理员

创建两个公有子网

Task	描述	所需技能
创建第一个公有子网。	1. 在 Amazon VPC 控制台上，选择子网，然后选择创建子网。 2. 对于名称标签，输入 `public-one` 。 3. 对于 VPC，选择 Lambda VPC。 4. 选择一个可用区并记录它。 5. 对于 IPv4 CIDR 块，输入，`10.0.0.0/28` 然后选择创建子网。	AWS 管理员
创建第二个公有子网。	1. 在 Amazon VPC 控制台上，选择子网，然后选择创建子网。 2. 对于名称标签，输入 `public-two` 。 3. 对于 VPC，选择 Lambda VPC。 4. **Important** 选择一个可用区并记录它。：您不能使用包含`public-one` 子网的可用区。 5. 对于 IPv4 CIDR 块，输入，`10.0.0.16/28` 然后选择创建子网。	AWS 管理员

创建两个私有子网

Task	描述	所需技能
创建第一个私有子网。	1. 在 Amazon VPC 控制台上，选择子网，然后选择创建子网。 2. 对于名称标签，输入 `private-one` 。 3. 对于 VPC，选择 Lambda VPC。 4. 选择包含您之前创建的 `public-one` 子网的可用区。 5. 对于 IPv4 CIDR 块，输入，`10.0.0.32/28` 然后选择创建子网。	AWS 管理员
创建第二个私有子网。	1. 在 Amazon VPC 控制台上，选择子网，然后选择创建子网。 2. 对于名称标签，输入 `private-two` 。 3. 对于 VPC，选择 Lambda VPC。 4. 选择包含您之前创建的 `public-two` 子网的相同可用区。 5. 对于 IPv4 CIDR 块，输入，`10.0.0.64/28` 然后选择创建子网。	AWS 管理员

为您的 NAT 网关创建两个弹性 IP 地址

Task	描述	所需技能
创建第一个弹性 IP 地址。	1. 在 Amazon VPC 控制台上，选择弹性 IPs，然后选择分配新地址。 2. 选择分配并为您新创建的弹性 IP 地址记录分配 ID。 **Note** 此弹性 IP 地址用于您的第一个 NAT 网关。	AWS 管理员
创建第二个弹性 IP 地址。	1. 在 Amazon VPC 控制台上，选择弹性 IPs，然后选择分配新地址。 2. 选择分配，并为第二个弹性 IP 地址记录分配 ID。 **Note** 此弹性 IP 地址用于您的第二个 NAT 网关。	AWS 管理员

创建互联网网关

Task	描述	所需技能
创建互联网网关。	1. 在 Amazon VPC 控制台上，选择互联网网关，然后选择创建互联网网关。	AWS 管理员

Task	描述	所需技能
	2. 输入 `Lambda internet gateway` 作为名称，然后选择创建互联网网关。务必记录互联网网关 ID。	
将互联网网关连接到 VPC。	选择刚刚创建的 Internet 网关，然后选择 Actions, Attach to VPC (操作，附加到 VPC)。	AWS 管理员

创建两个 NAT 网关

Task	描述	所需技能
创建第一个 NAT 网关。	1. 在 Amazon VPC 控制台上，选择 NAT 网关，然后选择创建 NAT 网关。 2. 输入 `nat-one`作为 NAT 网关名称。 3. 选择 `public-one` 作为子网，以在其中创建 NAT 网关。 4. 对于连接类型，选择公共。 5. 对于弹性 IP 分配 ID，选择您之前创建的第一个弹性 IP 地址，并将其与 NAT 网关关联。 6. 选择创建 NAT 网关。	AWS 管理员
创建第二个 NAT 网关。	1. 在 Amazon VPC 控制台上，选择 NAT 网关，然后选择创建 NAT 网关。 2. 输入 `nat-two`作为 NAT 网关名称。	AWS 管理员

Task	描述	所需技能
	3. 选择 public-two 作为子网，以在其中创建 NAT 网关。 4. 对于连接类型，选择公共。 5. 对于弹性 IP 分配 ID，选择您之前创建的第二个弹性 IP 地址，并将其与 NAT 网关关联。 6. 选择创建 NAT 网关。	

为您的公有子网与私有子网创建路由表

Task	描述	所需技能
为公有子网创建路由表。	1. 在 Amazon VPC 控制台中，选择路由表，然后选择创建路由表。 2. 输入 public-one-subnet 作为路由表名称，然后选择创建路由表。 3. 选择 public-one-subnet 路由表，选择编辑路由，然后选择添加路由。 4. 在目的地框中指定 0.0.0.0，然后在目标列表中选择互联网网关 ID。 5. 在子网关联选项卡上，选择编辑子网关联，选择 CIDR 范围为 10.0.0.0/28 的 public-one 子网，然后选择保存关联。 6. 选择保存更改。	AWS 管理员

Task	描述	所需技能
为 public-two 子网创建路由表。	1. 在 Amazon VPC 控制台中，选择路由表，然后选择创建路由表。 2. 输入 `public-two-subnet` 作为路由表名称，然后选择创建路由表。 3. 选择 `public-two-subnet` 路由表，选择编辑路由，然后选择添加路由。 4. 在目的地框中指定 `0.0.0.0`，然后在目标列表中选择互联网网关 ID。 5. 在子网关联选项卡上，选择编辑子网关联，选择 CIDR 范围为 `10.0.0.16/28` 的 `public-two` 子网，然后选择保存关联。 6. 选择保存更改。	AWS 管理员

Task	描述	所需技能
为 private-one 子网创建路由表。	1. 在 Amazon VPC 控制台中，选择路由表，然后选择创建路由表。 2. 输入 `private-one-subnet` 作为路由表名称，然后选择创建路由表。 3. 选择 `private-one-subnet` 路由表，选择编辑路由，然后选择添加路由。 4. 在目的地框中指定 `0.0.0.0`，然后在目标列表中选择 `public-one` 子网中的 NAT 网关。 5. 在子网关联选项卡上，选择编辑子网关联，选择 CIDR 范围为 `10.0.0.32/28` 的 `private-one` 子网，然后选择保存关联。 6. 选择保存更改。	AWS 管理员

Task	描述	所需技能
为 private-two 子网创建路由表。	1. 在 Amazon VPC 控制台中，选择路由表，然后选择创建路由表。 2. 输入 `private-two-subnet` 作为路由表名称，然后选择创建路由表。 3. 选择 `private-two-subnet` 路由表，选择编辑路由，然后选择添加路由。 4. 在目的地框中指定 `0.0.0.0`，然后在目标列表中选择 `public-two` 子网中的 NAT 网关。 5. 在子网关联选项卡上，选择编辑子网关联，选择 CIDR 范围为 `10.0.0.64/28` 的 `private-two` 子网，然后选择保存关联。 6. 选择保存更改。	AWS 管理员

创建 Lambda 函数，将其添加至 VPC，然后测试解决方案

Task	描述	所需技能
新建 Lambda 函数。	1. 打开 AWS Lambda 控制台，然后选择创建函数。 2. 在基本信息下方，在函数名称下输入 `Lambda test`，然后在运行时系统下选择所选语言。 3. 选择 Create function (创建函数)。	AWS 管理员

Task	描述	所需技能
将 Lambda 函数添加到您的 VPC。	1. 在 AWS Lambda 控制台上，选择函数，然后选择您之前创建的函数。 2. 选择 Configuration（配置），然后选择 VPC。 3. 选择编辑，然后选择 Lambda VPC 和两个私有子网。 4. 选择用于测试的默认安全组，然后选择保存。	AWS 管理员
编写代码来调用外部服务。	1. 使用您选择的编程语言编写代码，调用返回您的 IP 地址的外部服务。 2. 验证返回的 IP 地址是否与您的一个弹性 IP 地址相匹配。	AWS 管理员

相关资源

- 配置 Lambda 函数以访问 VPC 中的资源

迁移到 Amazon ECR 存储库时自动识别重复的容器映像

由 Rishabh Yadav (AWS) 和 Rishi Singla (AWS) 创作

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

该模式提供了一种自动解决方案，用于识别存储在不同容器存储库中的图像是否重复。当您计划将镜像从其他容器存储库迁移到亚马逊弹性容器注册表 (Amazon ECR) Container Registry 时，此检查非常有用。

有关基础信息，该模式还描述了容器镜像的组件，例如镜像摘要、清单和标签。当您计划迁移到 Amazon ECR 时，您可能会决定通过比较镜像摘要来跨容器注册表同步您的容器镜像。在迁移容器映像之前，您需要检查这些映像是否已存在于 Amazon ECR 存储库中，以防止重复。但是，通过比较图像摘要来检测重复可能很困难，这可能会导致在初始迁移阶段出现问题。 此模式比较了存储在不同容器注册表中的两个相似图像的摘要，并解释了摘要变化的原因，以帮助您准确比较图像。

先决条件和限制

- 活跃的 AWS 账户
- 访问 A mazon ECR 公共注册表
- 熟悉以下内容： AWS 服务
  - AWS CodeCommit
  - AWS CodePipeline
  - AWS CodeBuild
  - AWS Identity and Access Management (IAM)
  - Amazon Simple Storage Service（ Amazon S3 ）
- 已配置的 CodeCommit 凭证（参见说明）

架构

容器镜像组件

下图说明了容器镜像的某些组件。图后描述了这些组件。

术语和定义

以下术语是在开放容器倡议 (OCI) 图像规范中定义的。

- 注册表：一种用于图像存储和管理的服务。
- 客户端：一种与注册管理机构通信并处理本地图像的工具。
- 推送：将图像上传到注册表的过程。
- 提取：从注册表下载图像的过程。
- Blob：由注册表存储并可通过摘要进行寻址的二进制形式的内容。
- 索引：一种用于识别不同计算机平台（例如 x86-64 或 ARM 64 位）或媒体类型的多个图像清单的结构。有关更多信息，请参阅 OCI 图像索引规范。
- 清单：一个 JSON 文档，用于定义通过清单端点上传的图像或构件。清单可以使用描述符来引用存储库中的其他 Blob。有关更多信息，请参阅 OCI 图像清单规范。
- 文件系统层：系统库和图像的其他依赖项。
- 配置：包含工件元数据并在清单中引用的 blob。有关更多信息，请参阅 OCI 映像配置规范。
- 对象或构件：一种概念性内容项目，存储为 blob，并与带有配置的随附清单相关联。
- 摘要：根据清单内容的加密哈希值创建的唯一标识符。图像摘要有助于唯一标识不可变的容器镜像。使用摘要提取图像时，每次在任何操作系统或架构上都将下载相同的图像。有关更多信息，请参阅 OCI 图像规范。
- 标签：人类可读的清单标识符。与不可变的图像摘要相比，标签是动态的。指向图像的标签可以更改并从一张图像移动到另一张图像，尽管底层图像摘要保持不变。

目标架构

下图显示了该模式提供的解决方案的高级架构，该架构通过比较存储在 Amazon ECR 和私有存储库中的图像来识别重复的容器映像。

工具

AWS 服务

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在资源的整个生命周期中跨地区对其 AWS 账户 进行管理。
- AWS CodeBuild是一项完全托管的生成服务，可帮助您编译源代码、运行单元测试和生成可随时部署的工件。

- AWS CodeCommit是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。
- AWS CodePipeline帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件更改所需的步骤。
- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。

代码

此模式的代码可在存储库中找到，用于识别 GitHub 存储库之间重复的容器镜像。

最佳实践

- AWS CloudFormation 最佳实践
- AWS CodePipeline 最佳实践

操作说明

从 Amazon ECR 公有和私有存储库中提取容器镜像

Task	描述	所需技能

从 Amazon ECR 公共存储库中提取镜像。	在终端上，运行以下命令`amazonlinux` 从 Amazon ECR 公共存储库中提取映像。
	`$~ % docker pull public.ecr.aws/amazonlinux/amazonlinux:2018.03`
	将图像拉到本地计算机后，您将看到以下拉取摘要，它代表图像索引。
	`2018.03: Pulling from amazonlinux/amazonlinux`
		应用程序开发人员、AWS DevOps、AWS 管理员

Task	描述	所需技能
	4ddc0f8d367f: Pull complete Digest: sha256:f9 72d24199508c52de7a d37a298bda35d8a1bd 7df158149b381c03f6 c6e363b5 Status: Downloade d newer image for public.ecr.aws/ama zonlinux/amazonlin ux:2018.03 public.ecr.aws/a mazonlinux/amazonl inux:2018.03	

Task	描述	所需技能

将镜像推送到 Amazon ECR 私有存储库。	1. 创建名为美国东部（弗吉尼亚北部）区域`test_ecr_repository` 的私有 Amazon ECR 存储库 (`us-east-1`)。
	`$~ % aws ecr get-login -password --region us-east-1 | docker login --username AWS --password-stdin <account-id>.dkr.ecr.us-east-1.amazonaws.com`
	`Login Succeeded`
	其中`<account-id>` 指的是你的 AWS 账户。
	2. 标记您之前提取的本地图像。使用该值`public.ecr.aws/amazonlinux/amazonlinux:2018.03` 并将其推送到 Amazon ECR 私有存储库。
	`$~ % docker tag public.ecr.aws/amazonlinux/amazonlinux:2018.03 <account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest`
	`$~ % docker push <account-id>.dkr.ecr.us-east-1.amazo`
		AWS 管理员、AWS DevOps、应用程序开发者

Task	描述	所需技能
	naws.com/test_ecr_repository:latest 当您将映像推送到 Amazon ECR 存储库时，Docker 将推送底层映像而不是图像索引。 The push refers to repository [<account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository] d5655967c2c4: Pushed latest: digest: sha256:52db9000073d93b9bdee6a7246a68c35a741aaade05a8f4febba0bf795cdac02 size: 529	

Task	描述	所需技能

从 Amazon ECR 私有存储库中提取相同的图像。	1. 从终端运行以下命令将您之前推送到 Amazon ECR 私有存储库的映像拉取。
	
	`$~ % docker pull <account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest`
	`latest: Pulling from test_ecr_repository`
	`Digest: sha256:52db9000073d93b9bdee6a7246a68c35a741aaade05a8f4febba0bf795cdac02`
	`Status: Image is up to date for <account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest`
	`<account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest`
	
	此图像的摘要与您推送到 Amazon ECR 私有存储库的图像摘要相匹配，并且代表底层图像。此值与您从公共存储库中提取的图像索引不匹配。
	2. 要进行验证，请按摘要检索图像索引。
	
	`curl -k -H “Authorization: Bearer $TOKEN”`
		应用程序开发人员、AWS DevOps、AWS 管理员

Task	描述	所需技能

	https://public.ecr.aws/v2/amazonlinux/amazonlinux/manifests/sha256:f972d24199508c52de7ad37a298bda35d8a1bd7df158149b381c03f6c6e363b55
	{
	“schemaVersion”: 2,
	“mediaType”: “application/vnd.docker.distribution.manifest.list.v2+json”,
	“manifests”: [
	{
	“mediaType”: “application/vnd.docker.distribution.manifest.v2+json”,
	“size”: 529,
	“digest”: “sha256:52db9000073d93b9bdee6a7246a68c35a741aaade05a8f4febba0bf795cdac02",
	“platform”: {
	“architecture”: “amd64”,
	“os”: “linux”
	}
	}
	]
	}	

比较图像清单

Task	描述	所需技能

查找存储在 Amazon ECR 公共存储库中的图像的清单。	在终端上运行以下命令，`public.ecr.aws/amazonlinux/amazonlinux:2018.03` 从 Amazon ECR 公共存储库中提取图像清单。
	
	`$~ % docker manifest inspect public.ecr.aws/amazonlinux/amazonlinux:2018.03`
	`{`
	`"schemaVersion": 2,`
	`"mediaType": "application/vnd.docker.distribution.manifest.list.v2+json",`
	`"manifests": [`
	`{`
	`"mediaType": "application/vnd.docker.distribution.manifest.v2+json",`
	`"size": 529,`
	`"digest": "sha256:52db9000073d93b9bdee6a7246a68c35a741aaade05a8f4febba0bf795cdac02",`
	`"platform": {`
	`"architecture": "amd64",`
	`"os": "linux"`
	`}`
	`}`
	`]`
		AWS 管理员、AWS DevOps、应用程序开发者

Task	描述	所需技能
	} }	

Task	描述	所需技能

查找存储在 Amazon ECR 私有存储库中的图像的清单。	在终端运行以下命令从 Amazon ECR 私<account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest 有存储库中提取图像清单。
	
	`$~ % docker manifest inspect <account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest`
	
	`{ "schemaVersion": 2, "mediaType": "application/vnd.docker.distribution.manifest.v2+json", "config": { "mediaType": "application/vnd.docker.container.image.v1+json", "size": 1477, "digest": "sha256:f7cee5e1af28ad4e147589c474d399b12d9b551ef4c3e11e02d982fce5eebc68" }, "layers": [{ "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",`
		AWS DevOps、AWS 系统管理员、应用程序开发者

Task	描述	所需技能
	"size": 62267075, "digest": "sha256:4ddc0f8d367f424871a060e2067749f32bd36a91085e714dcb159952f2d71453" }] }	

Task	描述	所需技能
将 Docker 提取的摘要与 Amazon ECR 私有存储库中镜像的清单摘要进行比较。	另一个问题是，为什么 docker pull 命令提供的摘要与清单的图像摘要不同。`<account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest` 用于 docker pull 的摘要表示存储在注册表中的镜像清单的摘要。此摘要被视为哈希链的根，因为清单包含将要下载并导入到 Docker 中的内容的哈希值。 Docker 中使用的镜像 ID 可以在此清单中找到。`config.digest` 这代表了 Docker 使用的镜像配置。因此，你可以说清单是信封，图像是信封的内容。清单摘要始终与图片 ID 不同。但是，特定的清单应始终生成相同的图片 ID。由于清单摘要是一个哈希链，因此我们不能保证给定图片 ID 的哈希链总是相同的。在大多数情况下，它会生成相同的摘要，尽管 Docker 无法保证。清单摘要中可能存在的区别源于 Docker 没有在本地存储用 gzip 压缩的 blob。因此，尽管未压缩的内容保持不变，但导出图层可能会生成不同的摘要。图像 ID 验证未压缩的内容是否相同；也就是说，图像	AWS DevOps、AWS 系统管理员、应用程序开发者

Task	描述	所需技能
	ID 现在是内容可寻址标识符 ()。chainID 要确认此信息，您可以比较 Amazon ECR 公有和私有存储库上的 docker inspec t 命令的输出： 1. 在终端中为存储在 Amazon ECR 公共存储库中的图像运行以下命令。 ``` $~ % docker inspect public.ecr.aws/ama zonlinux/amazonlin ux:2018.03 ``` 有关命令的输出，请参阅“其他信息”部分。 2. 在终端中为存储在 Amazon ECR 私有存储库中的图像运行以下命令。 ``` $~ % docker inspect <account-id>.dkr.e cr.us-east-1.amazo naws.com/test_ecr_ repository:latest ``` 有关命令的输出，请参阅“其他信息”部分。 结果验证两个图像的图像 ID 摘要和图层摘要是否相同。	

Task	描述	所需技能
	身份证:f7cee5e1a f28ad4e147589c474d 399b12d9b551ef4c3e 11e02d982fce5eebc68 图层：d5655967c 2c4e8d68f8ec7cf753 218938669e6c16ac13 24303c073c736a2e2a2 此外，摘要基于本地管理的对象（本地文件是容器映像层的 tar）的字节或推送到注册表服务器的 blob。但是，当你将 blob 推送到注册表时，tar 会被压缩，并在压缩的 tar 文件中计算摘要。因此，docker 提取摘要值的差异源于在注册表（Amazon ECR 私有或公共）级别应用的压缩。 **Note** 此解释特定于使用 Docker 客户端。你不会在其他客户端（例如 nerdctl 或 Finc h）上看到这种行为，因为它们在推拉操作期间不会自动压缩图像。	

自动识别 Amazon ECR 公共存储库和私有存储库之间的重复映像

Task	描述	所需技能
克隆存储库。	将此模式的 Github 存储库克隆到本地文件夹： `$git clone https://github.com/aws-samples/automated-solution-to-identify-duplicate-container-images-between-repositories`	AWS 管理员，AWS DevOps
设置 CI/CD 管道。	GitHub 存储库包含一个`.yaml`文件，该文件用于创建 AWS CloudFormation 堆栈以在其中设置管道 AWS CodePipeline。 1. 登录 AWS Management Console 并打开AWS CloudFormation 控制台。 2. 使用模板`pipeline.yaml` 文件创建堆栈，该文件位于克隆存储库的`code`文件夹中。 3. 接受或更改参数的默认值。为以下字段指定值： • 堆栈名称 • `ArtifactStoreBucketName` — 将用于存储 AWS CodePipeline 工件的现有 S3 存储桶	AWS 管理员，AWS DevOps

Task	描述	所需技能
	• `OutputBucket` — 现有 S3 存储桶，将用于存储重复图像的 URIs • `SourceImageFile` — 名为的现有文本文件 `input.txt` ，其中包含 URIs 来自公共存储库的图像，将对照 Amazon ECR 私有存储库进行检查以检测重复情况 4. 查看并调整堆栈选项，然后选择 S ubm it 以运行模板。 该管道将设置为两个阶段（CodeCommit 以及 CodeBuild，如架构图所示），以识别私有存储库中也存在于公共存储库中的图像。管道配置有以下资源： • CodePipeline 用于编排部署管道。 • 用于 CodeCommit 存储 bash 脚本和输入文件的存储库。bash 脚本用于比较公共存储库和私有存储库 IDs 中的容器镜像以查找重复项。此检查是在单个存储库中指定的所有存储库 AWS 账户中执行的 AWS 区域。 • 一个调用 bash 脚本来识别已存在于 Amazon ECR 存储库中的图像的 CodeBuild 项目。	

Task	描述	所需技能
	• 允许访问所必需的 IAM 角色。 • 一个 S3 存储桶，用于存储包含图像的输出文件 URIs。 • 另一个用于存储 CodePipeline 工件的 S3 存储桶。	

Task	描述	所需技能
填充 CodeCommit 存储库。	要填充 CodeCommit 存储库，请执行以下步骤： 1. 打开CodeCommit 控制台并导航到您创建 CloudFormation 堆栈 AWS 区域 的位置。 2. 使用列表中的 CloudFormation 脚本找到您配置的存储库，选择 “克隆 URL”，然后复制 HTTPS URL 协议以连接到存储库。 3. 打开命令提示符并使用您在上一步中复制的 HTTPS 网址运行 git clone 命令。 4. 导航到根目录。创建一个名为的文件，`input.txt` 并使用您要在私有 Amazon ECR 存储库中搜索 URIs 的 Amazon ECR 公共映像注册表填充此文件。 5. 将文件`script.sh` `buildspec.yml` 、和`input.txt` 从存储库的本地副本复制到克隆 CodeCommit 的 GitHub 存储库中识别存储库之间重复的容器映像的自动解决方案。 6. 使用以下命令 CodeCommit 将文件上传到： `git add .`	AWS 管理员，AWS DevOps

Task	描述	所需技能
	`git commit -m “added input files”` `git push`	
清理。	为避免将来产生费用，请按照以下步骤删除资源： 1. 导航到存储项目的 S3 存储桶 CodePipeline ，然后清空该存储桶。 2. 导航到存储重复图像的 S3 存储桶 URIs，然后清空该存储桶。 3. 导航到 CloudFormation 控制台并删除您为设置管道而创建的堆栈。	AWS 管理员

故障排除

事务	解决方案
当您尝试从终端或命令行推送、拉取 CodeCommit 存储库或以其他方式与仓库交互时，系统会提示您提供用户名和密码，并且必须为您的 IAM 用户提供 Git 证书。	此错误的最常见原因如下： • 您的本地计算机运行的操作系统不支持凭据管理，或者未安装凭据管理实用程序。 • 您的 IAM 用户的 Git 证书尚未保存到其中一个凭证管理系统中。 根据您的操作系统和本地环境，您可能需要安装凭证管理器、配置操作系统随附的凭证管理器或自定义您的本地环境以使用凭证存储。例如，如果您的计算机运行的是 macOS，您可以使用 Keychain Access 实用程序存储您的凭证。如果您的计算机运行的是 Windows，您可

事务	解决方案
	以使用随 Windows 版 Git 安装的 Git Credential Manager。有关更多信息，请参阅文档中的使用 Git 凭据为 HTTPS 用户设置和 Git CodeCommit 文档中的凭据存储。
将图像推送到 Amazon ECR 存储库时，您会遇到 HTTP 403 或“没有基本身份验证凭证”错误。	即使你已使用 aws ecr 命令成功向 Docker 进行了身份验证，你也可能会在 docker push 或 docker pull 命令中遇到这些错误消息。 get-login-password已知原因有： • 您已通过身份验证到其他区域。有关更多信息，请参阅 Amazon ECR 文档中的私有注册表身份验证。 • 您已通过身份验证推送到您无权访问的存储库。有关更多信息，请参阅 Amazon ECR 文档中的私有存储库政策。 • 您的令牌已过期。使用该`GetAuthorizationToken` 操作获得的代币的默认到期时间为 12 小时。

相关资源

- 用于@@ 识别存储库（存储GitHub 库）之间重复的容器映像的自动化解决方案
- 亚马逊 ECR 公共画廊
- 亚马逊 ECR 中的私有图片（亚马逊 ECR 文档）
- AWS::CodePipeline::Pipeline 资源（AWS CloudFormation 文档）
- OCI 图像格式规范

其他信息

Docker 检查亚马逊 ECR 公共存储库中的图像的输出

```
[
    {
```

```
        "Id":
 "sha256:f7cee5e1af28ad4e147589c474d399b12d9b551ef4c3e11e02d982fce5eebc68",
        "RepoTags": [
            "<account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest",
            "public.ecr.aws/amazonlinux/amazonlinux:2018.03"
        ],
        "RepoDigests": [
            "<account-id>.dkr.ecr.us-east-1.amazonaws.com/
test_ecr_repository@sha256:52db9000073d93b9bdee6a7246a68c35a741aaade05a8f4febba0bf795cdac02",
            "public.ecr.aws/amazonlinux/
amazonlinux@sha256:f972d24199508c52de7ad37a298bda35d8a1bd7df158149b381c03f6c6e363b5"
        ],
        "Parent": "",
        "Comment": "",
        "Created": "2023-02-23T06:20:11.575053226Z",
        "Container":
 "ec7f2fc7d2b6a382384061247ef603e7d647d65f5cd4fa397a3ccbba9278367c",
        "ContainerConfig": {
            "Hostname": "ec7f2fc7d2b6",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [
                "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
            ],
            "Cmd": [
                "/bin/sh",
                "-c",
                "#(nop) ",
                "CMD [\"/bin/bash\"]"
            ],
            "Image":
 "sha256:c1bced1b5a65681e1e0e52d0a6ad17aaf76606149492ca0bf519a466ecb21e51",
            "Volumes": null,
            "WorkingDir": "",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": {}
        },
```

```
        "DockerVersion": "20.10.17",
        "Author": "",
        "Config": {
            "Hostname": "",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [
                "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
            ],
            "Cmd": [
                "/bin/bash"
            ],
            "Image":
 "sha256:c1bced1b5a65681e1e0e52d0a6ad17aaf76606149492ca0bf519a466ecb21e51",
            "Volumes": null,
            "WorkingDir": "",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": null
        },
        "Architecture": "amd64",
        "Os": "linux",
        "Size": 167436755,
        "VirtualSize": 167436755,
        "GraphDriver": {
            "Data": {
                "MergedDir": "/var/lib/docker/overlay2/
c2c2351a82b26cbdf7782507500e5adb5c2b3a2875bdbba79788a4b27cd6a913/merged",
                "UpperDir": "/var/lib/docker/overlay2/
c2c2351a82b26cbdf7782507500e5adb5c2b3a2875bdbba79788a4b27cd6a913/diff",
                "WorkDir": "/var/lib/docker/overlay2/
c2c2351a82b26cbdf7782507500e5adb5c2b3a2875bdbba79788a4b27cd6a913/work"
            },
            "Name": "overlay2"
        },
        "RootFS": {
            "Type": "layers",
            "Layers": [
```

```
 "sha256:d5655967c2c4e8d68f8ec7cf753218938669e6c16ac1324303c073c736a2e2a2"
            ]
        },
        "Metadata": {
            "LastTagTime": "2023-03-02T10:28:47.142155987Z"
        }
    }
]
```

Docker 检查亚马逊 ECR 私有存储库中的图像的输出

```
[
    {
        "Id":
 "sha256:f7cee5e1af28ad4e147589c474d399b12d9b551ef4c3e11e02d982fce5eebc68",
        "RepoTags": [
            "<account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest",
            "public.ecr.aws/amazonlinux/amazonlinux:2018.03"
        ],
        "RepoDigests": [
            "<account-id>.dkr.ecr.us-east-1.amazonaws.com/
test_ecr_repository@sha256:52db9000073d93b9bdee6a7246a68c35a741aaade05a8f4febba0bf795cdac02",
            "public.ecr.aws/amazonlinux/
amazonlinux@sha256:f972d24199508c52de7ad37a298bda35d8a1bd7df158149b381c03f6c6e363b5"
        ],
        "Parent": "",
        "Comment": "",
        "Created": "2023-02-23T06:20:11.575053226Z",
        "Container":
 "ec7f2fc7d2b6a382384061247ef603e7d647d65f5cd4fa397a3ccbba9278367c",
        "ContainerConfig": {
            "Hostname": "ec7f2fc7d2b6",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [
                "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
```

```
            ],
            "Cmd": [
                "/bin/sh",
                "-c",
                "#(nop) ",
                "CMD [\"/bin/bash\"]"
            ],
            "Image":
"sha256:c1bced1b5a65681e1e0e52d0a6ad17aaf76606149492ca0bf519a466ecb21e51",
            "Volumes": null,
            "WorkingDir": "",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": {}
        },
        "DockerVersion": "20.10.17",
        "Author": "",
        "Config": {
            "Hostname": "",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [
                "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
            ],
            "Cmd": [
                "/bin/bash"
            ],
            "Image":
"sha256:c1bced1b5a65681e1e0e52d0a6ad17aaf76606149492ca0bf519a466ecb21e51",
            "Volumes": null,
            "WorkingDir": "",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": null
        },
        "Architecture": "amd64",
        "Os": "linux",
        "Size": 167436755,
```

```
        "VirtualSize": 167436755,
        "GraphDriver": {
            "Data": {
                "MergedDir": "/var/lib/docker/overlay2/
c2c2351a82b26cbdf7782507500e5adb5c2b3a2875bdbba79788a4b27cd6a913/merged",
                "UpperDir": "/var/lib/docker/overlay2/
c2c2351a82b26cbdf7782507500e5adb5c2b3a2875bdbba79788a4b27cd6a913/diff",
                "WorkDir": "/var/lib/docker/overlay2/
c2c2351a82b26cbdf7782507500e5adb5c2b3a2875bdbba79788a4b27cd6a913/work"
            },
            "Name": "overlay2"
        },
        "RootFS": {
            "Type": "layers",
            "Layers": [

 "sha256:d5655967c2c4e8d68f8ec7cf753218938669e6c16ac1324303c073c736a2e2a2"
            ]
        },
        "Metadata": {
            "LastTagTime": "2023-03-02T10:28:47.142155987Z"
        }
    }
]
```

使用 Kubernetes 在亚马逊 EKS 工作节点上安装 SSM 代理 DaemonSet

由 Mahendra Revanasiddappa (AWS) 创建

摘要

注意，2021 年 9 月：经过优化的最新亚马逊 EKS 会自动 AMIs 安装 SSM 代理。有关更多信息，请参阅 2021 年 6 月的发行说明 AMIs。

在 Amazon Elastic Kubernetes Service(Amazon EKS) 中，由于安全指导方针，Worker 节点没有附加 Secure Shell (SSH) 密钥对。此模式显示了如何使用 Kubernetes DaemonSet 资源类型在所有工作节点上安装 AWS Systems Manager 代理（SSM 代理），而不是手动安装或替换节点的亚马逊系统映像 (AMI)。 DaemonSet 使用工作节点上的 cron 作业来安排 SSM 代理的安装。您也可以使用这种模式在 Worker 节点上安装其他软件包。

在对集群中的问题进行故障排除时，按需安装 SSM Agent 可让您与 Worker 节点建立 SSH 会话、收集日志或查看实例配置，而无需使用 SSH 密钥对。

先决条件和限制

先决条件

- 带有亚马逊弹性计算云 (Amazon EC2) 工作节点的现有 Amazon EKS 集群。
- 容器实例应获得与 SSM 服务通信的许可。AWS Identity and Access Management (IAM) 托管角色亚马逊为 SSM 代理SSMManagedInstanceCore提供了在 EC2 实例上运行所需的权限。有关更多信息，请参阅 AWS Systems Manager 文档。

限制

- 这种模式不适用于 AWS Fargate，因为 F DaemonSets argate 平台不支持。
- 此模式仅适用于基于 Linux 的 Worker 节点。
- P DaemonSet od 在特权模式下运行。如果 Amazon EKS 集群有阻止特权模式的容器组（pod），则不会安装 SSM Agent。

架构

下图阐明了该模式的架构。

工具

工具

- Kubectl 是命令行实用程序，用于与 Amazon EKS 集群交互。此模式用于`kubectl`在 Amazon EKS 集群 DaemonSet 上部署，该集群将在所有工作节点上安装 SSM 代理。
- Amazon EKS 让您在 AWS 上轻松运行 Kubernetes，而无需安装、操作和维护您自己的 Kubernetes 控制面板或节点。Kubernetes 是一个用于实现容器化应用程序的部署、扩缩和管理自动化的开源系统。
- AWS Systems Manager 会话管理器允许您通过交互式、一键式、基于浏览器的外壳或 AWS 命令行界面 (AWS CLIVMs) 来管理您的 EC2 实例、本地实例和虚拟机 ()。

代码

使用以下代码创建将在 Amazon EKS 集群上安装 SSM 代理的 DaemonSet 配置文件。按照操作说明部分中的说明操作。

```
cat << EOF > ssm_daemonset.yaml
apiVersion: apps/v1
kind: DaemonSet
metadata:
  labels:
    k8s-app: ssm-installer
  name: ssm-installer
  namespace: kube-system
spec:
  selector:
    matchLabels:
      k8s-app: ssm-installer
  template:
    metadata:
      labels:
        k8s-app: ssm-installer
    spec:
      containers:
      - name: sleeper
        image: busybox
        command: ['sh', '-c', 'echo I keep things running! && sleep 3600']
      initContainers:
      - image: amazonlinux
        imagePullPolicy: Always
```

```
        name: ssm
        command: ["/bin/bash"]
        args: ["-c","echo '* * * * * root yum install -y https://s3.amazonaws.com/
ec2-downloads-windows/SSMAgent/latest/linux_amd64/amazon-ssm-agent.rpm & rm -rf /etc/
cron.d/ssmstart' > /etc/cron.d/ssmstart"]
        securityContext:
          allowPrivilegeEscalation: true
        volumeMounts:
        - mountPath: /etc/cron.d
          name: cronfile
        terminationMessagePath: /dev/termination-log
        terminationMessagePolicy: File
      volumes:
      - name: cronfile
        hostPath:
          path: /etc/cron.d
          type: Directory
      dnsPolicy: ClusterFirst
      restartPolicy: Always
      schedulerName: default-scheduler
      terminationGracePeriodSeconds: 30
EOF
```

操作说明

设置 kubectl

Task	描述	所需技能
安装和配置 kubectl 以访问 EKS 集群。	如果 `kubectl` 尚未安装并配置为访问 Amazon EKS 集群，请参阅 Amazon EKS 文档中的安装 kubectl。	DevOps

部署 DaemonSet

Task	描述	所需技能
创建 DaemonSet 配置文件。	使用此模式前面代码部分中的代码创建名为的 DaemonSet	DevOps

Task	描述	所需技能
	配置文件ssm_daemonset.yaml ，该文件将部署到 Amazon EKS 集群。 启动的 pod DaemonSet 有一个主容器和一个init容器。主容器包含一个 sleep 命令。该 init 容器包括一个 command 部分，用于创建 cron 作业文件，以在 /etc/cron.d/ 路径上安装 SSM Agent。cron 作业仅运行一次，其创建的文件会在作业完成后自动删除。 容器初始化完成后，主容器将等待 60 分钟后再退出。60 分钟后，将启动新容器组（pod）。此容器组（pod）会安装 SSM Agent（如果缺失），或者将 SSM Agent 更新至最新版本。 如果需要，您可以修改 sleep 命令，以每天重启容器组（pod）一次或更频繁地运行。	

Task	描述	所需技能
在 Amazon EKS 集群 DaemonSet 上部署。	要在 Amazon EKS 集群上部署您在上一步中创建的 DaemonSet 配置文件，请使用以下命令： `kubectl apply -f ssm_daemonset.yaml` 此命令创建一个，DaemonSet 用于在工作节点上运行 pod 以安装 SSM 代理。	DevOps

相关资源

- 安装 kubectl(Amazon EKS 文档)
- 设置 Session Manager（AWS Systems Manager 文档）

使用在 Amazon EKS 工作节点上安装 SSM CloudWatch 代理和代理 preBootstrapCommands

由 Akkamahadevi Hiremath (AWS) 编写

摘要

此模式提供了在亚马逊 EKS 集群创建期间在亚马逊网络服务 (AWS) 云中的亚马逊 Elastic Kubernetes Service（Amazon EKS）工作节点上安装 AWS Systems Manager 代理（SSM 代理）和亚马逊 CloudWatch 代理的代码示例和步骤。您可以使用`eksctl`配置文件架构中的`preBootstrapCommands`属性安装 SSM CloudWatch 代理和代理（Weaveworks 文档）。然后，您可以使用 SSM 代理连接到您的工作节点，而无需使用亚马逊弹性计算云 (Amazon EC2) 密钥对。此外，您还可以使用 CloudWatch 代理来监控 Amazon EKS 工作节点上的内存和磁盘利用率。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- eksctl 命令行实用程序，已在 macOS、Linux 或 Windows 上安装与配置
- kubectl 命令行实用程序，已在 macOS、Linux 或 Windows 上安装与配置

限制

- 我们建议您避免向 `preBootstrapCommands` 属性中添加长时间运行的脚本，因为这会延迟节点在扩展活动期间加入 Amazon EKS 集群的时间。我们建议您改为创建自定义亚马逊机器映像（AMI）。
- 此模式仅适用于亚马逊 EC2 Linux 实例。

架构

技术堆栈

- Amazon CloudWatch
- Amazon Elastic Kubernetes Service(Amazon EKS)
- AWS Systems Manager Parameter Store

目标架构

下图显示了一个用户使用 SSM Agent 连接到 Amazon EKS Worker 节点的示例，该代理是使用 `preBootstrapCommands` 安装的。

图表显示了以下工作流：

1. 用户使用带有`preBootstrapCommands`属性的`eksctl`配置文件创建 Amazon EKS 集群，该文件将安装 SSM CloudWatch 代理和代理。
2. 之后由于扩展活动而加入集群的任何新实例都将使用预安装的 SSM 代理和 CloudWatch 代理创建。
3. 用户使用 SSM 代理连接 EC2 到 Amazon，然后使用代理监控内存和磁盘利用率。 CloudWatch

工具

- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和在 AWS 上运行的应用程序的指标。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可帮助您在 AWS 上运行 Kubernetes，而无需安装或维护您自己的 Kubernetes 控制面板或节点。
- AWS Systems Manager Parameter Store 提供安全的分层存储，用于配置数据管理和密钥管理。
- AWS Systems Manager 会话管理器可帮助您通过交互式、一键式、基于浏览器的外壳或 AWS 命令行界面 (AWS CLI) 来管理您的 EC2 实例、本地实例和虚拟机。
- eksctl 是一种用于在 Amazon EKS 上创建和管理 Kubernetes 集群的命令行实用程序。
- kubectl 是命令行实用程序，用于与集群 API 服务器通信。

操作说明

创建 Amazon EKS 集群。

Task	描述	所需技能
存储 CloudWatch 代理配置文件。	将 CloudWatch 代理配置文件存储在您要创建亚马逊 EKS 集群的 AWS 区域的 AWS Systems Manager Parameter	DevOps 工程师

Task	描述	所需技能
	St ore 中。为此，请在 AWS Systems Manager Parameter Store 中创建参数，并备注参数名称（例如`AmazonCloudwatch-linux`）。 有关更多信息，请参阅此模式的“其他信息”部分中的 CloudWatch 代理配置文件代码示例。	
创建 eksctl 配置文件与集群。	1. 创建包含 CloudWatch 代理和 SSM 代理安装步骤的`eksctl`配置文件。有关更多信息，请参阅此模式的其他信息部分中的 eksctl 配置文件示例代码。 2. 运行 `eksctl create cluster -f cluster.yaml` 命令以创建集群。	AWS DevOps

验证 SSM 代理和 CloudWatch 代理是否正常工作

Task	描述	所需技能
测试 SSM Agent。	使用 AWS Systems Manager 文档中的启动会话所涵盖的任何方法，利用 SSH 连接至您的 Amazon EKS 集群节点。	AWS DevOps
测试代 CloudWatch 理。	使用 CloudWatch 控制台验证代 CloudWatch 理： 1. 登录 AWS 管理控制台并打开CloudWatch 控制台。	AWS DevOps

Task	描述	所需技能
	2. 在导航窗格中，展开 Metrics（指标），然后选择 All metrics（所有指标）。 3. 在“浏览”选项卡的搜索框中，输入并选择CWAgent 指标以查看内存和磁盘指标。	

相关资源

- 在@@ 您的服务器上安装和运行 CloudWatch 代理（Amazon CloudWatch 文档）
- 创建 Systems Manager 参数（控制台）(AWS Systems Manager 文档)
- 创建 CloudWatch 代理配置文件（Amazon CloudWatch 文档）
- 启动会话 (AWS CLI)（AWS Systems Manager 文档）
- 启动会话（亚马逊 EC2 控制台）（AWS Systems Manager 文档）

其他信息

CloudWatch 代理配置文件示例

在以下示例中，CloudWatch 代理配置为监控 Amazon Linux 实例上的磁盘和内存利用率：

```
{
    "agent": {
        "metrics_collection_interval": 60,
        "run_as_user": "cwagent"
    },
    "metrics": {
        "append_dimensions": {
            "AutoScalingGroupName": "${aws:AutoScalingGroupName}",
            "ImageId": "${aws:ImageId}",
            "InstanceId": "${aws:InstanceId}",
            "InstanceType": "${aws:InstanceType}"
        },
        "metrics_collected": {
            "disk": {
```

```
                "measurement": [
                    "used_percent"
                ],
                "metrics_collection_interval": 60,
                "resources": [
                    "*"
                ]
            },
            "mem": {
                "measurement": [
                    "mem_used_percent"
                ],
                "metrics_collection_interval": 60
            }
        }
    }
}
```

eksctl 配置文件示例

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: test
  region: us-east-2
  version: "1.24"
managedNodeGroups:
  - name: test
    minSize: 2
    maxSize: 4
    desiredCapacity: 2
    volumeSize: 20
    instanceType: t3.medium
    preBootstrapCommands:
    - sudo yum install amazon-ssm-agent -y
    - sudo systemctl enable amazon-ssm-agent
    - sudo systemctl start amazon-ssm-agent
    - sudo yum install amazon-cloudwatch-agent -y
    - sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a fetch-
config -m ec2 -s -c ssm:AmazonCloudwatch-linux
    iam:
      attachPolicyARNs:
        - arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy
```

```
        - arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
        - arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
        - arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy
        - arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
```

其他代码详细信息

- preBootstrapCommands 属性的最后一行，AmazonCloudwatch-linux 是在 AWS Systems Manager Parameter Store 中创建的参数名称。您必须在创建 Amazon EKS 集群的相同 AWS 区域的 Parameter Store 中纳入 AmazonCloudwatch-linux。您也可指定文件路径，但我们建议使用 Systems Manager，以便更轻松地实现自动化和重复使用。
- 如果您在 eksctl 配置文件中使用 preBootstrapCommands，则会在 AWS 管理控制台中看到两个启动模板。第一个启动模板包含 preBootstrapCommands 中指定的命令。第二个模板包括 preBootstrapCommands 中指定的命令和默认 Amazon EKS 用户数据。此数据用于将节点加入集群。节点组的自动扩缩组使用此用户数据启动新实例。
- 如果您在 eksctl 配置文件中使用 iam 属性，则必须列出默认 Amazon EKS 策略，以及随附 AWS Identity and Access Management (IAM) 策略中所需任何其他策略。在 “创建 eksctl 配置文件和集群” 步骤的代码片段中，CloudWatchAgentServerPolicy添加了其他策略以确保 CloudWatch 代理和 SSM 代理按预期运行。AmazonSSMMangedInstanceCoreAmazonEKSWorkerNodePolicy、AmazonEKS_CNI_Policy、策略是 Amazon EKS 集群正常运行所需强制策略。

启用 Amazon EKS 自动模式时迁移 NGINX 入口控制器

由 Olawale Olaleye (AWS) 和 Shamanth Devagari (AWS) 创作

摘要

适用于亚马逊 Elastic Kubernetes Service（Amazon EKS）的 EKS 自动模式可以减少在 Kubernetes 集群上运行工作负载的运营开销。此模式还 AWS 允许您代表您设置和管理基础架构。在现有集群上启用 EKS 自动模式时，必须仔细规划 NGINX 入口控制器配置的迁移。这是因为无法直接传输网络负载均衡器。

当您在现有 Amazon EKS 集群中启用 EKS 自动模式时，您可以使用蓝/绿部署策略迁移 NGINX 入口控制器实例。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 运行 Kubernetes 版本 1.29 或更高版本的 Amazon EKS 集群
- 运行最低版本的 Amazon EKS 附加组件
- 最新版本的 kub ectl
- 现有的 NGINX 入口控制器实例
- （可选）Amazon Route 53 中的托管区域，用于基于 DNS 的流量转移

架构

蓝/绿部署是一种部署策略，在这种策略中，您可以创建两个独立但相同的环境。Blue/Green 部署提供近乎零的停机时间发布和回滚功能。基本思想是在运行不同版本的应用程序的两个相同环境之间转移流量。

下图显示了启用 EKS 自动模式时从两个不同的 NGINX Ingress Controller 实例迁移网络负载均衡器的情况。您可以使用蓝/绿部署在两个网络负载均衡器之间转移流量。

原始命名空间是蓝色命名空间。在启用 EKS 自动模式之前，这是原始 NGINX Ingress Controller 服务和实例的运行位置。原始服务和实例连接到 Network Load Balancer，该负载均衡器的 DNS 名称已在

Route 53 中配置。Loa AWS d Balancer Contro ller 在目标虚拟私有云 (VPC) 中部署了此网络负载均衡器。

该图显示了为蓝/绿部署设置环境的以下工作流程：

1. 在不同的命名空间（绿色命名空间）中安装和配置另一个 NGINX Ingress Controller 实例。
2. 在 Route 53 中，为新的 Network Load Balancer 配置 DNS 名称。

工具

AWS 服务

- 亚马逊 Elastic Kubernetes Service（亚马逊 EKS）可帮助你在上面运行 AWS Kubernetes，而无需安装或维护自己的 Kubernetes 控制平面或节点。
- 弹性负载均衡在多个目标上分配传入的应用程序或网络流量。例如，您可以跨亚马逊弹性计算云 (Amazon EC2) 实例、容器以及一个或多个可用区中的 IP 地址分配流量。
- Amazon Route 53 是一种可用性高、可扩展性强的 DNS Web 服务。
- Amazon Virtual Private Cloud（亚马逊 VPC）可帮助您将 AWS 资源启动到您定义的虚拟网络中。该虚拟网络类似于您在数据中心中运行的传统网络，并具有使用 AWS的可扩展基础设施的优势。

其他工具

- Helm 是一款适用于 Kubernetes 的开源软件包管理器，可帮助你在 Kubernetes 集群上安装和管理应用程序。
- kubectl：针对 Kubernetes 集群运行命令的命令行界面。
- NGINX Ingress Contro ller 将 Kubernetes 应用程序和服务与请求处理、身份验证、自助服务自定义资源和调试连接起来。

操作说明

查看现有环境

Task	描述	所需技能
确认原来的 NGINX 入口控制器实例运行正常。	输入以下命令以验证 `ingress-nginx` 命名空间	DevOps 工程师

Task	描述	所需技能

	中的资源是否正常运行。如果您已在其他命名空间中部署了 NGINX Ingress Controller，请在此命令中更新命名空间名称。
	
	`kubectl get all -n ingress-nginx`
	
	在输出中，确认 NGINX Ingress Controller Pod 处于运行状态。下面是一个示例输出：
	
	`NAME READY STATUS RESTARTS AGE pod/ingress-nginx-admission-create-xqn9d 0/1 Completed 0 88m pod/ingress-nginx-admission-patch-lhk4j 0/1 Completed 1 88m pod/ingress-nginx-controller-68f68f859-xrz74 1/1 Running 2 (10m ago) 72m NAME TYPE CLUSTER-IP EXTERNAL-IP`	

Task	描述	所需技能
	PORT(S) AGE service/ingress-nginx-controller LoadBalancer 10.100.67.255 k8s-ingressn-ingressn-abcdefg-12345.elb.eu-west-1.amazonaws.com 80:30330/TCP,443:31462/TCP 88m service/ingress-nginx-controller-admission ClusterIP 10.100.201.176 <none> 443/TCP 88m NAME READY UP-TO-DATE AVAILABLE AGE deployment.apps/ingress-nginx-controller 1/1 1 1 88m NAME DESIRED CURRENT READY AGE replicaset.apps/ingress-nginx-controller-68f68f859 1 1 1 72m replicaset.apps/ingress-nginx-controller-d8c96cf68 0	

Task	描述	所需技能

	`0 0 88m`
	
	`NAME`
	
	`STATUS COMPLETIONS DURATION AGE`
	`job.batch/ingress-nginx-admission-create Complete 1/1 4s 88m`
	`job.batch/ingress-nginx-admission-patch Complete 1/1 5s 88m`	

部署示例 HTTPd 工作负载以使用 NGINX 入口控制器

Task	描述	所需技能

创建 Kubernetes 资源。	输入以下命令来创建 Kubernetes 部署、服务和入口示例：
	
	`kubectl create deployment demo --image=httpd --port=80`
	
	`kubectl expose deployment demo`
	
	`kubectl create ingress demo --class=nginx \ --rule nginxautomode.local.dev/=demo:80`
		DevOps 工程师

Task	描述	所需技能

查看已部署的资源。	输入以下命令以查看已部署资源的列表：
	`kubectl get all,ingress`
	在输出中，确认示例 HTTPd pod 处于运行状态。下面是一个示例输出：
	`NAME READY STATUS RESTARTS AGE pod/demo-7d94f8cb4f-q68wc 1/1 Running 0 59m`
	`NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE service/demo ClusterIP 10.100.78.155 <none> 80/TCP 59m service/kubernetes ClusterIP 10.100.0.1 <none> 443/TCP 117m`
	`NAME READY UP-TO-DATE AVAILABLE AGE deployment.apps/demo 1/1 1 1 59m`
	`NAME DESIRED CURRENT READY AGE`
		DevOps 工程师

Task	描述	所需技能

	`replicaset.apps/demo-7d94f8cb4f 1 1 1 59m`
	
	`NAME CLASS HOSTS ADDRESS PORTS AGE`
	`ingress.networking.k8s.io/demo nginx nginxautomode.local.dev k8s-ingressn-ingressn-abcdefg-12345.elb.eu-west-1.amazonaws.com 80 56m`	

确认服务可以访问。	输入以下命令以确认可以通过网络负载均衡器的 DNS 名称访问该服务：
	
	`curl -H "Host: nginxautomode.local.dev" http://k8s-ingressn-ingressn-abcdefg-12345.elb.eu-west-1.amazonaws.com`
	
	以下是预期的输出：
	
	`<html><body><h1>It works!</h1></body></html>`
		DevOps 工程师

Task	描述	所需技能

（可选）创建 DNS 记录。	1. 按照使用 Amazon Route 53 控制台（Route 53 文档）创建记录中的说明为配置的域创建 DNS 记录。
	2. 输入以下命令以确认可通过配置的域名访问该服务：
	`curl "http://nginxautomode.local.dev/?[1-5]"`
	以下是预期的输出：
	`<html><body><h1>It works!</h1></body></html>`
	`<html><body><h1>It works!</h1></body></html>`
	`<html><body><h1>It works!</h1></body></html>`
	`<html><body><h1>It works!</h1></body></html>`
	`<html><body><h1>It works!</h1></body></html>`
		DevOps 工程师，AWS DevOps

在现有集群上启用 EKS 自动模式

Task	描述	所需技能
启用 EKS 自动模式。	按照在现有集群上启用 EKS 自动模式中的说明进行操作（Amazon EKS 文档）。	AWS DevOps

安装新的 NGINX 入口控制器

Task	描述	所需技能

配置一个新的 NGINX 入口控制器实例。	1. 下载 deploy.yaml 模板。
	2. 在您的首选编辑器中打开 deploy.yaml 模板。
	3. 在该`kind: Namespace` 部分中，输入命名空间的唯一名称，例如`ingress-nginx-v2` ：
	`apiVersion: v1 kind: Namespace metadata: labels: app.kubernetes.io/instance: ingress-nginx app.kubernetes.io/name: ingress-nginx name: ingress-nginx-v2`
	4. 将每个部分的`namespace` 值更新为新名称。
	5. 请在 `kind: Deployment` 部分执行以下操作：
	a. 为输入唯一值`--controller-class` ，例如`k8s.io/ingress-nginx-v2` 。
	b. 为输入唯一值`--ingress-class` ，例如`nginx-v2`。
	`apiVersion: apps/v1`
		DevOps 工程师

Task	描述	所需技能

	kind: Deployment
	  name: ingress-nginx-controller
	  namespace: ingress-nginx-v2
	...
	  spec:
	  containers:
	  - args:
	  - /nginx-ingress-controller
	  - --publish-service=$(POD_NAMESPACE)/ingress-nginx-controller
	  - --election-id=ingress-nginx-leader
	  - --controller-class=k8s.io/ingress-nginx-v2
	  - --ingress-class=nginx-v2
	
	6. 在该kind: IngressClass 部分中，为--controller-class 和--ingress-class 输入您在上一节中使用的相同值：
	
	apiVersion: networking.k8s.io/v1
	kind: IngressClass
	metadata:
	  labels:
	  app.kubernetes.io/component: controller	

Task	描述	所需技能

	<pre>app.kuber
	netes.io/instance:
	ingress-nginx
	app.kuber
	netes.io/name:
	ingress-nginx
	app.kuber
	netes.io/part-of:
	ingress-nginx
	app.kuber
	netes.io/version:
	1.12.0
	name: nginx-v2
	spec:
	controller: k8s.io/
	ingress-nginx-v2</pre>	

	7. 在下一节中，添加`loadBalancerClass: eks.amazonaws.com/nlb` 为 NGINX Ingress Controller 实例配置 Network Load Balancer：
	<pre>apiVersion: v1
	kind: Service
	metadata:
	name: ingress-n
	ginx-controller
	namespace: ingress-
	nginx-v2
	spec:
	...
	selector:
	app.kuber
	netes.io/component:
	controller
	app.kuber
	netes.io/instance:
	ingress-nginx</pre>	

Task	描述	所需技能

	<pre>app.kuber
	netes.io/name:
	ingress-nginx
	type: LoadBalancer
	loadBalancerClass:
	eks.amazonaws.com/
	nlb</pre>
	8. 保存并关闭 deploy.yaml 模板。	

部署新的 NGINX 实例控制器实例。	输入以下命令以应用修改后的清单文件：
	<pre>kubectl apply -f
	deploy.yaml</pre>
		DevOps 工程师

Task	描述	所需技能

确认成功部署。	输入以下命令以验证 `ingress-nginx-v2` 命名空间中的资源是否正常运行：
	`kubectl get all -n ingress-nginx-v2`
	在输出中，确认 NGINX Ingress Controller Pod 处于运行状态。下面是一个示例输出：
	`NAME READY STATUS RESTARTS AGE pod/ingress-nginx-admission-create-7shrj 0/1 Completed 0 24s pod/ingress-nginx-admission-patch-vkxr5 0/1 Completed 1 24s pod/ingress-nginx-controller-757bfcbc6d-4fw52 1/1 Running 0 24s NAME TYPE CLUSTER-IP EXTERNAL-IP`
		DevOps 工程师

Task	描述	所需技能
	PORT(S) AGE service/ingress- nginx-controller LoadBalancer 10.100.208.114 k8s-ingressn-ingre ssn-2e5e37fab6-848 337cd9c9d520f.elb.eu- west-1.amazonaws.com 80:31469/TCP,443:3 0658/TCP 24s service/ingress- nginx-controller- admission ClusterIP 10.100.150.114 <none> 443/TCP 24s NAME READY UP-TO-DATE AVAILABLE AGE deployment.apps/ ingress-nginx-co ntroller 1/1 1 1 24s NAME DESIRED CURRENT READY AGE replicaset.apps/ ingress-nginx-co ntroller-757bfcbc6d 1 1 1 24s	

Task	描述	所需技能
	`NAME STATUS COMPLETIONS DURATION AGE job.batch/ingress-nginx-admission-create Complete 1/1 4s 24s job.batch/ingress-nginx-admission-patch Complete 1/1 5s 24s`	
为示例 HTTPd 工作负载创建新的入口。	输入以下命令为现有示例 HTTPd 工作负载创建新的入口： `kubectl create ingress demo-new --class=nginx-v2 \ --rule nginxautomode.local.dev/=demo:80`	DevOps 工程师

Task	描述	所需技能

确认新的入口是否正常工作。	输入以下命令以确认新的入口是否正常工作：
	`curl -H "Host: nginxautomode.local.dev" k8s-ingressn-ingressn-2e5e37fab6-848337cd9c9d520f.elb.eu-west-1.amazonaws.com`
	以下是预期的输出：
	`<html><body><h1>It works!</h1></body></html>`
		DevOps 工程师

割接

Task	描述	所需技能
切换到新的命名空间。	1. （可选）按照编辑记录（Route 53 文档）中的说明更新 DNS 记录。 2. 确认新的 NGINX Ingress 控制器实例按预期运行后，请删除原来的实例。 3. 删除自行管理的 Loa AWS d Balancer 控制器。有关说明，请参阅从已弃用的 ALB 入口控制器迁移应用程序（Amazon EKS 文档）。 4. 耗尽托管节点组。有关说明，请参阅删除和清空节点组（eksctl 文档）。	AWS DevOps， DevOps 工程师

Task	描述	所需技能

查看两个入口。	输入以下命令以查看为示例 HTTPd 工作负载创建的两个入口：
	
	`kubectl get ingress`
	
	下面是一个示例输出：
	
	`NAME CLASS HOSTS ADDRESS PORTS AGE`
	`demo nginx nginxautomode.local.dev k8s-ingressn-ingressn-abcdefg-12345.elb.eu-west-1.amazonaws.com 80 95m`
	`demo-new nginx-v2 nginxautomode.local.dev k8s-ingressn-ingressn-2e5e37fab6-848337cd9c9d520f.elb.eu-west-1.amazonaws.com 80 33s`
		DevOps 工程师

相关资源

- 在@@ 现有集群上启用 EKS 自动模式（Amazon EKS 文档）
- 对@@ 由 Kubernetes 服务控制器在 Amazon EKS 中创建的负载均衡器进行故障排除（re: Post 知识中心）AWS

- NGINX 入口控制器（NGINX 文档）

将你的容器工作负载从 Azure 红帽 OpenShift (ARO) 迁移到 AWS 云端 Red Hat OpenShift 服务 (ROSA)

由 Naveen Ramasamy (AWS)、Gireesh Sreekantan (AWS) 和 Srikanth Rangavajhala (AWS) 创作

摘要

此模式提供了将容器工作负载从 Azure Red Hat OpenShift (ARO) 迁移到 AWS 云端 Red Hat OpenShift 服务 (ROSA) 的 step-by-step说明。ROSA 是一项托管 Kubernetes 服务，由红帽与之合作提供。 AWS它可以帮助您使用 Kubernetes 平台部署、管理和扩展容器化应用程序，并受益于红帽在 Kubernetes 和基础设施方面的专业知识。 AWS Cloud

将容器工作负载从 ARO、其他云或本地迁移到 ROSA 需要将应用程序、配置和数据从一个平台转移到另一个平台。这种模式有助于确保平稳过渡，同时优化 AWS Cloud 服务、安全性和成本效益。它涵盖了将工作负载迁移到 ROSA 集群的两种方法：CI/CD 和容器迁移工具包 (MTC)。

此模式涵盖了这两种方法。您选择的方法取决于迁移过程的复杂性和确定性。如果您完全控制应用程序的状态，并且可以通过管道保证设置的一致性，我们建议您使用 CI/CD 方法。但是，如果您的应用程序状态涉及不确定性、不可预见的变化或复杂的生态系统，我们建议您使用 MTC 作为可靠且受控的路径，将您的应用程序及其数据迁移到新集群。有关这两种方法的详细比较，请参阅 “其他信息” 部分。

迁移到 ROSA 的好处：

- ROSA AWS 作为本机服务无缝集成。可通过轻松访问 AWS Management Console 并通过单 AWS 账户笔计费。它与其他产品完全兼容 AWS 服务 ，并提供双方 AWS 和红帽的协作支持。
- ROSA 支持混合云和多云部署。它使应用程序能够在本地数据中心和多个云环境中一致运行。
- ROSA 受益于红帽对安全的关注，并提供基于角色的访问控制 (RBAC)、图像扫描和漏洞评估等功能，以确保容器环境的安全。
- ROSA 旨在轻松扩展应用程序，并提供高可用性选项。它允许应用程序根据需要进行扩展，同时保持可靠性。
- 与手动设置和管理方法相比，ROSA 可以自动化和简化 Kubernetes 集群的部署。这加快了开发和部署过程。
- ROSA 受益于 AWS Cloud 服务，并提供与数据库服务、存储解决方案和安全服务等 AWS 产品的无缝集成。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- 为 AWS 服务 此 ROSA 配置的权限依赖于提供功能。有关更多信息，请参阅 ROSA 文档中的先决条件。
- ROSA 已在 ROSA 控制台上启用。有关说明，请参阅 ROSA 文档。
- 已安装并配置了 ROSA 集群。有关更多信息，请参阅 ROSA 文档中的 ROSA 入门。要了解设置 ROSA 集群的不同方法，请参阅 AWS 规范性指导指南 ROSA 实现策略。
- 建立从本地网络到 AWS 通过 AWS Direct Connect（首选）或 AWS Virtual Private Network（AWS VPN）的网络连接。
- 亚马逊弹性计算云 (Amazon EC2) 实例或其他虚拟服务器，用于安装 OpenShift CLI (oc) 客户端`aws client`、ROSA 客户端和 Git 二进制文件等工具。

CI/CD 方法的其他先决条件：

- 访问本地 Jenkins 服务器，拥有创建新管道、添加阶段、添加 OpenShift 集群和执行构建的权限。
- 访问维护应用程序源代码的 Git 存储库，有权创建新的 Git 分支并向新分支执行提交。

MTC 方法的其他先决条件：

- 亚马逊简单存储服务 (Amazon S3) Service 存储桶，将用作复制存储库。
- 对源 ARO 集群的管理访问权限。这是设置 MTC 连接所必需的。

限制

- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额页面，然后选择服务的链接。

架构

ROSA 提供三种网络部署模式：公共、私有和AWS PrivateLink。 PrivateLink使红帽站点可靠性工程 (SRE) 团队能够使用连接到现有 VPC 中集群 PrivateLink 终端节点的私有子网来管理集群。

选择该 PrivateLink选项可提供最安全的配置。因此，我们建议将其用于敏感工作负载或严格的合规性要求。有关公网和私有网络部署选项的信息，请参阅红帽 OpenShift 文档。

> **Important**
>
> 您只能在安装时创建 PrivateLink 集群。安装 PrivateLink 后您无法更改要使用的群集。

下图说明了用于 AWS Direct Connect 连接到本地和 ARO 环境的 ROSA 集群的 PrivateLink 架构。

AWS 对 ROSA 的权限

要获 AWS 得 ROSA 的权限，我们建议您将 AWS Security Token Service (AWS STS) 与短期动态令牌一起使用。此方法使用最低权限的预定义角色和策略来授予 ROSA 在中操作的最低权限 AWS 账户，并支持 ROSA 安装、控制平面和计算功能。

CI/CD 管道重新部署

CI/CD pipeline redeployment is the recommended method for users who have a mature CI/CD管道。选择此选项后，您可以使用任何DevOps 部署策略将应用程序负载逐渐转移到 ROSA 上的部署。

> **Note**
>
> 这种模式假设了一个常见的用例，即你有一个本地 Git、Artif JFrog actory 和 Jenkins 管道。这种方法要求您建立从本地网络到 AWS 直通的网络连接 AWS Direct Connect，并且必须先设置 ROSA 集群，然后再按照 Epics 部分中的说明进行操作。有关详细信息，请参阅 “先决条件” 部分。

下图显示了此方法的工作流程。

MTC 方法

您可以使用容器迁移工具包 (MTC) 在不同的 Kubernetes 环境之间迁移容器化工作负载，例如从 ARO 迁移到 ROSA。MTC 通过自动执行多项关键任务并提供管理迁移生命周期的全面框架来简化迁移过程。

下图显示了此方法的工作流程。

工具

AWS 服务

- AWS DataSync是一项在线数据传输和发现服务，可帮助您在 AWS 存储服务之间移动文件或对象数据。
- AWS Direct Connect通过标准以太网光纤电缆将您的内部网络链接到某个 AWS Direct Connect 位置。通过此连接，您可以直接创建面向公众的虚拟接口， AWS 服务 同时绕过网络路径中的互联网服务提供商。
- AWS PrivateLink帮助您创建从您的虚拟私有云 (VPCs) 到 VPC 外部服务的单向私有连接。
- AWS 云端 Red Hat OpenShift 服务 (ROSA) 是一项托管服务，可帮助红帽 OpenShift 用户在上构建、扩展和管理容器化应用程序。 AWS
- AWS Security Token Service (AWS STS) 可帮助您为用户申请临时的有限权限证书。

其他工具

- 容器迁移工具包 (MTC) 提供了用于将容器化应用程序从 ARO 迁移到 ROSA 的控制台和 API。

最佳实践

- 为了提高弹性，如果您有安全合规性工作负载，请设置一个使用的 PrivateLink多可用区 ROSA 集群。有关更多信息，请参阅 ROSA 文档。

> **Note**
>
> PrivateLink 安装后无法进行配置。

- 用于复制存储库的 S3 存储桶不应公开。使用相应的 S3 存储桶策略来限制访问权限。
- 如果您选择 MTC 方法，请使用阶段迁移选项来缩短直接转换期间的停机时间。
- 在配置 ROSA 集群之前和之后，请检查您的服务配额。如有必要，请根据您的要求申请增加配额。有关更多信息，请参阅服务配额文档。
- 查看 ROSA 安全指南并实施安全最佳实践。
- 我们建议您在安装后删除默认集群管理员。有关更多信息，请参阅红帽 OpenShift 文档。
- 使用机器池自动扩展缩小 ROSA 集群中未使用的工作节点以优化成本。欲了解更多信息，请参阅 ROSA 研讨会。

- 使用红帽 OpenShift 容器平台成本管理服务，更好地了解和跟踪云和容器的成本。欲了解更多信息，请参阅 ROSA 研讨会。
- 使用监控和审计 ROSA 集群基础设施服务和应用程序 AWS 服务。欲了解更多信息，请参阅 ROSA 研讨会。

操作说明

选项 1：使用 CI/CD 管道

Task	描述	所需技能
将新的 ROSA 集群添加到 Jenkins。	1. 在 Jenkins 控制台上，选择管理 Jenkins，配置系统。 2. 在管理 Jenkins 页面的OpenShift 插件部分，选择添加新集群。 3. 提供所需的信息，例如集群名称、API 服务器 URL 和令牌信息，以便向 ROSA 进行身份验证。	AWS 管理员、AWS 系统管理员、AWS DevOps
将oc客户端添加到你的 Jenkins 节点。	1. 在 Jenkins 控制台上，选择管理 Jenkins，全局工具配置。 2. 在 “OpenShift 客户端工具” 部分，安装与 ROSA 集群版本相同的 OpenShift CLI (oc) 客户端版本。	AWS 管理员、AWS 系统管理员、AWS DevOps
创建一个新的 Git 分支。	在您的 Git 存储库中为创建一个新分支`rosa-dev`。此分支将 ROSA 的代码或配置参数更改与现有管道分开。	AWS DevOps

Task	描述	所需技能
为 ROSA 添加图片。	在构建阶段，使用不同的标签来识别从 ROSA 管道构建的映像。	AWS 管理员、AWS 系统管理员、AWS DevOps
创建管道。	创建一个与现有管道相似的新 Jenkins 管道。对于此管道，请使用您之前创建的 `rosa-dev` Git 分支，并确保包含与现有管道相同的 Git 签出、代码扫描和构建阶段。	AWS 管理员、AWS 系统管理员、AWS DevOps
添加 ROSA 部署阶段。	在新管道中，添加一个部署到 ROSA 集群的阶段，并引用您添加到 Jenkins 全局配置中的 ROSA 集群。	AWS 管理员、AWS DevOps、AWS 系统管理员
开始新构建。	在 Jenkins 中，选择您的管道并选择 “立即构建”，或者通过在 Git 中提交对`rosa-dev`分支的更改来开始新的构建。	AWS 管理员、AWS DevOps、AWS 系统管理员
验证部署。	使用 oc 命令或 ROSA 控制台验证应用程序是否已部署到您的目标 ROSA 集群上。	AWS 管理员、AWS DevOps、AWS 系统管理员
将数据复制到目标集群。	对于有状态的工作负载，请使用 AWS DataSync 诸如 rsync 之类的开源实用程序将数据从源集群复制到目标集群，也可以使用 MTC 方法。有关更多信息，请参阅 AWS DataSync 文档。	AWS 管理员、AWS DevOps、AWS 系统管理员

Task	描述	所需技能
测试您的应用程序。	1. 使用 oc route 命令或 ROSA 控制台检索应用程序的路由 URL。 2. 使用路由 URL 测试您的应用程序。	AWS 管理员、AWS DevOps、AWS 系统管理员
切开。	如果测试成功，请使用相应的 Amazon Route 53 策略将流量从 ARO 托管的应用程序转移到 Rosa 托管的应用程序。完成此步骤后，您的应用程序的工作负载将完全过渡到 ROSA 集群。	AWS 管理员、AWS 系统管理员

选项 2：使用 MTC

Task	描述	所需技能
安装 MTC 操作员。	在 ARO 和 ROSA 集群上安装 MTC 运算符： 1. 在 ARO 或 ROSA 控制台中，导航到操作员OperatorHub页面。 2. 在 “按关键字筛选” 框中，查找或键入 MTC。 3. 选择 MTC 运算符以显示其他信息。 4. 阅读有关操作员的信息后，选择 “安装”。	AWS 管理员、AWS DevOps、AWS 系统管理员
配置到复制存储库的网络流量。	如果您使用的是代理服务器，请将其配置为允许复制存储库和集群之间的网络流量。复制	AWS 管理员、AWS DevOps、AWS 系统管理员

Task	描述	所需技能
	存储库是 MTC 用来迁移数据的中间存储对象。迁移期间，源集群和目标集群必须具有对复制存储库的网络访问权限。	
将源集群添加到 MTC。	在 MTC Web 控制台上，添加 ARO 源集群。	AWS 管理员、AWS DevOps、AWS 系统管理员
添加 Amazon S3 作为您的复制存储库。	在 MTC Web 控制台上，将 Amazon S3 存储桶（参见先决条件）添加为复制存储库。	AWS 管理员、AWS DevOps、AWS 系统管理员
制定迁移计划。	在 MTC Web 控制台上，创建迁移计划并将数据传输类型指定为 “复制”。这将指示 MTC 将数据从源 (ARO) 集群复制到 S3 存储桶，然后从存储桶复制到目标 (ROSA) 集群。	AWS 管理员、AWS DevOps、AWS 系统管理员
运行迁移计划。	使用 Stag e 或 Cutover 选项运行迁移计划： • 选择 Stag e 可在不停止应用程序的情况下将数据复制到目标集群。在直接转换迁移之前，您可以运行多阶段迁移来复制大部分数据。这缩短了切换迁移的持续时间。 • 选择 Cutover 可在将资源移至目标群集的同时停止源群集上的应用程序。 为防止应用程序停止，可以清除 “迁移期间在源集群上暂停事务” 复选框。	AWS 管理员、AWS DevOps、AWS 系统管理员

故障排除

事务	解决方案
连接问题	当您将容器工作负载从 ARO 迁移到 ROSA 时，您可能会遇到连接问题，应解决这些问题以确保成功迁移。要解决迁移期间的这些连接问题（在本表中列出），必须进行细致的规划、与网络和安全团队的协调以及全面的测试。实施渐进式迁移策略并在每个步骤中验证连接，将有助于最大限度地减少潜在的干扰，并确保从 ARO 平稳过渡到 ROSA。
网络配置差异	ARO 和 ROSA 的网络配置可能有所不同，例如虚拟网络 (VNet) 设置、子网和网络策略。为了在服务之间进行适当的通信，请确保两个平台之间的网络设置保持一致。
安全组和防火墙规则	ROSA 和 ARO 可能有不同的默认安全组和防火墙设置。请务必调整和更新这些规则，以允许必要的流量在迁移期间保持容器和服务之间的连接。
IP 地址和 DNS 变更	迁移工作负载时，IP 地址和 DNS 名称可能会发生变化。重新配置依赖静态 IPs 或特定 DNS 名称的应用程序。
外部服务访问权限	如果您的应用程序依赖外部服务（例如数据库或） APIs，则可能需要更新其连接设置以确保它们可以与 ROSA 的新服务进行通信。
Azure 私有链接配置	如果您在 ARO 中使用 Azure 私有链接或私有端点服务，则需要在 ROSA 中设置等效功能，以确保资源之间的私有连接。
AWS VPN 或 AWS Direct Connect 设置	如果您的本地网络与 ARO 之间存在 AWS Direct Connect 连接 AWS VPN 或连接，则需要与

事务	解决方案
	ROSA 建立类似的连接，以便与本地资源进行不间断的通信。
入口和负载均衡器设置	ARO 和 ROSA 的入口控制器和负载均衡器的配置可能有所不同。重新配置这些设置以保持对服务的外部访问权限。
证书和 TLS 处理	如果您的应用程序使用 SSL 证书或 TLS，请确保证书有效且在 ROSA 中配置正确。
容器注册表访问权限	如果您的容器托管在外部容器注册表中，请为 ROSA 设置正确的身份验证和访问权限。
监控和日志记录	更新监控和日志配置以反映 ROSA 上的新基础架构，这样您就可以继续有效地监控容器的运行状况和性能。

相关资源

AWS参考文献

- 什么是 AWS 云端 Red Hat OpenShift 服务？（ROSA 文档）
- 开始使用 ROSA（ROSA 文档）
- AWS 云端 Red Hat OpenShift 服务 实施策略（AWS 规范性指导）
- AWS 云端 Red Hat OpenShift 服务 现在 GA（AWS 博客文章）
- ROSA 工作坊
- ROSA 常见问题
- ROSA 研讨会常见问题
- ROSA 定价

红帽 OpenShift 文档

- 快速安装集群 AWS
- 在受限网络 AWS 中安装集群

- 在现有 VPC AWS 上安装集群
- 使用模板在用户配置的基础架构上 AWS 安装集群 CloudFormation
- 在具有用户配置基础 AWS 架构的受限网络中安装集群
- 使用自定义功能安装 AWS 集群
- OpenShift CLI 入门

其他信息

在 MTC 和 CI/CD 管道重新部署选项之间进行选择

将应用程序从一个 OpenShift 集群迁移到另一个集群需要仔细考虑。理想情况下，您希望使用 CI/CD 管道来重新部署应用程序并处理永久卷数据的迁移，从而实现平稳过渡。但是，实际上，集群上正在运行的应用程序会随着时间的推移而受到不可预见的变化的影响。这些更改可能导致应用程序逐渐偏离其原始部署状态。MTC 为命名空间的确切内容不确定且将所有应用程序组件无缝迁移到新集群很重要的场景提供了解决方案。

做出正确的选择需要评估您的具体场景并权衡每种方法的好处。通过这样做，您可以确保成功实现符合您的需求和优先事项的无缝迁移。以下是在这两个选项之间进行选择的其他指南。

CI/CD 管道重新部署

如果使用管道可以放心地重新部署应用程序，则推荐使用 CI/CD 管道方法。这样可以确保迁移是可控的、可预测的，并且与您现有的部署实践保持一致。选择此方法时，您可以使用蓝/绿部署或金丝雀部署策略将负载逐渐转移到 ROSA 上的部署。在这种情况下，此模式假设 Jenkins 正在从本地环境协调应用程序部署。

优势：

- 您不需要对源 ARO 集群具有管理权限，也不需要在源集群或目标集群上部署任何操作员。
- 这种方法可以帮助您使用 DevOps 策略逐步切换流量。

劣势：

- 测试应用程序的功能需要付出更多的努力。
- 如果您的应用程序包含永久性数据，则需要额外的步骤才能使用 AWS DataSync 或其他工具来复制数据。

MTC 迁移

在现实世界中，正在运行的应用程序可能会发生意想不到的变化，从而导致它们偏离初始部署。如果您不确定源集群上应用程序的当前状态，请选择 MTC 选项。例如，如果您的应用程序生态系统跨越各种组件、配置和数据存储卷，我们建议您选择 MTC，以确保覆盖应用程序及其整个环境的完整迁移。

优势：

- MTC 提供工作负载的完整备份和恢复。
- 迁移工作负载时，它会将永久性数据从源复制到目标。
- 它不需要访问源代码存储库。

劣势：

- 您需要管理权限才能在源集群和目标集群上安装 MTC 操作员。
- 该 DevOps 团队需要接受培训，才能使用 MTC 工具和执行迁移。

优化 AWS App2Container 生成的 Docker 映像

由 Varun Sharma (AWS) 创建

摘要

AWS App2Container 是一款命令行工具，无需更改代码即可帮助将本地或虚拟机上运行的现有应用程序转换至容器。

根据应用程序类型，App2Container 采用一种保守的方法来识别依赖项。在进程模式下，应用程序服务器上的所有非系统文件都包含在容器映像中。在这种情况下，可能会生成相当大的映像。

此模式提供了一种优化由 App2Container 生成容器映像的方法。它适用于 App2Container 处理模式下发现的所有 Java 应用程序。模式中定义的工作流旨在在应用程序服务器运行。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 在 Linux 服务器上的应用服务器上运行的 Java 应用程序
- 已安装和设置 App2Container，满足所有先决条件，位于 Linux 服务器上

架构

源技术堆栈

- 在 Linux 服务器上运行的 Java 应用程序

目标技术堆栈

- 由 App2Container 生成的 Docker 映像

目标架构流程

1. 发现在应用程序服务器上运行的应用程序，然后分析这些应用程序。
2. 容器化应用程序。

3. 评估 Docker 映像大小。如果映像太大，请继续执行步骤 4。
4. 使用 shell 脚本（附件）来识别大文件。
5. 更新 `analysis.json` 文件中的 `appExcludedFiles` 和 `appSpecificFiles` 列表。

工具

工具

- AWS App2Container – AWS App2Container（A2C）是一款命令行工具，可帮助您直接迁移在本地部署的数据中心或在虚拟机上运行的应用程序，以便它们在由 Amazon Elastic Container Service (Amazon ECS) 或 Amazon Elastic Kubernetes Service (Amazon EKS) 托管的容器中运行。

代码

附上了 `optimizeImage.sh` shell 脚本和示例 `analysis.json` 文件。

该 `optimizeImage.sh` 文件是实用程序脚本，用于查看 App2Container 生成文件 `ContainerFiles.tar` 的内容。审查可以识别出较大、且可以排除的文件或子目录。该脚本是以下 tar 命令的包装器。

```
tar -Ptvf <path>|tr -s ' '|cut -d ' ' -f3,6| awk '$2 ~/<filetype>$/'| awk '$2 ~/
^<toplevel>/'| cut -f1-<depth> -d'/'|awk '{ if ($1>= <size>) arr[$2]+=$1 } END { for
 (key in arr) { if(<verbose>) printf("%-50s\t%-50s\n", key, arr[key]) else printf("%s,
\n", key) } } '|sort -k2 -nr
```

在 tar 命令中，脚本使用以下值：

`path`	`ContainerFiles.tar` 的路径
`filetype`	要匹配的文件类型
`toplevel`	顶层顶层目录匹配
`depth`	绝对路径深度
`size`	每个文件的大小

脚本执行以下操作：

1. 它使用 `tar -Ptvf` 列出文件而不提取它们。
2. 它按文件类型筛选文件，顺序从顶级目录开始。
3. 它根据深度生成绝对路径，以作为索引。
4. 根据索引和存储，提供了子目录的总大小。
5. 它打印子目录的大小。

您也可在 tar 命令中手动替换值。

操作说明

发现、分析以及容器化应用程序

Task	描述	所需技能
探索本地 Java 应用程序。	若要发现应用程序服务器上运行的所有应用程序，请运行以下命令。 `sudo app2container inventory`	AWS DevOps
分析所发现的应用程序。	要使用在清单阶段获得的 `application-id` 分析每个应用程序，请运行以下命令。 `sudo app2container analyze --application-id <java-app-id>`	AWS DevOps
对分析的应用程序执行容器化。	如要容器化应用程序，请运行以下命令。 `sudo app2container containerize --application-id <application-id>`	AWS DevOps

Task	描述	所需技能
	该命令在工作区位置生成 Docker 映像和 tar 包。 如果 Docker 映像太大，则请继续执行下一步。	

识别 appExcludedFiles 并 appSpecificFiles 从 App2Container 中提取的 tar 文件

Task	描述	所需技能
确定构件 tar 文件的大小。	确定 `{workspace}/{java-app-id}/Artifacts` 中的 `ContainerFiles.tar` 文件，其中 `workspace` 是 App2Container 工作空间，`java-app-id` 是应用程序 ID。 `./optimizeImage.sh -p /{workspace}/{java-app-id}/Artifacts/ContainerFiles.tar -d 0 -t / -v` 这是优化后 tar 文件的总大小。	AWS DevOps

列出/目录下的子目录及大小。	要确定 / 顶级目录下主要子目录的大小，请运行以下命令。
	
	`./optimizeImage.sh -p /{workspace}/{java-app-id}/Artifacts/ContainerFiles.tar -d 1 -t / -s 1000000 -v`
	`/var`
	`554144711`
		AWS DevOps

Task	描述	所需技能

	<pre>/usr
	     2097300819
	/tmp
	       18579660
	/root
	        43645397
	/opt
	       222320534
	/home
	       65212518
	/etc
	        11357677</pre>	

标识/目录下的大型子目录。	对于上一个命令中列出的每个主要子目录，请确定其子目录大小。使用 `-d` 增加深度，使用 `-t` 指示顶级目录。 例如，将 `/var` 用作顶级目录。在 `/var` 下，标识所有大型子目录及其大小。 <pre>./optimizeImage.sh -p /{workspace}/{java-app-id}/Artifacts/ContainerFiles.tar -d 2 -t /var -s 1000000 -v</pre> 对上一步中列出的每个子目录（例如 `/usr`、`/tmp`、`/opt` 和 `/home`）重复此过程。	AWS DevOps

Task	描述	所需技能
分析/目录下的每个子目录中的大文件夹。	对于上一步中列出的每个子目录，请确定运行此应用程序所需所有文件夹。 例如，使用上一步中的子目录，列出 /var 目录中的所有子目录及其大小。确定应用程序所需所有子目录。 `/var/tmp 237285851 /var/lib 24489984 /var/cache 237285851` 若要排除应用程序不需要的子目录，请在 analysis.json 文件中将这些子目录添加至 containerParameters 的 appExcludedFiles 部分中 随附示例 analysis.json 文件。	AWS DevOps

Task	描述	所需技能
从 AppExcludes 列表中识别所需文件。	对于添加至 AppExcludes 列表中的每个子目录，请标识该子目录中应用程序所需所有文件。在 analysis.json 文件，在 `containerParameters` 的 `appSpecificFiles` 部分添加指定文件或子目录。 例如，如果 `/usr/lib` 目录已添加到排除列表中，但应用程序需要 `/usr/lib/jvm` 目录，则将其 `/usr/lib/jvm` 添加至该 `appSpecificFiles` 部分。	AWS DevOps

再次提取应用程序，并对其进行容器化

Task	描述	所需技能
对分析的应用程序执行容器化。	运行以下命令以应用配置文件。 `sudo app2container containerize --application-id <application-id>` 该命令在工作区位置生成 Docker 映像和 tar 包。	AWS DevOps
确定构件 tar 文件的大小。	确定 `{workspace}/{java-app-id}/Artifacts` 中的 `ContainerFiles.tar` 文件，其中 `workspace` 是 App2Container 工作空	AWS DevOps

Task	描述	所需技能
	间， `java-app-id` 是应用程序 ID。 `./optimizeImage.sh -p /{workspace}/{java-app-id}/Artifacts/ContainerFiles.tar -d 0 -t / -v` 这是优化后 tar 文件的总大小。	
运行 Docker 映像。	要验证映像启动时是否有错误，请使用以下命令在本地运行 Docker 映像。 要识别容器的 `imageId`，请使用 `docker images |grep java-app-id` 。 若要运行容器，请使用 `docker run -d <image id>`。	AWS DevOps

相关资源

- 什么是 App2Container？
- AWS App2Container — 一款适用于 Java 和.NET 应用程序的全新容器化工具（博文）

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用节点关联性、污点和容忍度将 Kubernetes 容器组（pod）置于 Amazon EKS 上

由 Hitesh Parikh (AWS) 和 Raghu Bhamidimarri (AWS) 编写

摘要

此模式演示了如何使用 Kubernetes 节点亲和性、节点污点以及容器组（pod）容忍度 在 Amazon Web Services (AWS) Cloud 上的 Amazon Elastic Kubernetes Service (Amazon EKS) 集群中的特定 Worker 节点上调度应用程序容器组（pod）。

污点是一种节点属性，它使节点能够拒绝一组容器组（pod）。容忍度是一个容器组（pod）属性，它允许 Kubernetes 调度器在具有匹配污点的节点上调度容器组（pod）。

但是，仅凭容忍度并不能阻止调度器将容器组（pod）放在没有任何污点的 Worker 节点上。例如，具有容忍度的计算密集型容器组（pod）可能会无意中被调度到通用无污染的节点上。在这种情况下，容器组（pod）的节点亲和性属性会指示调度器将容器组（pod）放在符合节点亲和性中指定的节点选择标准的节点上。

污点、容忍度和节点亲和性共同指示调度器在具有匹配污点的节点和与容器组（pod）上指定的节点亲和性节点选择标准相匹配的节点标签上一致地调度容器组（pod）。

此模式提供了 Kubernetes 部署清单文件示例，以及创建 EKS 集群、部署应用程序和验证容器组（pod）放置位置的步骤。

先决条件和限制

先决条件

- 已配置凭证的 Amazon Web Services account 在您的 Amazon Web Services account 上创建资源
- AWS 命令行界面（AWS CLI）
- eksctl
- kubectl
- 安装了 Docker（对于正在使用的操作系统），引擎已启动（有关 Docker 许可要求的信息，请参阅 Docker 网站）
- Java 版本 11 或更高版本
- 在你最喜欢的集成开发环境 (IDE) 上运行的 Java 微服务；例如 IntelliJ IDEA 社区版或 Eclipse（如果你没有 Java 微服务，请参阅在 A mazon EKS 上部署示例 Java 微服务模式和使用 Spring 微服务获取创建微服务的帮助）

限制

- 此模式不提供 Java 代码，并且假设您已经熟悉 Java。若要创建基本 Java 微服务，请参阅在 Amazon EKS 上部署示例 Java 微服务。
- 本文中的步骤创建会产生成本的 AWS 资源。确保在完成实施与验证模式步骤后清理 AWS 资源。

架构

目标技术堆栈

- Amazon EKS
- Java
- Docker
- Amazon Elastic Container Registry (Amazon ECR)

目标架构

该解决方案架构图显示 Amazon EKS 具有两个容器组（pod）（部署 1 和部署 2）和两个节点组（ng1 和 ng2），每个节点组有两个节点。容器组（pod）和节点具有以下属性。

	部署 1 容器组（pod）	部署 2 容器组（pod）	节点组 1 (ng1)	节点组 2 (ng2)
容忍度	键：classified_workload，值：true，效果：NoSchedule 键：machine_learning_workload，值：true，效果：NoSchedule	无		
节点亲和性	密钥：alpha.eksctl.io/	无	nodeGroups.name = ng1	

	nodegroup-name = ng1;		
污点		键：classid ed_workload，值：true，效果：NoSchedule 键：machine _learning _workload，值：true，效果：NoSchedule	无

1. 部署 1 容器组（pod）定义了容忍度和节点亲和性，这会指示 Kubernetes 调度器将部署容器组（pod）放在节点组 1(ng1) 节点上。
2. 节点组 2 (ng2) 没有与部署 1 的节点亲和性节点选择器表达式相匹配的节点标签，因此容器组（pod）不会被调度在 ng2 节点上。
3. 部署 2 容器组（pod）在部署清单中没有任何容忍度或节点亲和性。由于节点上有污点，调度器将拒绝在节点组 1 上调度部署 2 容器组（pod）。
4. 部署 2 容器组（pod）将改为放在节点组 2 上，因为这些节点没有任何污点。

这种模式表明，通过使用污点和容忍度，再加上节点亲和性，您可以控制容器组（pod）在特定 Worker 节点集上的放置。

工具

Amazon Web Services

- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可帮助您在 AWS 上运行 Kubernetes，而无需安装或维护您自己的 Kubernetes 控制面板或节点。

- eksctl 在 AWS 上等同于 kubectl，可帮助创建 EKS。

其他工具

- Docker 是一组平台即服务（PaaS）产品，它们使用操作系统级别的虚拟化技术在容器中交付软件。
- kubectl：针对 Kubernetes 集群运行命令的命令行界面。

操作说明

创建 EKS 集群

Task	描述	所需技能

创建集群 .yaml 文件。	使用以下代码创建名为 `cluster.yaml` 的文件。
	
	`apiVersion: eksctl.io/v1alpha5`
	`kind: ClusterConfig`
	
	`metadata:`
	`  name: eks-taint-demo`
	`  region: us-west-1`
	
	`# Unmanaged nodegroups with and without taints.`
	`nodeGroups:`
	`  - name: ng1`
	`  instanceType: m5.xlarge`
	`  minSize: 2`
	`  maxSize: 3`
	`  taints:`
	`  - key: classified_workload`
	`  value: "true"`
	`  effect: NoSchedule`
	`  - key: machine_learning_workload`
		应用程序所有者、AWS DevOps、云管理员、 DevOps 工程师

Task	描述	所需技能

	<pre>value: "true"
	effect:
	NoSchedule
	
	- name: ng2
	instanceType:
	m5.xlarge
	minSize: 2
	maxSize: 3</pre>	

使用 eksctl 创建集群。	运行 `cluster.yaml` 文件以创建 EKS 集群。创建集群可能耗时数分钟。
	
	<pre>eksctl create cluster -f
	cluster.yaml</pre>
		AWS DevOps、AWS 系统管理员、应用程序开发者

创建映像并将其上传至 Amazon ECR

Task	描述	所需技能
创建 Amazon ECR 私有存储库。	要创建 Amazon ECR 存储库，请参阅创建私有存储库。请注意存储库 URI。	AWS DevOps， DevOps 工程师，应用程序开发人员

创建 Dockerfile。	如果您有要用于测试模式的现有 Docker 容器映像，您可跳过此步骤。
	
	若要创建 Dockerfile，请使用以下代码段作为参考。如果遇到错误，请参阅故障排除部分。
	
	<pre>FROM adoptopenjdk/openj
	dk11:jdk-11.0.14.1_1-
	alpine</pre>
		AWS DevOps， DevOps 工程师

Task	描述	所需技能
	``` RUN apk add maven WORKDIR /code # Prepare by downloadi ng dependencies ADD pom.xml /code/pom .xml RUN ["mvn", "dependen cy:resolve"] RUN ["mvn", "verify"] # Adding source, compile and package into a fat jar ADD src /code/src RUN ["mvn", "package"] EXPOSE 4567 CMD ["java", "-jar", "target/eksExample -jar-with-dependen cies.jar"] ```	
创建 pom.xml 和源文件，然后构建和推送 Docker 映像。	要创建 `pom.xml` 文件和 Java 源文件，请参阅在 Amazon EKS 上部署示例 Java 微服务模式。 使用该模式中的指令构建和推送 Docker 映像。	AWS DevOps， DevOps 工程师，应用程序开发人员

部署到 Amazon EKS

Task	描述	所需技能
创建部署 .yaml 文件。	要创建 `deployment.yaml` 文件，请使用其他信息部分中的代码。 在代码中，节点亲和性的关键是您在创建节点组时创建的任何标签。此模式使用 eksctl 创建默认标签。有关自定义标签的信息，请参阅 Kubernetes 文档中的将容器组（pod）分配至节点。 节点亲和性键的值是 `cluster.yaml` 创建的节点组的名称。 若要获取污点的键和值，请运行以下命令。 `kubectl get nodes -o json | jq '.items[].spec.taints'` 映像是您在上一个步骤中创建的 Amazon ECR 存储库 URI。	AWS DevOps， DevOps 工程师，应用程序开发人员
部署文件。	若要部署到 Amazon EKS，请运行以下命令。 `kubectl apply -f deployment.yaml`	应用程序开发人员、 DevOps 工程师、AWS DevOps

Task	描述	所需技能

检查部署。	1. 要检查容器组（pod）是否准备就绪，请运行以下命令。
	
	`kubectl get pods -o wide`
	
	如果容器组（pod）已准备就绪，则输出内容应如下所示，且 STATUS 为“运行中”。
	
	`NAME READY STATUS RESTARTS AGE IP NODE NOMINATED NODE READINESS GATES`
	`<pod_name> 1/1 Running 0 12d 192.168.18.50 ip-192-168-20-110.us-west-1.compute.internal <none> <none>`
	
	记下容器组（pod）的名称和节点的名称。您可跳过下一步。
	2. （可选）要获取有关容器组（pod）的更多详细信息并检查容器组（pod）的容忍度，请运行以下命令。
	
	`kubectl describe pod <pod_name>`
		应用程序开发人员、 DevOps 工程师、AWS DevOps

Task	描述	所需技能

	输出示例参见其他信息部分。
	3. 要验证容器组（pod）在节点上的放置位置是否正确，请运行以下命令。
	`kubectl describe node <node name> | grep -A 1 "Taints"`
	确认节点上的污点与容忍度相匹配，并且节点上的标签与 deployment.yaml 中定义的节点亲和性相匹配。
	应将具有容忍度和节点亲和性的容器组（pod）放在具有匹配污点和节点亲和性标签的节点上。前述命令为您提供了节点上的污点。下面是一个示例输出。
	`kubectl describe node ip-192-168-29-181.us-west-1.compute.internal | grep -A 1 "Taints"`
	`Taints: classifled_workload=true:NoSchedule`
	`machine_learning_workload=true:NoSchedule`	

Task	描述	所需技能

	此外，运行以下命令来检查放置容器组（pod）的节点上是否有与节点亲和性节点标签相匹配的标签。
	`kubectl get node <node name> --show-labels`
	4. 要验证应用程序是否正在执行其预期任务，请运行以下命令查看容器组（pod）日志。
	`kubectl logs -f <name-of-the-pod>`	

Task	描述	所需技能

在没有容忍度和节点亲和性的情况下创建第二个部署 .yaml 文件。	此额外步骤是为了验证当部署清单文件中没有指定节点亲和性或容忍度时，生成的容器组（pod）不会被调度到有污点的节点上。(它应该被调度到没有任何污点的节点上)。使用以下代码创建名为 `deploy_no_taint.yaml` 的新部署文件。
	
	`apiVersion: apps/v1`
	`kind: Deployment`
	`metadata:`
	`  name: microservice-deployment-non-tainted`
	`spec:`
	`  replicas: 1`
	`  selector:`
	`  matchLabels:`
	`  app.kubernetes.io/name: java-microservice-no-taint`
	`  template:`
	`  metadata:`
	`  labels:`
	`  app.kubernetes.io/name: java-microservice-no-taint`
	`  spec:`
	`  containers:`
	`  - name: java-microservice-container-2`
	`  image: <account_number>.dkr.ecr<region>.amazonaws.com/<repository_name>:latest`
	`  ports:`
		应用程序开发人员、AWS DevOps、 DevOps 工程师

Task	描述	所需技能
	`- container` `Port: 4567`	

部署第二个部署 .yaml 文件，并验证容器组（pod）的位置	1. 运行以下命令。
	`kubectl apply -f deploy_no_taint.yaml`
	2. 部署成功后，运行与之前运行的相同命令来检查容器组（pod）在没有污点的节点组中的位置。
	`kubectl describe node <node_name> | grep "Taints"`
	输出应为以下内容。
	`Taints: <none>`
	测试已完成。
		应用程序开发人员、AWS DevOps、 DevOps 工程师

清理资源

Task	描述	所需技能
清除资源。	要避免对保持运行的资源产生 AWS 费用，请使用以下命令。 `eksctl delete cluster --name <Name of the cluster> --region <region-code>`	AWS DevOps，应用程序开发者

故障排除

事务	解决方案
如果您的系统使用 arm64 架构（特别是如果您在 M1 Mac 上运行此架构），则其中一些命令可能无法运行。以下行可能会出错。 `FROM adoptopenjdk/openjdk11:jdk-11.0.14.1_1-alpine`	如果您在运行 Dockerfile 时遇到错误，请将 FROM 行替换为以下行。 `FROM bellsoft/liberica-openjdk-alpine-musl:17`

相关资源

- 在 Amazon EKS 上部署示例 Java 微服务
- 创建 Amazon ECR 私有存储库
- 将容器组（pod）分配给节点（Kubernetes 文档）
- 污点和容忍度（Kubernetes 文档）
- Amazon EKS
- Amazon ECR
- AWS CLI
- Docker
- IntelliJ IDEA CE
- Eclipse

其他信息

部署 .yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: microservice-deployment
spec:
  replicas: 1
  selector:
```

```
    matchLabels:
      app.kubernetes.io/name: java-microservice
  template:
    metadata:
      labels:
        app.kubernetes.io/name: java-microservice
    spec:
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
              - matchExpressions:
                  - key: alpha.eksctl.io/nodegroup-name
                    operator: In
                    values:
                      - <node-group-name-from-cluster.yaml>
      tolerations: #only this pod has toleration and is viable to go to ng with taint
        - key: "<Taint key>" #classified_workload in our case
          operator: Equal
          value: "<Taint value>" #true
          effect: "NoSchedule"
        - key: "<Taint key>" #machine_learning_workload in our case
          operator: Equal
          value: "<Taint value>" #true
          effect: "NoSchedule"
      containers:
        - name: java-microservice-container
          image: <account_number>.dkr.ecr<region>.amazonaws.com/
<repository_name>:latest
          ports:
            - containerPort: 4567
```

描述容器组（pod）示例输出

```
Name:         microservice-deployment-in-tainted-nodes-5684cc495b-vpcfx
Namespace:    default
Priority:     0
Node:         ip-192-168-29-181.us-west-1.compute.internal/192.168.29.181
Start Time:   Wed, 14 Sep 2022 11:06:47 -0400
Labels:       app.kubernetes.io/name=java-microservice-taint
              pod-template-hash=5684cc495b
Annotations:  kubernetes.io/psp: eks.privileged
Status:       Running
```

```
IP:           192.168.13.44
IPs:
  IP:           192.168.13.44
Controlled By:  ReplicaSet/microservice-deployment-in-tainted-nodes-5684cc495b
Containers:
  java-microservice-container-1:
    Container ID:
 docker://5c158df8cc160de8f57f62f3ee16b12725a87510a809d90a1fb9e5d873c320a4
    Image:          934188034500.dkr.ecr.us-east-1.amazonaws.com/java-eks-apg
    Image ID:       docker-pullable://934188034500.dkr.ecr.us-east-1.amazonaws.com/
java-eks-apg@sha256:d223924aca8315aab20d54eddf3443929eba511b6433017474d01b63a4114835
    Port:           4567/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Wed, 14 Sep 2022 11:07:02 -0400
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-ddvvw (ro)
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
Volumes:
  kube-api-access-ddvvw:
    Type:                     Projected (a volume that contains injected data from
 multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    ConfigMapOptional:       <nil>
    DownwardAPI:             true
QoS Class:                   BestEffort
Node-Selectors:              <none>
Tolerations:                 classifled_workload=true:NoSchedule
                             machine_learning_workload=true:NoSchedule
                             node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                             node.kubernetes.io/unreachable:NoExecute op=Exists for
 300s
Events:                      <none>
```

跨账户或区域复制已筛选的 Amazon ECR 容器映像

由 Abdal Garuba (AWS) 编写

摘要

Amazon Elastic Container Registry (Amazon ECR) 可以使用跨区域和跨账户复制功能，跨 Amazon Web Services (AWS) 区域和 Amazon Web Services account 本地复制映像存储库中的所有容器映像。（有关更多信息，请参阅 AWS Blog 文章 Amazon ECR 中的跨区域复制已实现。）但是，无法根据任何标准筛选在 AWS 区域或账户之间复制的映像。

此示例介绍了如何根据映像标签模式在 Amazon Web Services account 和区域之间复制存储在 Amazon ECR 中的容器映像。该模式使用 Amazon CloudWatch Events 来监听具有预定义自定义标签的图像的推送事件。推送事件启动一个 AWS CodeBuild 项目并将图像详细信息传递给该项目。该 CodeBuild 项目根据提供的详细信息将图像从源 Amazon ECR 注册表复制到目标注册表。

此模式跨账户复制具有特定标签的映像。例如您可使用此模式仅将可用于生产的安全映像复制到生产 Amazon Web Services account。在开发账户中，在对映像进行全面测试后，您可向安全映像添加预定义的标签，并使用此模式中的步骤将标记的映像复制到生产账户。

先决条件和限制

先决条件

- 用于源和目标 Amazon ECR 注册表的有效 Amazon Web Services account
- 此模式所用工具的管理权限
- Docker 已安装在本地计算机上以进行测试
- AWS 命令行界面（AWS CLI），用于对 Amazon ECR 进行身份验证

限制

- 此模式仅在一个 AWS 区域中监视源注册表的推送事件。您可将此模式部署到其他区域，以监视这些区域中的注册表。
- 在这种模式中，一个 Amazon CloudWatch Events 规则监听单个图像标签模式。如果要检查多个模式，则可以添加事件来监听其他映像标签模式。

架构

目标架构

自动化和扩缩

这种模式可以通过基础设施即代码（IaC）脚本实现自动化，并可以大规模部署。要使用 AWS CloudFormation 模板部署此模式，请下载附件并按照 “其他信息” 部分的说明进行操作。

您可以将多个 Amazon Events CloudWatch 事件（具有不同的自定义事件模式）指向同一 AWS CodeBuild 项目以复制多个图像标签模式，但您需要按如下方式更新`buildspec.yaml`文件（包含在附件和 “工具” 部分中）中的辅助验证以支持多种模式。

```
...
if [[ ${IMAGE_TAG} != release-* ]]; then
...
```

工具

Amazon 服务

- IAM – AWS Identity and Access Management (IAM)让您安全地控制对 Amazon Web Services 和资源的访问。在这种模式中，您需要创建跨账户 IAM 角色，AWS 在 CodeBuild 将容器映像推送到目标注册表时将担任该角色。
- Amazon ECR – Amazon Elastic Container Registry (Amazon ECR) 是一个完全托管式容器注册表，可以轻松地在任何地方存储、管理、共享和部署容器映像和构件。将图像推送到源注册表的操作会将系统事件详细信息发送到 Amazon Events 获取 CloudWatch 的事件总线。
- AWS CodeBuild — AWS CodeBuild 是一项完全托管的持续集成服务，它为执行编译源代码、运行测试和生成准备部署的工件等任务提供计算能力。此模式使用 AWS CodeBuild 执行从源 Amazon ECR 注册表到目标注册表的复制操作。
- CloudWatch 事件 — Amazon CloudWatch Events 提供一系列描述了 AWS 资源变化的系统事件。此模式使用规则将 Amazon ECR 推送操作与特定映像标签模式进行匹配。

工具

- Docker CLI – Docker 是一款可以更轻松地创建和管理容器的工具。容器将应用程序及其所有依赖项打包到一个单元或者包，可以轻松部署在支持容器运行时系统的任何平台上。

代码

您可通过两种方式实现此模式：

- 自动设置：部署附件中提供的两个 AWS CloudFormation 模板。有关说明，请参阅其他信息部分。
- 手动设置：按照操作说明部分的步骤执行操作。

示例 buildspec.yaml

如果您使用的是随此模式提供的 CloudFormation 模板，则该`buildspec.yaml`文件将包含在 CodeBuild 资源中。

```
  version: 0.2
  env:
    shell: bash
  phases:
    install:
      commands:
        - export CURRENT_ACCOUNT=$(echo ${CODEBUILD_BUILD_ARN} | cut -d':' -f5)
        - export CURRENT_ECR_REGISTRY=${CURRENT_ACCOUNT}.dkr.ecr.
${AWS_REGION}.amazonaws.com
        - export DESTINATION_ECR_REGISTRY=${DESTINATION_ACCOUNT}.dkr.ecr.
${DESTINATION_REGION}.amazonaws.com
    pre_build:
      on-failure: ABORT
      commands:
        - echo "Validating Image Tag ${IMAGE_TAG}"
        - |
          if [[ ${IMAGE_TAG} != release-* ]]; then
            aws codebuild stop-build --id ${CODEBUILD_BUILD_ID}
            sleep 60
            exit 1
          fi
        - aws ecr get-login-password --region ${AWS_REGION} | docker login -u AWS --
password-stdin ${CURRENT_ECR_REGISTRY}
        - docker pull ${CURRENT_ECR_REGISTRY}/${REPO_NAME}:${IMAGE_TAG}
    build:
      commands:
        - echo "Assume cross-account role"
        - CREDENTIALS=$(aws sts assume-role --role-arn ${CROSS_ACCOUNT_ROLE_ARN} --
role-session-name Rolesession)
        - export AWS_DEFAULT_REGION=${DESTINATON_REGION}
        - export AWS_ACCESS_KEY_ID=$(echo ${CREDENTIALS} | jq -r
 '.Credentials.AccessKeyId')
```

```
      - export AWS_SECRET_ACCESS_KEY=$(echo ${CREDENTIALS} | jq -r
'.Credentials.SecretAccessKey')
      - export AWS_SESSION_TOKEN=$(echo ${CREDENTIALS} | jq -r
'.Credentials.SessionToken')
      - echo "Logging into cross-account registry"
      - aws ecr get-login-password --region ${DESTINATION_REGION} | docker login -u
AWS --password-stdin ${DESTINATION_ECR_REGISTRY}
      - echo "Check if Destination Repository exists, else create"
      - |
        aws ecr describe-repositories --repository-names ${REPO_NAME} --region
${DESTINATION_REGION} \
        || aws ecr create-repository --repository-name ${REPO_NAME} --region
${DESTINATION_REGION}
      - echo "retag image and push to destination"
      - docker tag ${CURRENT_ECR_REGISTRY}/${REPO_NAME}:${IMAGE_TAG}
${DESTINATION_ECR_REGISTRY}/${REPO_NAME}:${IMAGE_TAG}
      - docker push ${DESTINATION_ECR_REGISTRY}/${REPO_NAME}:${IMAGE_TAG}
```

操作说明

创建 IAM 角色。

Task	描述	所需技能

创建 “ CloudWatch 活动” 角色。	在源 AWS 账户中，创建一个 IAM 角色供亚马逊 CloudWatch 活动代替。该角色应具有启动 AWS CodeBuild 项目的权限。
	
	要使用 AWS CLI 创建角色，请按照 IAM 文档中的说明进行操作。
	
	示例信任策略(`trustpolicy.json`)：
	
	`{`
	`  "Version": "2012-10-17",`
	`  "Statement": {`
		AWS 管理员、AWS DevOps、AWS 系统管理员、云管理员、云架构师、DevOps 工程师

Task	描述	所需技能

	<pre> "Effect": "Allow",
	    "Principal":
	 {"Service": "events.a
	mazonaws.com"},
	    "Action": "sts:Assu
	meRole"
	  }
	}</pre>
	示例权限策略(permissio npolicy.json)：
	<pre>{
	  "Version": "2012-10-
	17",
	  "Statement": {
	    "Effect": "Allow",
	    "Action": "codebuil
	d:StartBuild",
	    "Resource":
	 "<CodeBuild Project
	 ARN>"
	  }
	}</pre>	

Task	描述	所需技能
创建 CodeBuild 角色。	按照 IAM 文档中的说明创建一个 IAM 角色让 AWS CodeBuild 代替。角色还必须具有以下权限： • 代入目标跨账户角色权限 • 创建日志组和日志流，以及放置日志事件的权限 • 通过向角色添加亚马逊EC2ContainerRegistryReadOnly托管策略，获得所有 A mazon ECR 存储库的只读权限 • 允许停止 CodeBuild 示例信任策略(`trustpolicy.json`)： `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "Service": "codebuild.amazonaws.com" }, "Action": "sts:AssumeRole" } ] }`	AWS 管理员、AWS DevOps、AWS 系统管理员、云管理员、云架构师、DevOps 工程师

Task	描述	所需技能
	示例权限策略(permissionpolicy.json): { "Version": "2012-10-17", "Statement": [{ "Action": ["codebuild:StartBuild", "codebuild:StopBuild", "codebuild:Get*", "codebuild:List*", "codebuild:BatchGet*"], "Resource": "*", "Effect": "Allow" }, { "Action": ["logs:CreateLogGroup", "logs:CreateLogStream", "logs:PutLogEvents"], "Resource": "*",	

Task	描述	所需技能
	<pre>"Effect": "Allow" }, { "Action": "sts:AssumeRole", "Resource": "<ARN of destination role>", "Effect": "Allow", "Sid": "AssumeCrossAccountArn" }] }</pre> 按如下方式将托管策略 AmazonEC2ContainerRegistryReadOnly 附加到 CLI 命令： <pre>~$ aws iam attach-role-policy \ --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \ --role-name <name of CodeBuild Role></pre>	

Task	描述	所需技能

创建跨账户角色。	在目标 AWS 账户中，为源账户的 AWS CodeBuild 角色创建一个 IAM 角色。跨账户角色应允许容器映像创建新的存储库并将容器映像上传至 Amazon ECR。
	
	要使用 AWS CLI 创建 IAM 角色，请按照 IAM 文档中的说明进行操作。
	
	要允许上一步 CodeBuild 中的 AWS 项目，请使用以下信任策略：
	
	`{ "Version": "2012-10-17", "Statement": { "Effect": "Allow", "Principal": { "AWS": "<ARN of source codebuild role>" }, "Action": "sts:AssumeRole" } }`
	
	要允许上一步中的 AWS CodeBuild 项目将图像保存在目标注册表中，请使用以下权限策略：
	
	`{ "Version": "2012-10-17", "Statement": [`
		AWS 管理员、AWS DevOps、云管理员、云架构师、 DevOps 工程师、AWS 系统管理员

Task	描述	所需技能

	{
	"Action": [
	"ecr:GetDownloadUrlForLayer",
	"ecr:BatchCheckLayerAvailability",
	"ecr:PutImage",
	"ecr:InitiateLayerUpload",
	"ecr:UploadLayerPart",
	"ecr:CompleteLayerUpload",
	"ecr:GetRepositoryPolicy",
	"ecr:DescribeRepositories",
	"ecr:GetAuthorizationToken",
	"ecr:CreateRepository"
	],
	"Resource": "*",
	"Effect": "Allow"
	}
	]
	}	

创建 CodeBuild 项目

Task	描述	所需技能

创建 CodeBuild 项目。	按照 AWS CodeBuild 文档中的说明在源账户中创建 AWS CodeBuild 项目。该项目应与源注册表位于同一区域。
	
	项目配置如下：
	
	• 环境类型：`LINUX CONTAINER`
	• 服务角色：`CodeBuild Role`
	• 特权模式：`true`
	• 环境映像：`aws/codebuild/standard:x.x`（使用最新可用映像）
	• 环境变量：
	  • `CROSS_ACCOUNT_ROLE_ARN`：跨账户角色的 Amazon 资源名称（ARN）
	  • `DESTINATION_REGION`：跨账户区域的名称
	  • `DESTINATION_ACCOUNT`：目标账户的号码
	• 构建规范：使用工具部分中列出的 `buildspec.yaml` 文件。
		AWS 管理员、AWS DevOps、AWS 系统管理员、云管理员、云架构师、DevOps 工程师

创建事件

Task	描述	所需技能
创建事件规则。	由于该模式使用内容筛选功能，因此您需要使用 Amazon 创建事件 EventBridge。按照 EventBridge 文档中的说明创建事件和目标，并进行一些修改： • 对于 Define pattern（定义模式），选择 Event pattern（事件模式），然后选择 Custom pattern（自定义模式）。 • 将以下自定义事件模式示例代码复制到提供的文本框中： `{ "source": ["aws.ecr"], "detail-type": ["ECR Image Action"], "detail": { "action-type": ["PUSH"], "result": ["SUCCESS"], "image-tag": [{ "prefix": "release-"}] } }` • 对于选择目标，选择 AWS CodeBuild 项目，然后粘贴您在上一篇长篇故事中创建	AWS 管理员、AWS DevOps、AWS 系统管理员、云管理员、云架构师、DevOps 工程师

Task	描述	所需技能

	的 AWS CodeBuild 项目的 ARN。
	• 对于 Configure input（配置输入），选择 Input Transformer（输入转换器）。
	  • 在输入路径文本框，粘贴：
	
	  • 在输入模板文本框，粘贴：
	
	• 选择使用现有角色，然后选择您之前在创建 IAM 角色长篇故事中创建 CloudWatch 的事件角色的名称。	

```
 {"IMAGE_TAG":"$.de
tail.image-tag","R
EPO_NAME":"$.detai
l.repository-name"
}
```

```
 {"environmentVaria
blesOverride":
 [ {"name":
 "IMAGE_TAG",
 "value":<IMAGE_TAG
>},{"name":"REPO_N
AME","value":<REPO
_NAME>}]}
```

验证

Task	描述	所需技能
向 Amazon ECR 进行身份验证。	按照 Amazon ECR 文档中的步骤，向来源和目标注册表进行身份验证。	AWS 管理员、AWS DevOps、AWS 系统管理员、云管理员、 DevOps 工程师、云架构师
测试映像复制。	在您的源账户中，将容器映像推送到新的或现有的 Amazon ECR 源存储库，其映像标签前缀为 `release-`。要推送映像，请按照 Amazon ECR 文档中的步骤进行操作。 您可以在CodeBuild 控制台中监控 CodeBuild 项目的进度。 CodeBuild 项目成功完成后，登录目标 AWS 账户，打开 Amazon ECR 控制台，并确认目标的 Amazon ECR 注册表中存在该映像。	AWS 管理员、AWS DevOps、AWS 系统管理员、云管理员、云架构师、DevOps 工程师
测试映像排除。	在您的源账户中，使用没有自定义前缀的映像标签将容器映像推送到新的或现有的 Amazon ECR 源存储库。 确认 CodeBuild 项目尚未启动，并且目标注册表中未显示任何容器映像。	AWS 管理员、AWS DevOps、AWS 系统管理员、云管理员、云架构师、DevOps 工程师

相关资源

- 入门 CodeBuild
- 开始使用亚马逊 EventBridge

- Ama EventBridge zon 事件模式中基于内容的筛选
- 使用 IAM 角色委托跨 Amazon Web Services account 的访问权限
- 私有映像复制

其他信息

若要自动部署此模式的资源，请执行以下步骤：

1. 下载附件并提取两个 CloudFormation 模板：`part-1-copy-tagged-images.yaml`和`part-2-destination-account-role.yaml`。
2. 登录 AWS CloudFormation 控制台，`part-1-copy-tagged-images.yaml`在与源 Amazon ECR 注册表相同的 AWS 账户和区域中进行部署。根据需要更新参数。模板部署以下资源：
   - 亚马逊 CloudWatch 活动 IAM 角色
   - AWS CodeBuild 项目 IAM 角色
   - AWS CodeBuild 项目
   - AWS CloudWatch 活动规则
3. 记下输出选项卡中的 `SourceRoleName` 值。在下一个步骤中，您需要用到此值。
4. 在要将 Amazon ECR 容器映像复制到的 AWS 账户中部署第二个 CloudFormation 模板。`part-2-destination-account-role.yaml`根据需要更新参数。为 `SourceRoleName` 参数指定第 3 步的值 。此模板部署跨账户 IAM 角色。
5. 验证映像复制和排除，如操作说明部分的最后一步所述。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

在不重启容器的情况下轮换数据库凭证

由 Josh Joy(AWS) 编写

摘要

在 Amazon Web Services (AWS) Cloud 上，您可使用 AWS Secrets Manager 在数据库凭证的整个生命周期中轮换、管理和检索数据库凭证。用户和应用程序通过调用 Secrets Manager API 来检索密钥，而不需要对明文形式的敏感信息进行硬编码。

如果您使用容器处理微服务工作负载，则可以将凭证安全地存储在 AWS Secrets Manager 中。为了将配置与代码分开，这些凭证通常被注入至容器中。但是，定期自动轮换您的凭证非常重要。支持撤销后刷新凭证的能力也很重要。同时，应用程序需要能够轮换凭证，同时减少对下游可用性的任何潜在影响。

此示例介绍了如何在不要求容器重启的情况下在容器中轮换使用 AWS Secrets Manager 保护的密钥。此外，这种模式通过使用 Secrets Manager 客户端缓存组件减少了对 Secrets Manager 的凭证查找次数。当您使用客户端缓存组件刷新应用程序中的凭证时，无需重新启动容器即可获取轮换后的凭证。

这种方法适用于 Amazon Elastic Kubernetes Service (Amazon EKS)和 Amazon Elastic Container Service (Amazon ECS)。

涵盖了两种场景。在单用户场景中，通过检测过期的凭证，在秘密轮转时刷新数据库凭证。指示凭证缓存刷新密钥，然后应用程序重新建立数据库连接。客户端缓存组件在应用程序中缓存凭证，有助于避免在每次凭证查询时联系 Secrets Manager。凭证在应用程序内轮换，无需通过重启容器来强制刷新凭证。

第二种情况是通过在两个用户之间交替轮换密钥。拥有两名活跃用户可以减少停机的可能性，因为一名用户的凭证始终处于活动状态。当您的大型部署中的集群中可能存在较小的凭证更新传播延迟时，双用户凭证轮换会很有用。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 在 Amazon EKS 或 Amazon ECS 的容器中运行的应用程序。
- 凭证存储在 Secrets Manager 中，且启用轮换。

- 如果部署双用户解决方案，则存储在 Secrets Manager 中的第二组凭证。代码示例可以在 GitHub 存储库 aws-secrets-manager-rotation- lambdas 中找到。
- Amazon Aurora 数据库。

限制

- 本示例针对 Python 应用程序。对于 Java 应用程序，您可使用 Java 客户端缓存组件或 Secrets Manager 的 JDBC 客户端缓存库。

架构

目标架构

场景 1 - 轮换单个用户的凭证

在第一种情况下，Secrets Manager 会定期轮换单个数据库凭证。应用程序容器在 Fargate 中运行。建立第一个数据库连接后，应用程序容器会获取 Aurora 的数据库凭证。然后，Secrets Manager 缓存组件会缓存凭证，以便将来建立连接。轮换期过后，凭证将过期，数据库将返回身份验证错误。然后，应用程序通过 Secrets Manager 客户端缓存组件获取轮换后的凭证，使缓存失效，并更新凭证缓存。

在这种情况下，在轮换凭证且过时的连接使用过时的凭证时，中断可能最小。这个问题可通过使用双用户场景来解决。

场景 2 - 轮换两个用户的凭证

在第二种情况下，Secrets Manager 会定期轮换两个数据库用户凭证（Alice 和 Bob）。应用程序容器在 Fargate 集群运行。建立第一个数据库连接后，应用程序容器将获取第一个用户 (Alice) 的 Aurora 数据库凭证。然后，Secrets Manager 缓存组件会缓存凭证，以便将来建立连接。

尽管有两个用户和凭证，但只有一个有效的凭证由 Secrets Manager 管理。在这种情况下，缓存组件会定期过期并获取最新的凭证。如果 Secrets Manager 轮换周期长于缓存超时时间，则缓存组件会为第二个用户 (Bob) 获取轮换后的凭证。例如如果缓存过期时间以分钟为单位，轮换周期以天为单位，则缓存组件将在定期刷新缓存时获取新的凭证。通过这种方式，可以最大限度地减少停机时间，因为每个用户的凭证在一次 Secrets Manager 轮换中都处于活动状态。

自动化和扩缩

您可以使用 AWS 通过使用基础设施即代码 CloudFormation来部署此模式。这将生成和创建应用程序容器，创建 Fargate 任务，将容器部署至 Fargate 中，并使用 Aurora 设置和配置 Secrets Manager。有关 step-by-step部署说明，请参阅自述文件。

工具

工具

- AWS Secrets Manager 启用将硬编码凭证（包括密码）替换为对 Secrets Manager 的 API 调用，以便检索密钥。由于 Secrets Manager 可以根据计划自动轮换密钥，因此您可将长期密钥替换为短期密钥，从而降低泄露的风险。
- Docker 帮助开发人员将任何应用程序作为轻量级、便携且自给自足的容器打包、运输和运行。

代码

Python 代码示例

此模式使用 Secrets Manager 的 Python 客户端缓存组件在建立数据库连接时检索身份验证凭证。客户端缓存组件有助于避免每次联系 Secrets Manager。

现在，当轮换周期过后，缓存的凭证将过期，并且连接到数据库将导致身份验证错误。对于 MySQL，身份验证错误代码为 1045。此示例使用适用于 MySQL 的 Amazon Aurora，但您可使用其他引擎，例如 PostgreSQL。出现身份验证错误时，数据库连接异常处理代码将捕获错误。然后，它通知 Secrets Manager 客户端缓存组件刷新密钥，然后重新进行身份验证，并重新建立数据库连接。如果您使用 PostgreSQL 或其他引擎，则必须查找相应的身份验证错误代码。

容器应用程序现在可以使用轮换后的密码更新数据库密码，而无需重新启动容器。

将以下代码放入处理数据库连接的应用程序代码中。此示例使用 Django，它使用数据库包装器对数据库后端进行子类化，用于连接。如果您使用不同的编程语言或数据库连接库，请参阅您的数据库连接库以查看如何子类化数据库连接检索。

```
    def get_new_connection(self, conn_params):
        try:
            logger.info("get connection")
            databasecredentials.get_conn_params_from_secrets_manager(conn_params)
            conn =super(DatabaseWrapper,self).get_new_connection(conn_params)
            return conn
        except MySQLdb.OperationalError as e:
            error_code=e.args[0]
            if error_code!=1045:
```

```
                raise e

            logger.info("Authentication error. Going to refresh secret and try again.")
            databasecredentials.refresh_now()
            databasecredentials.get_conn_params_from_secrets_manager(conn_params)
            conn=super(DatabaseWrapper,self).get_new_connection(conn_params)
            logger.info("Successfully refreshed secret and established new database
connection.")
            return conn
```

AWS CloudFormation 和 Python 代码

- https://github.com/aws-samples/aws-secrets-manager-credential-rotation-without-container-restart

操作说明

在凭证轮换期间保持应用程序的可用性

Task	描述	所需技能
安装缓存组件。	下载并安装适用于 Python 的 Secrets Manager 客户端缓存组件。有关下载链接，请参阅相关资源部分。	开发人员
缓存工作凭证。	使用 Secrets Manager 客户端缓存组件在本地缓存工作凭证。	开发人员
更新应用程序代码，以在数据库连接出现未经授权的错误时刷新凭证。	更新应用程序代码，以使用 Secrets Manager 获取和刷新数据库凭证。添加处理未经授权的错误代码的逻辑，然后获取新轮换的凭证。请参阅 Python 代码示例部分。	开发人员

相关资源

创建 Secrets Manager 密钥

- 在 AWS KMS 中创建密钥
- 创建和管理 AWS Secrets Manager 密钥

创建 Amazon Aurora 集群

- 创建 Amazon RDS 数据库实例

创建 Amazon ECS 组件

- 使用经典控制台创建集群
- 创建 Docker 映像
- 创建私有存储库
- Amazon ECR 私有注册表
- 推送 Docker 映像
- Amazon ECS 任务定义
- 在经典控制台中创建 Amazon ECS 服务

下载和安装 Secrets Manager 客户端缓存组件

- Python 缓存客户端

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 Amazon ECS Anywhere 在亚马逊 WorkSpaces 上运行亚马逊 ECS 任务

由 Akash Kumar (AWS) 创建

摘要

Amazon Elastic Container Service (Amazon ECS) Anywhere 支持在任何环境中部署 Amazon ECS 任务，包括Amazon Web Services (AWS) 托管基础设施和客户托管基础设施。您可以使用在云中运行且始终保持最新状态的完全由 AWS 托管的控制面板来执行此操作。

企业经常使用 Amazon WorkSpaces 开发基于容器的应用程序。这需要亚马逊弹性计算云 (Amazon EC2) 或带有亚马逊 ECS 集群的 AWS Fargate 来测试和运行 ECS 任务。现在，通过使用 Amazon ECS Anywhere，您可以将亚马逊 WorkSpaces 作为外部实例直接添加到 ECS 集群，并且可以直接运行任务。这可以缩短您的开发时间，因为您可以在 Amazon 上使用本地的 ECS 集群测试容器 WorkSpaces。您还可以节省使用 EC2 或 Fargate 实例来测试容器应用程序的成本。

此模式展示了如何使用 Amazon ECS Anywhere 在亚马逊 WorkSpaces 上部署 ECS 任务。它设置 ECS 集群并使用 AWS Directory Service Simple AD 来启动 WorkSpaces。然后，示例 ECS 任务在中启动 NGINX。 WorkSpaces

先决条件和限制

- 一个有效的 Amazon Web Services account
- AWS 命令行界面（AWS CLI）
- 在您的计算机上配置的 AWS 凭证

架构

目标技术堆栈

- 虚拟私有云（VPC）
- Amazon ECS 集群
- Amazon WorkSpaces
- 具有 Simple AD 的 AWS Directory Service

目标架构

该架构包括以下服务与资源：

- 自定义 VPC 中具有公有子网和私有子网的 ECS 集群
- 在 VPC 中使用 Simple AD 为用户提供访问亚马逊的权限 WorkSpaces
- 亚马逊使用 Simple A WorkSpaces D 在 VPC 中进行了配置
- AWS Systems Manager 已激活，可将亚马逊添加 WorkSpaces 为托管实例
- 亚马逊使用 Amazon ECS 和 AWS Systems Manager 代理（SSM 代理） WorkSpaces 添加到 Systems Manager 和 ECS 集群中
- 要在 ECS 集群中运行的 WorkSpaces ECS 任务示例

工具

- AWS Directory Service Simple Active Directory（Simple AD） 是由 Samba 4 Active Directory Compatible Server 提供支持的一个独立托管目录。Simple AD 提供了 AWS 托管 Microsoft AD 提供的部分功能，包括管理用户和安全连接亚马逊 EC2 实例的功能。
- Amazon Elastic Container Service (Amazon ECS)是一项快速且可扩展的容器管理服务，可帮助运行、停止和管理集群上的容器。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Systems Manager 可帮助您管理在 Amazon Web Services Cloud 中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。
- 亚马逊 WorkSpaces可帮助您为用户配置基于云的虚拟 Microsoft Windows 或 Amazon Linux 桌面，即WorkSpaces。 WorkSpaces 无需购买和部署硬件或安装复杂的软件。

操作说明

设置 ECS 集群

Task	描述	所需技能
创建和配置 ECS 集群。	要创建 ECS 集群，按照 AWS 文档中的说明进行操作，包括以下步骤： • 对于选择集群兼容性，请选择仅联网，这将支持 Amazon WorkSpace 作为 ECS 集群的外部实例。 • 选择 Create a new VPC (创建新的 VPC)。	云架构师

启动 Amazon WorkSpaces

Task	描述	所需技能
设置 Simple AD 并启动亚马逊 WorkSpaces。	要为您新创建的 VPC 预置 Simple AD 目录并启动 Amazon WorkSpaces，请按照 AWS 文档中的说明进行操作。	云架构师

为混合环境设置 AWS Systems Manager

Task	描述	所需技能
下载随附脚本。	在本地计算机上，下载附件部分中的 `ssm-trust-policy.json` 和 `ssm-activation.json` 文件。	云架构师
添加 IAM 角色。	根据业务需求添加环境变量。	云架构师

Task	描述	所需技能

	`export AWS_DEFAU LT_REGION=${AWS_RE GION_ID}`
	`export ROLE_NAME= ${ECS_TASK_ROLE}`
	`export CLUSTER_NAME= ${ECS_CLUSTER_NAME}`
	`export SERVICE_N AME=${ECS_CLUSTER_ SERVICE_NAME}`
	
	运行以下命令。
	
	`aws iam create-role -- role-name $ROLE_NAME --assume-role-policy- document file://ssm- trust-policy.json`	

将 Amazon SSMManaged InstanceCore 策略添加到 IAM 角色中。	运行以下命令。 `aws iam attach-role- policy --role-name $ROLE_NAME --policy- arn arn:aws:iam::aws:p olicy/AmazonSSMMan agedInstanceCore`	云架构师
将 Amazon EC2 Container Servicefor EC2 角色策略添加到 IAM 角色。	运行以下命令。 `aws iam attach-role- policy --role-name $ROLE_NAME --policy- arn arn:aws:iam::aws:p olicy/service-role /AmazonEC2Containe rServiceforEC2Role`	云架构师

Task	描述	所需技能
验证 IAM 角色。	要验证 IAM 角色，请运行以下命令。 `aws iam list-attached-role-policies --role-name $ROLE_NAME`	云架构师
激活 Systems Manager。	运行以下命令。 `aws ssm create-activation --iam-role $ROLE_NAME | tee ssm-activation.json`	云架构师

添加 WorkSpaces 到 ECS 集群

Task	描述	所需技能
Connect 到你的 WorkSpaces。	要连接和设置您的 WorkSpaces，请按照 AWS 文档中的说明进行操作。	应用程序开发人员
下载 ecs-anywhere 安装脚本。	在命令提示符下，运行以下 命令。 `curl -o "ecs-anywhere-install.sh" "https://amazon-ecs-agent-packages-preview.s3.us-east-1.amazonaws.com/ecs-anywhere-install.sh" && sudo chmod +x ecs-anywhere-install.sh`	应用程序开发人员
检查 Shell 脚本完整性。	（可选）运行以下命令。	应用程序开发人员

Task	描述	所需技能
	`curl -o "ecs-anywhere-install.sh.sha256" "https://amazon-ecs-agent-packages-preview.s3.us-east-1.amazonaws.com/ecs-anywhere-install.sh.sha256" && sha256sum -c ecs-anywhere-install.sh.sha256`	
在 Amazon Linux 上添加 EPEL 存储库。	若要添加企业 Linux (EPEL) 存储库，请运行命令 `sudo amazon-linux-extras install epel -y`。	应用程序开发人员
安装 Amazon ECS Anywhere。	要运行安装脚本，请使用以下命令。 `sudo ./ecs-anywhere-install.sh --cluster $CLUSTER_NAME --activation-id $ACTIVATION_ID --activation-code $ACTIVATION_CODE --region $AWS_REGION`	

Task	描述	所需技能
查看 ECS 集群的实例信息。	要检查 Systems Manager 和 ECS 集群实例信息并验证 WorkSpaces 已添加到集群上，请在本地计算机上运行以下命令。 `aws ssm describe-instance-informati on" && "aws ecs list-container-instances --cluster $CLUSTER_NAME`	应用程序开发人员

为添加 ECS 任务 WorkSpaces

Task	描述	所需技能
创建任务执行 IAM 角色。	从附件部分下载 `task-execution-assume-role.json` 和 `external-task-definition.json` 。 在本地计算机上运行以下命令。 `aws iam --region $AWS_DEFAULT_REGIO N create-role --role-name $ECS_TASK _EXECUTION_ROLE --assume-role-policy-document file://ta sk-execution-assume-role.json`	云架构师
将策略添加到该角执行色。	运行以下命令。	云架构师

Task	描述	所需技能
	`aws iam --region $AWS_DEFAULT_REGION attach-role-policy --role-name $ECS_TASK_EXECUTION_ROLE --policy-arn arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy`	
创建任务角色。	运行以下命令。 `aws iam --region $AWS_DEFAULT_REGION create-role --role-name $ECS_TASK_EXECUTION_ROLE --assume-role-policy-document file://task-execution-assume-role.json`	云架构师
将任务定义注册到集群。	在本地计算机上运行以下命令。 `aws ecs register-task-definition --cli-input-json file://external-task-definition.json`	云架构师

Task	描述	所需技能
运行任务。	在本地计算机上运行以下命令。 `aws ecs run-task --cluster $CLUSTER_NAME --launch-type EXTERNAL --task-definition nginx`	云架构师

验证任务运行状态。	若要获取任务 ID，请运行以下命令。
	`export TEST_TASKID=$(aws ecs list-tasks --cluster $CLUSTER_NAME | jq -r '.taskArns[0]')`
	使用任务 ID 运行以下命令。
	`aws ecs describe-tasks --cluster $CLUSTER_NAME --tasks ${TEST_TASKID}`
		云架构师

在上验证任务 WorkSpace。	要检查 NGINX 是否在上运行 WorkSpace，请运行命令。 `curl http://localhost:8080`	应用程序开发人员

相关资源

- ECS 集群
- 设置混合环境
- Amazon WorkSpaces
- Simple AD

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

在亚马 EC2 逊 Linux 实例上运行 ASP.NET Core Web API Docker 容器

由 Vijai Anand Ramalingam (AWS) 和 Sreelaxmi Pai (AWS) 编写

摘要

这种模式适用于开始在 Amazon Web Services (AWS) Cloud 上对其应用程序进行容器化的人员。当您开始在云上容器化应用程序时，通常没有设置容器编排平台。这种模式可以帮助您在 AWS 上快速设置基础设施来测试您的容器化应用程序，而无需复杂的容器编排基础设施。

现代化进程第一步是改造应用程序。如果是旧版.NET 框架应用程序，则必须先将其运行时系统更改为 ASP.NET Core。然后执行以下操作：

- 创建 Docker 容器镜像
- 从该镜像运行 Docker 容器。
- 在将应用程序部署到任何容器编排平台（例如 Amazon Elastic Container Service (Amazon ECS) 或 Amazon Elastic Kubernetes Service (Amazon EKS)）之前验证应用程序。

此模式涵盖了在亚马逊弹性计算云 (Amazon EC2) Linux 实例上构建、运行和验证现代应用程序开发的各个方面。

先决条件和限制

先决条件

- 一个活动 Amazon Web Services (AWS) 账户。
- 要在这种模式下创建 AWS 资源，需要具有足够的 IAM 访问权限的 AWS Identity and Access Management (IAM) 角色。
- Visual Studio Community 2022 或更新版本已下载并安装
- 已升级为 ASP.NET Core 框架项目
- 存储 GitHub 库

产品版本

- Visual Studio Community 2022 及以上版本

架构

目标架构

此模式使用 A WS CloudFormation 模板创建高可用架构，如下图所示。Amazon EC2 Linux 实例在私有子网中启动。AWS Systems Manager 会话管理器用于访问私有 Amazon EC2 Linux 实例和测试 Docker 容器中运行的 API。

1. 通过 Session Manager 访问 Linux 实例

工具

Amazon Web Services

- AWS 命令行界面 - AWS 命令行界面（AWS CLI）是一种开源工具，它使您能够使用命令行 Shell 中的命令与 Amazon Web Services 交互。仅需最少的配置，即可使用 AWS CLI 开始运行命令，以便从终端程序中的命令提示符实现与基于浏览器的 AWS 管理控制台 所提供的功能等同的功能：
- AWS 管理控制台 - AWS 管理控制台是一款 Web 应用程序，其中包含并引用了多种用于管理 AWS 资源的服务控制台。首次登录时，您会看到控制台主页。主页提供了对每个服务控制台的访问权限，并提供了访问执行 AWS 相关任务所需信息的单一位置。
- AWS Systems Manager 会话管理器 – 会话管理器是一项完全托管的 AWS Systems Manager 功能。使用会话管理器，您可以管理您的亚马逊弹性计算云 (Amazon EC2) 实例。Session Manager 提供安全且可审计的节点管理，无需打开入站端口、维护堡垒主机或者管理 SSH 密钥。

其他工具

- Visual Studio 2022 – Visual Studio 2022 是一种集成式开发环境（IDE）。
- Docker - Docker 是一组平台即服务（PaaS）产品，它们使用操作系统级别的虚拟化技术在容器中交付软件。

代码

```
FROM mcr.microsoft.com/dotnet/aspnet:5.0 AS base
 WORKDIR /app
EXPOSE 80
EXPOSE 443
```

```
FROM mcr.microsoft.com/dotnet/sdk:5.0 AS build
WORKDIR /src
COPY ["DemoNetCoreWebAPI/DemoNetCoreWebAPI.csproj", "DemoNetCoreWebAPI/"]
RUN dotnet restore "DemoNetCoreWebAPI/DemoNetCoreWebAPI.csproj"
COPY . .
WORKDIR "/src/DemoNetCoreWebAPI"
RUN dotnet build "DemoNetCoreWebAPI.csproj" -c Release -o /app/build

FROM build AS publish
RUN dotnet publish "DemoNetCoreWebAPI.csproj" -c Release -o /app/publish

FROM base AS final
WORKDIR /app
COPY --from=publish /app/publish .
ENTRYPOINT ["dotnet", "DemoNetCoreWebAPI.dll"]
```

操作说明

开发ASP.NET Core web API

Task	描述	所需技能
使用 Visual Studio 创建 ASP.NET Core web API	要创建示例 ASP.NET Core web API，请执行以下操作： 1. 打开 Visual Studio 2022 2. 选择 Create a new project。 3. 选择 ASP.NET Core Web API 项目模板，然后选择下一步。 4. 对于项目名称，输入 DemoNetCoreWebAPI，然后选择下一步。 5. 选择创建。 6. 要在本地运行项目，请按 F5。	应用程序开发人员

Task	描述	所需技能

	7. 验证默认 WeatherForecastAPI 端点是否使用 Swagger 返回结果。
	8. 打开命令提示符，导航到.csproj 项目文件夹，然后运行以下命令将新的 Web API 推送到您的存储库。GitHub
	`git add --all`
	`git commit -m “Initial Version”`
	`git push`	

Task	描述	所需技能

创建 Dockerfile。	要创建新文件，请执行以下操作之一：
	• 使用代码部分中的示例 Dockerfile 手动创建 Dockerfile。根据要求，选择相应的 .NET 基础镜像。有关.NET 和 ASP.NET Core 相关镜像的信息，请参见 Docker 中心。
	• 使用 Visual Studio 和 Docker 桌面创建 Docker Desktop。在解决方案资源管理器中，右键单击项目，选择添加-> Docker Support。对于目标操作系统，选择 Linux。确保新的 Dockerfile 与解决方案文件 (.sln) 路径相同。
	
	要将更改推送到您的 GitHub 存储库，请运行以下命令。
	`git add --all`
	`git commit -m “Dockerfile added”`
	`git push`
		应用程序开发人员

设置亚马逊 EC2 Linux 实例

Task	描述	所需技能
设置基础设施。	启动 A WS CloudFormation 模板以创建基础设施，其中包括以下内容： • 虚拟私有云（VPC），使用 AWS VPC 快速入门，具有跨越两个可用区的两个公有子网和两个私有子网。 • 启用 AWS Systems Manager 所需 IAM 角色。 • 在其中一个私有子网中，一个带有最新 SSM 代理 Amazon Linux 2 演示实例。尽管此实例没有任何来自 Internet 的直接连接，但可以使用 AWS Systems Manager 会话管理器安全地访问它，而无需堡垒主机。 **Note** 亚马逊 Linux 2 的支持已接近终止。欲了解更多信息，请参阅亚马逊 Linux 2 FAQs。 要详细了解如何使用会话管理器访问私有 Amazon EC2 实例，无需堡垒主机，请参阅迈向无堡垒世界博客文章。	应用程序开发人员、AWS 管理员、AWS DevOps

Task	描述	所需技能
登录亚马逊 EC2 Linux 实例。	要连接到私有子网中的 Amazon EC2 Linux 实例，请执行以下操作： 1. 打开 Amazon EC2 控制台。 2. 在导航窗格中，选择实例。 3. 选择 Amazon Linux 2 演示实例，然后选择 Connect。 **Note** 亚马逊 Linux 2 的支持已接近终止。欲了解更多信息，请参阅亚马逊 Linux 2 FAQs。 4. 选择 Session Manager(会话管理器)。 5. 选择连接以打开终端窗口。 6. 运行以下命令。 `sudo su`	应用程序开发人员

Task	描述	所需技能

安装并运行 Docker	要在亚马逊 EC2 Linux 实例中安装并启动 Docker，请执行以下操作：
	1. 要安装，请运行以下命令：
	`yum install -y docker`
	2. 要重新启动 Docker 服务，运行以下命令：
	`service docker start`
	3. 要验证 Docker 的安装，请运行以下命令。
	`docker info`
		应用程序开发人员、AWS 管理员、AWS DevOps

Task	描述	所需技能

安装 Git 并克隆存储库。	要在 Amazon EC2 Linux 实例上安装 Git 并从中克隆存储库 GitHub，请执行以下操作。
	1. 要安装 Git，请运行以下命令。
	`yum install git -y`
	2. 要克隆存储库，请运行以下命令。
	`git clone https://github.com/<username>/<repo-name>.git`
	3. 要导航到 Dockerfile，请运行以下命令。
	`cd <repo-name>/DemoNetCoreWebAPI/`
		应用程序开发人员、AWS 管理员、AWS DevOps

Task	描述	所需技能

运行 Docker 容器。	要构建 Docker 镜像并在亚马逊 EC2 Linux 实例中运行容器，请执行以下操作：
	1. 要创建 Docker 映像，请运行以下命令。
	`docker build -t aspnetcorewebapiimage -f Dockerfile .`
	2. 要获取 Docker 映像的列表，请运行以下命令。
	`docker images`
	3. 运行以下命令来创建和运行容器。
	`docker run -d -p 80:80 --name aspnetcorewebapicontainer aspnetcorewebapiimage`
		应用程序开发人员、AWS 管理员、AWS DevOps

测试 web API

Task	描述	所需技能
使用 curl 命令测试 Web API。	使用以下命令可运行 RSA 测试。 `curl -X GET "http://localhost/WeatherForecast" -H  "accept: text/plain"`	应用程序开发人员

Task	描述	所需技能
	验证 API 响应。 **Note** 当你在本地运行 Swagger 时，你可以从 Swagger 获取每个端点的 curl 命令。	

清理资源

Task	描述	所需技能
删除资源	删除堆栈，以移除所有资源。这确保您不会为未使用的任何服务付费。	AWS 管理员，AWS DevOps

相关资源

- 使用 PuTTY 从 Windows 连接到 Linux 实例
- 使用 ASP.NET Core 创建 Web API
- 走向没有堡垒的世界

使用 AWS Fargate 大规模运行消息驱动型工作负载

创建者：Stan Zubarev（AWS）

摘要

此模式展示了如何使用容器和 AWS Fargate 在 Amazon Web Services Cloud 中大规模运行消息驱动型工作负载。

当应用程序处理的数据量超过基于函数的无服务器计算服务的限制时，使用容器来处理数据可能会有所帮助。例如，如果应用程序需要的计算容量或处理时间超过 AWS Lambda 提供的容量或处理时间，则使用 Fargate 可以提高性能。

以下示例设置使用中的 AWS 云开发套件 (AWS CDK) 在 TypeScript AWS 云中配置和部署以下资源：

- Fargate 服务
- Amazon Simple Queue Service（Amazon SQS）队列
- Amazon DynamoDB 表。
- 亚马逊 CloudWatch 控制面板

Fargate 服务接收和处理来自 Amazon SQS 队列的消息，然后将其存储在 Amazon DynamoDB 表中。您可以使用控制面板监控 Fargate 处理了多少 Amazon SQS 消息以及 Fargate 创建了多少 DynamoDB 项目。 CloudWatch

> **Note**
>
> 您还可以使用此模式的示例代码在事件驱动的无服务器架构中构建更复杂的数据处理工作负载。有关更多信息，请参阅使用 AWS Fargate 大规模运行事件驱动型和计划性工作负载。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- AWS 命令行界面（AWS CLI）的最新版本，已在本地计算机上安装并配置
- Git，已在本地计算机上安装并配置
- AWS CDK，已在本地计算机上安装并配置

- Go，已在本地计算机上安装并配置
- Docker，已在本地计算机上安装并配置

架构

目标技术堆栈

- Amazon SQS
- AWS Fargate
- Amazon DynamoDB

目标架构

下图显示了使用 Fargate 在 Amazon Web Services Cloud 中大规模运行消息驱动型工作负载的示例工作流程：

图表显示了以下工作流：

1. Fargate 服务使用 Amazon SQS 长轮询来接收来自某个 Amazon SQS 队列的消息。
2. 然后，Fargate 服务会处理 Amazon SQS 消息并将其存储在 DynamoDB 表中。

自动化和扩缩

要自动扩缩 Fargate 任务数量，您可以配置 Amazon Elastic Container Service（Amazon ECS）服务自动扩缩。最佳做法是根据应用程序的 Amazon SQS 队列中可见消息的数量来配置扩展策略。

有关更多信息，请参阅 Amazon A EC2 uto Scaling 用户指南中的基于亚马逊 SQS 进行扩展。

工具

Amazon Web Services

- AWS Fargate 可帮助您运行容器，而无需管理服务器或亚马逊弹性计算云 (Amazon EC2) 实例。它与 Amazon Elastic Container Service（Amazon ECS）配合使用。
- Amazon Simple Queue Service (Amazon SQS) 提供了一个安全、持久且可用的托管队列，它可帮助您集成和分离分布式软件系统与组件。

- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和在 AWS 上运行的应用程序的指标。

代码

此模式的代码可在 GitHub sqs-fargate-ddb-cdk-go 存储库中找到。

操作说明

使用 AWS CDK 创建和部署资源

Task	描述	所需技能
克隆 GitHub 存储库。	通过运行以下命令将 GitHub sqs-fargate-ddb-cdk-go 存储库克隆到本地计算机： `git clone https://github.com/aws-samples/sqs-fargate-ddb-cdk-go.git`	应用程序开发人员
验证 AWS CLI 是否配置为正确的 Amazon Web Services account，以及 AWS CDK 是否具有所需权限。	要检查您的 AWS CLI 配置设置是否正确，您可以运行以下 Amazon Simple Storage Service（Amazon S3）的 ls 命令： `aws s3 ls` 此过程还要求 AWS CDK 拥有在 Amazon Web Services account 中预调配基础设施的权限。要授予所需权限，您必须在 AWS CLI 中创建命名的 AWS 配置文件并将其导出为AWS_PROFILE环境变量。	应用程序开发人员

Task	描述	所需技能
	Note 如果您之前未在 AWS 账户中使用过 AWS CDK，则必须先配置所需的 AWS CDK 资源。有关更多信息，请参阅 AWS CDK v2 开发人员指南中的引导。	

将 AWS CDK 堆栈部署到您的 Amazon Web Services account。	1. 通过运行以下 AWS CLI 命令来构建容器镜像：
	`docker build -t go-fargate .`
	2. 通过运行以下命令打开 AWS CDK 目录：
	`cd cdk`
	3. 通过运行以下命令，安装所需 npm 模块：
	`npm i`
	4. 通过运行以下命令将 AWS CDK 模式部署到 Amazon Web Services account ：
	`cdk deploy --profile ${AWS_PROFILE}`
		应用程序开发人员

测试设置

Task	描述	所需技能
向 Amazon SQS 队列发送测试消息。	有关说明，请参阅 Amazon SQS 开发人员指南中的将消息发送到队列（控制台）。 测试 Amazon SQS 消息示例 `{ "message": "hello, Fargate" }`	应用程序开发人员
验证测试消息是否显示在 Fargate 服务的日志中。 CloudWatch	按照《Amazon ECS 开发人员指南》中查看 CloudWatch 日志中的说明进行操作。请务必查看 go-service-clusterECS 集群中go-fargate-service日志组的日志。	应用程序开发人员
验证测试消息是否显示在 DynamoDB 表中。	1. 打开 DynamoDB 控制台。 2. 在左侧导航窗格中，选择表。然后，从列表中选择下表：sqs-fargate-ddb-table。 3. 选择 Explore table items（浏览表项目）。 4. 确认测试消息显示在已返回的项目列表中。	应用程序开发人员
验证 Fargate 服务是否正在向日志发送消息。 CloudWatch	1. 打开 CloudWatch 管理控制台。 2. 在左侧导航窗格中，选择控制面板。	应用程序开发人员

Task	描述	所需技能
	3. 在 “自定义仪表板” 列表中，选择名为的仪表板go-service-dashboard。 4. 验证测试消息是否显示在日志中。 **Note** AWS CDK 会在您的 AWS 账户中自动创建 CloudWatch 控制面板。	

清理

Task	描述	所需技能

删除 AWS CDK 堆栈。	1. 通过运行以下命令打开 AWS CLI 中的 AWS CDK 目录：
	
	`cd cdk`
	
	2. 通过运行以下命令删除 AWS CDK 堆栈：
	
	`cdk destroy --profile ${AWS_PROFILE}`
		应用程序开发人员

验证 AWS CDK 堆栈是否已删除。	要确保堆栈已删除，请运行以下命令： `aws cloudformation list-stacks --query \`	应用程序开发人员

Task	描述	所需技能
	`"StackSummaries[?contains(StackName,'SqsFargate')].StackStatus" \ --profile ${AWS_PROFILE}` 如果堆栈已删除，则命令输出中返回的 `StackStatus` 值为 `DELETE_COMPLETE` 。 有关更多信息，请参阅 AWS CloudFormation 用户指南 PowerShell中的 AWS CLI 和 CloudFormation 堆栈操作命令示例。	

相关资源

- 配置 AWS CLI（版本 2 的 AWS CLI 用户指南）
- API 参考（AWS CDK API 参考）
- 适用于 Go 的 AWS SDK 版本 2（Go 文档）

使用带 AWS Fargate 的 Amazon EFS on Amazon EKS，运行带持久数据存储的有状态工作负载

由里卡多·莫赖斯 (AWS)、罗德里戈·贝尔萨 (AWS) 和卢西奥·佩雷拉 (AWS) 创作

摘要

此模式为使用 AWS Fargate 配置计算资源来启用亚马逊弹性文件系统 (Amazon EFS) 作为在亚马逊弹性 Kubernetes Service (Amazon EKS) 上运行的容器的存储设备提供了指导。

此模式中描述的设置遵循安全最佳实践，默认提供静态安全性和传输安全性。若要加密您的 Amazon EFS 文件系统，请使用 AWS Key Management Service (AWS KMS) 密钥，但您也可以指定密钥别名来调度 KMS 密钥创建过程。

您可以按照此模式中的步骤为 proof-of-concept (PoC) 应用程序创建命名空间和 Fargate 配置文件，安装用于将 Kubernetes 集群与 Amazon EFS 集成的亚马逊 EFS 容器存储接口 (CSI) 驱动程序，配置存储类并部署 PoC 应用程序。这些步骤会生成 Amazon EFS 文件系统，该文件系统在多个 Kubernetes 工作负载之间共享，在 Fargate 上运行。此模式附带自动执行这些步骤的脚本。

如果您想在容器化应用程序中保持数据持久性，并希望避免在扩展操作期间丢失数据，则可以使用这种模式。例如：

- DevOps 工具 — 常见的场景是开发持续集成和持续交付（CI/CD) strategy. In this case, you can use Amazon EFS as a shared file system to store configurations among different instances of the CI/CD tool or to store a cache (for example, an Apache Maven repository) for pipeline stages among different instances of the CI/CD工具。
- Web 服务器 — 常见的场景是使用 Apache 作为 HTTP 网络服务器。您可以将 Amazon EFS 用作共享文件系统，以存储在 Web 服务器的不同实例之间共享的静态文件。在此示例场景中，直接修改文件系统，而不是将静态文件融入 Docker 映像中。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 具有 Kubernetes 版本 1.17 或更高版本的现有 Amazon EKS 集群（已测试至 1.27 版本）
- 现有 Amazon EFS 文件系统，用于绑定 Kubernetes StorageClass 并动态配置文件系统

- 集群管理权限
- 配置为指向所需 Amazon EKS 集群的上下文

限制

- 配合使用 Amazon EKS 与 Fargate 时需要考虑某些限制。例如，不支持使用某些 Kubernetes 结构，例如 DaemonSets 和特权容器。有关 Fargate 限制的更多信息，请参阅亚马逊 EKS 文档中的 AWS Fargate 注意事项。
- 此模式代码支持运行 Linux 或 macOS 的工作站。

产品版本

- AWS 命令行界面（AWS CLI）版本 2 或更高版本
- Amazon EFS CSI 驱动程序版本 1.0 或更高版本（已测试至 2.4.8 版本）
- eksctl 版本 0.24.0 或更高版本（已测试至 0.158.0 版本）
- jq 版本 1.6 或更高版本
- kubectl 版本 1.17 或更高版本（已在 1.27 版本之前测试）
- Kubernetes 版本 1.17 或更高版本（已在 1.27 版本之前测试）

架构

目标架构由以下基础架构组成：

- 虚拟私有云（VPC）
- 两个可用区
- 带有提供互联网访问的 NAT 网关的公有子网
- 一个带有 Amazon EKS 集群和 Amazon EFS 挂载目标（也称为挂载点）的私有子网
- VPC 级别的 Amazon EFS

以下是 Amazon EKS 集群的环境基础架构：

- 适用于命名空间级别的 Kubernetes 结构的 AWS Fargate 配置文件

- 一个 Kubernetes 命名空间，其中包含以下内容：
  - 两个应用程序 pod 分布在可用区中
  - 在集群级别绑定到永久卷 (PV) 的一个永久卷声明 (PVC)
- 绑定到命名空间中的 PVC 并指向集群外部私有子网中的 Amazon EFS 挂载目标的集群范围的 PV

工具

Amazon Web Services

- AWS 命令行接口 (AWS CLI) L ine AWS CLI 是一种开源工具，可用于通过命令行与 AWS 服务进行交互。
- Amazon Elastic File System (Amazon EFS) 可帮助您在 Amazon Web Services Cloud 中创建和配置共享文件系统。在此模式中，提供了可用于 Amazon EKS 的简单、可扩展、完全托管和共享的文件系统。
- Amazon Elastic Kubernetes Service（ Amazon EKS）可帮助您在 AWS 上运行 Kubernetes，而无需安装或操作自己的集群。
- AWS Fargate 是一款适用于亚马逊 EKS 的无服务器计算引擎。其为 Kubernetes 应用程序创建和管理计算资源。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥，以帮助保护您的数据。

其他工具

- Docker 是一组平台即服务（PaaS）产品，它们使用操作系统级别的虚拟化技术在容器中交付软件。
- eksctl 是一种用于在 Amazon EKS 上创建和管理 Kubernetes 集群的命令行实用程序。
- kubectl：针对 Kubernetes 集群运行命令的命令行界面。
- jq 是一个用于解析 JSON 的命令行工具。

代码

此模式的代码在使用 AWS Fargate 存储库的 Amazon EFS GitHub 持久性配置中提供。这些脚本按照 epic 的顺序整`epic01`理`epic06`，按照 Epics 部分中此模式的顺序排列。

最佳实践

目标架构包括以下服务和组件，它遵循了 AWS Well-Architected Framework 最佳实践：

- Amazon EFS 提供了一个简单、可扩展、完全托管的弹性 NFS 文件系统。这用作在容器组（pod）（分布在所选 Amazon EKS 集群的私有子网中）中运行的 PoC 应用程序的所有复制的共享文件系统。
- 每个私有子网的 Amazon EFS 挂载目标。其为集群的虚拟私有云（VPC）中的每个可用区域提供冗余。
- 运行 Kubernetes 工作负载的 Amazon EKS。您必须预置 Amazon EKS 集群，然后才能使用此模式，如先决条件部分所述。
- 为存储在 Amazon EFS 文件系统中的内容提供静态加密的 AWS KMS。
- Fargate 管理容器的计算资源，因此您可以专注于业务需求而不是基础设施负担。Fargate 配置文件是为所有私有子网创建的。其为集群的虚拟私有云（VPC）中的每个可用区域提供冗余。
- Kubernetes Pods，用于验证内容是否可以由应用程序的不同实例共享、使用和编写。

操作说明

配置 Amazon EKS 集群（可选）

Task	描述	所需技能
创建 Amazon EKS 集群。	**Note** 如果您已经部署了集群，请跳至下一个长篇故事。在您的现有 AWS 账户中创建一个 Amazon EKS 集群。在GitHub 目录中，使用其中一种模式使用 Terraform 或 eksctl 部署 Amazon EKS 集群。有关更多信息，请参阅 Amazon EKS 文档中的创建 Amazon EKS 集群。在 Terraform 模式中，还有一些示例展示了如何：将 Fargate 配	AWS 管理员、Terraform 或 eksctl 管理员、Kubernetes 管理员

Task	描述	所需技能
	置文件链接到你的 Amazon EKS 集群、创建 Amazon EFS 文件系统以及在你的 Amazon EKS 集群中部署 Amazon EFS CSI 驱动程序。	

Task	描述	所需技能

导出环境变量。	运行 env.sh 脚本。这提供了后续步骤所需的信息。
	
	`source ./scripts/env.sh`
	`Inform the AWS Account ID:`
	`<13-digit-account-id>`
	`Inform your AWS Region:`
	`<aws-Region-code>`
	`Inform your Amazon EKS Cluster Name:`
	`<amazon-eks-cluster-name>`
	`Inform the Amazon EFS Creation Token:`
	`<self-genereated-uuid>`
	
	如果尚未注明，则可以使用以下 CLI 命令获取上面要求的所有信息。
	
	`# ACCOUNT ID`
	`aws sts get-caller-identity --query "Account" --output text`
	
	`# REGION CODE`
	`aws configure get region`
	
	`# CLUSTER EKS NAME`
	`aws eks list-clusters --query "clusters" --output text`
	
	`# GENERATE EFS TOKEN`
		AWS 系统管理员

Task	描述	所需技能
	`uuidgen`	

创建 Kubernetes 命名空间和关联 Fargate 配置文件

Task	描述	所需技能

为应用程序工作负载创建 Kubernetes 命名空间和 Fargate 配置文件。	创建命名空间以接收与 Amazon EFS 交互的应用程序工作负载。运行 `create-k8s-ns-and-linked-fargate-profile.sh` 脚本。您可以选择使用自定义命名空间名称或默认提供的命名空间`poc-efs-eks-fargate` 。
	
	使用自定义应用程序命名空间名称：
	
	`export $APP_NAMESPACE=<CUSTOM_NAME>`
	`./scripts/epic01/create-k8s-ns-and-linked-fargate-profile.sh \`
	`-c "$CLUSTER_NAME" -n "$APP_NAMESPACE"`
	
	如果没有自定义的应用程序命名空间名称：
	
	`./scripts/epic01/create-k8s-ns-and-linked-fargate-profile.sh \`
	`-c "$CLUSTER_NAME"`
		拥有授予权限的 Kubernetes 用户

Task	描述	所需技能
	其中，$CLUSTER_NAME 是 Amazon EKS 集群的名称。该-n <NAMESPACE> 参数是可选的；如果未被通知，则将提供默认生成的命名空间名称。	

创建 Amazon EFS 文件系统

Task	描述	所需技能
生成一个唯一的代币。	Amazon EFS 需要一个创建令牌，以确保幂等操作（使用相同的创建令牌调用该操作没有效果）。要满足此要求，您必须通过可用技术生成唯一的令牌。例如，您可以生成通用唯一标识符 (UUID) 以用作创建令牌。	AWS 系统管理员
创建 Amazon EFS 文件系统。	创建用于接收应用程序工作负载读取和写入数据文件的文件系统。您可以创建加密或非加密文件系统。（根据最佳实践标准，此模式的代码将创建加密系统，以默认启用静态加密。）您可以使用唯一的对称 AWS KMS 密钥来加密您的文件系统。如果未指定自定义密钥，则使用 AWS 托管密钥。 为 Amazon EFS 生成唯一令牌后，使用 create-efs.sh 脚本创建加密或非加密 Amazon EFS 文件系统。	AWS 系统管理员

Task	描述	所需技能

	使用静态加密，无 KMS 密钥：
	
	`./scripts/epic02/create-efs.sh \ -c "$CLUSTER_NAME" \ -t "$EFS_CREATION_TOKEN"`
	
	其中，`$CLUSTER_NAME` 是 Amazon EKS 集群的名称，`$EFS_CREATION_TOKEN` 是文件系统的唯一创建令牌。
	
	使用静态加密，有 KMS 密钥：
	
	`./scripts/epic02/create-efs.sh \ -c "$CLUSTER_NAME" \ -t "$EFS_CREATION_TOKEN" \ -k "$KMS_KEY_ALIAS"`
	
	其中，`$CLUSTER_NAME` 是 Amazon EKS 集群的名称，`$EFS_CREATION_TOKEN` 是文件系统的唯一创建令牌，`$KMS_KEY_ALIAS` 是 KMS 密钥的别名。
	
	不使用加密：
	
	`./scripts/epic02/create-efs.sh -d \`	

Task	描述	所需技能
	`-c "$CLUSTER_NAME" \ -t "$EFS_CREATION_TOKEN"` 其中，`$CLUSTER_NAME` 是 Amazon EKS 集群的名称，`$EFS_CREATION_TOKEN` 是文件系统的唯一创建令牌，`-d` 禁用静态加密。	
创建安全组。	创建允许 Amazon EKS 集群访问 Amazon EFS 文件系统的安全组。	AWS 系统管理员
更新安全组的入站规则	更新安全组的入站规则，以允许对传入流量进行以下设置： • TCP 协议 — 端口 2049 • 来源 — 包含 Kubernetes 集群的 VPC 中私有子网的 CIDR 块范围	AWS 系统管理员
为每个私有子网添加挂载目标。	对于 Kubernetes 集群的每个私有子网，请为文件系统和安全组创建挂载目标。	AWS 系统管理员

将 Amazon EFS 组件安装至 Kubernetes 集群

Task	描述	所需技能
部署 Amazon EFS CSI 驱动程序。	将 Amazon EFS CSI 驱动程序部署至集群。驱动程序根据应用程序创建的永久卷声明来配置存储。运行该`create-k8s-efs-csi-sc.sh` 脚本将	拥有授予权限的 Kubernetes 用户

Task	描述	所需技能
	Amazon EFS CSI 驱动程序和存储类部署到集群中。 `./scripts/epic03/create-k8s-efs-csi-sc.sh` 此脚本使用该`kubectl`实用程序，因此请确保已配置上下文并指向所需的 Amazon EKS 集群。	
部署存储类。	将 Amazon EFS 配置器 (efs.csi.aws.com) 的存储类部署至集群中。	拥有授予权限的 Kubernetes 用户

将 PoC 应用程序安装至 Kubernetes 集群

Task	描述	所需技能
部署持久卷。	部署持久卷，并将其链接至创建的存储类和 Amazon EFS 文件系统的 ID。应用程序使用持久卷读取和写入内容。您可以在存储字段中为持久卷指定任何大小。Kubernetes 需要此字段，但由于 Amazon EFS 是弹性文件系统，因此其不会强制实施任何文件系统容量。您可以部署加密或不加密的持久卷。（根据最佳实践标准，Amazon EFS CSI 驱动程序默认启用加密。）运行`deploy-poc-app.sh` 脚本以部署永久	拥有授予权限的 Kubernetes 用户

Task	描述	所需技能

	卷、永久卷声明和两个工作负载。
	
	使用传输中加密：
	
	`./scripts/epic04/deploy-poc-app.sh \ -t "$EFS_CREATION_TOKEN"`
	
	其中，`$EFS_CREATION_TOKEN` 是文件系统的唯一创建令牌。
	
	不使用传输中加密：
	
	`./scripts/epic04/deploy-poc-app.sh -d \ -t "$EFS_CREATION_TOKEN"`
	
	其中，`$EFS_CREATION_TOKEN` 是文件系统的唯一创建令牌，`-d` 禁用传输中加密。	

部署应用程序请求的持久卷声明。	部署应用程序请求的持久卷声明，并将其链接到存储类别。使用与之前创建的持久卷相同的访问模式。您可以在存储字段中为持久卷声明指定任何大小。Kubernetes 需要此字段，但由于 Amazon EFS 是弹性文件系统，因此其不会强制实施任何文件系统容量。	拥有授予权限的 Kubernetes 用户

Task	描述	所需技能
部署工作负载 1。	部署可代表应用程序工作负载 1 的容器组（pod）。此工作负载将内容写入文件/data/out1.txt 。	拥有授予权限的 Kubernetes 用户
部署工作负载 2。	部署可代表应用程序工作负载 2 的容器组（pod）。此工作负载将内容写入文件/data/out2.txt 。	拥有授予权限的 Kubernetes 用户

验证文件系统的持久性、耐用性和可共享性

Task	描述	所需技能
检查的状态PersistentVolume 。	输入以下命令以检查的状态PersistentVolume 。 `kubectl get pv` 有关输出示例，请参阅 “其他信息” 部分。	拥有授予权限的 Kubernetes 用户
检查的状态PersistentVolumeClaim 。	输入以下命令以检查的状态PersistentVolumeClaim 。 `kubectl -n poc-efs-eks-fargate get pvc` 有关输出示例，请参阅 “其他信息” 部分。	拥有授予权限的 Kubernetes 用户
验证工作负载 1 是否可写入文件系统。	输入以下命令以验证工作负载 1 是否正在写入/data/out1.txt 。	拥有授予权限的 Kubernetes 用户

Task	描述	所需技能

	`kubectl exec -ti poc-app1 -n poc-efs-eks-fargate -- tail -f /data/out1.txt`
	
	结果类似于以下内容：
	
	`...`
	`Thu Sep  3 15:25:07 UTC 2023 - PoC APP 1`
	`Thu Sep  3 15:25:12 UTC 2023 - PoC APP 1`
	`Thu Sep  3 15:25:17 UTC 2023 - PoC APP 1`
	`...`	

验证工作负载 2 是否可写入文件系统。	输入以下命令以验证工作负载 2 是否正在写入内容`/data/out2.txt` 。
	
	`kubectl -n $APP_NAMESPACE exec -ti poc-app2 -- tail -f /data/out2.txt`
	
	结果类似于以下内容：
	
	`...`
	`Thu Sep  3 15:26:48 UTC 2023 - PoC APP 2`
	`Thu Sep  3 15:26:53 UTC 2023 - PoC APP 2`
	`Thu Sep  3 15:26:58 UTC 2023 - PoC APP 2`
	`...`
		拥有授予权限的 Kubernetes 用户

Task	描述	所需技能

验证工作负载 1 是否可以读取由工作负载 2 写入的文件。	输入以下命令以验证工作负载 1 是否可以读取工作负载 2 写入的/data/out2.txt 文件。
	`kubectl exec -ti poc-app1 -n poc-efs-eks-fargate -- tail -n 3 /data/out2.txt`
	结果类似于以下内容：
	`...`
	`Thu Sep  3 15:26:48 UTC 2023 - PoC APP 2`
	`Thu Sep  3 15:26:53 UTC 2023 - PoC APP 2`
	`Thu Sep  3 15:26:58 UTC 2023 - PoC APP 2`
	`...`
		拥有授予权限的 Kubernetes 用户

Task	描述	所需技能

验证工作负载 2 是否可以读取由工作负载 1 写入的文件。	输入以下命令以验证工作负载 2 是否可以读取工作负载 1 写入的/data/out1.txt 文件。
	`kubectl -n $APP_NAMESPACE exec -ti poc-app2 -- tail -n 3 /data/out1.txt`
	结果类似于以下内容：
	`... Thu Sep 3 15:29:22 UTC 2023 - PoC APP 1 Thu Sep 3 15:29:27 UTC 2023 - PoC APP 1 Thu Sep 3 15:29:32 UTC 2023 - PoC APP 1 ...`
		拥有授予权限的 Kubernetes 用户

Task	描述	所需技能

验证移除应用程序组件后文件是否保留。	接下来，使用脚本删除应用程序组件（永久卷、永久卷声明和 pod），并验证文件/data/out1.txt 和/data/out2.txt 是否保留在文件系统中。使用以下命令运行 validate-efs-content.sh 脚本。
	`./scripts/epic05/validate-efs-content.sh \`
	`-t "$EFS_CREATION_TOKEN"`
	其中，$EFS_CREATION_TOKEN 是文件系统的唯一创建令牌。
	结果类似于以下内容：
	`pod/poc-app-validation created`
	`Waiting for pod get Running state...`
	`Waiting for pod get Running state...`
	`Waiting for pod get Running state...`
	`Results from execution of 'find /data' on validation process pod:`
	`/data`
	`/data/out2.txt`
	`/data/out1.txt`
		拥有授予权限的 Kubernetes 用户、系统管理员

监控操作

Task	描述	所需技能
监控应用程序日志。	作为第二天操作的一部分，请将应用程序日志发送到 Amazon CloudWatch 进行监控。	AWS 系统管理员，被授予权限的 Kubernetes 用户
使用 Container Insights 监控 Amazon EKS 和 Kubernetes 容器。	作为第二天操作的一部分，使用亚马逊容器洞察监控亚马逊 EKS 和 Kubernetes 系统。CloudWatch 此工具从不同级别和维度的容器化应用程序收集、聚合以及汇总指标。有关更多信息，请参阅相关资源部分。	AWS 系统管理员，被授予权限的 Kubernetes 用户
使用监控 Amazon EFS CloudWatch。	作为第二天操作的一部分，使用 Amazon 监控文件系统 CloudWatch，Amazon 会收集来自 Amazon EFS 的原始数据并将其处理为可读的近乎实时的指标。有关更多信息，请参阅相关资源部分。	AWS 系统管理员

清理资源

Task	描述	所需技能
清理所有已创建的模式资源。	完成本模式后，清理所有资源，以避免产生 AWS 费用。使用 PoC 应用程序后，运行`clean-up-resources.sh` 脚本以删除所有资源。完成以下选项之一。	拥有授予权限的 Kubernetes 用户、系统管理员

Task	描述	所需技能

	使用静态加密，有 KMS 密钥：
	
	`./scripts/epic06/clean-up-resources.sh \ -c "$CLUSTER_NAME" \ -t "$EFS_CREATION_TOKEN" \ -k "$KMS_KEY_ALIAS"`
	
	其中，`$CLUSTER_NAME` 是 Amazon EKS 集群的名称，`$EFS_CREATION_TOKEN` 是文件系统的创建令牌，`$KMS_KEY_ALIAS` 是 KMS 密钥的别名。
	
	不使用静态加密：
	
	`./scripts/epic06/clean-up-resources.sh \ -c "$CLUSTER_NAME" \ -t "$EFS_CREATION_TOKEN"`
	
	其中，`$CLUSTER_NAME` 是 Amazon EKS 集群的名称，`$EFS_CREATION_TOKEN` 是文件系统的创建令牌。	

相关资源

参考

- 适用于@@ 亚马逊 EKS 的 AWS Fargate 现在支持亚马逊 EFS（公告）
- 如何在 AWS Fargate 上使用 Amazon EKS 捕获应用程序日志（博客文章）
- 使用容器见解（Amazon CloudWatch 文档）
- 在亚马逊 EKS 和 Kubernetes 上设置容器见解（亚马逊文档） CloudWatch
- 亚马逊 EKS 和 Kubernetes 容器洞察指标（亚马逊文档） CloudWatch
- 使用@@ 亚马逊监控亚马逊 EFS CloudWatch（亚马逊 EFS 文档）

GitHub 教程和示例

- 静态预置
- 传输中加密
- 通过多个容器组（pod）访问文件系统
- 在 Amazon EFS 中使用 StatefulSets
- 安装子路径
- 使用 Amazon EFS 接入点
- 适用于 Terraform 的 Amazon EKS 蓝图

必要工具

- 正在安装 AWS CLI 版本 2
- 正在安装 eksctl
- 正在安装 kubectl
- 正在安装 jq

其他信息

以下是该`kubectl get pv`命令的输出示例。

```
NAME         CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM
            STORAGECLASS   REASON   AGE
poc-app-pv   1Mi        RWX            Retain           Bound    poc-efs-eks-fargate/
poc-app-pvc   efs-sc                  3m56s
```

以下是该`kubectl -n poc-efs-eks-fargate get pvc`命令的输出示例。

```
NAME          STATUS   VOLUME       CAPACITY   ACCESS MODES   STORAGECLASS   AGE
poc-app-pvc   Bound    poc-app-pv   1Mi        RWX            efs-sc         4m34s
```

使用 Amazon EKS Pod Identity 和 KEDA 在 Amazon EKS 中设置事件驱动的自动缩放

由 Dipen Desai (AWS)、Abhay Diwan (AWS)、Kamal Joshi (AWS) 和 Mahendra Revanasiddappa (AWS) 创作

摘要

诸如亚马逊 Elastic Kubernetes Service（Amazon EKS）之类的编排平台简化了基于容器的应用程序的生命周期管理。这有助于组织专注于构建、保护、操作和维护基于容器的应用程序。随着事件驱动的部署变得越来越普遍，组织越来越频繁地根据各种事件源扩展 Kubernetes 部署。此方法与 auto Scaling 相结合，可提供按需计算资源和针对应用程序逻辑量身定制的高效扩展，从而显著节省成本。

KEDA 是一款基于 Kubernetes 的事件驱动型自动扩缩器。KEDA 可帮助您根据需要处理的事件数量扩展 Kubernetes 中的任何容器。它是轻量级的，可以与任何 Kubernetes 集群集成。它还适用于标准的 Kubernetes 组件，例如水平容器自动缩放 (HPA)。KEDA 还提供 TriggerAuthentication，这是一项可帮助您委托身份验证的功能。它允许您描述与 ScaledObject 和部署容器分开的身份验证参数。

AWS 提供支持各种 Kubernetes 部署选项的 AWS Identity and Access Management (IAM) 角色，包括亚马逊 EKS、Amazon EKS Anywhere、 AWS 云端 Red Hat OpenShift 服务 (ROSA) 和亚马逊弹性计算云 (亚马逊) 上的自行管理的 Kubernetes 集群。 EC2这些角色使用 IAM 结构（例如 OpenID Connect (OIDC) 身份提供商和 IAM 信任策略）在不同的环境中运行，无需直接依赖 Amazon EKS 服务或。 APIs有关更多信息，请参阅 Amazon EKS 文档中的服务账户的 IAM 角色。

Amazon EKS Pod Identity 简化了 Kubernetes 服务账户在不需要 OIDC 提供商的情况下担任 IAM 角色的流程。它提供了管理应用程序凭证的功能。您可以将 IAM 角色与 Kubernetes 服务账户关联，然后将您的 Pod 配置为使用服务账户，而不是创建 AWS 证书并将其分配给容器或使用 Amazon EC2 实例的角色。这可以帮助您在多个集群中使用 IAM 角色，并允许在 IAM 角色之间重复使用权限策略，从而简化策略管理。

通过使用 Amazon EKS Pod Identity 实施 KEDA，企业可以实现高效的事件驱动型自动扩展和简化的凭证管理。应用程序可根据需求进行扩展，从而优化资源利用率并降低成本。

这种模式可以帮助你将 Amazon EKS Pod 身份与 KEDA 集成。它展示了如何使用`keda-operator`服务帐号和委托身份验证。`TriggerAuthentication`它还描述了如何在 KEDA 操作员的 IAM 角色和应用程序的 IAM 角色之间建立信任关系。这种信任关系允许 KEDA 监控事件队列中的消息并调整目标 Kubernetes 对象的缩放比例。

先决条件和限制

先决条件

- AWS Command Line Interface (AWS CLI) 版本 2.13.17 或更高版本，已安装
- Python 版本 3.11.5 或更高版本，已安装
- 适用于 Python (Boto3) 的 AWS SDK 版本 1.34.135 或更高版本，已安装
- Helm 版本 3.12.3 或更高版本，已安装
- kubectl 版本 1.25.1 或更高版本，已安装
- 已安装 Docker Engine 版本 26.1.1 或更高版本
- 已创建 Amazon EKS 集群版本 1.24 或更高版本
- 满足创建 Amazon EKS Pod 身份代理的先决条件

限制

- 您需要在`keda-operator`角色和角色之间建立信任关系。`keda-identity`此模式的 “长篇故事” 部分提供了说明。

架构

在此模式中，您可以创建以下 AWS 资源：

- Amazon Elastic Container Registry (Amazon ECR) 存储库 — 在这种模式中，这个存储库被命名为。`keda-pod-identity-registry`此私有存储库用于存储示例应用程序的 Docker 镜像。
- 亚马逊简单队列服务 (Amazon SQS) Simple Q ueue 队列 — 在此模式中，该队列被命名。`event-messages-queue`队列充当消息缓冲区，用于收集和存储传入的消息。KEDA 监控队列指标，例如消息计数或队列长度，并根据这些指标自动扩展应用程序。
- 应用程序的 IAM 角色-在此模式中，此角色被命名`keda-identity`。该`keda-operator`角色扮演此角色。此角色允许访问 Amazon SQS 队列。
- KEDA 操作员的 IAM 角色 — 在此模式中，此角色被命名`keda-operator`。KEDA 操作员使用此角色进行所需的 AWS API 调用。此角色有权代入该`keda-identity`角色。由于`keda-operator`和角色之间的信任关系，该`keda-identity`keda-operator角色具有 Amazon SQS 权限。

通过`TriggerAuthentication`和 `ScaledObject` Kubernetes 自定义资源，操作员使用该`keda-identity`角色连接亚马逊 SQS 队列。根据队列大小，KEDA 会自动扩展应用程序部署。它会为队列

中每 5 条未读消息添加 1 个 pod。在默认配置中，如果 Amazon SQS 队列中没有未读消息，则应用程序会缩减到 0 个 pod。KEDA 操作员按您指定的时间间隔监控队列。

下图显示了您如何使用 Amazon EKS Pod Identity 为该`keda-operator`角色提供对 Amazon SQS 队列的安全访问权限。

图表显示了以下工作流：

1. 你在 Amazon EKS 集群中安装 Amazon EKS Pod 身份代理。
2. 您可以在 Amazon EKS 集群的 KEDA 命名空间中部署 KEDA 操作员。
3. 您在目标中创建`keda-operator`和 `keda-identity` IAM 角色 AWS 账户。
4. 您可以在 IAM 角色之间建立信任关系。
5. 您在`security`命名空间中部署应用程序。
6. KEDA 操作员轮询亚马逊 SQS 队列中的消息。
7. KEDA 启动 HPA，它会根据队列大小自动扩展应用程序。

工具

AWS 服务

- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- 亚马逊 Elastic Kubernetes Service（亚马逊 EKS）可帮助你在上面运行 AWS Kubernetes，而无需安装或维护自己的 Kubernetes 控制平面或节点。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- Amazon Simple Queue Service (Amazon SQS) 提供了一个安全、持久且可用的托管队列，它可帮助您集成和分离分布式软件系统与组件。

其他工具

- KEDA 是一款基于 Kubernetes 的事件驱动型自动扩缩器。

代码存储库

此模式的代码可在使用 EK S Pod Id GitHub entity 和 KEDA 存储库的事件驱动的自动缩放中找到。

最佳实践

我们建议您遵循以下最佳实践：

- 亚马逊 EKS 最佳实践
- IAM 安全最佳实操
- 亚马逊 SQS 最佳实践

操作说明

创建 AWS 资源

Task	描述	所需技能
为 KEDA 操作员创建 IAM 角色。	1. 登录 AWS Management Console，然后打开 IAM 控制台。 2. 在导航窗格中，选择角色。 3. 选择创建角色。 4. 选择 Custom trust policy（自定义信任策略）角色类型。 5. 在“自定义信任策略”部分，为此角色输入以下自定义信任策略： `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow",`	AWS 管理员

Task	描述	所需技能

	<pre>"Principa
	l": {
	
	"Service": "pods.eks
	.amazonaws.com"
	},
	"Action":
	[
	
	"sts:AssumeRole",
	
	"sts:TagSession"
	]
	}
	]
	}</pre> 6. 在添加权限页面上，选择下一步。您不向该角色添加任何策略。
	7. 对于 Role name（角色名称），输入 `keda-operator` 。
	8. 请选择 Create role（创建角色）。	

Task	描述	所需技能
为示例应用程序创建 IAM 角色。	1. 在 IAM 控制台的导航窗格中，选择角色。 2. 选择创建角色。 3. 选择 Custom trust policy（自定义信任策略）角色类型。 4. 在自定义信任策略部分，输入此角色的以下自定义信任策略。<account number>用您的目标账号替换： `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "Service": "pods.eks.amazonaws.com", "AWS": "arn:aws:iam::<account number>:role/keda-operator" }, "Action": [ "sts:AssumeRole", "sts:TagSession" ] } ]`	AWS 管理员

Task	描述	所需技能
	} 5. 在添加权限页面上，向角色添加以下 AWS 托管策略： • AmazonSQSReadOnlyAccess • AWSLambdaSQSQueueExecutionRole 6. 选择下一步。 7. 对于 Role name（角色名称），输入 keda-identity 。 8. 请选择 Create role（创建角色）。	
创建 Amazon SQS 队列。	1. 打开 Amazon SQS 控制台。 2. 选择创建队列。 3. 对于 Type（类型），选择 Standard（标准）。 4. 在 “创建队列” 页面上，在 “名称” 栏中输入event-messages-queue 。 5. 选择创建队列。您无需更改此队列的任何默认设置。	常规 AWS

Task	描述	所需技能
创建 Amazon ECR 存储库。	1. 打开 Amazon ECR 控制台。 2. 选择创建存储库。 3. 在 “存储库名称” 中，输入`keda-pod-identity-registry` 。 4. 选择创建存储库。您无需更改此存储库的任何默认设置。	常规 AWS

设置 Amazon EKS 集群

Task	描述	所需技能
部署 Amazon EKS Pod 身份代理。	对于目标 Amazon EKS 集群，设置 Amazon EKS Pod 身份代理。按照亚马逊 EKS 文档中设置 Amazon EKS Pod 身份代理中的说明进行操作。	AWS DevOps

部署 KEDA。	1. 输入以下命令在目标 Amazon EKS 集群上部署 KEDA：
	`# Add Helm Repo for Keda`
	`helm repo add kedacore https://kedacore.github.io/charts`
	`# Update Helm repo`
	`helm repo update`
	`# Install Keda`
	`helm install keda kedacore/keda --`
		DevOps 工程师

Task	描述	所需技能

	`namespace keda --create-namespace`
	
	有关更多信息，请参阅 KEDA 文档中的使用 Helm 进行部署。
	2. 成功部署后，在输出中，验证是否为 KEDA 操作员创建了三个部署。以下是输出示例：
	
	`NAME READY UP-TO-DATE AVAILABLE AGE`
	`keda-admission-webhooks 1/1 1 1 89s`
	`keda-operator 1/1 1 1 89s`
	`keda-operator-metrics-apiserver 1/1 1 1 89s`	

Task	描述	所需技能
将 IAM 角色分配给 Kubernetes 服务账户。	按照亚马逊 EKS 文档中向 Kubernetes 服务账户分配 IAM 角色中的说明进行操作。使用以下值： • 对于 IAM 角色，请输入`keda-operator` 。 • 对于 Kubernetes 命名空间，请输入。`keda` • 对于 Kubernetes 服务账号，请输入。`keda-operator`	AWS DevOps
创建 命名空间。	输入以下命令在目标 Amazon EKS 集群中创建`security`命名空间： `kubectl create ns security`	DevOps 工程师

部署示例应用程序。

Task	描述	所需技能
克隆应用程序文件。	输入以下命令，使用 EKS Pod Identity 和 KEDA 存储库从中克隆事件驱动的自动缩放：GitHub `git clone https://github.com/aws-samples/event-driven-autoscaling-using-podidentity-and-keda.git`	DevOps 工程师

Task	描述	所需技能

构建 Docker 映像。	1. 输入以下命令以导航到克隆的存储库：
	`cd event-driven-autoscaling-using-podidentity-and-keda`
	2. 输入以下命令为示例应用程序构建 Docker 镜像：
	`docker build -t keda-pod-identity-registry .`
		DevOps 工程师

Task	描述	所需技能

将 Docker 映像推送到 Amazon ECR。	1. 在构建 Docker 镜像的终端中，输入以下命令登录 Amazon ECR。<AWS_ACCOUNT_ID> 用 AWS 环境中的值替换<AWS_REGION> 和：
	`aws ecr get-login-password \ --region <AWS_REGION> | docker login \ --username AWS \ --password-stdin <AWS_ACCOUNT_ID>.dkr.ecr.<AWS_REGION>.amazonaws.com`
	2. 输入以下命令为图像添加标签。<AWS_ACCOUNT_ID> 用 AWS 环境中的值替换<AWS_REGION> 和：
	`docker tag keda-pod-identity-registry:latest <AWS_ACCOUNT_ID>.dkr.ecr.<AWS_REGION>.amazonaws.com/keda-pod-identity-registry:latest`
	3. 输入以下命令将图像推送到 Amazon ECR。<AWS_ACCOUNT_ID> 用 AWS 环境中的值替换<AWS_REGION> 和：
		DevOps 工程师

Task	描述	所需技能
	`docker push <AWS_ACCOUNT_ID>.dkr.ecr.<AWS_REGION>.amazonaws.com/keda-pod-identity-registry:latest` **Note** 导航到 Amazon ECR 存储库页面，然后选择“查看推送命令”，即可找到推送命令。	
部署示例应用程序。	1. 在克隆的存储库中，打开 depl oy.yaml 文件。 2. `<AWS_REGION>` 用环境中的值替换`<AWS_ACCOUNT_ID>` 和。 3. 保存并关闭 deploy.yaml 文件。 4. 输入以下命令在目标 Amazon EKS 集群上部署示例应用程序： `kubectl apply -f deploy.yaml` 此命令在集群中创建部署和服务帐户。	DevOps 工程师

Task	描述	所需技能

将 IAM 角色分配给应用程序服务账户。	执行以下任一操作，将 `keda-identity` IAM 角色与示例应用程序的服务账号关联：
	• 按照亚马逊 EKS 文档中向 Kubernetes 服务账户分配 IAM 角色中的说明进行操作。使用以下值：
	  • 对于 IAM 角色，请输入`keda-identity` 。
	  • 对于 Kubernetes 命名空间，请输入。`security`
	  • 对于 Kubernetes 服务账号，请输入。`my-sqs-read-msgs`
	• 输入以下 AWS CLI 命令。`<cluster-name>` 替换为目标 Amazon EKS 集群的名称，并`<role-ARN>` 替换为角色的亚马逊资源名称 (ARN)：`keda-identity`
	`aws eks create-pod-identity-association \ --cluster-name <cluster-name> \ --role-arn <role-ARN> \ --namespace security \ --service-account my-sqs-read-msgs`
		DevOps 工程师

Task	描述	所需技能
部署ScaledObject 和TriggerAuthentication 。	1. 在克隆的存储库中，打开 keda .yaml 文件。 2. {{AWS_ACCOUNT_ID}} 替换为目标的 ID AWS 账户。 3. {{AWS_REGION}} 用你的目标替换 AWS 区域。 4. （可选）在第 21—24 行中，更新ScaledObject 扩展策略的参数。有关这些参数的更多信息，请参阅以下内容： • 轮询间隔 • 冷却期 • idleReplicaCount • minReplicaCount 5. 保存并关闭 keda.y aml 文件。 6. 输入以下命令来部署ScaledObject 和TriggerAuthentication 资源： `kubectl -n security apply -f keda.yaml`	DevOps 工程师

测试 auto 缩放

Task	描述	所需技能
向 Amazon SQS 队列发送消息。	1. 输入以下命令以导航到克隆的存储库：	DevOps 工程师

Task	描述	所需技能

	`cd event-driven-autoscaling-using-podidentity-and-keda`
	2. 输入以下命令将测试消息发送到 Amazon SQS 队列：
	`python sqs_send_msg.py`
	sqs_send_msg.py 脚本充当生成用于测试自动缩放的消息的应用程序。
	Note
	如果你运行的是 Python 3，请输入`python3 sqs_send_msg.py` 。	

Task	描述	所需技能

监控应用程序窗格。	1. 在不同的终端中，输入以下命令来监控 pod：
	`kubectl -n security get po`
	2. 在 Amazon SQS 队列中，每 5 条未读消息，KEDA 就会添加一个容器。在上一个命令的输出中，确认正在添加新的 pod。以下是输出示例：
	`kubectl -n security get po`
	`NAME READY STATUS RESTARTS AGE`
	`q-read-797f4c7589-2bj76 1/1 Running 0 2s`
	`q-read-797f4c7589-4zxph 1/1 Running 0 49s`
	`q-read-797f4c7589-cg9dt 1/1 Running 0 18s`
	`q-read-797f4c7589-slc69 1/1 Running 0 33s`
	3. 完成测试后，在原始终端中输入 CTRL + C (Windows) 或 CMD + C (macOS)。这
		DevOps 工程师

Task	描述	所需技能
	将停止 python sqs_send_msg.py 脚本。	

故障排除

事务	解决方案

KEDA 操作员无法扩展应用程序。	输入以下命令查看 `keda-operator` IAM 角色的日志：
	`kubectl logs -n keda -l app=keda-operator -c keda-operator`
	
	如果有`HTTP 403`响应码，则应用程序和 KEDA 缩放器没有足够的权限访问 Amazon SQS 队列。完成以下步骤：
	1. 查看该`keda-identity` 角色的 IAM 策略和声明，确认已授予该队列读取权限。
	2. 验证 IAM 角色之间的信任关系。以下是示例：
	`{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "Service": "pods.eks.amazonaws.com" }, "Action": [ "sts:AssumeRole", "sts:TagSession" ]`

事务	解决方案

	<pre> }
	    ]
	}</pre>
	如果出现Assume-Role 错误，则 A mazon EKS 节点 IAM 角色将无法代入为定义的 IAM 角色TriggerAuthentication 。完成以下步骤：
	1. 输入以下命令以删除 keda-operator pod 并创建新的 Pod：
	<pre>kubectl delete pod keda-operator-
	<alphenumeric-value> --namespace
	keda</pre>
	2. 输入以下命令以检查 Pod 所采用的身份：
	<pre>kubectl describe pod <keda-operator-
	pod-name> --namespace keda</pre>
	3. 当 Pod 成功重启后，请确认在 Pod 描述中添加了以下两个变量：
	• AWS_CONTAINER_CREDENTIALS_FULL_URI
	• AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE

相关资源

- 设置亚马逊 EKS Pod 身份代理（亚马逊 EKS 文档）
- 部署 KEDA（KEDA 文档）
- ScaledObject 规格（KEDA 文档）
- 使用进行身份验证 TriggerAuthentication（KEDA 文档）

使用 PGO 简化 PostgreSQL 在 Amazon EKS 上的部署

由 Shalaka Dengale (AWS) 创建

摘要

这种模式将来自 Crunchy Data (PGO) 的 Postgres 操作员与亚马逊 Elastic Kubernetes Service（Amazon EKS）集成在一起，以简化云原生环境中的 PostgreSQL 部署。PGO 为在 Kubernetes 中管理 PostgreSQL 数据库提供了自动化和可扩展性。当你将 PGO 与 Amazon EKS 结合使用时，它会形成一个强大的平台，用于高效部署、管理和扩展 PostgreSQL 数据库。

这种集成提供了以下主要好处：

- 自动部署：简化 PostgreSQL 集群的部署和管理。
- 自定义资源定义 (CRDs)：使用 Kubernetes 原语进行 PostgreSQL 管理。
- 高可用性：支持自动故障转移和同步复制。
- 自动备份和恢复：简化备份和恢复流程。
- 水平扩展：启用 PostgreSQL 集群的动态扩展。
- 版本升级：便于滚动升级，停机时间最短。
- 安全：强制执行加密、访问控制和身份验证机制。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- AWS 命令行界面（AWS CLI）版本 2，已在 Linux、macOS 或 Windows 上安装并配置。
- AWS CLI CLI Config， AWS 用于从命令行连接资源。
- eksctl，在 Linux、macOS 或 Windows 上安装和配置。
- `kubectl`，已安装并配置为访问 Amazon EKS 集群资源。有关更多信息，请参阅亚马逊 EKS 文档中的设置 kubectl 和 eksc tl。
- 您的计算机终端已配置为访问 Amazon EKS 集群。有关更多信息，请参阅 Amazon EKS 文档中的配置您的计算机以与集群通信。

产品版本

- Kubernetes 版本 1.21—1.24 或更高版本（请参阅 PGO 文档）。
- PostgreSQL 版本 10 或更高版本。此模式使用 PostgreSQL 版本 16。

限制

- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额页面，然后选择服务的链接。

架构

目标技术堆栈

- Amazon EKS
- Amazon Virtual Private Cloud（Amazon VPC）
- 亚马逊弹性计算云（亚马逊 EC2）

目标架构

此模式构建的架构包含具有三个节点的 Amazon EKS 集群。每个节点在后端的一组 EC2 实例上运行。这个 PostgreSQL 设置遵循主副本架构，这对于读取密集型用例特别有效。例架包括以下组件：

- 主数据库容器 (pg-primary) 托管主 PostgreSQL 实例，所有写入操作都将定向到该实例。
- 辅助副本容器（pg-replica）托管 PostgreSQL 实例，这些实例从主数据库复制数据并处理读取操作。
- PgBouncer是 PGO 中包含的 PostgreSQL 数据库的轻量级连接池器。它位于客户端和 PostgreSQL 服务器之间，充当数据库连接的中介。
- PGO 可以在此 K ubernetes 环境中自动部署和管理 PostgreSQL 集群。
- Patroni 是一款开源工具，用于管理和自动执行 PostgreSQL 的高可用性配置。它包含在 PGO 中。当你在 Kubernetes 中将 Patroni 与 PGO 配合使用时，它在确保 PostgreSQL 集群的弹性和容错能力方面起着至关重要的作用。有关更多信息，请参阅 Patroni 文档。

该工作流程包括以下步骤：

- 部署 PGO 操作员。你在在 Amazon EKS 上运行的 Kubernetes 集群上部署 PGO 操作员。这可以通过使用 Kubernetes 清单或 Helm 图表来完成。这种模式使用 Kubernetes 清单。
- 定义 PostgreSQL 实例。当操作符运行时，您可以创建自定义资源 (CRs) 来指定 PostgreSQL 实例的所需状态。这包括存储、复制和高可用性设置等配置。
- 操作员管理。您可以通过 Kubernetes API 对象与操作员进行交互，例如 CRs 创建、更新或删除 PostgreSQL 实例。
- 监控和维护。您可以监控在 Amazon EKS 上运行的 PostgreSQL 实例的运行状况和性能。操作员通常会提供指标和日志以进行监控。您可以根据需要执行例行维护任务，例如升级和修补。有关更多信息，请参阅 Amazon EKS 文档中的监控集群性能和查看日志。
- 扩展和备份：您可以使用操作员提供的功能来扩展 PostgreSQL 实例并管理备份。

这种模式不包括监控、维护和备份操作。

自动化和扩缩

- 您可以使用自动 AWS CloudFormation 创建基础架构。有关更多信息，请参阅 Amazon EKS 文档 AWS CloudFormation中的使用创建 Amazon EKS 资源。
- 您可以使用 GitVersion 或 Jenkins 内部版本号来自动部署数据库实例。

工具

AWS 服务

- 亚马逊 Elastic Kubernetes Service（亚马逊 EKS）可帮助你在上面运行 AWS Kubernetes，而无需安装或维护自己的 Kubernetes 控制平面或节点。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行 shell 中的命令进行交互。

其他工具

- eksctl 是一个简单的命令行工具，用于在 Amazon EKS 上创建集群。
- kubectl 是针对 Kubernetes 集群运行命令的命令行实用程序。
- PGO 可在 Kubernetes 中自动化和扩展 PostgreSQL 数据库的管理。

最佳实践

请遵循以下最佳实践，以确保平稳高效的部署：

- 保护您的 EKS 集群。为您的 EKS 集群实施安全最佳实践，例如为服务账户 AWS Identity and Access Management (IRSA)、网络策略和 VPC 安全组使用 (IRSA) 角色。限制对 EKS 集群 API 服务器的访问，并使用 TLS 加密节点与 API 服务器之间的通信。
- 确保在亚马逊 EKS 上运行的 PGO 和 Kubernetes 之间的版本兼容性。某些 PGO 功能可能需要特定的 Kubernetes 版本或引入兼容性限制。有关更多信息，请参阅 PGO 文档中的组件和兼容性。
- 为 PGO 部署@@ 规划资源分配，包括 CPU、内存和存储。考虑 PGO 及其管理的 PostgreSQL 实例的资源需求。监控资源使用情况并根据需要扩展资源。
- 专为高可用性而设计。设计您的 PGO 部署以实现高可用性，从而最大限度地减少停机时间并确保可靠性。跨多个可用区部署 PGO 的多个 PGO 副本以实现容错能力。
- 为 PGO 管理的 PostgreSQL 数据库@@ 实施备份和恢复程序。使用 PGO 提供的功能或与 Kubernetes 和 Amazon EKS 兼容的第三方备份解决方案。
- 为 PGO 部署@@ 设置监控和日志记录，以跟踪性能、运行状况和事件。使用 Prometheus 等工具监控指标，使用 Grafana 等工具进行可视化。配置日志以捕获 PGO 日志以进行故障排除和审计。
- 正确@@ 配置网络以允许 PGO、PostgreSQL 实例和 Kubernetes 集群中的其他服务之间进行通信。使用 Amazon VPC 联网功能和 Kubernetes 联网插件（例如 Calico 或 A mazon VPC CNI）执行网络策略和流量隔离。
- 考虑性能、耐久性和可扩展性等因素，为 PostgreSQL 数据库@@ 选择合适的存储选项。使用亚马逊 Elastic Block Store (Amazon EBS) 卷 AWS 或托管存储服务进行永久存储。有关更多信息，请参阅亚马逊 E KS 文档中的使用亚马逊 EBS 存储 Kubernetes 卷。
- 使用基础设施即代码 (IaC) 工具，例如 AWS CloudFormation 在 Amazon EKS 上自动部署和配置 PGO。将基础设施组件（包括 EKS 集群、网络和 PGO 资源）定义为一致性、可重复性和版本控制的代码。

操作说明

创建 IAM 角色

Task	描述	所需技能
创建一个 IAM 角色。	1. 使用以下命令创建 IAM 角色 AWS CLI：	AWS 管理员

Task	描述	所需技能

	<pre>aws iam create-role \
	    --role-name
	 {YourRoleName} \
	    --assume-role-
	policy-document '{
	        "Version":
	 "2012-10-17",
	        "Statement":
	 [
	            {
	
	 "Effect": "Allow",
	
	 "Principal": {
	
	 "Service": "eks.amaz
	onaws.com"
	                },
	
	 "Action": "sts:Assu
	meRole"
	            }
	        ]
	    }' && \
	aws iam attach-role-
	policy \
	    --role-name
	 {YourRoleName}\
	    --policy-arn
	 arn:aws:iam::aws:p
	olicy/AmazonEKSClu
	sterPolicy && \
	aws iam attach-role-
	policy \
	    --role-name
	 {YourRoleName}\
	    --policy-arn
	 arn:aws:iam::aws:p
	olicy/AmazonEKSSer
	vicePolicy && \</pre>	

Task	描述	所需技能

	`aws iam attach-role-policy \ --role-name {YourRoleName}\ --policy-arn arn:aws:iam::aws:policy/CloudWatchFullAccess`
	2. 在以下内容中查看该角色 AWS Management Console：
	a. 打开 IAM 管理控制台。
	b. 选择 “角色”，然后搜索您创建的角色名称。
	c. 验证是否已附加以下策略：
	`AmazonEKSClusterPolicy`
	`AmazonEKSServicePolicy`
	`CloudWatchFullAccess`	

创建 Amazon EKS 集群。

Task	描述	所需技能
创建 Amazon EKS 集群。	如果您已经部署了集群，请跳过此步骤。否则，请使用 `eksctl` Terraform 或在当前 AWS 账户 集群中部署 Amazon EKS 集群。	AWS 管理员、Terraform 或 eksctl 管理员、Kubernetes 管理员

Task	描述	所需技能

	AWS CloudFormation此模式eksctl用于集群部署。
	
	Note
	此模式使用亚马逊 EC2 作为 Amazon EKS 的节点组。如果要使用 AWS Fargate，请参阅 eksctl 文档中的managedNodeGroups 配置。
	
	1. 使用以下eksctl输入文件生成集群。
	
	sample-cluster.yaml：
	
	`apiVersion: eksctl.io/v1alpha5`
	`kind: ClusterConfig`
	`metadata:`
	`  name: postgresql`
	`  region: us-east-1`
	`  version: "1.29"`
	`accessConfig:`
	`  authenticationMode: API_AND_CONFIG_MAP`
	`availabilityZones:`
	`- us-east-1a`
	`- us-east-1b`
	`- us-east-1c`
	`nodeGroups:`
	`  - name: ng-1`	

Task	描述	所需技能

	<pre> instanceType:
	 m5.16xlarge
	    desiredCapacity:
	 2
	  - name: ng-2
	    instanceType:
	 m5.16xlarge
	    desiredCapacity:
	 2
	  - name: ng-3
	    instanceType:
	 m5.16xlarge
	    desiredCapacity:
	 2
	vpc:
	  cidr: 192.168.0
	.0/16
	  clusterEndpoints:
	    publicAccess:
	 true
	  nat:
	    gateway: HighlyAva
	ilable
	iamIdentity
	Mappings:
	  - arn: arn:aws:i
	am::<account-id>:r
	ole/<role-name> #
	 update the IAM role
	 ARN created in step 1
	    username: <user-
	name> # Enter the
	 user name per your
	 choice
	    noDuplicateARNs:
	 false</pre>
	2. 运行以下命令创建集群（提供文件的文件路径）：`sample-cluster.yaml`	

Task	描述	所需技能
	`eksctl create cluster -f sample-cluster.yaml`	
验证集群的状态。	运行以下命令以查看集群中节点的当前状态： `kubectl get nodes` 如果您遇到错误，请参阅 Amazon EKS 文档的疑难解答部分。	AWS 管理员、Terraform 或 eksctl 管理员、Kubernetes 管理员

创建 OIDC 身份提供商

Task	描述	所需技能
启用 IAM OIDC 提供商。	作为 Amazon EBS 容器存储接口 (CSI) 驱动程序的先决条件，您的集群必须有一个现有的 IAM OpenID Connect (OIDC) 提供商。 使用以下命令启用 IAM OIDC 提供商： `eksctl utils associate-iam-oidc-provider --region={region} --cluster={YourClusterNameHere} --approve`	AWS 管理员

Task	描述	所需技能
	有关此步骤的更多信息，请参阅 Amazon EKS 文档。	

为 Amazon EBS CSI 驱动程序创建 IAM 角色。	使用以下`eksctl`命令为 CSI 驱动程序创建 IAM 角色：
	`eksctl create iamserviceaccount \`
	`  --region {RegionName} \`
	`  --name ebs-csi-controller-sa \`
	`  --namespace kube-system \`
	`  --cluster {YourClusterNameHere} \`
	`  --attach-policy-arn arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy \`
	`  --approve \`
	`  --role-only \`
	`  --role-name AmazonEKS_EBS_CSI_DriverRole`
	如果您使用加密的 Amazon EBS 驱动器，则必须进一步配置策略。有关说明，请参阅 Amazon EBS SCI 驱动程序文档。
		AWS 管理员

Task	描述	所需技能
添加 Amazon EBS CSI 驱动程序。	使用以下`eksctl`命令添加 Amazon EBS CSI 驱动程序： `eksctl create addon \ --name aws-ebs-csi-driver \ --cluster <YourClusterName> service-account-role-arn arn:aws:iam::$(aws sts get-caller-identity \ --query Account \ --output text):role/AmazonEKS_EBS_CSI_DriverRole \ --force`	AWS 管理员

安装 PGO

Task	描述	所需技能
克隆 PGO 存储库。	克隆 PGO 的 GitHub 存储库： `git clone https://github.com/CrunchyData/postgres-operator-examples.git`	AWS DevOps
提供创建服务帐号的角色详细信息。	要授予 Amazon EKS 集群访问所需 AWS 资源的权限，请指定您之前在文件中创建的 OIDC 角色的亚马逊资源名称 (ARN)。`service_account.yaml` 此文件位于存储库的命名空间文件夹中。	AWS 管理员、Kubernetes 管理员

Task	描述	所需技能

	`cd postgres-operator-examples`
	
	`---`
	`metadata:`
	`  annotations:`
	`  eks.amazonaws.com/role-arn: arn:aws:iam::<accountId>:role/<role_name> # Update the OIDC role ARN created earlier`	

Task	描述	所需技能

创建命名空间和 PGO 先决条件。	1. 运行以下命令以创建 命名空间：
	`kubectl apply -k kustomize/install/namespace`
	这为 PGO 建立了一个专用的命名空间。如有必要，您可以修改`namespace.yml` 文件并为命名空间指定不同的名称。
	2. 运行以下命令将默认配置应用于集群：
	`kubectl apply --server-side -k kustomize/install/default`
	`kustomize/install/default` 为 Kubernetes 基于角色的访问控制 (RBAC)、自定义资源定义 (CRD) 和 Kubernetes 管理器文件提供了默认配置。
		Kunernetes 管理员

验证 Pod 的创建情况。	验证命名空间和默认配置是否已创建： `kubectl get pods -n postgres-operator`	AWS 管理员、Kubernetes 管理员

Task	描述	所需技能
验证 PVCs。	使用以下命令验证永久卷声明 (PVCs)： `kubectl describe pvc -n postgres-operator`	AWS 管理员、Kubernetes 管理员

创建和部署操作员

Task	描述	所需技能

创建运算符。	修改位于的文件内容，使其与以下内容/kustomize/postgres/postgres.yaml 相匹配：
	`spec:`
	`  instances:`
	`  - name: pg-1`
	`  replicas: 3`
	`  patroni:`
	`  dynamicConfiguration:`
	`  postgresql:`
	`  pg_hba:`
	`  - "host all all 0.0.0.0/0 trust"`
	`  # this line enabled logical replication with programmatic access`
	`  - "host all postgres 127.0.0.1/32 md5"`
	`  synchronous_mode: true`
	`  users:`
	`  - name: replicator`
	`  databases:`
		AWS 管理员、数据库管理员、Kubernetes 管理员

Task	描述	所需技能

	<pre>- testdb
	options: "REPLICATION"</pre>
	这些更新执行以下操作：
	• 调整 PostgreSQL 配置设置以方便访问 PostgreSQL 实例。
	• 包括复制用户、数据库用户和超级用户的配置，以启用流式复制、数据库访问和集群管理。	

部署操作员。	部署 PGO 操作员，以便在 Kubernetes 环境中简化 PostgreSQL 数据库的管理和操作： <pre>kubectl apply -k kustomize/postgres</pre>	AWS 管理员、数据库管理员、Kubernetes 管理员

Task	描述	所需技能

验证部署。	1. 验证操作员是否已部署：
	`kubectl get pods -n postgres-operator --selector=postgres-operator.crunchydata.com/instance-set \ -L postgres-operator.crunchydata.com/role`
	2. 验证是否已创建与操作员容器关联的服务资源：
	`kubectl get svc -n postgres-operator`
	从命令输出中，记下主副本 (`primary_pod_name`) 和只读副本 (`read_pod_name`)。您将在接下来的步骤中使用它们。
		AWS 管理员、数据库管理员、Kubernetes 管理员

验证流式传送复制

Task	描述	所需技能
向主副本写入数据。	使用以下命令连接到 PostgreSQL 主副本并将数据写入数据库： `kubectl exec -it <primary_pod_name> bash -n postgres-operator`	AWS 管理员、Kubernetes 管理员

Task	描述	所需技能

	`psql`
	
	`CREATE TABLE customers (firstname text, customer_id serial, date_created timestamp );`
	`\dt`	

确认只读副本具有相同的数据。	连接到 PostgreSQL 只读副本并检查流式复制是否正常运行：
	
	`kubectl exec -it {read_pod_name} bash -n postgres-operator`
	
	`psql`
	
	`\dt`
	
	只读副本应具有您在上一步中在主副本中创建的表。
		AWS 管理员、Kubernetes 管理员

故障排除

事务	解决方案
吊舱无法启动。	• 使用以下命令检查 pod 状态： `kubectl get pods -n your-namespace` • 检查日志中是否存在任何错误：

事务	解决方案

	`kubectl logs your-pod-name -n your-namespace`
	• 检查 pod 事件中是否存在与你的 pod 相关的任何异常事件：
	`kubectl describe pod your-pod-name -n your-namespace`

副本明显落后于主数据库。	• 检查复制延迟：
	`SELECT * FROM pg_stat_replication;`
	• 确保副本有足够的 CPU 和内存资源。检查资源限制：
	`kubectl describe pod your-replica-pod -n your-namespace`
	• 验证存储后端的性能是否处于最佳状态。磁盘 I/O 缓慢可能会导致复制延迟。

您无法了解 PostgreSQL 集群的性能和运行状况。	• 启用 Amazon CloudWatch Logs 并确保将日志发送到亚马逊 CloudWatch 进行分析。有关更多信息，请参阅 Amazon EKS 文档。 • 检查 **`pg_stat_activity`** ： `SELECT * FROM pg_stat_activity;`

事务	解决方案

复制不起作用。	• 通过查看以下内容中的复制设置来检查主配置postgresql.conf ：
	`wal_level = replica`
	`max_wal_senders = 10`
	`wal_keep_size = 64  # or wal_keep_segments in older versions`
	• 确认pg_hba.conf 其中包含复制权限：
	`host replication replica_user all md5`
	• 检查副本配置。确保在副本上正确设置recovery.conf 或等效设置（standby.signal 和primary_conninfo ）。

相关资源

- 亚马逊 Elastic Kubernetes Service（AWS 部署选项概述白皮书）
- AWS CloudFormation（AWS 部署选项概述白皮书）
- 开始使用亚马逊 EKS — eksctl（亚马逊 EKS 用户指南）
- 设置 kubectl 和 eksctl（亚马逊 EKS 用户指南）
- 为 OpenID Connect 联合创建角色（IAM 用户指南）
- 配置 AWS CLI（AWS CLI 用户指南）的设置
- 适用于 Kubernetes 的 Crunchy Postgres 文档
- Crunch & Learn：适用于 Kubernetes 5.0 的 Crunchy Postgres（视频）

使用应用程序负载均衡器在 Amazon ECS 中使用双向 TLS 简化应用程序身份验证

由 Olawale Olaleye (AWS) 和 Shamanth Devagari (AWS) 创作

摘要

这种模式可帮助您简化应用程序身份验证，并使用应用程序负载均衡器 (ALB) 在亚马逊弹性容器服务 (Amazon ECS) 中使用双向 TLS 来减轻安全负担。使用 ALB，您可以对来自的 X.509 客户端证书进行身份验证。 AWS Private Certificate Authority这种强大的组合有助于实现服务之间的安全通信，从而减少应用程序中对复杂身份验证机制的需求。此外，该模式使用亚马逊弹性容器注册表 (Amazon ECR) Container Registry 来存储容器映像。

此模式中的示例使用公共图库中的 Docker 镜像最初创建示例工作负载。随后，将构建新的 Docker 镜像以存储在 Amazon ECR 中。要获取来源，可以考虑基于 Git 的系统，例如 GitHub、或 Bitbucket GitLab，或者使用亚马逊简单存储服务 Amazon S3 (Amazon S3)。要构建 Docker 镜像，请考虑使用 AWS CodeBuild 作为后续镜像。

先决条件和限制

先决条件

- AWS 账户 具有部署 AWS CloudFormation 堆栈访问权限的活跃用户。确保您拥有 AWS Identity and Access Management (IAM) 用户或角色的部署权限 CloudFormation。
- AWS Command Line Interface (AWS CLI) 已安装。使用或通过在`~/.aws/credentials`文件中设置环境变量，在本地计算机 AWS CLI 或环境中@@ 配置您的 AWS 凭证。
- 已安装 OpenSSL。
- 安装了 Docker。
- 熟悉工具中 AWS 服务 描述的内容。
- 熟悉 Docker 和 NGINX。

限制

- Application Load Balancer 的双向 TLS 仅支持 X.509v3 客户端证书。不支持 X.509v1 客户端证书。
- 此模式的代码存储库中提供的 CloudFormation 模板不包括将 CodeBuild 项目配置为堆栈的一部分。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅按地区划分的AWS 服务。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

产品版本

- Docker 版本 27.3.1 或更高版本
- AWS CLI 版本 2.14.5 或更高版本

架构

下图显示了此模式的架构组件。

图表显示了以下工作流：

1. 创建 Git 仓库，并将应用程序代码提交到存储库。
2. 在中创建私有证书颁发机构 (CA) AWS Private CA。
3. 创建 CodeBuild 项目。由提交更改触发，创建 Docker 映像并将构建的映像发布到 Amazon ECR。CodeBuildproject
4. 从 CA 复制证书链和证书正文，然后将证书包上传到 Amazon S3。
5. 使用您上传到 Amazon S3 的 CA 捆绑包创建信任存储。将信任存储库与 Application Load Balancer (ALB) 上的双向 TLS 侦听器相关联。
6. 使用私有 CA 为容器工作负载颁发客户端证书。还可以使用创建私有 TLS 证书 AWS Private CA。
7. 将私有 TLS 证书导入 AWS Certificate Manager (ACM)，然后将其与 ALB 一起使用。
8. 中的容器工作负载在与中的容器工作负载通信时，`ServiceTwo`使用颁发的客户端证书向 ALB 进行身份验证。`Service0ne`
9. 中的容器工作负载在与中的容器工作负载通信时，`Service0ne`使用颁发的客户端证书向 ALB 进行身份验证。`ServiceTwo`

自动化和扩缩

这种模式可以通过使用 SDK 中的 CloudFormation AWS Cloud Development Kit (AWS CDK) 、或 API 操作来配置 AWS 资源来实现完全自动化。

您可以使用 AWS CodePipeline 来实现持续集成和持续部署 (CI/CD) 管道，用于自动执行容器映像构建过程并将新版本部署到 Amazon ECS 集群服务。 CodeBuild

工具

AWS 服务

- AWS Certificate Manager (ACM) 可帮助您创建、存储和续订保护您的网站和应用程序的公共和私有 SSL/TLS X.509 证书和密钥。 AWS
- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。
- AWS CodeBuild是一项完全托管的生成服务，可帮助您编译源代码、运行单元测试和生成可随时部署的工件。
- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- Amazon Elastic Container Service (Amazon ECS) 是一项高度可扩展、快速的容器管理服务，用于在集群上运行、停止和管理容器。您可以在由 AWS Fargate管理的无服务器基础架构上运行任务和服务。或者，为了更好地控制您的基础设施，您可以在您管理的亚马逊弹性计算云 (Amazon EC2) 实例集群上运行任务和服务。
- Amazon ECS Exec 允许您直接与容器交互，而无需先与主机容器操作系统交互、打开入站端口或管理 SSH 密钥。您可以使用 ECS Exec 在 Amazon EC2 实例上或上运行的容器中运行命令或获取外壳程序。 AWS Fargate
- 弹性负载均衡(ELB) 将传入的应用程序或网络流量分配到多个目标。例如，您可以在一个或多个可用区内的 Amazon EC2 实例、容器和 IP 地址之间分配流量。ELB 监控其注册目标的运行状况，并仅将流量路由到健康的目标。当您的传入流量随时间变化时，ELB 会扩展您的负载均衡器。它可以自动扩展到大多数工作负载。
- AWS Fargate无需管理服务器或 Amazon EC2 实例，即可帮助您运行容器。Fargate 与亚马逊 ECS 和亚马逊 Elastic Kubernetes Service（亚马逊 EKS）兼容。您可以使用 Fargate 启动类型或 Fargate 容量提供程序来运行 Amazon ECS 任务和服务。为此，请将您的应用程序打包到容器中，指定 CPU 和内存要求，定义网络和 IAM 策略，然后启动应用程序。每个 Fargate 任务都具有自己的隔离边界，不与其他任务共享底层内核、CPU 资源、内存资源或弹性网络接口。
- AWS Private Certificate Authority允许创建私有证书颁发机构 (CA) 层次结构，包括根和下级结构 CAs，而无需运营本地 CA 的投资和维护成本。

其他工具

- Docker 是一组平台即服务（PaaS）产品，它们使用操作系统级别的虚拟化技术在容器中交付软件。
- GitHubGitLab、和 Bitbucket 是一些常用的基于 Git 的源代码控制系统，用于跟踪源代码更改。
- NGINX Open Sou rce 是一个开源负载均衡器、内容缓存和 Web 服务器。此模式将其用作 Web 服务器。
- OpenSSL 是一个开源库，它提供由 TLS 和 CMS 的 OpenSSL 实现所使用的服务。

代码存储库

此模式的代码可在 Amazon-ECS 存储库中的 GitHub mtls with-Application-Load-Balancer-Amazon-ECS 存储库中找到。

最佳实践

- 使用 Amazon ECS Exec 运行命令或获取在 Fargate 上运行的容器的外壳。您还可以使用 ECS Exec 帮助收集用于调试的诊断信息。
- 使用安全组和网络访问控制列表 (ACLs) 来控制服务之间的入站和出站流量。Fargate 任务从您的虚拟私有云 (VPC) 中配置的子网接收一个 IP 地址。

操作说明

创建存储库

Task	描述	所需技能
下载源代码。	要下载此模式的源代码，请复刻或克隆 GitHub MTLS with-Amazon- ECS 存储库中的应用程序负载平衡器。	DevOps 工程师
创建一个 Git 存储库。	要创建包含 Dockerfile 和文件的 Git 存储库，请使用以下步骤：`buildspec.yaml` 1. 在虚拟环境中创建一个文件夹。用你的项目名称来命名。 2. 在本地计算机上打开终端，然后导航到该文件夹。 3. 要将 MTLS with-Application-Load-Balancer-Amazon-EC S 存储库克隆到您的项目目录，请输入以下命令：	DevOps 工程师

Task	描述	所需技能
	`git clone https://github.com/aws-samples/mTLS-with-Application-Load-Balancer-in-Amazon-ECS.git`	

创建 CA 并生成证书

Task	描述	所需技能
在中创建私有 CA AWS Private CA。	要创建私有证书颁发机构 (CA)，请在终端中运行以下命令。用您自己的值替换示例变量中的值。 `export AWS_DEFAULT_REGION="us-west-2"` `export SERVICES_DOMAIN="www.example.com"` `export ROOT_CA_ARN=`aws acm-pca create-certificate-authority \` `--certificate-authority-type ROOT \` `--certificate-authority-configuration \` `"KeyAlgorithm=RSA_2048,` `SigningAlgorithm=SHA256WITHRSA,` `Subject={` `Country=US,` `State=WA,`	DevOps 工程师，AWS DevOps

Task	描述	所需技能

	``
	        Locality=
	Seattle,
	        Organizat
	ion=Build on AWS,
	        Organizat
	ionalUnit=mTLS Amazon
	 ECS and ALB Example,
	        CommonName=
	${SERVICES_DOMAIN}}" \
	        --query
	 CertificateAuthori
	tyArn --output text`
	`` 有关更多详细信息，请参阅 AWS 文档AWS Private CA中的创建私有 CA。	

Task	描述	所需技能
创建并安装您的私有 CA 证书。	要为您的私有根 CA 创建和安装证书，请在终端中运行以下命令： 1. 生成证书签名请求（CSR）。 ``` ROOT_CA_CSR=`aws acm-pca get-certificate-authority-csr \     --certificate-authority-arn ${ROOT_CA_ARN} \     --query Csr --output text` ``` 2. 颁发根证书。 ``` AWS_CLI_VERSION=$(aws --version 2>&1 | cut -d/ -f2 | cut -d. -f1) [[${AWS_CLI_VERSION} -gt 1]] && ROOT_CA_CSR="$(echo ${ROOT_CA_CSR} | base64)" ROOT_CA_CERT_ARN=`aws acm-pca issue-certificate \ --certificate-authority-arn ${ROOT_CA_ARN} \ --template-arn arn:aws:acm-pca:::template/RootCACertificate/V1 \ --signing-algorithm SHA256WITHRSA \ ```	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	--validity Value=10,Type=YEARS \ --csr "${ROOT_CA_CSR}" \ --query CertificateArn --output text` 3. 检索根证书。 ROOT_CA_CERT=`aws acm-pca get-certificate \ --certificate-arn ${ROOT_CA_CERT_ARN} \ --certificate-authority-arn ${ROOT_CA_ARN} \ --query Certificate --output text` # store for later use aws acm-pca get-certificate \ --certificate-arn ${ROOT_CA_CERT_ARN} \ --certificate-authority-arn ${ROOT_CA_ARN} \ --query Certificate --output text > ca-cert.pem 4. 导入根 CA 证书以将其安装在 CA 上。 [[${AWS_CLI_VERSION} -gt 1]] && ROOT_CA_CERT="$(echo		

Task	描述	所需技能
	``` ${ROOT_CA_CERT} | base64)" aws acm-pca import-ce rtificate-authorit y-certificate \ --certificate- authority-arn $ROOT_CA_ARN \ --certificate "${ROOT_CA_CERT}" ``` 有关更多详细信息，请参阅 AWS 文档中的安装 CA 证书。	
申请托管证书。	要申请私有证书 AWS Certificate Manager 以用于您的私有 ALB，请使用以下命令： ``` export TLS_CERTI FICATE_ARN=`aws acm request-certificate \ --domain-name "*. ${DOMAIN_DOMAIN}" \ --certificate-auth ority-arn ${ROOT_CA _ARN} \ --query Certifica teArn --output text` ```	DevOps 工程师，AWS DevOps

Task	描述	所需技能

使用私有 CA 颁发客户证书。	• 要为这两项服务创建证书签名请求 (CSR)，请使用以下 AWS CLI 命令：
	
	`openssl req -out client_csr1.pem -new -newkey rsa:2048 -nodes -keyout client_private-key1.pem`
	
	`openssl req -out client_csr2.pem -new -newkey rsa:2048 -nodes -keyout client_private-key2.pem`
	
	此命令返回两个服务的 CSR 和私钥。
	
	• 要为服务颁发证书，请运行以下命令以使用您创建的私有 CA：
	
	`SERVICE_ONE_CERT_ARN=`aws acm-pca issue-certificate \ --certificate-authority-arn ${ROOT_CA_ARN} \ --csr fileb://client_csr1.pem \ --signing-algorithm "SHA256WITHRSA" \`
		DevOps 工程师，AWS DevOps

Task	描述	所需技能
	--validity Value=5,Type="YEARS" --query CertificateArn --output text`  echo "SERVICE_ONE_CERT_ ARN: ${SERVICE _ONE_CERT_ARN}"  aws acm-pca get-certi ficate \ --certificate-auth ority-arn ${ROOT_CA _ARN} \ --certificate-arn ${SERVICE_ONE_CERT _ARN} \ | jq -r '.Certifi cate' > client_ce rt1.cert  SERVICE_TWO_CERT_ ARN=`aws acm-pca issue- certificate \ --certificate-auth ority-arn ${ROOT_CA _ARN} \ --csr fileb://c lient_csr2.pem \ --signing-algorith m "SHA256WITHRSA" \ --validity Value=5,Type="YEARS" --query CertificateArn --output text` echo "SERVICE_TWO_CERT_ ARN: ${SERVICE _TWO_CERT_ARN}" aws acm-pca get-certi ficate \	

Task	描述	所需技能
	`--certificate-auth ority-arn ${ROOT_CA _ARN} \ --certificate-arn ${SERVICE_TWO_CERT _ARN} \ | jq -r '.Certifi cate' > client_ce rt2.cert` 有关更多信息，请参阅 AWS 文档中的颁发私有终端实体证书。	

预置 Amazon Web Service

Task	描述	所需技能
使用 CloudFormation 模板 AWS 服务 进行配置。	要配置虚拟私有云 (VPC)、亚马逊 ECS 集群、亚马逊 ECS 服务、Application Load Balancer 和亚马逊弹性容器注册表 (Amazon ECR) Container Registry，请使用该模板。CloudFormation	DevOps 工程师
获取变量。	确认您有一个运行两个服务的 Amazon ECS 集群。要检索资源详细信息并将其存储为变量，请使用以下命令： `export LoadBalancerDNS= $(aws cloudformation describe-stacks -- stack-name ecs-mtls \ --output text \`	DevOps 工程师

Task	描述	所需技能

	`--query 'Stacks[0 ].Outputs[?OutputK ey==`LoadBalancerD NS`].OutputValue')`
	
	`export ECRReposi toryUri=$(aws cloudformation describe-stacks -- stack-name ecs-mtls \ --output text \ --query 'Stacks[0 ].Outputs[?OutputK ey==`ECRRepository Uri`].OutputValue')`
	
	`export ECRReposi toryServiceOneUri= $(aws cloudformation describe-stacks -- stack-name ecs-mtls \ --output text \ --query 'Stacks[0 ].Outputs[?OutputK ey==`ECRRepository ServiceOneUri`].Ou tputValue')`
	
	`export ECRReposi toryServiceTwoUri= $(aws cloudformation describe-stacks -- stack-name ecs-mtls \ --output text \ --query 'Stacks[0 ].Outputs[?OutputK ey==`ECRRepository ServiceTwoUri`].Ou tputValue')`
	
	`export ClusterName= $(aws cloudformation`	

Task	描述	所需技能
	describe-stacks -- stack-name ecs-mtls \ --output text \ --query 'Stacks[0].Outputs[?OutputK ey==`ClusterName`] .OutputValue') export BucketName= $(aws cloudformation describe-stacks -- stack-name ecs-mtls \ --output text \ --query 'Stacks[0].Outputs[?OutputK ey==`BucketName`]. OutputValue') export Service1L istenerArn=$(aws cloudformation describe-stacks -- stack-name ecs-mtls \ --output text \ --query 'Stacks[0].Outputs[?OutputK ey==`Service1Liste nerArn`].OutputVal ue') export Service2L istenerArn=$(aws cloudformation describe-stacks -- stack-name ecs-mtls \ --output text \ --query 'Stacks[0].Outputs[?OutputK ey==`Service2Liste nerArn`].OutputVal ue')	

Task	描述	所需技能
创建 CodeBuild 项目。	要使用 CodeBuild 项目为您的 Amazon ECS 服务创建 Docker 镜像，请执行以下操作： 1. 登录 AWS Management Console，然后在 https://console.aws.amazon.com/codesuite/codebuild/上打开 CodeBuild 控制台。 2. 创建新 项目 对于 “源”，选择您创建的 Git 存储库。有关不同类型的 Git 存储库集成的信息，请参阅 AWS 文档中的使用连接。 3. 确认已启用特权模式。要构建 Docker 镜像，必须使用此模式。否则，将无法成功构建映像。 4. 使用为每项服务共享的自定义`buildspec.yaml` 文件。 5. 提供项目名称和描述的值。 有关更多详细信息，请参阅 AWS 文档AWS CodeBuild中的创建构建项目。	AWS DevOps， DevOps 工程师

Task	描述	所需技能

构建 Docker 镜像。	您可以使用 CodeBuild 来执行映像构建过程。 CodeBuild 需要权限才能与 Amazon ECR 交互和使用 Amazon S3。
	
	作为该过程的一部分，将构建 Docker 映像并将其推送至 Amazon ECR 注册表。有关模板和代码的详细信息，请参阅其他信息。
	
	（可选）要出于测试目的在本地构建，请使用以下命令：
	
	`# login to ECR`
	`aws ecr get-login`
	`-password | docker`
	`login --username`
	`AWS --password-stdin`
	`$ECRRepositoryUri`
	
	`# build image for`
	`service one`
	`cd /service1`
	`aws s3 cp s3://$Buc`
	`ketName/serviceone/`
	`service1/ --recursive`
	`docker build -t`
	`$ECRRepositoryServ`
	`iceOneUri .`
	`docker push $ECRRepos`
	`itoryServiceOneUri`
	
	`# build image for`
	`service two`
	`cd ../service2`
	`aws s3 cp s3://$Buc`
	`ketName/servicetwo/`
	`service2/ --recursive`
		DevOps 工程师

Task	描述	所需技能
	`docker build -t $ECRRepositoryServiceTwoUri .` `docker push $ECRRepositoryServiceTwoUri`	

启用双向 TLS

Task	描述	所需技能
将 CA 证书上传到亚马逊 S3。	要将 CA 证书上传到 Amazon S3 存储桶，请使用以下示例命令： `aws s3 cp ca-cert.pem s3://$BucketName/acm-trust-store/`	AWS DevOps， DevOps 工程师
创建信任存储库。	要创建信任存储库，请使用以下示例命令： `TrustStoreArn=`aws elbv2 create-trust-store --name acm-pca-trust-certs \` `--ca-certificates-bundle-s3-bucket $BucketName \` `--ca-certificates-bundle-s3-key acm-trust-store/ca-cert.pem --query 'TrustStores[].TrustStoreArn' --output text``	AWS DevOps， DevOps 工程师

Task	描述	所需技能

上传客户证书。	要将 Docker 镜像的客户端证书上传到 Amazon S3，请使用以下示例命令：
	
	`# for service one`
	`aws s3 cp client_cert1.cert s3://$BucketName/serviceone/`
	`aws s3 cp client_private-key1.pem s3://$BucketName/serviceone/`
	
	`# for service two`
	`aws s3 cp client_cert2.cert s3://$BucketName/servicetwo/`
	`aws s3 cp client_private-key2.pem s3://$BucketName/servicetwo/`
		AWS DevOps， DevOps 工程师

Task	描述	所需技能

修改监听器。	要在 ALB 上启用双向 TLS，请使用以下命令修改 HTTPS 侦听器：
	
	`aws elbv2 modify-listener \`
	`  --listener-arn $Service1ListenerArn \`
	`  --certificates CertificateArn=$TLS_CERTIFICATE_ARN_TWO \`
	`  --ssl-policy ELBSecurityPolicy-2016-08 \`
	`  --protocol HTTPS \`
	`  --port 8080 \`
	`  --mutual-authentication Mode=verify,TrustStoreArn=$TrustStoreArn,IgnoreClientCertificateExpiry=false`
	
	`aws elbv2 modify-listener \`
	`  --listener-arn $Service2ListenerArn \`
	`  --certificates CertificateArn=$TLS_CERTIFICATE_ARN_TWO \`
	`  --ssl-policy ELBSecurityPolicy-2016-08 \`
	`  --protocol HTTPS \`
	`  --port 8090 \`
	`  --mutual-authentication Mode=veri`
		AWS DevOps， DevOps 工程师

Task	描述	所需技能
	fy,TrustStoreArn=$ TrustStoreArn,Igno reClientCertificat eExpiry=false 有关更多信息，请参阅 AWS 文档中的在 App lication Load Balancer 上配置双向 TLS。	

更新服务

Task	描述	所需技能
更新 Amazon ECS 任务定义。	要更新 Amazon ECS 任务定义，请修改新修订版中的image参数。 要获取相应服务的值，请使用您在前面的步骤中构建的新 Docker 镜像 Uri 更新任务定义：或 echo $ECRRepos itoryServiceOneUri echo $ECRRepos itoryServiceTwoUri "containerDefiniti ons": [{ "name": "nginx", "image": "public.ecr.aws/ng inx/nginx:latest", # <----- change to new Uri "cpu": 0,	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	有关更多信息，请参阅 AWS 文档中的使用控制台更新 Amazon ECS 任务定义。	
更新 Amazon ECS 服务。	使用最新的任务定义更新服务。此任务定义是新建的 Docker 镜像的蓝图，它包含双向 TLS 身份验证所需的客户端证书。 要更新服务，请按以下步骤操作： 1. 在 https://console.aws.amazon.com/ecs/v2 中打开 Amazon ECS 控制台。 2. 在 Clusters（集群）页面上，选择集群。 3. 在集群详细信息页面的服务部分，选中服务旁边的复选框，然后选择更新。 4. 要让您的服务开始新的部署，请选择 Force new deployment（强制执行新部署）。 5. 在 “任务定义” 中，选择任务定义系列和最新修订版。 6. 选择更新。 对其他服务重复上述步骤。	AWS 管理员、AWS DevOps、 DevOps 工程师

访问应用程序

Task	描述	所需技能
复制应用程序 URL。	使用 Amazon ECS 控制台查看任务。当任务状态更新为正在运行时，选择该任务。在 “任务” 部分，复制任务 ID。	AWS 管理员，AWS DevOps

测试您的应用程序。	要测试您的应用程序，请使用 ECS Exec 访问任务。
	1. 对于服务一，请使用以下命令：
	`container="nginx"`
	`ECS_EXEC_TASK_ARN="<TASK ARN>"`
	`aws ecs execute-command --cluster $ClusterName \ --task $ECS_EXEC_TASK_ARN \ --container $container \ --interactive \ --command "/bin/bash"`
	2. 在服务一任务的容器中，使用以下命令输入内部负载均衡`url` 器和指向服务二的侦听器端口。然后指定客户端证书的路径以测试应用程序：
	`curl -kvs https://<internal-alb-url>:8090 --key /usr/local/share/ca-certifi`
		AWS 管理员，AWS DevOps

Task	描述	所需技能

	cates/client.key
	  --cert /usr/loca
	l/share/ca-certifi
	cates/client.crt
	
	3. 在服务二任务的容器中，使用以下命令输入内部负载均衡url器和指向服务一的侦听器端口。然后指定客户端证书的路径以测试应用程序：
	
	curl -kvs https://<
	internal-alb-url>:
	8080 --key /usr/loca
	l/share/ca-certifi
	cates/client.key
	  --cert /usr/loca
	l/share/ca-certifi
	cates/client.crt
	
	Note
	curl 命令中的-k标志（作为的一部分-kvs）禁用 SSL 证书验证。当使用与您的域名匹配的 SSL 证书时，您可以删除此标志，从而启用正确的证书验证。	

相关资源

Amazon ECS 文档

- 使用控制台创建 Amazon ECS 任务定义
- 创建容器镜像以在 Amazon ECS 上使用
- Amazon ECS 集群
- 适用于 Amazon ECS AWS Fargate
- Amazon ECS 联网最佳实践
- 亚马逊 ECS 服务定义参数

其他 AWS 资源

- 如何使用 AWS 私有 CA 在 Application Load Balancer 上配置 mTLS？ (AWS re:Post)

其他信息

编辑 Dockerfile

以下代码显示了您在服务 1 的 Dockerfile 中编辑的命令：

```
FROM public.ecr.aws/nginx/nginx:latest
WORKDIR /usr/share/nginx/html
RUN echo "Returning response from Service 1: Ok" > /usr/share/nginx/html/index.html
ADD client_cert1.cert client_private-key1.pem /usr/local/share/ca-certificates/
RUN chmod -R 400 /usr/local/share/ca-certificates/
```

以下代码显示了您在服务 2 的 Dockerfile 中编辑的命令：

```
FROM public.ecr.aws/nginx/nginx:latest
WORKDIR /usr/share/nginx/html
RUN echo "Returning response from Service 2: Ok" > /usr/share/nginx/html/index.html
ADD client_cert2.cert client_private-key2.pem /usr/local/share/ca-certificates/
RUN chmod -R 400 /usr/local/share/ca-certificates/
```

如果您使用构建 Docker 镜像 CodeBuild，则该buildspec文件会使用 CodeBuild 内部版本号将镜像版本唯一标识为标签值。您可以根据自己的要求更改buildspec文件，如以下buildspec 自定义代码所示：

```
version: 0.2

phases:
```

```
  pre_build:
    commands:
      - echo Logging in to Amazon ECR...
      - aws ecr get-login-password --region $AWS_DEFAULT_REGION | docker login --
username AWS --password-stdin $ECR_REPOSITORY_URI
      - COMMIT_HASH=$(echo $CODEBUILD_RESOLVED_SOURCE_VERSION | cut -c 1-7)
      - IMAGE_TAG=${COMMIT_HASH:=latest}
  build:
    commands:
        # change the S3 path depending on the service
      - aws s3 cp s3://$YOUR_S3_BUCKET_NAME/serviceone/ $CodeBuild_SRC_DIR/ --
recursive
      - echo Build started on `date`
      - echo Building the Docker image...
      - docker build -t $ECR_REPOSITORY_URI:latest .
      - docker tag $ECR_REPOSITORY_URI:latest $ECR_REPOSITORY_URI:$IMAGE_TAG
  post_build:
    commands:
      - echo Build completed on `date`
      - echo Pushing the Docker images...
      - docker push $ECR_REPOSITORY_URI:latest
      - docker push $ECR_REPOSITORY_URI:$IMAGE_TAG
      - echo Writing image definitions file...
      # for ECS deployment reference
      - printf '[{"name":"%s","imageUri":"%s"}]' $CONTAINER_NAME $ECR_REPOSITORY_URI:
$IMAGE_TAG > imagedefinitions.json

artifacts:
  files:
    - imagedefinitions.json
```

更多模式

- 使用 CAST Highlight 评测迁移至 Amazon Web Services Cloud 的应用程序就绪情况
- 自动删除 AWS CloudFormation 堆栈和相关资源
- 使用和自动进行动态管道管理，以便在 Gitflow 环境中部署修补程序解决方案 AWS Service CatalogAWS CodePipeline
- 自动使用 AWS CDK 为微服务构建 CI/CD 管道与 Amazon ECS 集群
- 使用 GitHub Actions 和 Terraform 构建 Docker 镜像并将其推送到 Amazon ECR
- 对经过 Blu Age 现代化改造的大型机工作负载进行容器化
- 使用 Firelens 日志路由器为 Amazon ECS 创建自定义日志解析器
- 在 Amazon ECS 上部署 Java 微服务 CI/CD 管道
- 使用实例配置文件从 AWS Cloud9 部署 Amazon EKS EC2 集群
- 使用 Terraform 为容器化 Blu Age 应用程序部署环境
- 使用 Amazon 中的推理管道将预处理逻辑部署到单个终端节点的 ML 模型中 SageMaker
- 使用 AWS 代码服务和 AWS KMS 多区域密钥，管理微服务到多个账户和区域的蓝/绿部署
- 通过使用 AWS CDK 设置 Amazon ECS Anywhere 来管理本地容器应用程序
- 从 Oracle 迁移 GlassFish 到 AWS Elastic Beanstalk
- 在 Amazon ECS 上从 Oracle 迁移 WebLogic 到 Apache Tomcat (ToMee)
- 设置最小可行数据空间，以便在组织之间共享数据
- 在 AWS 上实现 ASP.NET Web 表单应用程序的现代化
- 使用 AWS 和 AWS CloudFormation Config 监控亚马逊 ECR 存储库的通配符权限
- 使用 AWS CDK 在 Amazon ECS Anywhere 上为混合工作负载设置 CI/CD 管道和 GitLab
- 在 Amazon S3 中设置 Helm v3 图表存储库
- 使用证书管理器和 “让我们加 end-to-end密” 为 Amazon EKS 上的应用程序设置加密
- 使用 Flux 简化 Amazon EKS 多租户应用程序部署
- 使用 SageMaker AI 和 Hydra 简化从本地开发到可扩展实验的机器学习工作流程
- 使用 AWS Lambda 以六边形架构构建 Python 项目
- 使用 LocalStack 和 Terraform 测试来测试 AWS 基础架构
- 在 Amazon 上训练和部署支持 GPU 的自定义机器学习模型 SageMaker
- 使用 AWS Fargate WaitCondition 挂钩构造协调资源依赖关系和任务执行
- 使用 Amazon Bedrock 代理通过基于文本的提示在 Amazon EKS 中自动创建访问入口控制

无服务器

主题

- 使用 AWS Amplify 构建无服务器 React Native 移动应用程序
- 在单个控制面板上管理多个 SaaS 产品的租户
- 在组织中创建跨账户 Amazon EventBridge 连接
- 使用 Kinesis Data Streams 和 Firehose 将 DynamoDB 记录传送到亚马逊 S3 AWS CDK
- 在 Amazon API Gateway 中使用自定义域名实现基于路径的 API 版本控制
- 将 psycopg2 库导入到以与你的 PostgreS AWS Lambda QL 数据库进行交互
- 将 Amazon API Gateway 与亚马逊 SQS 集成以处理异步 REST APIs
- 使用 Amazon API Gateway 和 AWS Lambda 异步处理事件
- 使用 Amazon API Gateway 和 Amazon DynamoDB Streams 异步处理事件
- 使用 Amazon API Gateway、Amazon SQS 和 AWS Fargate 异步处理事件
- 从 AWS Step Functions 同步运行 AWS Systems Manager Automation 任务
- 在 AWS Lambda 函数中使用 Python 并行读取 S3 对象
- 将遥测数据从发送到 AWS Lambda 以 OpenSearch 进行实时分析和可视化
- 为基于蜂窝的架构设置无服务器蜂窝路由器
- 通过 VPC 终端节点设置对 Amazon S3 存储桶的私有访问权限
- 使用 Amazon Bedroc AWS Step Functions k 对中的状态进行故障排除
- 使用无服务器方法将 Amazon Web Services 串在一起
- 更多模式

使用 AWS Amplify 构建无服务器 React Native 移动应用程序

由 Deekshitulu Pentakota (AWS) 编写

摘要

此示例介绍了如何使用 AWS Amplify 和以下 Amazon Web Services 为 React Native 移动应用程序创建无服务器后端：

- AWS AppSync
- Amazon Cognito
- Amazon DynamoDB

使用 Amplify 配置和部署应用程序的后端后，Amazon Cognito 将对应用程序用户进行身份验证并授权他们访问应用程序。 AppSync 然后，AWS 与前端应用程序和后端 DynamoDB 表进行交互以创建和获取数据。

> **Note**
>
> 这种模式使用一个简单的 “ToDoList” 应用程序作为示例，但你可以使用类似的过程来创建任何 React Native 移动应用程序。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- Amplify 命令行界面 (Amplify CLI)，已安装并配置
- XCode （任何版本）
- Microsoft Visual Studio（任何版本、任何代码编辑器、任何文本编辑器）
- 熟悉 Amplify
- 熟悉 Amazon Cognito
- 熟悉 AWS AppSync
- 熟悉 DynamoDB
- 熟悉 Node.js

- 熟悉 npm
- 熟悉 React 和 React Native
- 熟悉 JavaScript 和 ECMAScript 6 () ES6
- 熟悉 GraphQL

架构

下图显示了在 Amazon Web Services Cloud 中运行 React Native 移动应用程序后端的示例架构：

该图显示以下架构：

1. Amazon Cognito 对应用程序用户进行身份验证，并授权他们访问应用程序。
2. 为了创建和获取数据，AWS AppSync 使用 GraphQL API 与前端应用程序和后端 DynamoDB 表进行交互。

工具

Amazon Web Services

- AWS Amplify 是一组专门构建的工具和功能，可帮助前端 Web 和移动开发人员快速地在 AWS 上构建全栈应用程序。
- AWS AppSync 提供了可扩展的 GraphQL 接口，可帮助应用程序开发人员合并来自多个来源的数据，包括亚马逊 DynamoDB、AWS Lambda 和 HTTP。 APIs
- Amazon Cognito 为您的 Web 和移动应用程序提供身份验证、授权和用户管理。
- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。

代码

此模式中使用的示例应用程序的代码可在 GitHub aws-amplify-react-native-ios-todo-app 存储库中找到。若要使用示例文件，请按照此模式的操作说明部分进行操作。

操作说明

创建并运行您的 React Native 应用程序

Task	描述	所需技能
设置 React Native 开发环境。	有关说明，请参阅 React Native 文档中的设置开发环境。	应用程序开发人员

在 iOS 模拟器中创建并运行 ToDoList React Native 移动应用程序。	1. 在新的终端窗口中运行以下命令，在本地环境中创建新的 React Native 移动应用程序项目目录：
	`npx react-native init ToDoListAmplify`
	2. 通过运行以下命令导航到项目根目录：
	`cd ToDoListAmplify`
	3. 运行以下命令来运行应用程序：
	`npx react-native run-ios`
		应用程序开发人员

为应用程序初始化新后端环境

Task	描述	所需技能
创建支持 Amplify 中的应用程序所需后端服务。	1. 在本地环境中，从项目的根目录 (ToDoListAmplify) 运行以下命令： `amplify init`	应用程序开发人员

Task	描述	所需技能

	2. 将显示一条提示，要求您提供有关该应用程序的信息。基于自己的用例来输入所需信息。然后按 Enter。
	
	对于此模式中使用的 ToDoList 应用程序设置，请应用以下示例配置。
	
	示例 React Native Amplify 应用程序配置
	
	`? Name: ToDoListAmplify`
	
	`? Environment: dev`
	
	`? Default editor: Visual Studio Code`
	
	`? App type: javascript`
	
	`? Javascript framework: react-native`
	
	`? Source Directory Path: src`
	
	`? Distribution Directory Path: /`
	
	`? Build Command: npm run-script build`
	
	`? Start Command: npm run-script start`
	
	`? Select the authentication method you want to use: AWS profile`	

Task	描述	所需技能
	``` ? Please choose the profile you want to use: default ``` 有关更多信息，请参阅 Amplify Dev Center 文档中的创建新的 Amplify 后端。 **Note** 该`amplify init`命令使用 AWS 预置以下资源 CloudFormation： • 适用于经过身份验证和未经身份验证的用户的 AWS Identity and Access Management（IAM）角色（Auth 角色 和 Unauth 角色） • 用于部署 Amazon Simple Storage Service (Amazon S3) 存储桶（用于此模式的示例应用程序 Amplify-meta.json）存储桶 • Amplify Hosting 中的后端环境	

将Amazon Cognito 身份验证添加到您的 Amplify React Native 应用程序中

Task	描述	所需技能

创建 Amazon Cognito 身份验证服务。	1. 在本地环境中，从项目的根目录 (ToDoListAmplify) 运行以下命令：
	
	`amplify add auth`
	2. 将出现一条提示，要求您提供有关身份验证服务的配置设置的信息。基于自己的用例来输入所需信息。然后按 Enter。
	
	对于此模式中使用的 ToDoList 应用程序设置，请应用以下示例配置。
	
	身份验证服务配置设置示例
	
	`? Do you want to use the default authentication and security configuration? \`
	`Default configuration`
	
	`? How do you want users to be able to sign in? \`
	`Username`
	
	`? Do you want to configure advanced settings? \`
	`No, I am done`
		应用程序开发人员

Task	描述	所需技能
	Note 该`amplify add auth`命令在项目根目录的本地文件夹（放大）中创建必要的文件夹、文件和依赖文件。对于此模式中使用的 ToDoList 应用程序设置，aws-exports.js 就是为此目的而创建的。	

将 Amazon Cognito 服务部署到 Amazon Web Services Cloud 端。	1. 在项目的根目录中，运行以下 Amplify CLI 命令：
	`amplify push`
	2. 将出现确认部署提示。输入是。然后按 Enter。
	Note
	要查看项目中已部署的服务，请运行以下命令前往 Amplify 控制台：
	`amplify console`
		应用程序开发人员

Task	描述	所需技能

为 React Native 安装所需的 Amplify 库和 iOS 的 CocoaPods 依赖项。	1. 通过从项目的根目录运行以下命令以安装所需 Amplify 开源客户端库：
	
	`npm install aws-amplify aws-amplify-react-native \ amazon-cognito-identity-js @react-native-community/netinfo \ @react-native-async-storage/async-storage`
	
	2. 通过运行以下命令安装 iOS 所需的 CocoaPods 依赖项：
	
	`npx pod-install`
		应用程序开发人员

Task	描述	所需技能

导入并配置 Amplify 服务。	在应用程序的入口点文件（例如 App.js）中，通过输入以下代码行来导入和加载 Amplify 服务的配置文件：
	`import Amplify from 'aws-amplify'`
	`import config from './src/aws-exports'`
	`Amplify.configure(config)`
	Note
	如果您在应用程序的入口点文件中导入 Amplify 服务后收到错误，请停止该应用程序。然后，打开 XCode 并选择项目的 iOS 文件夹中的 ToDoListAmplify.xcworkspace，然后运行该应用程序。
		应用程序开发人员

Task	描述	所需技能

更新应用程序的入口点文件，以使用 withAuthenticator 高阶组件 (HOC)。	**Note**
	`withAuthenticator` HOC 只需使用几行代码，即可在您的应用程序中提供登录、注册和忘记密码的工作流程。有关更多信息，请参阅 Amplify 开发中心的选项 1：使用预构建的用户界面组件。另外，React 文档中的高阶组件。
	
	1. 在应用程序的入口点文件（例如 App.js）中，通过输入以下代码行来导入 `withAuthenticator` HOC：
	`import { withAuthenticator } from 'aws-amplify-react-native'`
	2. 输入以下代码以导出 withAuthenticator HOC：
	`export default withAuthenticator(App)`
	
	withAuthenticator HOC 代码示例
		应用程序开发人员

Task	描述	所需技能

	``
	import Amplify from
	'aws-amplify'
	import config from './
	src/aws-exports'
	Amplify.configur
	e(config)
	import { withAuthe
	nticator } from
	'aws-amplify-react-
	native';
	
	const App = () => {
	return null;
	};
	
	export default withAuthe
	nticator(App);
	``
	
	Note
	在 iOS 模拟器中，该应用程序会显示 Amazon Cognito 服务提供的登录屏幕。	

Task	描述	所需技能
测试身份验证服务设置。	使用 iOS 模拟器，执行以下操作： 1. 使用真实的电子邮件地址在应用程序中创建新账户。然后，验证码将发送至注册的电子邮件中。 2. 使用您在验证电子邮件中收到的代码验证所设置的账户。 3. 输入您创建的用户名和密码。然后，选择登录，出现欢迎屏幕。 **Note** 您也可以打开 Amazon Cognito 控制台，检查身份池中是否创建了新用户。	应用程序开发人员

将 AWS AppSync API 和 DynamoDB 数据库连接到应用程序

Task	描述	所需技能
创建 AWS AppSync API 和 DynamoDB 数据库。	1. 向您的应用程序添加 AWS AppSync API，并通过从项目的根目录运行以下 Amplify CLI 命令来自动预配置 DynamoDB 数据库： `amplify add api`	应用程序开发人员

Task	描述	所需技能

	2. 将出现一条提示，要求您提供有关 API 和数据库配置设置的信息。基于自己的用例来输入所需信息。然后按 Enter。Amplify CLI 在文本编辑器中打开 GraphQL 架构文件。
	
	对于此模式中使用的 ToDoList 应用程序设置，请应用以下示例配置。
	
	API 与数据库配置设置示例
	
	`? Please select from one of the below mentioned services: \ GraphQL`
	
	`? Provide API name: todolistamplify`
	
	`? Choose the default authorization type for the API \ Amazon Cognito User Pool`
	
	`Do you want to use the default authentication and security configuration`
	
	`? Default configuration How do you want users to be able to sign in? \ Username`	

Task	描述	所需技能

	Do you want to configure advanced settings? \
	No, I am done.
	
	? Do you want to configure advanced settings for the GraphQL API \
	No, I am done.
	
	? Do you have an annotated GraphQL schema? \
	No
	
	? Choose a schema template: \
	Single object with fields (e.g., “Todo” with ID, name, description)
	
	? Do you want to edit the schema now? \
	Yes
	
	示例 GraphQL 架构
	
	type Todo @model {
	id: ID!
	name: String!
	description: String
	}	

Task	描述	所需技能

部署 AWS AppSync API。	1. 在项目根目录中，运行以下 Amplify CLI 命令：
	
	`amplify push`
	
	2. 将出现一条提示，要求您提供有关 API 和数据库配置设置的更多信息。基于自己的用例来输入所需信息。然后按 Enter。您的应用程序现在可以与 AWS AppSync API 进行交互了。
	
	对于此模式中使用的 ToDoList 应用程序设置，请应用以下示例配置。
	
	AWS AppSync API 配置设置示例
	
	Note
	以下配置在 AWS AppSync 中创建 GraphQL API，在 Dynamo DB 中创建 Todo 表。
	
	`? Are you sure you want to continue? Yes`
	`? Do you want to generate code for your newly created GraphQL API Yes`
		应用程序开发人员

Task	描述	所需技能
	``` ? Choose the code generation language target javascript ? Enter the file name pattern of graphql queries, mutations and subscriptions src/ graphql/**/*.js ? Do you want to generate/update all possible GraphQL operations - \ queries, mutations and subscriptions Yes ? Enter maximum statement depth \ [increase from default if your schema is deeply nested] 2 ```	
将应用程序的前端连接到 AWS AppSync API。	要使用此模式中提供的示例 ToDoList 应用程序，请从 aws-amplify-react-native-ios-todo-app GitHub 存储库中的 App.js 文件中复制代码。然后，将示例代码集成至您的本地环境中。 存储库 App.js 文件中提供的示例代码执行以下操作： • 显示用于创建带有标题和描述字段的ToDo 项目的表单 • 显示待办事项列表（标题与描述） • 使用 `aws-amplify` 方法发布和获取数据	应用程序开发人员

相关资源

- AWS Amplify
- Amazon Cognito
- AWS AppSync
- Amazon DynamoDB
- React（React 文档）

在单个控制面板上管理多个 SaaS 产品的租户

由 Ramanna Avancha (AWS)、Jenifer Pascal (AWS)、Kishan Kavala (AWS) 和 Anusha Mandava (AWS) 创建

摘要

此模式展示了如何在 Amazon Web Services Cloud 的单个控制面板上跨多个软件即服务（SaaS）产品管理租户生命周期。提供的参考架构可以帮助组织减少在其单个 SaaS 产品中实施冗余的共享功能，并大规模提高治理效率。

大型企业可以在不同的业务部门拥有多个 SaaS 产品。这些产品通常需要预配，以供不同订阅级别的外部租户使用。如果没有通用的租户解决方案，IT 管理员必须花时间管理多个 SaaS 中的无差异功能 APIs，而不是专注于核心产品功能开发。

此模式中提供的通用租户解决方案可帮助集中管理组织的许多共享 SaaS 产品功能，包括：

- 安全
- 租户预配
- 租户数据存储
- 租户通信
- 产品管理
- 日志记录和监控

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 了解 Amazon Cognito 或第三方身份提供者（IdP）
- 了解 Amazon API Gateway
- 了解 AWS Lambda
- 了解 Amazon DynamoDB
- 了解 AWS Identity and Access Management（AWS IAM）
- 了解 AWS Step Functions
- 对 AWS CloudTrail 和亚马逊的了解 CloudWatch

- 了解 Python 库和代码
- SaaS 知识 APIs，包括不同类型的用户（组织、租户、管理员和应用程序用户）、订阅模式和租户隔离模型
- 了解组织的多产品 SaaS 要求和多租户订阅

限制

- 此模式不包括通用租户解决方案与单个 SaaS 产品之间的集成。
- 此模式仅在单个 AWS 区域中部署 Amazon Cognito 服务。

架构

目标技术堆栈

- Amazon API Gateway
- Amazon Cognito
- AWS CloudTrail
- Amazon CloudWatch
- Amazon DynamoDB
- IAM
- AWS Lambda
- Amazon Simple Storage Service (Amazon S3)
- Amazon Simple Notification Service (Amazon SNS)
- AWS Step functions

目标架构

下图显示了在 Amazon Web Services Cloud 中的单个控制面板上跨多个 SaaS 产品管理租户生命周期的示例工作流程。

图表显示了以下工作流：

1. AWS 用户通过调用 API Gateway 端点来启动租户预置、产品预置或与管理相关的操作。
2. 用户通过从 Amazon Cognito 用户池或其他 IdP 检索到的访问令牌进行身份验证。

3. 单个预置或管理任务由与 API Gateway API 端点集成的 Lambda 函数运行。
4. 通用租户解决方案（针对租户、产品和用户）的管理 APIs 收集了所有必需的输入参数、标题和令牌。然后，管理部门 APIs 调用关联的 Lambda 函数。
5. 管理 APIs 和 Lambda 函数的 IAM 权限均由 IAM 服务进行验证。
6. Lambda 函数在 DynamoDB 和 Amazon S3 中存储和检索目录（针对租户、产品和用户）中的数据。
7. 验证权限后，将调用 AWS Step Functions 工作流来执行特定任务。图中的示例显示了租户预配工作流。
8. 各个 AWS Step Functions 工作流任务在预定的工作流（状态机）中运行。
9. 运行与每个工作流任务关联的 Lambda 函数所需任何基本数据都将从 DynamoDB 或 Amazon S3 中进行检索。其他 AWS 资源可能需要使用 AWS CloudFormation 模板进行配置。
10. 如果需要，工作流会发送请求，为特定 SaaS 产品预置其他 AWS 资源到该产品的 Amazon Web Services account。
11. 当请求成功或失败时，工作流会将状态更新作为消息发布到 Amazon SNS 主题。
12. Amazon SNS 订阅了 Step Functions 工作流程的 Amazon SNS 主题。
13. 然后，Amazon SNS 将工作流状态更新发送回 AWS 用户。
14. 每个 AWS 服务的操作日志，包括 API 调用的审计记录，都将发送到 CloudWatch。可以在中 CloudWatch 为每个用例配置特定的规则和警报。
15. 日志存档在 Amazon S3 存储桶中，以便进行审计。

自动化和扩缩

此模式使用 CloudFormation 模板来帮助自动部署通用租户解决方案。该模板还可以帮助您快速向上或向下扩展相关资源。

有关更多信息，请参阅 AWS CloudFormation 用户指南中的使用 AWS CloudFormation 模板。

工具

Amazon Web Services

- Amazon API Gateway 可帮助您创建、发布、维护、监控和保护任何规模的 RES WebSocket APIs T、HTTP。
- Amazon Cognito 为您的 Web 和移动应用程序提供身份验证、授权和用户管理。
- AWS CloudTrail 可帮助您审计 AWS 账户的治理、合规和运营风险。

- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和在 AWS 上运行的应用程序的指标。
- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Storage Service (Amazon S3)是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。
- AWS Step Functions 是一项无服务器编排服务，可让您搭配使用 AWS Lambda 函数和其他 Amazon Web Services 来构建业务关键型应用程序。

最佳实践

此模式中的解决方案使用单个控制面板来管理多个租户的载入，并预配对多个 SaaS 产品的访问权限。控制面板可帮助管理用户管理其他四个特定于功能的面板：

- 安全面板
- 工作流面板
- 通信面板
- 日志和监控面板

操作说明

配置安全面板

Task	描述	所需技能
确定多租户 SaaS 平台的要求。	为以下各项制定详细要求： • 租户 • Users • 角色	云架构师、AWS 系统管理员

Task	描述	所需技能
	• SaaS 产品 • 订阅 • 配置文件交换	
设置 Amazon Cognito 服务。	按照 Amazon Cognito 开发人员指南中 Amazon Cognito 入门的说明进行操作。	云架构师
配置所需 IAM policy。	为您的使用案例创建所需 IAM policy。然后，将策略映射到 Amazon Cognito 中的 IAM 角色上。 有关更多信息，请参阅 Amazon Cognito 开发人员指南中的使用策略管理访问和基于角色的访问控制。	云管理员、云架构师、AWS IAM 安全
配置所需 API 权限。	使用 IAM 角色和策略以及 Lambda 授权方设置 API Gateway 访问权限。 有关说明，请参阅 Amazon API Gateway 开发人员指南的以下部分： • 使用 IAM 权限控制对 API 的访问 • 使用 API Gateway Lambda 授权方	云管理员、云架构师

配置数据面板

Task	描述	所需技能
创建所需数据目录。	1. 创建 DynamoDB 表以存储用户目录的数据。请确保包含用户属性和角色。此外，请确保对目录表执行数据建模，以维护每个用户和角色的必需属性和可选属性。 2. 创建 DynamoDB 表以存储产品目录的数据。确保对 SaaS 产品的特定用例进行建模。 3. 创建 DynamoDB 表以存储租户目录的数据。请确保为租户、产品和多 SaaS 订阅的许可以及标记设置订阅模型。 有关更多信息，请参阅 Amazon DynamoDB 开发人员指南中的设置 DynamoDB。	数据库管理员

配置控制面板

Task	描述	所需技能
创建 Lambda 函数和 API Gateway APIs 以运行所需的控制平面任务。	创建单独的 Lambda 函数和 API Gateway APIs 来添加、删除和管理以下内容： • Users • 租户 • 产品	应用程序开发人员

Task	描述	所需技能
	有关更多信息，请参阅 AWS Lambda 开发人员指南中的将 AWS Lambda 与 Amazon API Gateway 协作使用。	

配置工作流面板

Task	描述	所需技能
确定 AWS Step Functions 工作流必须运行的任务。	确定并记录以下各项的详细 AWS Step Functions 工作流程要求： • Users • 租户 • 产品 **Important** 确保主要利益相关者批准这些要求。	应用程序所有者
创建所需 AWS Step Functions 工作流。	1. 在 AWS Step Functions 中为用户、租户和产品创建所需工作流。有关更多信息，请参阅 AWS Step Functions 开发人员指南。 2. 确定重试和错误处理机制。有关更多信息，请参阅 AWS Blog 上的处理错误、重试和向 Step Function 状态机添加警报。	应用程序开发人员、构建主管

Task	描述	所需技能
	3. 使用 Lambda 函数实施工作流步骤。有关更多信息，请参阅 AWS Step Functions 开发人员指南中的创建使用 Lambda 的 Step Functions 状态机。 4. 根据需要将任何外部服务与 AWS Step Functions 集成。 5. 在 DynamoDB 表中维护每个工作流的状态，并使用 Amazon SNS 传达每个工作流的状态。	

配置通信面板

Task	描述	所需技能
创建 Amazon SNS 主题。	创建 Amazon SNS 主题以接收有关以下内容的通知： • 工作流状态 • 错误 • 重试 有关更多信息，请参阅 Amazon SNS 开发人员指南中的创建 Amazon SNS 主题。	应用程序所有者、云架构师
为每个 Amazon SNS 主题订阅端点。	要接收发布至某个 Amazon SNS 主题的消息，您必须为每个主题订阅一个端点。	应用程序开发人员、云架构师

Task	描述	所需技能
	有关更多信息，请参阅 Amazon SNS 开发人员指南中的订阅 Amazon SNS 主题。	

配置日志和监控面板

Task	描述	所需技能
为公共租户解决方案的每个组件激活日志记录。	在组件级别为创建的公共租户解决方案中的每个资源激活日志记录。 有关说明，请参阅： • 如何开启 CloudWatch 日志以排除我的 API Gateway REST API 或 WebSocket API 故障？ (AWS Knowledge Center) • 使用日志进行 CloudWatch 日志记录（AWS Step Functions 开发者指南） • Python 中的 AWS Lambda 函数日志记录（AWS Lambda 开发人员指南） • Amazon Cognito 中的日志和监控（Amazon Cognito 开发人员指南） • 使用亚马逊进行监控 CloudWatch（亚马逊 DynamoDB 开发者指南）	应用程序开发人员、AWS 系统管理员、云管理员

Task	描述	所需技能
	Note 您可以使用 IAM 策略将每个资源的日志整合到一个集中式日志记录账户中。有关更多信息，请参阅集中式日志记录和多账户安全防护机制。	

预配和部署通用租户解决方案

Task	描述	所需技能
创建 CloudFormation 模板。	使用 CloudFormation 模板自动部署和维护完整的通用租户解决方案及其所有组件。 有关更多信息，请参阅 A WS CloudFormation 用户指南。	应用程序开发者、 DevOps 工程师、 CloudFormation 开发者

相关资源

- 用 Amazon Cognito 用户池作为授权方控制对 REST API 的访问（Amazon API Gateway 开发人员指南）。
- 使用 API Gateway Lambda 授权方（Amazon API Gateway 开发人员指南）
- Amazon Cognito 用户池（Amazon Cognito 开发人员指南）
- 跨账户跨区域 CloudWatch 控制台（A mazon CloudWatch 用户指南）

在组织中创建跨账户 Amazon EventBridge 连接

由山姆·威尔逊 (AWS) 和罗伯特·斯通 (AWS) 创作

摘要

大型分布式系统使用亚马逊 EventBridge 在 AWS Organizations 组织中的各个 Amazon Web Services (AWS) 账户之间传达状态变化。但是， EventBridge 通常只能定位同一个端点或消费者 AWS 账户。不同账户中的事件总线除外。该事件总线是一个有效的目标。要在另一个账户中使用来自事件总线的事件，必须将事件从源账户的事件总线推送到目标账户的事件总线。为避免在管理不同应用程序中的关键事件时遇到困难 AWS 账户，请使用此模式中介绍的推荐方法。

这种模式说明了如何实现一个事件驱动的架构 EventBridge ，该架构涉及组织 AWS 账户 中的多个事件 AWS Organizations 。该模式使用 AWS Cloud Development Kit (AWS CDK) 工具包和 AWS CloudFormation.

EventBridge 提供无服务器事件总线，可帮助您接收、过滤、转换、路由和传递事件。作为事件驱动架构的关键组成部分， EventBridge 它支持将消息的生成者和消息的使用者分开。在单个账户中，这是直截了当的。多账户结构需要额外的注意事项，才能将一个账户中的事件总线上的事件用于同一组织内的其他账户。

有关生产者和消费者的账户特定注意事项的信息，请参阅 “其他信息” 部分。

先决条件和限制

先决条件

- 至少有两个关联的 AWS Organizations 组织 AWS 账户
- 两者中都有一个 AWS Identity and Access Management (IAM) 角色 AWS 账户 ，允许您 AWS 账户 通过使用在两者中配置基础设施 AWS CloudFormation
- Git 安装在了本地
- AWS Command Line Interface (AWS CLI) 已在本地安装
- AWS CDK 在本地安装并在两者中都已引导 AWS 账户

产品版本

此模式是使用以下工具和版本构建和测试的：

- AWS CDK Toolkit 2.126.0
- Node.js 18.19.0
- npm 10.2.3
- Python 3.12

这种模式应该适用于任何版本的 AWS CDK v2 或 npm。Node.js 版本 13.0.0 到 13.6.0 不兼容。 AWS CDK

架构

目标架构

下图显示了从一个账户推送事件并在另一个账户中使用该事件的架构工作流程。

该工作流程包含以下步骤：

1. Source 账户中的 Producer AWS Lambda 函数在账户的事件总线上放置一个 EventBridge 事件。
2. 跨账户 EventBridge 规则将事件路由到目标账户中的 EventBridge 事件总线。
3. 目标账户中的 EventBridge 事件总线具有调用使用者 Lambda 函数的目标 Lambda 规则。

最佳做法是使用死信队列 (DLQ) 来处理使用者 L ambda 函数的失败调用。但是，为清楚起见，此解决方案中省略了 DLQ。要详细了解如何在工作流程中实现 DLQ 并提高工作流程从故障中恢复的能力，请参阅实现 AWS Lambda 错误处理模式博客文章。

自动化和扩缩

AWS CDK 自动配置所需的架构。 EventBridge 可以扩展到每秒数千条记录，具体取决于 AWS 区域. 有关更多信息，请参阅 Amazon EventBridge 配额文档。

工具

AWS 服务

- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，可帮助您在代码中定义和配置 AWS Cloud 基础架构。此模式使用 AWS CDK Toolkit，这是一款命令行云开发套件，可帮助您与 AWS CDK 应用程序进行交互。

- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如，AWS Lambda 函数、使用 API 目的地的 HTTP 调用端点或其他 AWS 账户目的地的事件总线。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Organizations是一项账户管理服务，可帮助您将多个账户整合 AWS 账户 到一个由您创建和集中管理的组织中。

其他工具

- Node.js 是一个事件驱动的 JavaScript 运行时环境，专为构建可扩展的网络应用程序而设计。
- npm 是在 Node.js 环境中运行的软件注册表，用于共享或借用软件包以及管理私有软件包的部署。
- Python 是通用的计算机编程语言。

代码存储库

此模式的代码可在 GitHub cross-account-eventbridge-in- organization存储库中找到。

最佳实践

有关使用时的最佳实践 EventBridge，请参阅以下资源：

- Amazon EventBridge 事件模式的最佳实践
- 在 Amazon 中定义规则时的最佳实践 EventBridge

操作说明

准备您的本地 AWS CDK 部署环境

Task	描述	所需技能
为源账户和目标账户配置本地凭证。	查看设置新配置和凭据，并使用对您的环境最有意义的身份验证和凭据方法。	应用程序开发人员

Task	描述	所需技能
	Important 请务必同时 AWS CLI 为源帐户和目标帐户身份验证进行配置。 这些说明假设您在本地配置了两个 AWS 配置文件：`sourceAccount` 和`destinationAccount` 。	

两者都 AWS 账户引导。	要引导账户，请运行以下命令：
	
	`cdk bootstrap --profile sourceAccount`
	`cdk bootstrap --profile destinationAccount`
		应用程序开发人员

克隆模式代码。	要克隆存储库，请运行以下命令：
	
	`git clone git@github.com:aws-samples/aws-cdk-examples.git`
	
	然后，将目录更改为新克隆的项目文件夹：
	
	`cd aws-cdk-examples/python/cross-account-eventbridge-in-organization`
		应用程序开发人员

部署 ProducerStack 到源账户

Task	描述	所需技能
cdk.json使用您的 AWS Organizations 和账户详细信息进行修改。	在项目的根文件夹中，对以下内容进行以下更改cdk.json： • organization_id – 部署中涉及的账户的 Organizations ID • event_bus_name –CrossAccount ，或者你可以重命名它 • rules[].targets[].arn – 消费账户（目标账户）的 AWS 账户 ID	应用程序开发人员
部署 ProducerStack 资源。	从项目的根目录运行以下命令： `cdk deploy ProducerStack --profile sourceAccount` 出现提示时，接受通过 AWS CloudFormation创建的新 IAM 角色和其他与安全相关的权限。	应用程序开发人员
验证 ProducerStack 资源是否已部署。	要验证资源，请执行以下操作： 1. 在 “ AWS Management Console 源帐户” 上，选择CloudFormation。 2. 从堆栈列表中选择ProducerStack。	应用程序开发人员

Task	描述	所需技能
	3. 在 “堆栈信息” 选项卡上，验证堆栈状态是否为CREATE_COMPLETE 。或者，在资源选项卡上，查看已配置的资源。	

部署 ConsumerStack 到目标账户

Task	描述	所需技能
部署 ConsumerStack 资源。	从项目的根目录运行以下命令： `cdk deploy ConsumerStack --profile destinationAccount` 出现提示时，接受通过 AWS CloudFormation创建的新 IAM 角色和其他与安全相关的权限。	应用程序开发人员
验证 ConsumerStack 资源是否已部署	1. 在目标账户的控制台上，选择CloudFormation。 2. 从堆栈列表中选择ConsumerStack。 3. 在 “堆栈信息” 选项卡上，验证堆栈状态是否为CREATE_COMPLETE 。或者，在资源选项卡上，查看已配置的资源。	应用程序开发人员

制作和消费活动

Task	描述	所需技能
调用 Producer Lambda 函数。	1. 在源账户的控制台上，选择 Lambda。 2. 从函数列表中选择 ProducerStack-ProducerLambda XXXX（XX XX 表示由 AWS CDK 自动生成的字符序列）。 3. 选择测试选项卡。 4. 在 “测试事件” 部分中，选择 “测试”。 事件 JSON 文本区域内容可以是作为负载提供给 Lambda 函数的任何有效 JSON。在这种情况下，默认提供的 JSON 就足够了。 5. 验证 “正在执行函数：成功” 消息是否显示在 “测试事件” 部分上方的绿色横幅中。	应用程序开发人员
验证是否已收到该事件。	1. 在目标账户的控制台上，选择 Lambda。 2. 从函数列表中选择 ConsumerStack-ConsumerLambda XXXX（XX XX 表示由 AWS CDK 自动生成的字符序列）。 3. 选择监控选项卡。 4. 在 “监控” 部分，选择 “查看 CloudWatch 日志”。 5. 在新打开的选项卡中，选择最新日志流的日志流名称。	应用程序开发人员

Task	描述	所需技能
	6. 验证是否出现了如下所示的日志语句： `[DEBUG] 2024-04-08T19:08:10.091Z 9c16844a-f9de-444d-b621-86afe64d4cc8 Event: {'version':'0', 'id':'0b9faa96-973f-8be2-ecf8-75e4f328b980', 'detail-type':'TestType', 'source':'Producer', 'account' :'XXXXXXXXXXXX', 'time':'2024-04-08T19:08:09Z', 'region':'us-east-1', 'resources': [], 'detail': {'key1':'value1', 'key2':'value2', 'key3':'value3'}}`	

清理

Task	描述	所需技能
摧毁 ConsumerStack 资源。	如果您使用此模式作为测试，请清理已部署的资源以避免产生额外成本。 从项目的根目录运行以下命令：	应用程序开发人员

Task	描述	所需技能
	`cdk destroy ConsumerStack --profile destinationAccount` 系统将提示您确认删除堆栈。	
摧毁 ProducerStack 资源。	从项目的根目录运行以下命令： `cdk destroy ProducerStack --profile sourceAccount` 系统将提示您确认删除堆栈。	应用程序开发人员

故障排除

事务	解决方案
目标账户中未收到任何事件。	1. 验证提供的组织 ID 是否正确。 2. 验证来源账户是否属于所提供的组织。 3. 验证源帐户中的事件总线规则是否映射到目标帐户中的正确信息。
从控制台调用 Lambda 函数会返回以下错误：： `User: arn:aws:iam::123456789012:user/XXXXX is not authorized to perform: lambda:Invoke`	请联系您的 AWS 账户 管理员以获取对 `ProducerStack-ProducerLambdaXXXX` Lambda 函数的相应`lambda:Invoke` 操作权限。

相关资源

参考

- AWS Organizations 用户指南

- 亚马逊 EventBridge 事件模式
- Amazon 中的规则 EventBridge

教程和视频

- 教程：创建和配置组织
- AWS re: Invent 2023-亚马逊的高级事件驱动模式 (01-R) EventBridge COM3

其他信息

制作人规则

在 Source 账户中，创建了一个 EventBridge 事件总线来接受来自生产者的消息（如架构部分所示）。在此事件总线上创建了附带 IAM 权限的规则。这些规则基于以下`cdk.json`结构将目标账户中的 EventBridge 事件总线作为目标：

```
"rules": [
  {
    "id": "CrossAccount",
    "sources": ["Producer"],
    "detail_types": ["TestType"],
    "targets": [
      {
        "id": "ConsumerEventBus",
        "arn": "arn:aws:events:us-east-2:012345678901:event-bus/CrossAccount"
      }
    ]
  }
]
```

对于每条消耗的事件总线，必须包括事件模式和目标事件总线。

事件模式

事件模式筛选此规则将应用于哪些事件。在本示例中，事件源和记录`detail_types`标识了哪些事件要从源账户的事件总线传输到目标账户的事件总线。

目标事件总线

此规则针对另一个账户中存在的事件总线。需要完整的arn（Amazon 资源名称）来唯一标识目标事件总线，并且id是使用的逻辑 ID AWS CloudFormation。在创建目标规则时，目标事件总线不必实际存在。

目标账户特定的注意事项

在目标账户中，创建了一个 EventBridge 事件总线，用于接收来自源账户的事件总线的消息。要允许从来源账户发布事件，您必须创建基于资源的策略：

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowOrgToPutEvents",
    "Effect": "Allow",
    "Principal": "*",
    "Action": "events:PutEvents",
    "Resource": "arn:aws:events:us-east-2:012345678901:event-bus/CrossAccount",
    "Condition": {
      "StringEquals": {
        "aws:PrincipalOrgID": "o-XXXXXXXXX"
      }
    }
  }]
}
```

授予权限尤为重要，该events:PutEvents权限允许同一组织中的任何其他账户将活动发布到此活动总线。设置aws:PrincipalOrgId为组织 ID 可授予所需的权限。

事件模式

您可以修改包含的事件模式以满足您的用例：

```
rule = events.Rule(
    self,
    self.id + 'Rule' + rule_definition['id'],
    event_bus=event_bus,
    event_pattern=events.EventPattern(
        source=rule_definition['sources'],
        detail_type=rule_definition['detail_types'],
    )
)
```

为了减少不必要的处理，事件模式应指定只有目标账户要处理的事件才会传输到目标账户的事件总线。

基于资源的政策

此示例使用组织 ID 来控制允许哪些账户在目标账户的事件总线上放置事件。考虑使用限制性更强的策略，例如指定来源账户。

EventBridge 配额

请记住以下配额：

- 每个事件总线 300 个规则是默认配额。如有必要，可以对其进行扩展，但它应该适合大多数用例。
- 每条规则允许的最大目标为五个。我们建议应用程序架构师为每个目标账户使用不同的规则，以支持对事件模式的精细控制。

使用 Kinesis Data Streams 和 Firehose 将 DynamoDB 记录传送到亚马逊 S3 AWS CDK

由 Shashank Shrivastava (AWS) 和 Daniel Matuki da Cunha (AWS) 创作

摘要

此模式提供了使用亚马逊 Kinesis Data Streams 和 Amazon Data Firehose 将记录从亚马逊 DynamoDB 传输到亚马逊简单存储服务 (Amazon S3) 的示例代码和应用程序。该模式的方法使用 AWS Cloud Development Kit (AWS CDK) L3 结构，并包括一个示例，说明如何在将数据传输到 Amazon Web Services (AWS) 云上的目标 S3 存储桶 AWS Lambda 之前执行数据转换。

Kinesis Data Streams 记录 DynamoDB 表中的项目级别修改，并将它们按要求复制到 Kinesis Data Stream。您的应用程序可以访问 Kinesis 数据流，近实时查看项目级别的更改。Kinesis Data Streams 还提供对其他亚马逊 Kinesis 服务的访问权限，例如 Firehose 和适用于 Apache Flink 的亚马逊托管服务。这意味着您可构建应用程序，以提供实时控制面板、生成警报、实施动态定价和广告以及执行复杂数据分析。

您可将此模式用于数据集成用例。例如，运输车辆或工业设备可将大量数据发送至 DynamoDB 表中。然后，可以转换这些数据，并将其存储至 Amazon S3 中托管的数据湖中。然后，您可以使用无服务器服务（例如亚马逊 Athena、Amazon Redshift Spectrum、Amazon Rekognition 和 ）查询和处理数据并预测任何潜在的缺陷。 AWS Glue

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- AWS Command Line Interface (AWS CLI)，已安装并配置。有关更多信息，请参阅 AWS CLI 文档AWS CLI中的入门。
- Node.js (18.x+) 和 npm，已安装和配置。有关更多信息，请参阅 npm 文档中的下载和安装 Node.js 和 npm。
- aws-cdk (2.x+)，已安装并配置。有关更多信息，请参阅 AWS CDK 文档AWS CDK中的入门。
- 在本地 GitHub aws-dynamodb-kinesisfirehose-s计算机上克隆和配置的 3 次摄取存储库。
- DynamoDB 表现有示例数据。数据必须采用以下格式：`{"SourceDataId": {"S": "123"},"MessageData":{"S": "Hello World"}}`

架构

下图显示了使用 Kinesis Data Streams 和 Firehose 将记录从 DynamoDB 传输到 Amazon S3 的示例工作流程。

图表显示了以下工作流：

1. 使用 Amazon API Gateway 为 DynamoDB 代理，以摄取数据。您也可以使用任何其他来源，将数据采集至 DynamoDB。
2. 在 Kinesis Data Streams 中近乎实时生成项目级更改，然后传送至 Amazon S3。
3. Kinesis Data Streams 将记录发送到 Firehose 进行转换和交付。
4. Lambda 函数将记录从 DynamoDB 记录格式转换至 JSON 格式，后者仅包含记录项目属性名称和值。

工具

AWS 服务

- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，可帮助您在代码中定义和配置 AWS 云基础设施。
- AWS CDK Toolkit 是一个命令行云开发套件，可帮助您与 AWS CDK 应用程序进行交互。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行外壳中的命令进行交互。
- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。

代码存储库

此模式的代码可在 GitHub aws-dynamodb-kinesisfirehose-s3 摄取存储库中找到。

操作说明

设置和配置 Sample 代码

Task	描述	所需技能

安装依赖项。	在本地计算机上，通过运行以下命令，为 `pattern/aws-dynamodb-kinesisstreams-s3` 和 `sample-application` 目录中的 `package.json` 文件安装依赖项：
	`cd <project_root>/pattern/aws-dynamodb-kinesisstreams-s3`
	`npm install && npm run build`
	`cd <project_root>/sample-application/`
	`npm install && npm run build`
		应用程序开发人员，常规 AWS

生成 CloudFormation 模板。	1. 运行 `cd <project_root>/sample-application/` 命令。 2. 运行`cdk synth`命令以生成 CloudFormation 模板。 3. `AwsDynamodbKinesisfirehoseS3IngestionStack.template.js`	应用程序开发人员、常规 AWS、AWS DevOps

Task	描述	所需技能
	on 输出存储在 cdk.out 目录中。 4. 使用 AWS CDK 或 AWS Management Console 在中处理模板 CloudFormation。	

部署资源

Task	描述	所需技能
检查和部署资源。	1. 运行cdk diff命令以识别由 AWS CDK 构造创建的资源类型。 2. 运行 cdk deploy 命令以部署资源。	应用程序开发人员、常规 AWS、AWS DevOps

将数据摄取至 DynamoDB 表中以测试解决方案

Task	描述	所需技能
将您的示例数据摄取至 DynamoDB 表中。	在中运行以下命令，向你的 DynamoDB 表发送请求：AWS CLI `aws dynamodb put-item --table-name <your_table_name> --item '{"<table_partition_key>": {"S": "<partition_key_ID>"},"MessageData":{"S": "<data>"}}'`	应用程序开发人员

Task	描述	所需技能
	示例： `aws dynamodb put-item --table-name SourceData_table --item '{"SourceDataId": {"S": "123"},"MessageData":{"S": "Hello World"}}'` 默认情况下，如果操作成功，`put-item` 不返回任何值作为输出。如果操作失败，则会返回错误。数据存储在 DynamoDB 中，然后发送到 Kinesis Data Streams 和 Firehose。 **Note** 您可以使用不同的方法将数据添加到 DynamoDB 表中。有关更多信息，请参阅 DynamoDB 文档中的将数据加载到表中。	

Task	描述	所需技能
验证是否在 S3 存储桶中创建了新对象。	登录 AWS Management Console 并监控 S3 存储桶，以验证是否使用您发送的数据创建了新对象。 有关更多信息，请参阅 GetObjectAmazon S3 文档中的。	应用程序开发人员，常规 AWS

清理资源

Task	描述	所需技能
清理资源。	运行 `cdk destroy` 命令以删除此模式使用的所有资源。	应用程序开发人员，常规 AWS

相关资源

- s@@ 3-static-site-stack .ts（GitHub 存储库）
- aws-apigateway-dynamodb 模块（GitHub 存储库）
- aws-kinesisstreams-kinesisfirehose-s3 模块（GitHub 存储库）
- 更改 DynamoDB Streams 的数据采集（DynamoDB 文档）
- 使用 Kinesis Data Streams 捕获对 DynamoDB 的更改（DynamoDB 文档）

在 Amazon API Gateway 中使用自定义域名实现基于路径的 API 版本控制

由 Corey Schnedl (AWS)、Anbazhagan Ponnuswamy (AWS)、Marcelo Barbosa (AWS)、Gaurav Samudra (AWS)、Mario Lopez Martinez (AWS) 和 Abhilash Vinod (AWS) 创作

摘要

此模式演示了如何使用自定义域的 API 映射功能为 Amazon API Gateway 实施基于路径的 API 版本控制解决方案。

Amazon API Gateway 是一项完全托管的服务，您可以使用它来创建、发布、维护、监控和保护 APIs 任何规模。通过使用该服务的自定义域名功能，您可以创建更简单、更直观的自定义域名 URLs ，以便提供给 API 用户。您可以使用 API 映射将 API 阶段连接到自定义域名。创建域名并配置 DNS 记录后，您可以使用 API 映射 APIs 通过您的自定义域名向您发送流量。

在 API 公开可用后，消费者就会使用它。随着公共 API 的发展，其服务合同也在不断演变，以反映新的特性和功能。但是，更改或移除现有功能是不明智的。任何重大更改都可能影响消费者的应用程序并在运行时中断它们。API 版本控制对于避免破坏向后兼容性和破坏合约非常重要。

您需要制定明确的 API 版本控制策略，以帮助消费者采用这些策略。使用基于路径 APIs 的版本控制 URLs 是最直接和最常用的方法。在这种版本控制中，版本被明确定义为 API URIs 的一部分。以下示例 URLs 显示消费者如何使用 URI 为其请求指定 API 版本：

```
https://api.example.com/api/v1/orders
```

```
https://api.example.com/api/v2/orders
```

```
https://api.example.com/api/vX/orders
```

此模式使用为您的 AWS Cloud Development Kit (AWS CDK) API 构建、部署和测试基于路径的可扩展版本控制解决方案的示例实现。 AWS CDK 是一个开源软件开发框架，用于使用熟悉的编程语言对云应用程序资源进行建模和配置。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- 使用此模式的示例存储库和使用 Amazon API Gateway 自定义域名功能需要拥有域的所有权。您可以使用 Amazon Route 53 为您的组织创建和管理您的域名。有关如何使用 Route 53 注册或转移域的信息，请参阅 Route 53 文档中的注册新域名。

- 在为 API 设置自定义域名之前，必须准备好 SSL/TLS 证书。 AWS Certificate Manager
- 您必须创建或更新 DNS 提供程序的资源记录以映射到您的 API 端点。如果没有这样的映射，绑定到自定义域名的 API 请求就无法到达 API Gateway。

限制

- 私有域名不支持自定义域名 APIs。
- 自定义域名在所有域名中必须是 AWS 区域 唯一的 AWS 账户。
- 要配置具有多个级别的 API 映射，您必须使用区域自定义域名并使用 TLS 1.2 安全策略。
- 在 API 映射中，自定义域名和映射的域名 APIs 必须相同 AWS 账户。
- API 映射必须仅包含字母、数字和以下字符：`$-_.+!*'()/`
- API 映射中路径的最大长度为 300 个字符。
- 每个域名可以有 200 个具有多个级别的 API 映射。
- 您只能使用 TLS 1.2 安全策略 APIs 将 HTTP 映射到区域自定义域名。
- 你不能映射 WebSocket APIs 到与 HTTP API 或 REST API 相同的自定义域名。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅按地区划分的AWS 服务。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

产品版本

- 此示例实现在 2.149.0 TypeScript 版本AWS CDK 中使用。

架构

下图显示了架构工作流程。

该图阐释了以下内容：

1. API 用户使用自定义域名向 Amazon API Gateway 发送请求。
2. API Gateway 根据请求网址中指示的路径，动态地将用户的请求路由到 API Gateway 的相应实例和阶段。下表显示了如何将不同的基于 URL 的路径路由到不同的 API Gateway 实例的特定阶段的示例。

API	舞台	路径	默认终端节点
计算 APIv1	api	apiv1	已启用
计算 APIv2	api	apiv2	已启用
计算方法 APIv X	api	apiVX	已启用

3. 目标 API Gateway 实例处理请求并将结果返回给用户。

自动化和扩缩

我们建议您为每个 API 版本使用单独的 AWS CloudFormation 堆栈。通过这种方法，您可以在自定义域 API 映射功能 APIs 可以路由到的后端之间实现完全隔离。这种方法的一个优点是，可以独立部署或删除不同版本的 API，而不会带来修改其他 API 的风险。这种方法通过隔离 CloudFormation 堆栈来提高弹性。此外，它还为您的 API 提供了不同的后端选项 AWS Lambda，例如、 AWS Fargate、HTTP 端点和的 AWS 服务操作。

您可以将 Git 分支策略（例如 Gitflow）与隔离 CloudFormation 堆栈结合使用，来管理部署到不同版本的 API 的源代码。通过使用此选项，您可以维护不同版本的 API，而无需为新版本复制源代码。使用 Gitflow，您可以在执行发布时为 git 存储库中的提交添加标签。因此，您可以获得与特定版本相关的源代码的完整快照。由于需要执行更新，您可以查看特定版本中的代码，进行更新，然后将更新的源代码部署到与相应主版本一致的 CloudFormation 堆栈中。这种方法降低了破坏其他 API 版本的风险，因为每个 API 版本都有独立的源代码并部署到不同的 CloudFormation 堆栈中。

工具

AWS 服务

- Amazon API Gateway 可帮助您创建、发布、维护、监控和保护任何规模的 RES WebSocket APIs T、HTTP。
- AWS Certificate Manager (ACM) 可帮助您创建、存储和续订保护您的网站和应用程序的公共和私有 SSL/TLS X.509 证书和密钥。 AWS
- AWS Cloud Development Kit (AWS CDK)是一个开源软件开发框架，用于在代码中定义您的云基础架构并通过它进行配置 AWS CloudFormation。此模式的示例实现使用 AWS CDK in TypeScript。在 in AWS CDK 中 TypeScript 使用熟悉的工具，包括 Microsoft TypeScript 编译器 (`tsc`)、Node.js 和节点包管理器 (npm)。如果你愿意，你可以使用 Yarn，尽管此模式中的示例使用npm了。构成AWS构造库的模块通过npm 存储库 npm js.org 分发。

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Route 53 是一种可用性高、可扩展性强的 DNS Web 服务。
- AWS WAF是一种 Web 应用程序防火墙，可帮助您监控转发到受保护的 Web 应用程序资源的 HTTP 和 HTTPS 请求。

其他工具

- Bruno 是一款开源、对 git 友好的 API 测试客户端。
- cdk-nag 是一个开源实用程序，它使用规则包检查 AWS CDK 应用程序的最佳实践。

代码存储库

此模式的代码可在 GitHub path-based-versioning-with-api- gateway 存储库中找到。

最佳实践

- 使用强大的持续集成和持续交付 (CI/CD) 管道，自动测试和部署使用构建的 CloudFormation 堆栈。AWS CDK有关此建议的更多信息，请参阅 Well-Architect AWS ed 指南 DevOps 。
- AWS WAF 是一款托管防火墙，可轻松与 Amazon API Gateway 等服务集成。尽管 AWS WAF 这不是此版本控制模式起作用的必要组件，但我们建议将其作为安全最佳实践包含 AWS WAF 在 API Gateway 中。
- 鼓励 API 使用者定期升级到最新版本的 API，以便可以高效地弃用和移除旧版本的 API。
- 在生产环境中使用此方法之前，请为您的 API 实施防火墙和授权策略。
- 使用最低权限访问模式实现对 AWS 资源管理的访问权限。 AWS 账户
- 要对使用构建的应用程序强制执行最佳实践和安全建议 AWS CDK，我们建议您使用 cdk-na g 实用程序。

操作说明

准备好您的本地环境

Task	描述	所需技能
克隆存储库。	要克隆示例应用程序存储库，请运行以下命令： `git clone https://github.com/aws-samples/path-based-versioning-with-api-gateway`	应用程序开发人员
导航到克隆的存储库。	要导航到克隆的存储库文件夹位置，请运行以下命令： `cd api-gateway-custom-domain-versioning`	应用程序开发人员
安装所需的依赖项。	要安装所需的依赖项，请运行以下命令： `npm install`	应用程序开发人员

部署 CloudFormation 路由堆栈

Task	描述	所需技能
启动路由堆栈的部署。	要启动 CloudFormation 路由堆栈的部署`CustomDomainRouterStack`，请运行以下命令，`example.com` 替换为您拥有的域名：	应用程序开发人员

Task	描述	所需技能
	`npx cdk deploy CustomDomainRouterStack --parameters PrerequisiteDomainName=example.com` **Note** 只有成功执行以下域 DNS 验证任务，才能成功部署堆栈。	

验证域所有权

Task	描述	所需技能
验证您的域名的所有权。	在您证明关联域的所有权之前，证书将一直处于 “待验证” 状态。 要证明所有权，请将 CNAME 记录添加到与该域关联的托管区域。有关更多信息，请参阅 AWS Certificate Manager 文档中的 DNS 验证。 添加适当的记录可以使`CustomDomainRouterStack` 部署成功。	应用程序开发者、AWS 系统管理员、网络管理员
创建别名记录以指向您的 API Gateway 自定义域。	成功颁发并验证证书后，创建一条指向您的 Amazon API Gateway 自定义域名网址的 DNS 记录。	应用程序开发者、AWS 系统管理员、网络管理员

Task	描述	所需技能
	自定义域 URL 由配置自定义域名时唯一生成，并指定为 CloudFormation 输出参数。以下是该记录的示例： 路由策略：简单路由 记录名称：`demo.api-gateway-custom-domain-versioning.example.com` 别名：是 记录类型：指向 AWS 资源的“A”类型的 DNS 记录 值：`d-xxxxxxxxxx.execute-api.xx-xxxx-x.amazonaws.com` TTL（秒）：30 0	

部署 CloudFormation 堆栈并调用 API

Task	描述	所需技能
部署 `ApiStackV1` 堆栈。	要部署`ApiStackV1` 堆栈，请使用以下命令： `npm run deploy-v1` 以下 CDK 代码添加了 API 映射：	应用程序开发人员

Task	描述	所需技能
	`var apiMapping = new CfnApiMapping(this, "ApiMapping", { apiId: this.lambdaRestApi.restApiId, domainName: props.customDomainName.domainName, stage: "api", apiMappingKey: "api/v1", });`	
部署 ApiStackV2 堆栈。	要部署ApiStackV2 堆栈，请使用以下命令： `npm run deploy-v2`	应用程序开发人员
调用 API。	要使用 Bruno 调用 API 并测试 API 端点，请参阅其他信息中的说明。	应用程序开发人员

清理资源

Task	描述	所需技能
清理资源。	要销毁与此示例应用程序关联的资源，请使用以下命令： `npx cdk destroy --all` **Note** 请务必清理为域所有权验证过程手动添加的	应用程序开发人员

Task	描述	所需技能
	所有 Route 53 DNS 记录。	

故障排除

事务	解决方案
部署超CustomDomainRouterStack 时，因为无法验证证书。	确保按照之前的任务中所述添加了正确的 DNS 验证 CNAME 记录。添加 DNS 验证记录后，您的新证书可能会在 30 分钟内继续显示待验证状态。有关更多信息，请参阅 AWS Certificate Manager 文档中的 DNS 验证。

相关资源

- 使用 Amazon 实现基于标头的 API Gateway 版本控制 CloudFront — 这篇 AWS Compute 博客文章提供了一种基于标头的版本控制策略，作为该模式中概述的基于路径的版本控制策略的替代方案。
- AWS CDK 研讨会 — 此入门研讨会重点介绍如何使用构建和部署应用程序 AWS Cloud Development Kit (AWS CDK)。 AWS 本研讨会支持 Go、Python 和 TypeScript。

其他信息

使用 Bruno 测试你的 API

我们建议您使用开源 API 测试工具 Bruno 来验证基于路径的路由是否适用于示例应用程序。此模式提供了一个样本集合，便于测试您的示例 API。

要调用和测试您的 API，请按以下步骤操作：

1. 安装 Bruno。
2. 打开 Bruno。
3. 在此模式的代码存储库中，选择 Bruno/Sample-API-Gateway-Custom-Domain-Versioning 并打开该集合。

4. 要查看用户界面 (UI) 右上角的 “环境” 下拉列表，请选择集合中的任何请求。
5. 在 “环境” 下拉列表中，选择 “配置”。
6. 用您的自定义域名替换该`REPLACE_ME_WITH_YOUR_DOMAIN`值。
7. 选择 “保存”，然后关闭 “配置” 部分。
8. 对于沙盒环境，请确认已选中 “活动” 选项。
9. 使用所选请求的 -> 按钮调用您的 API。
10. 请注意与 V2 相比，V1 中如何处理验证（传入非数字值）。

要查看示例 API 调用的屏幕截图以及 V1 和 V2 验证的比较，请参阅在此模式的代码存储库中的`README.md`文件中测试您的示例 API。

将 psycopg2 库导入到以与你的 PostgreS AWS Lambda QL 数据库进行交互

由 Louis Hourcade (AWS) 创作

摘要

Psycopg 是一款适用于 Python 的 PostgresSQL 数据库适配器。开发人员使用该psycopg2库来编写与 PostgreSQL 数据库交互的 Python 应用程序。

在 Amazon Web Services (AWS) 上，开发人员还使用AWS Lambda为应用程序或后端服务运行代码。Lambda 是一种无服务器、事件驱动的计算服务，无需预置或管理服务器即可运行代码。

默认情况下，当您创建使用 Python 运行时（版本 3.9、3.8 或 3.7）的新函数时，Lambda 运行时环境是根据提供的 L ambda 基础映像创建的。 AWS基础映像中不包含库，例如pandas或psycopg2。要使用库，您需要将其捆绑到自定义包中并将其附加到 Lambda。

有多种方法可以捆绑和附加库，包括以下几种：

- 从 .zip 文件档案中部署您的 Lambda 函数。
- 从自定义容器镜像部署您的 Lambda 函数。
- 创建一个 Lambda 层，并将其附加到您的 Lambda 函数。

此模式演示了前两个选项。

使用.zip 部署包，将pandas库添加到您的 Lambda 函数相对简单。在您的 Linux 计算机上创建一个文件夹，将 Lambda 脚本与pandas库和库的依赖项一起添加到该文件夹，压缩该文件夹，然后将其作为您的 Lambda 函数的源提供。

尽管使用.zip 部署包是一种常见的做法，但这种方法不适用于该psycopg2库。此模式首先显示您使用.zip 部署包将psycopg2库添加到 Lambda 函数时遇到的错误。然后，该模式显示了如何从 Dockerfile 部署 Lambda 并编辑 Lambda 映像以使库正常运行。psycopg2

有关该模式部署的三种资源的信息，请参阅 “其他信息” 部分。

先决条件和限制

先决条件

- AWS 账户 具有足够权限部署此模式所用 AWS 资源的活跃用户
- AWS Cloud Development Kit (AWS CDK) 通过运行全局安装 `npm install -g aws-cdk`

- 一个 Git 客户端
- Python
- Docker

限制

- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额页面，然后选择服务的链接。

产品版本

- AWS Lambda 运行时版本：Python 3.8（该模式可以适用于其他 Python 版本。）
- Psycopg2 版本 2.9.3
- Pandas 版本 1.5.2

架构

解决方案概述

为了说明在 Lambda 中使用该psycopg2库时可能面临的挑战，该模式部署了两个 Lambda 函数：

- 一个使用.zip 文件创建的 Python 3.8 运行时的 Lambda 函数。使用 pip 将psycopg2和pandas库安装到此.zip 部署包中。
- 一个使用 Python 3.8 运行时的 Lambda 函数，该函数是从 DockerFile 创建的。Dockerfile 会将psycopg2和pandas库安装到 Lambda 容器镜像中。

第一个 Lambda 函数将pandas库及其依赖项安装在.zip 文件中，Lambda 可以使用该库。

第二个 Lambda 函数演示，通过为 Lambda 函数构建容器映像，您可以在 Lambda 中运行和pandas库psycopg2。

工具

AWS 服务

- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，可帮助您用代码定义和配置 AWS 云基础设施。

- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。

其他工具

- Docker 是一组平台即服务（PaaS）产品，它们使用操作系统级别的虚拟化技术在容器中交付软件。
- p@@ andas 是一款基于 Python 的开源工具，用于数据分析和操作。
- Psycopg 是一款适用于 Python 语言的 PostgreSQL 数据库适配器，专为多线程应用程序而设计。此模式使用 Psycopg 2。
- Python 是通用的计算机编程语言。

代码存储库

此模式的代码可在上的 import-psycopg2--存储库中找到。in-lambda-to-interact with-postgres-database GitHub

最佳实践

此模式为您提供了一个使用 AWS CDK 从 Dockerfile 创建 Lambda 函数的工作示例。如果您在应用程序中重复使用此代码，请确保部署的资源满足所有安全要求。使用诸如 Checkov 之类的工具，它可以在部署基础架构之前扫描云基础设施配置，以发现配置错误。

操作说明

克隆存储库并配置部署

Task	描述	所需技能

克隆存储库。	要在本地计算机上克隆 GitHub 存储库，请运行以下命令：
	`git clone https://github.com/aws-samples/import-psycopg2-in-lambda-to-interact-with-postgres-database.git`
	`cd AWS-lambda-psycopg2`
		常规 AWS

Task	描述	所需技能

配置您的部署。	使用有关您的 AWS 账户以下信息的app.py文件进行编辑：
	`aws_acccount = "AWS_ACCOUNT_ID"`
	`region = "AWS_REGION"`
	`# Select the CPU architecture you are using to build the image (ARM or X86)`
	`architecture = "ARM"`
		常规 AWS

引导您的 AWS 账户并部署应用程序

Task	描述	所需技能
引导你的 AWS 账户.	如果您尚未引导 AWS 环境，请使用您的 AWS 账户 AWS 凭证运行以下命令： `cdk bootstrap aws://<tooling-account-id>/<aws-region>`	常规 AWS
部署代码。	要部署 AWS CDK 应用程序，请运行以下命令： `cdk deploy AWSLambdaPyscopg2`	常规 AWS

通过 AWS 管理控制台测试 Lambda 函数

Task	描述	所需技能
测试根据.zip 文件创建的 Lambda 函数。	要测试通过.zip 文件创建的 Lambda 函数，请执行以下操作： 1. 登录控制台，然后在上打开 Lambda 控制台。https://console.aws.amazon.com/lambda/ 2. 选择 `lambda-from-zip` Lambda 函数。 3. 创建一个测试事件来调用该函数。 4. 调用该函数时，该函数应引发包含以下消息的错误： `"errorMessage": Unable to import module 'lambda_code': libpq.so.5: cannot open shared object, "stackTrace": [] "errorType": Runtime.ImportModuleError",` 5. 打开 Amazon CloudWatch 控制台，网址为https://console.aws.amazon.com/cloudwatch/。 CloudWatch 日志显示pandas库已成功导入，但psycopg2库导入失败。	常规 AWS

Task	描述	所需技能
	由于 Lambda 在默认镜像中找不到所需的 PostgreSQL 库，因此它无法使用该库。`psycopg2`	

Task	描述	所需技能

测试从 Dockerfile 创建的 Lambda 函数。	要在 Lambda 函数中使用该psycopg2库，您必须编辑 Lambda 亚马逊系统映像 (AMI)。
	
	要测试从 Dockerfile 创建的 Lambda 函数，请执行以下操作：
	
	1. 登录控制台，然后打开 Lambda 控制台。
	2. 选择 `lambda-from-docker` Lambda 函数。
	3. 创建一个测试事件来调用该函数。
	4. 调用时，该函数应成功运行。
	
	以下代码显示了 AWS CDK 模板创建的 Dockerfile：
	
	`# Start from lambda Python3.8 image`
	`FROM public.ecr.aws/lambda/python:3.8`
	
	`# Copy the lambda code, together with its requirements`
	`COPY lambda/requirements.txt ${LAMBDA_TASK_ROOT}`
	`COPY lambda/lambda_code.py ${LAMBDA_TASK_ROOT}`
		常规 AWS

Task	描述	所需技能
	``` # Install postgresql- devel in your image RUN yum install -y gcc postgresql-devel # install the requireme nts for the Lambda code RUN pip3 install -r requirements.txt --target "${LAMBDA _TASK_ROOT}" # Command can be overwritten by providing a different command in the template directly. CMD ["lambda_code.hand ler"] ``` Dockerfile 采用提供的 Python 3.8 运行时的 Lamb AWS da 镜像并安装 p ostgresql-devel，其中包含编译直接与 PostgreSQL 管理服务器交互的应用程序所需的库。D ockerfile 还会安装文件中指示的`pandas`requireme nts.txt 和`psycopg2`库。	

相关资源

- AWS CDK 文档
- AWS Lambda 文档

其他信息

在这种模式中，AWS CDK 模板提供了一个包含三种资源的 AWS 堆栈：

- Lambda 函数的AWS Identity and Access Management (IAM) 角色。
- 一个带有 Python 3.8 运行时的 Lambda 函数。该函数是从部署包中`Constructs/lambda/lambda_deploy.zip`部署的。
- 一个带有 Python 3.8 运行时的 Lambda 函数。该函数是从 Dockerfile 中部署到该文件夹下的`Constructs`

两个 Lambda 函数的脚本都会检查`pandas`和`psycopg2`库是否成功导入：

```
import pandas
print("pandas successfully imported")

import psycopg2
print("psycopg2 successfully imported")

def handler(event, context):
    """Function that checks whether psycopg2  and pandas are successfully imported or
 not"""
    return {"Status": "psycopg2 and pandas successfully imported"}
```

`lambda_deploy.zip`部署包是使用 `Constructs/lambda/build.sh` bash 脚本构建的。此脚本创建文件夹、复制 Lambda 脚本、安装`pandas`和`psycopg2`库并生成.zip 文件。要自己生成.zip 文件，请运行此 bash 脚本并重新部署堆栈。 AWS CDK

Dockerfile 以为带有 Python 3.8 运行时的 Lambda AWS 提供的基础映像开头。Dockerfile 会在默认映像之上安装`pandas`和`psycopg2`库。

此模式显示了一种在 Lambda 中使用`psycopg2`库的方法，即从 Dockerfile 创建函数并向 Lambda 映像添加所需的依赖项。有关实现此目的的其他方法，请参阅 GitHub awslambda-ps ycopg2 存储库。

将 Amazon API Gateway 与亚马逊 SQS 集成以处理异步 REST APIs

由纳塔利娅·科兰托尼奥·法韦罗（AWS）和古斯塔沃·马蒂姆（AWS）创作

摘要

部署 REST 时 APIs，有时需要公开客户端应用程序可以发布的消息队列。例如，您可能遇到第三方延迟 APIs 和响应延迟的问题，或者您可能希望避免数据库查询的响应时间，或者避免在存在大量并发请求时扩展服务器 APIs。在这些情况下，发布到队列的客户端应用程序只需要知道 API 已收到数据，而不是在收到数据后会发生什么。

此模式使用亚马逊 API Gateway 向亚马逊简单队列服务 (Amazon SQS) Simple Q ueue Service 发送消息来创建 REST API 终端节点。它在两个服务之间创建了 easy-to-implement集成，从而避免了直接访问 SQS 队列。

先决条件和限制

- 一个活跃的 AWS 账户

架构

该图说明了以下步骤：

1. 使用 Postman、其他 API 或其他技术等工具请求 POST REST API 端点。
2. API Gateway 在队列中发布一条消息，该消息在请求正文中接收。
3. Amazon SQS 收到消息并向 API Gateway 发送答案，并附上成功或失败代码。

工具

- Amazon API Gateway 可帮助您创建、发布、维护、监控和保护任何规模的 RES WebSocket APIs T、HTTP。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- Amazon Simple Queue Service (Amazon SQS) 提供了一个安全、持久且可用的托管队列，它可帮助您集成和分离分布式软件系统与组件。

操作说明

创建 SQS 队列

Task	描述	所需技能
创建队列。	要创建接收来自 REST API 的消息的 SQS 队列，请执行以下操作： 1. 登录到您的 AWS 账户。 2. 打开 Amazon SQS 控制台，网址为。https://console.aws.amazon.com/sqs/ 3. 选择创建队列。 4. 在 “创建队列” 页面上，AWS 区域 从 “区域” 下拉列表中选择正确的队列。 5. 对于 “类型”，保留默认设置（标准）。 6. 输入队列的名称。 7. 保留所有其他设置的默认值。 8. 选择创建队列。	应用程序开发人员

提供对亚马逊 SQS 的访问权限

Task	描述	所需技能
创建一个 IAM 角色。	此 IAM 角色授予 API Gateway 资源对亚马逊 SQS 的完全访问权限。	应用程序开发者、AWS 管理员

Task	描述	所需技能
	1. 使用 https://console.aws.amazon.com/iam/ 打开 IAM 控制台。 2. 在导航窗格中，选择 Roles（角色）和 Create role（创建角色）。 3. 对于 Trusted entity type（可信实体类型），选择 AWS 服务。 4. 对于用例，从下拉列表中选择 API Gatew ay，然后选择下一步，下一步。 5. 在角色名称中，输入 AWSGatewayRoleForSQS 和可选描述，然后选择创建角色。 6. 在“角色”窗格中，搜索 SAWSGatewayRoleForQS，然后选中其复选框。 7. 在权限策略部分中，选择添加权限、附加策略。 8. 搜索 Amazon A SQSFull cces s 并将其选中。 9. 选择添加权限。 10. 在 AWSGatewayRoleForSQS 的“摘要”部分，复制亚马逊资源编号 (ARN)。您将在后面的步骤中使用此 ID。	

创建 REST API

Task	描述	所需技能
创建 REST API。	这是 HTTP 请求发送到的 REST API。 1. 打开 API Gateway 控制台，网址为https://console.aws.amazon.com/apigateway/。 2. 在 REST API 部分中，选择构建。 3. 在 API 名称中，输入您的 API 的名称和可选描述，保留所有其他默认设置，然后选择创建 API。	应用程序开发人员
将 API Gateway 连接到亚马逊 SQS。	此步骤允许消息从 HTTP 请求正文内部流向 Amazon SQS。 1. 在 API Gateway 控制台上，选择您创建的 API。 2. 在 “资源” 页面的 “方法” 部分，选择 “创建方法”。 3. 对于方法类型，选择 POST。 4. 对于集成类型，选择AWS 服务。 5. 对于 AWS 区域，请选择您创建 SQS 队列的区域。 6. 对于 AWS 服务，选择简单队列服务 (SQS) Simple Queue Service。 7. 对于 HTTP 方法，请选择 POST。	应用程序开发人员

Task	描述	所需技能
	8. 对于 “操作类型”，选择 “使用路径覆盖”。 9. 对于 “路径覆盖”，输入/<AWS account ID><name of SQS queue>。 10对于执行角色，请粘贴您之前创建的角色的 ARN。 11选择创建方法。	

测试 REST API

Task	描述	所需技能
测试 REST API。	运行测试以检查是否缺少配置： 1. 在 API Gateway 控制台上，选择您创建的 REST API。 2. 在资源窗格中，选择 POST 方法。 3. 选择测试选项卡。（如果未显示选项卡，请使用向右箭头。） 4. 在请求正文中，粘贴以下 JSON 代码： `{ "message": "lorem ipsum" }` 5. 选择测试。	应用程序开发人员

Task	描述	所需技能
	您将收到类似于以下内容的错误： `<UnknownOperationException/>`	

Task	描述	所需技能
更改 API 集成，将请求正确转发给 Amazon SQS。	完成配置以修复集成错误： 1. 在 API Gateway 控制台上，选择您创建的 API，然后选择 POST。 2. 方法执行部分显示了 API Gateway 和 Amazon SQS 之间的直观映射。在此部分中，选择集成请求，然后选择编辑。 3. 展开 HTTP 标头部分，然后选择添加请求标头参数。 • 在“名称”中，指定“内容类型”。 • 对于“映射来源”，输入“应用程序/ x-www-form-urlencoded”。确保包括单引号。 • 选中“缓存”复选框。 4. 展开“映射模板”部分。 • 选择 Add mapping template (添加映射模板)。 • 对于内容类型，输入 application /j son。 • 对于模板正文，请粘贴以下代码： `Action=SendMessage&MessageBody=$input.body` • 选择保存。	应用程序开发人员

Task	描述	所需技能

在 Amazon SQS 中测试和验证消息。	运行测试以确认测试成功完成：
	1. 在 API Gateway 控制台上，选择您创建的 REST API。
	2. 在资源窗格中，选择 POST 方法。
	3. 选择测试选项卡。（如果未显示选项卡，请使用向右箭头。）
	4. 在请求正文中，粘贴以下 JSON 代码：
	`{ "message": "lorem ipsum" }`
	5. 选择测试。
	6. 打开 Amazon SQS 控制台。
	7. 在导航窗格中，选择队列，然后选择您的队列。
	8. 选择发送和接收消息。
	9. 选择轮询消息。
	10选择消息。它应显示以下内容：
	`Body { "message": "lorem ipsum" }`
		应用程序开发人员

Task	描述	所需技能

使用特殊字符测试 API Gateway。	运行包含消息中不可接受的特殊字符（例如 &）的测试：
	1. 在 API Gateway 控制台上，选择你的 API。
	2. 使用以下 JSON 代码重复前面步骤中的测试：
	`{ "message": "lorem ipsum &" }`
	3. 选择测试。
	您将收到如下错误消息：
	`{ "Error": { "Code": "AccessDenied", "Message": "Access to the resource https://sqs.us-east-2.amazonaws.com/976166761794/Apg2 is denied.", "Type": "Sender" }, "RequestId": "e83c9c67-bcf6-5e9a-91e9-c737094b17ab" }`
	这是因为默认情况下，邮件正文中不支持特殊字符。在下一
		应用程序开发人员

Task	描述	所需技能
	步中，您将配置 API Gateway 以支持特殊字符。有关内容类型转换的更多信息，请参阅 API Gateway 文档。	

Task	描述	所需技能

更改 API 配置以支持特殊字符。	调整配置以接受消息中的特殊字符：
	1. 在 API Gateway 控制台上，选择您创建的 API，然后选择 POST。
	2. 选择集成请求，然后选择编辑。
	3. 将 “内容处理” 更改为 “转换为文本”。
	4. 在 “映射模板” 部分中：
	• 对于内容类型，输入 application /j son。
	• 在模板正文中，指定：
	`Action=SendMessage&MessageBody=$util.urlEncode($input.body)`
	• 选择保存。
	5. 选择测试选项卡。
	6. 在请求正文中，输入之前的 JSON 代码：
	`{ " message": "lorem ipsum &" }`
	7. 选择测试。
	8. 打开 Amazon SQS 控制台。
	9. 选择您的队列，然后依次选择 “发送和接收消息”、“轮
		应用程序开发人员

Task	描述	所需技能
	询留言”、“像以前一样留言”。 新消息应包含特殊字符。	

部署 REST API

Task	描述	所需技能
部署 API。	要部署 REST API，请执行以下操作 1. 打开 API Gateway 控制台。 2. 选择 API。 3. 选择部署 API。有关此步骤的更多信息，请参阅 API Gateway 文档。	应用程序开发人员
使用外部工具进行测试。	使用外部工具运行测试以确认消息已成功接收： 1. 打开诸如 Postman、Insomnia 或 curl 之类的工具。 2. 运行你的 API。 3. 打开 Amazon SQS 控制台。 4. 选择您的队列。 5. 加载消息以查看新消息。	应用程序开发人员

清除

Task	描述	所需技能
删除 API。	在 API Gateway 控制台上，选择您创建的 API，然后选择删除。	应用程序开发人员
删除 IAM 角色。	在 IAM 控制台的角色窗格中，选择 AWSGatewayRoleForSQS，然后选择删除。	应用程序开发人员
删除 SQS 队列。	在 Amazon SQS 控制台的队列窗格中，选择您创建的 SQS 队列，然后选择删除。	应用程序开发人员

相关资源

- SQS-SendMessage（API Gateway 文档）
- API Gateway 中的内容类型转换（API Gateway 文档）
- $util 变量（API Gateway 文档）
- 如何将 API Gateway REST API 与亚马逊 SQS 集成并解决常见错误？（re AWS : post 文章）

使用 Amazon API Gateway 和 AWS Lambda 异步处理事件

由安德里亚·梅罗尼（AWS）、纳迪姆·马吉德（AWS）、Mariem Kthiri（AWS）和迈克尔·沃尔纳（AWS）创作

摘要

Amazon API Gateway 是一项完全托管的服务，开发人员可以使用它来创建、发布、维护、监控和保护 APIs 任何规模。它可以处理接受和处理多达数十万个并发 API 调用所涉及的任务。

API Gateway 的一个重要服务配额是集成超时。超时是指在 REST API 返回错误之前，后端服务必须返回响应的最长时间。对于同步工作负载，29 秒的硬限制通常是可以接受的。但是，对于那些想要将 API Gateway 用于异步工作负载的开发者来说，这个限制是一个挑战。

此模式显示了使用 API Gateway 和 AWS Lambda异步处理事件的架构示例。该架构支持运行时长不超过 15 分钟的处理作业，并使用基本的 REST API 作为接口。

Projen 与 T AWS Cloud Development Kit (AWS CDK) oo lkit、Docker 和 Node.js 结合使用 AWS 账户，用于设置本地开发环境并将示例架构部署到目标。Projen 通过预提交和用于代码质量保证、安全扫描和单元测试的工具自动设置 Python 虚拟环境。有关更多信息，请参阅 “工具” 部分。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 您的工作站上安装了以下工具：
  - AWS Cloud Development Kit (AWS CDK) 工具包版本 2.85.0
  - Docker 版本 20.10 .21
  - Node.js 版本 18.13.0
  - Projen 版本 0.71. 111
  - Python 版本 3.9.16

限制

- 任务的最大运行时间受到 Lambda 函数的最大运行时间（15 分钟）的限制。
- 并发任务请求的最大数量受到 Lambda 函数预留并发性的限制。

架构

下图显示了任务 API 与事件处理和错误处理 Lambda 函数以及存储在 Amazon 事件档案中的事件之间的交互。 EventBridge

典型的工作流程包括以下步骤：

1. 您通过 AWS Identity and Access Management (IAM) 进行身份验证并获取安全证书。
2. 您向作/jobs业 API 端点发送 HTTP POST 请求，在请求正文中指定任务参数。
3. 作业 API 是一个 API Gateway REST API，它会向你返回一个包含任务标识符的 HTTP 响应。
4. 作业 API 异步调用事件处理 Lambda 函数。
5. 事件处理函数处理事件，然后将任务结果放入作业 Amazon DynamoDB 表中
6. 您向作/jobs/{jobId}业 API 端点发送 HTTP GET 请求，第 3 步中的任务标识符为{jobId}。
7. 作业 API 查询 jobs DynamoDB 表以检索任务结果。
8. 作业 API 会返回包含任务结果的 HTTP 响应。
9. 如果事件处理失败，则事件处理函数会将事件发送到错误处理函数。
10. 错误处理函数将作业参数放在 DynamoD jobs B 表中。
11. 您可以通过向作业 API 端点发送 HTTP GET 请求来检索/jobs/{jobId}任务参数。
12. 如果错误处理失败，则错误处理函数会将事件发送到 EventBridge 事件存档。

    您可以使用重播存档的事件 EventBridge。

工具

Amazon Web Services

- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，可帮助您在代码中定义和配置 AWS Cloud 基础架构。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您通过命令行外壳中的命令与 AWS 服务进行交互。
- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。

- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如，Lambda 函数、使用 API 目标的 HTTP 调用终端节点或其他中的事件总线。AWS 账户
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。

其他工具

- autopep8 会根据 Python 增强提案 (PEP) 8 风格指南自动格式化 Python 代码。
- Bandit 会扫描 Python 代码以查找常见的安全问题。
- C@@ ommitizen 是一个 Git 提交检查器和生成器。CHANGELOG
- cfn-lint 是个傻瓜 AWS CloudFormation
- Checkov 是一种静态代码分析工具，用于检查基础设施即代码 (IaC) 是否存在安全性和合规性错误配置。
- jq 是一个用于解析 JSON 的命令行工具。
- Postman 是一个 API 平台。
- 预提交是一个 Git 挂钩管理器。
- Projen 是一个项目生成器。
- pytest 是一个 Python 框架，用于编写可读的小型测试。

代码存储库

此示例架构代码可以在使用 API Gateway 和 Lambda 进行 GitHub 异步事件处理存储库中找到。

最佳实践

- 此示例架构不包括对已部署基础设施的监控。如果您的用例需要监控，请评估添加 CDK 监控结构或其他监控解决方案。
- 此示例架构使用 IAM 权限来控制对作业 API 的访问权限。任何有权假设的人`JobsAPIInvokeRole`都可以调用作业 API。因此，访问控制机制是二进制的。如果您的用例需要更复杂的授权模型，请使用不同的访问控制机制进行评估。
- 当用户向`/jobs`作业 API 端点发送 HTTP `POST` 请求时，将在两个不同的级别对输入数据进行验证：
  - 亚马逊 API Gateway 负责第一个请求的验证。

- 事件处理函数执行第二个请求。

  当用户向/jobs/{jobId}作业 API 端点发出 HTTP GET 请求时，不会执行任何验证。如果您的用例需要额外的输入验证和更高的安全级别，请评估如何使用 AWS WAF 来保护您的 API。

操作说明

设置环境

Task	描述	所需技能
克隆存储库。	要在本地克隆存储库，请运行以下命令： `git clone https://github.com/aws-samples/asynchronous-event-processing-api-gateway-lambda-cdk.git`	DevOps 工程师

设置项目。	将目录更改为存储库根目录，然后使用 P rojen 设置 Python 虚拟环境和所有工具：
	`cd asynchronous-event-processing-api-gateway-api-gateway-lambda-cdk`
	`npx projen`
		DevOps 工程师

安装预提交挂钩。	要安装预提交挂钩，请执行以下操作： 1. 激活 P ython 虚拟环境： `source .env/bin/activate` 2. 安装预提交挂钩：	DevOps 工程师

Task	描述	所需技能
	`pre-commit install` `pre-commit install --hook-type commit-msg`	

部署示例架构

Task	描述	所需技能
Bootstrap AWS CDK。	要 AWS CDK 在中进行引导 AWS 账户，请运行以下命令： `AWS_PROFILE=$YOUR_AWS_PROFILE npx projen bootstrap`	AWS DevOps
部署示例架构。	要在中部署示例架构 AWS 账户，请运行以下命令： `AWS_PROFILE=$YOUR_AWS_PROFILE npx projen deploy`	AWS DevOps

测试架构

Task	描述	所需技能
安装测试先决条件。	在你的工作站上安装 AWS Command Line Interface (AWS CLI)、Postman 和 j q。 建议使用 Postman 来测试此示例架构，但这不是强制性的。如果您选择其他 API 测试工	DevOps 工程师

Task	描述	所需技能
	具，请确保它支持AWS 签名版本 4 身份验证，并参考可通过导出 REST API 来检查的公开的 API 端点。	
假设JobsAPIInvokeRole 。	假JobsAPIInvokeRole 设打印为 deploy 命令的输出： `CREDENTIALS=$(AWS_ PROFILE=$<YOUR_AWS _PROFILE> aws sts assume-role \ --no-cli-pager \ --role-arn $<JOBS_AP I_INVOKE_ROLE_ARN> \ --role-session-name JobsAPIInvoke) export AWS_ACCES S_KEY_ID=$(cat $CREDENTIALS \| jq '.Credentials''.Ac cessKeyId') export AWS_SECRE T_ACCESS_KEY=$(cat $CREDENTIALS \| jq '.Credentials''.Se cretAccessKey') export AWS_SESSI ON_TOKEN==$(cat $CREDENTIALS | jq '.Credentials''.Se ssionToken')`	AWS DevOps

Task	描述	所需技能
配置 Postman。	1. 要导入存储库中包含的 Postman 集合，请按照 Postman 文档中的说明进行操作。 2. 使用以下值@@ 设置JobsAPI变量： • accessKey – assume-role 命令中Credentials.AccessKeyId 属性的值 • baseUrl– deploy 命令的JobsApiJobsAPIEndpoint 输出值，不带尾部斜杠 • region– 示例架构的部署 AWS 区域 位置的价值 • seconds– 示例作业的输入参数值。它必须是正整数 • secretKey – assume-role 命令中Credentials.SecretAccessKey 属性的值 • sessionToken – assume-role 命令中Credentials.SessionToken 属性的值	AWS DevOps

Task	描述	所需技能
测试示例架构。	要测试示例架构，请向作业 API 发送请求。有关更多信息，请参阅 Postman 文档。	DevOps 工程师

故障排除

事务	解决方案
由于 Amazon Lo CloudWatch gs 日志组/aws/apigateway/JobsAPIAccessLogs 已经存在，因此销毁和随后重新部署示例架构会失败。	1. 如有必要，请将您的日志数据导出到 Amazon S3。 2. 删除 CloudWatch 日志日志组/aws/apigateway/JobsAPIAccessLogs 。 3. 重新部署示例架构。

相关资源

- API Gateway 映射模板和访问日志变量参考
- 设置后端 Lambda 函数的异步调用

使用 Amazon API Gateway 和 Amazon DynamoDB Streams 异步处理事件

由 Andrea Meroni (AWS)、Alessandro Trisolini (AWS)、Nadim Majed (AWS)、Mariem Kthiri (AWS) 和迈克尔·沃尔纳 (AWS) 创作

摘要

Amazon API Gateway 是一项完全托管的服务，开发人员可以使用它来创建、发布、维护、监控和保护 APIs 任何规模。它可以处理接受和处理多达数十万个并发 API 调用所涉及的任务。

API Gateway 的一个重要服务配额是集成超时。超时是指在 REST API 返回错误之前，后端服务必须返回响应的最长时间。对于同步工作负载，29 秒的硬限制通常是可以接受的。但是，对于那些想要将 API Gateway 用于异步工作负载的开发者来说，这个限制是一个挑战。

此模式显示了使用 API Gateway、Amazon DynamoDB Streams 和异步处理事件的示例架构。 AWS Lambda该架构支持使用相同的输入参数运行并行处理作业，并且使用基本的 REST API 作为接口。在此示例中，使用 Lambda 作为后端将任务的持续时间限制为 15 分钟。您可以通过使用替代服务来处理传入的事件（例如 AWS Fargate）来规避此限制。

Projen 与 T AWS Cloud Development Kit (AWS CDK) oo lkit、Docker 和 Node.js 结合使用 AWS 账户，用于设置本地开发环境并将示例架构部署到目标。Projen 通过预提交和用于代码质量保证、安全扫描和单元测试的工具自动设置 Python 虚拟环境。有关更多信息，请参阅 “工具” 部分。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 您的工作站上安装了以下工具：
  - AWS Cloud Development Kit (AWS CDK) 工具包版本 2.85.0 或更高版本
  - Docker 版本 20.10.21 或更高版本
  - Node.js 版本 18 或更高版本
  - Projen 版本 0.71.111 或更高版本
  - Python 版本 3.9.16 或更高版本

限制

- 为了避免限制，建议的 DynamoDB Streams 读取器的最大数量为两个。

- 任务的最大运行时间受到 Lambda 函数的最大运行时间（15 分钟）的限制。
- 并发任务请求的最大数量受到 Lambda 函数预留并发性的限制。

架构

架构

下图显示了任务 API 与 DynamoDB Streams 以及事件处理和错误处理 Lambda 函数的交互，事件存储在亚马逊事件档案中。 EventBridge

典型的工作流程包括以下步骤：

1. 您通过 AWS Identity and Access Management (IAM) 进行身份验证并获取安全证书。
2. 您向作/jobs业 API 端点发送 HTTP POST 请求，在请求正文中指定任务参数。
3. 作业 API 会向您返回包含任务标识符的 HTTP 响应。
4. 作业 API 将任务参数放在 jobs_table Amazon DynamoDB 表中。
5. jobs_tableDynamoDB 表 DynamoDB 流调用事件处理 Lambda 函数。
6. 事件处理 Lambda 函数处理事件，然后将任务结果放入 DynamoDB 表中。jobs_table为了帮助确保结果一致，事件处理函数实现了乐观锁定机制。
7. 您向作/jobs/{jobId}业 API 端点发送 HTTP GET 请求，步骤 3 中的任务标识符为{jobId}。
8. 作业 API 查询 jobs_table DynamoDB 表以检索任务结果。
9. 作业 API 会返回包含任务结果的 HTTP 响应。
10. 如果事件处理失败，则事件处理函数的源映射会将事件发送到错误处理亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 主题。
11. 错误处理 SNS 主题异步将事件推送到错误处理函数。
12. 错误处理函数将作业参数放在 DynamoD jobs_table B 表中。

    您可以通过向作业 API 端点发送 HTTP GET 请求来检索/jobs/{jobId}任务参数。
13. 如果错误处理失败，错误处理功能会将事件发送到 Ama EventBridge zon 档案。

    您可以使用重播存档的事件 EventBridge。

工具

Amazon Web Services

- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，可帮助您用代码定义和配置 AWS 云基础设施。
- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如，AWS Lambda 函数、使用 API 目标的 HTTP 调用端点或其他 Amazon Web Services account 中的事件总线。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。

其他工具

- autopep8 会根据 Python 增强提案 (PEP) 8 风格指南自动格式化 Python 代码。
- Bandit 会扫描 Python 代码以查找常见的安全问题。
- C@@ ommitizen 是一个 Git 提交检查器和生成器。CHANGELOG
- cfn-lint 是个傻瓜 AWS CloudFormation
- Checkov 是一种静态代码分析工具，用于检查基础设施即代码 (IaC) 是否存在安全性和合规性错误配置。
- jq 是一个用于解析 JSON 的命令行工具。
- Postman 是一个 API 平台。
- p@@ re-comm it 是一个 Git 挂钩管理器。
- Projen 是一个项目生成器。
- pytest 是一个 Python 框架，用于编写可读的小型测试。

代码存储库

此示例架构代码可以在使用 API Gateway 和 DynamoDB Streams 进行 GitHub 异步处理存储库中找到。

最佳实践

- 此示例架构不包括对已部署基础设施的监控。如果您的用例需要监控，请评估添加 CDK 监控结构或其他监控解决方案。
- 此示例架构使用 IAM 权限来控制对作业 API 的访问权限。任何有权假设的人`JobsAPIInvokeRole`都可以调用作业 API。因此，访问控制机制是二进制的。如果您的用例需要更复杂的授权模型，请使用不同的访问控制机制进行评估。
- 当用户向`/jobs`作业 API 端点发送 HTTP `POST` 请求时，将在两个不同的级别对输入数据进行验证：
  - API Gateway 负责第一个请求的验证。
  - 事件处理函数执行第二个请求。

  当用户向`/jobs/{jobId}`作业 API 端点发出 HTTP `GET` 请求时，不会执行任何验证。如果您的用例需要额外的输入验证和更高的安全级别，请评估使用 AWS WAF 来保护您的 API。
- 为避免限制，DynamoDB Streams 文档不鼓励用户使用两个以上的使用者阅读来自同一个直播分片的分片。为了扩大消费者数量，我们建议使用亚马逊 Kinesis Data Stre ams。
- 本示例中使用了@@ 乐观锁定来确保 `jobs_table` DynamoDB 表中项目的一致更新。根据用例要求，您可能需要实现更可靠的锁定机制，例如悲观锁定。

操作说明

设置环境

Task	描述	所需技能
克隆存储库。	要在本地克隆存储库，请运行以下命令： `git clone https://github.com/aws-samples/asynchronous-event-processing-api-gateway-dynamodb-streams-cdk.git`	DevOps 工程师

Task	描述	所需技能

设置项目。	将目录更改为存储库根目录，然后使用 P rojen 设置 Python 虚拟环境和所有工具：
	`cd asynchronous-event-processing-api-gateway-api-gateway-dynamodb-streams-cdk`
	`npx projen`
		DevOps 工程师

安装预提交挂钩。	要安装预提交挂钩，请执行以下操作：
	1. 激活 P ython 虚拟环境：
	`source .env/bin/activate`
	2. 安装预提交挂钩：
	`pre-commit install`
	`pre-commit install --hook-type commit-msg`
		DevOps 工程师

部署示例架构

Task	描述	所需技能
Bootstrap AWS CDK。	要AWS CDK在中进行引导 AWS 账户，请运行以下命令： `AWS_PROFILE=$YOUR_AWS_PROFILE npx projen bootstrap`	AWS DevOps

Task	描述	所需技能
部署示例架构。	要在中部署示例架构 AWS 账户，请运行以下命令： `AWS_PROFILE=$YOUR_AWS_PROFILE npx projen deploy`	AWS DevOps

测试架构

Task	描述	所需技能
安装测试先决条件。	在你的工作站上安装 AWS Command Line Interface (AWS CLI)、Postman 和 j q。 建议使用 Postman 来测试此示例架构，但这不是强制性的。如果您选择替代的 API 测试工具，请确保它支持 AWS Signature 版本 4 身份验证，并参考可通过导出 REST API 检查的公开的 API 终端节点。	DevOps 工程师
假设`JobsAPIInvokeRole` 。	假设`JobsAPIInvokeRole` 那是作为`deploy`命令的输出打印出来的： `CREDENTIALS=$(AWS_PROFILE=$<YOUR_AWS_PROFILE> aws sts assume-role \ --no-cli-pager \ --role-arn $<JOBS_API_INVOKE_ROLE_ARN> \`	AWS DevOps

Task	描述	所需技能
	--role-session-name JobsAPIInvoke) export AWS_ACCESS_KEY_ID=$(cat $CREDENTIALS \| jq '.Credentials''.AccessKeyId') export AWS_SECRET_ACCESS_KEY=$(cat $CREDENTIALS \| jq '.Credentials''.SecretAccessKey') export AWS_SESSION_TOKEN==$(cat $CREDENTIALS | jq '.Credentials''.SessionToken')	

Task	描述	所需技能

配置 Postman。	• 要导入存储库中包含的 Postman 集合，请按照 Postman 文档中的说明进行操作。
	• 使用以下值设置JobsAPI变量：
	  • accessKey – assume-role 命令中Credentials.AccessKeyId 属性的值。
	  • baseUrl– deploy 命令JobsApiJobsAPIEndpoint 输出的值，不带尾部斜杠。
	  • region– 您在 AWS 区域 何处部署示例架构的价值。
	  • seconds– 示例作业的输入参数值。它必须是正整数。
	  • secretKey – assume-role 命令中Credentials.Secret AccessKey 属性的值。
	  • sessionToken – assume-role 命令中Credentials.SessionToken 属性的值。
		AWS DevOps

测试示例架构。	要测试示例架构，请向作业 API 发送请求。有关更多信息，请参阅 Postman 文档。	DevOps 工程师

故障排除

事务	解决方案
由于 Amazon Lo CloudWatch gs 日志组/aws/apigateway/JobsAPIAccessLogs 已经存在，因此销毁和随后重新部署示例架构会失败。	1. 如有必要，请将您的日志数据导出到亚马逊简单存储服务 (Amazon S3)。 2. 删除 CloudWatch 日志日志组/aws/apigateway/JobsAPIAccessLogs 。 3. 重新部署示例架构。

相关资源

- API Gateway 映射模板和访问日志变量参考
- 更改 DynamoDB Streams 的数据采集
- 使用版本号进行乐观锁定
- 使用 Kinesis Data Streams 捕获 DynamoDB 的更改

使用 Amazon API Gateway、Amazon SQS 和 AWS Fargate 异步处理事件

由 Andrea Meroni (AWS)、Alessandro Trisolini (AWS)、Nadim Majed (AWS)、Mariem Kthiri (AWS) 和迈克尔·沃尔纳 (AWS) 创作

摘要

Amazon API Gateway 是一项完全托管的服务，开发人员可以使用它来创建、发布、维护、监控和保护 APIs 任何规模。它可以处理接受和处理多达数十万个并发 API 调用所涉及的任务。

API Gateway 的一个重要服务配额是集成超时。超时是指在 REST API 返回错误之前，后端服务必须返回响应的最长时间。对于同步工作负载，29 秒的硬限制通常是可以接受的。但是，对于那些想要将 API Gateway 用于异步工作负载的开发者来说，这个限制是一个挑战。

此模式显示了使用 API Gateway、Amazon Simple Queue Service 和（亚马逊 SQS）异步处理事件的架构示例。 AWS Fargate该架构支持在没有持续时间限制的情况下运行处理作业，并且使用基本的 REST API 作为接口。

Projen 与、Docker 和 Node.js 结合使用 AWS 账户，用于设置本地开发环境并将示例架构部署到目标。AWS Cloud Development Kit (AWS CDK)Projen 通过预提交和用于代码质量保证、安全扫描和单元测试的工具自动设置 Python 虚拟环境。有关更多信息，请参阅 “工具” 部分。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 您的工作站上安装了以下工具：
  - AWS Cloud Development Kit (AWS CDK) 工具包版本 2.85.0 或更高版本
  - Docker 版本 20.10.21 或更高版本
  - Node.js 版本 18 或更高版本
  - Projen 版本 0.71.111 或更高版本
  - Python 版本 3.9.16 或更高版本

限制

- 并发任务限制为每分钟 500 个任务，这是 Fargate 可以配置的最大任务数。

架构

下图显示了作业 API 与 jobs Amazon DynamoDB 表、事件处理 Fargate 服务和错误处理函数的交互。 AWS Lambda 事件存储在 Amazon EventBridge 事件档案中。

典型的工作流程包括以下步骤：

1. 您通过 AWS Identity and Access Management (IAM) 进行身份验证并获取安全证书。
2. 您向作/jobs业 API 端点发送 HTTP POST 请求，在请求正文中指定任务参数。
3. 作业 API 是一个 API Gateway REST API，它会向你返回一个包含任务标识符的 HTTP 响应。
4. 作业 API 向 SQS 队列发送一条消息。
5. Fargate 从 SQS 队列中提取消息，处理事件，然后将任务结果放入 DynamoDB 表中。jobs
6. 您向作/jobs/{jobId}业 API 端点发送 HTTP GET 请求，第 3 步中的任务标识符为{jobId}。
7. 作业 API 查询 jobs DynamoDB 表以检索任务结果。
8. 作业 API 会返回包含任务结果的 HTTP 响应。
9. 如果事件处理失败，SQS 队列会将事件发送到死信队列 (DLQ)。
10. EventBridge 事件启动错误处理函数。
11. 错误处理函数将作业参数放在 DynamoD jobs B 表中。
12. 您可以通过向作业 API 端点发送 HTTP GET 请求来检索/jobs/{jobId}任务参数。
13. 如果错误处理失败，则错误处理函数会将事件发送 EventBridge 到存档。

    您可以使用重播存档的事件 EventBridge。

工具

Amazon Web Services

- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，可帮助您在代码中定义和配置 AWS Cloud 基础架构。
- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- AWS Fargate无需管理服务器或亚马逊弹性计算云 (Amazon EC2) 实例，即可帮助您运行容器。它与 Amazon Elastic Container Service（Amazon ECS）配合使用。

- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如，Lambda 函数、使用 API 目标的 HTTP 调用终端节点或其他中的事件总线。AWS 账户
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Queue Service (Amazon SQS) 提供了一个安全、持久且可用的托管队列，它可帮助您集成和分离分布式软件系统与组件。

其他工具

- autopep8 会根据 Python 增强提案 (PEP) 8 风格指南自动格式化 Python 代码。
- Bandit 会扫描 Python 代码以查找常见的安全问题。
- C@@ ommitizen 是一个 Git 提交检查器和生成器。CHANGELOG
- cfn-lint 是个傻瓜 AWS CloudFormation
- Checkov 是一种静态代码分析工具，用于检查基础设施即代码 (IaC) 是否存在安全性和合规性错误配置。
- jq 是一个用于解析 JSON 的命令行工具。
- Postman 是一个 API 平台。
- p@@ re-comm it 是一个 Git 挂钩管理器。
- Projen 是一个项目生成器。
- pytest 是一个 Python 框架，用于编写可读的小型测试。

代码存储库

此示例架构代码可以在使用 API Gateway 和 SQS 进行 GitHub 异步处理存储库中找到。

最佳实践

- 此示例架构不包括对已部署基础设施的监控。如果您的用例需要监控，请评估添加 CDK 监控结构或其他监控解决方案。
- 此示例架构使用 IAM 权限来控制对作业 API 的访问权限。任何有权假设的人JobsAPIInvokeRole都可以调用作业 API。因此，访问控制机制是二进制的。如果您的用例需要更复杂的授权模型，请使用不同的访问控制机制进行评估。
- 当用户向/jobs作业 API 端点发送 HTTP POST 请求时，将在两个不同的级别对输入数据进行验证：

- API Gateway 负责第一个请求的验证。
- 事件处理函数执行第二个请求。

  当用户向/jobs/{jobId}作业 API 端点发出 HTTP GET 请求时，不会执行任何验证。如果您的用例需要额外的输入验证和更高的安全级别，请评估使用 AWS WAF 来保护您的 API。

操作说明

设置环境

Task	描述	所需技能
克隆存储库。	要在本地克隆存储库，请运行以下命令： `git clone https://github.com/aws-samples/asynchronous-event-processing-api-gateway-sqs-cdk.git`	DevOps 工程师

设置项目。	将目录更改为存储库根目录，然后使用 P rojen 设置 Python 虚拟环境和所有工具：
	`cd asynchronous-event-processing-api-gateway-api-gateway-sqs-cdk`
	`npx projen`
		DevOps 工程师

安装预提交挂钩。	要安装预提交挂钩，请执行以下操作： 1. 激活 P ython 虚拟环境： `source .env/bin/activate`	DevOps 工程师

Task	描述	所需技能

	2. 安装预提交挂钩：
	`pre-commit install`
	`pre-commit install --hook-type commit-msg`	

部署示例架构

Task	描述	所需技能
Bootstrap AWS CDK。	要AWS CDK在中进行引导 AWS 账户，请运行以下命令： `AWS_PROFILE=$YOUR_AWS_PROFILE npx projen bootstrap`	AWS DevOps
部署示例架构。	要在中部署示例架构 AWS 账户，请运行以下命令： `AWS_PROFILE=$YOUR_AWS_PROFILE npx projen deploy`	AWS DevOps

测试架构

Task	描述	所需技能
安装测试先决条件。	在你的工作站上安装 AWS Command Line Interface (AWS CLI)、Postman 和 j q。 建议使用 Postman 来测试此示例架构，但这不是强制性的。	DevOps 工程师

Task	描述	所需技能
	如果您选择其他 API 测试工具，请确保它支持AWS 签名版本 4 身份验证，并参考可通过导出 REST API 来检查的公开的 API 端点。	
假设JobsAPIInvokeRole .	假设JobsAPIInvokeRole 那是作为deploy命令的输出打印出来的： `CREDENTIALS=$(AWS_PROFILE=$<YOUR_AWS_PROFILE> aws sts assume-role \ --no-cli-pager \ --role-arn $<JOBS_API_INVOKE_ROLE_ARN> \ --role-session-name JobsAPIInvoke) export AWS_ACCESS_KEY_ID=$(cat $CREDENTIALS \| jq '.Credentials''.AccessKeyId') export AWS_SECRET_ACCESS_KEY=$(cat $CREDENTIALS \| jq '.Credentials''.SecretAccessKey') export AWS_SESSION_TOKEN==$(cat $CREDENTIALS | jq '.Credentials''.SessionToken')`	AWS DevOps

Task	描述	所需技能

配置 Postman。	• 要导入存储库中包含的 Postman 集合，请按照 Postman 文档中的说明进行操作。
	• 使用以下值设置JobsAPI变量：
	  • accessKey – assume-role 命令中Credentials.AccessKeyId 属性的值。
	  • baseUrl– deploy 命令JobsApiJobsAPIEndpoint 输出的值，不带尾部斜杠。
	  • region– 您在 AWS 区域 何处部署示例架构的价值。
	  • seconds– 示例作业的输入参数值。它必须是正整数。
	  • secretKey – assume-role 命令中Credentials.Secret AccessKey 属性的值。
	  • sessionToken – assume-role 命令中Credentials.SessionToken 属性的值。
		AWS DevOps

测试示例架构。	要测试示例架构，请向作业 API 发送请求。有关更多信息，请参阅 Postman 文档。	DevOps 工程师

故障排除

事务	解决方案
由于 Amazon Lo CloudWatch gs 日志组/aws/apigateway/JobsAPIAccessLogs 已经存在，因此销毁和随后重新部署示例架构会失败。	1. 如有必要，请将您的日志数据导出到亚马逊简单存储服务 (Amazon S3) S ervice。 2. 删除 CloudWatch 日志日志组/aws/apigateway/JobsAPIAccessLogs 。 3. 重新部署示例架构。
由于CloudWatch 日志日志组/aws/ecs/EventProcessingServiceLogs 已经存在，因此销毁和随后重新部署示例架构会失败。	1. 如有必要，请将您的日志数据导出到 Amazon S3。 2. 删除 CloudWatch 日志日志组 /aws/ecs/EventProcessingServiceLogs. 3. 重新部署示例架构。

相关资源

- API Gateway 映射模板和访问日志变量参考
- 如何将 API Gateway REST API 与亚马逊 SQS 集成，并解决常见错误？

从 AWS Step Functions 同步运行 AWS Systems Manager Automation 任务

创建者：Elie El khoury（AWS）

摘要

此模式说明了如何 AWS Step Functions 与集成 AWS Systems Manager。它使用 AWS SDK 服务集成，使用状态机工作流程中的任务令牌调用 Systems Manager startAutomationExecutionAPI，然后暂停直到调用成功或失败时令牌返回。为了演示集成，此模式在或文档周围实现了一个自动化文档 (runbook) 包装器，并使用它`.waitForTaskToken`来同步调用`AWS-RunShellScript`或`AWS-RunPowerShellScript`。`AWS-RunShellScript` `AWS-RunPowerShellScript`有关 Step Functions 中软件开发 AWS 工具包服务集成的更多信息，请参阅AWS Step Functions 开发者指南。

Step Function s 是一项低代码的可视化工作流服务，您可以使用它来构建分布式应用程序、自动执行 IT 和业务流程，以及使用 AWS 服务构建数据和机器学习管道。工作流程可以管理故障、重试、并行化、服务集成和可观测性，因此您可以专注于更高价值的业务逻辑。

自动化是一项功能，可简化亚马逊弹性计算云（亚马逊）、亚马逊关系数据库服务（Amazon RDS EC2）、Amazon Redshift和亚马逊简单存储服务（Amazon S3） AWS 服务 等的常见维护、部署和补救任务。 AWS Systems Manager自动化使您可以精确控制自动化的并发性。例如，您可以指定同时定位多少资源，以及在停止自动化之前可能发生的错误数。

有关实施的详细信息，包括运行手册步骤、参数和示例，请参阅其他信息部分。

先决条件和限制

先决条件

- 一个活跃的 AWS 账户
- AWS Identity and Access Management (IAM) 访问 Step Functions 和 Systems Manager 的权限
- EC2 实例上安装了 Systems Manager 代理（SSM 代理）
- Systems Manager 的 IAM 实例配置文件附加到您计划运行运行手册的实例上
- 具有以下 IAM 权限（遵循最低权限原则）的 Step Functions 角色：

```
{
            "Effect": "Allow",
            "Action": "ssm:StartAutomationExecution",
            "Resource": "*"
 }
```

产品版本

- SSM 文档架构版本 0.3 或更高版本
- SSM Agent 版本 2.3.672.0 或更高版本

架构

目标技术堆栈

- AWS Step Functions
- AWS Systems Manager 自动化

目标架构

自动化和扩缩

- 此模式提供了一个 AWS CloudFormation 模板，您可以使用该模板在多个实例上部署运行手册。（请参阅 GitHub Step Functions 和 Systems Manager 实现存储库。）

工具

AWS 服务

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在资源的整个生命周期中跨地区对其 AWS 账户 进行管理。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Step Functions是一项无服务器编排服务，可帮助您组合 AWS Lambda 功能和其他功能 AWS 服务 来构建关键业务应用程序。
- AWS Systems Manager 可帮助您管理在 AWS Cloud中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。

代码

此模式的代码可在 GitHub Step Functions 和 Systems Manager 实现存储库中找到。

操作说明

创建运行手册

Task	描述	所需技能
下载 CloudFormation 模板。	从 GitHub 存储库的`cloudformation` 文件夹下载`ssm-automation-documents.cfn.json` 模板。	AWS DevOps
创建运行手册。	登录 AWS Management Console，打开AWS CloudFormation 控制台，然后部署模板。有关部署 CloudFormation 模板的更多信息，请参阅 CloudFormation 文档中的在AWS CloudFormation 控制台上创建堆栈。 该 CloudFormation 模板部署了三种资源： • `SfnRunCommandByInstanceIds` — 运行手册允许你运行`AWS-RunShellScript` 或使用`AWS-RunPowerShellScript` 实例 IDs。 • `SfnRunCommandByTargets` — 运行手册允许你跑`AWS-RunPowerShellScript` 步`AWS-RunShellScript` 或使用目标。 • `SSMSyncRole` — 运行手册承担的 IAM 角色。	AWS DevOps

创建示例状态机

Task	描述	所需技能
创建测试状态机。	按照《AWS Step Functions 开发者指南》中的说明创建和运行状态机。对于定义，请使用以下代码。请务必使用您账户中启用系统管理器的有效实例的 ID 更新该 `InstanceIds` 值。 `{ "Comment": "A description of my state machine", "StartAt": "StartAutomationWaitForCallBack", "States": { "StartAutomationWaitForCallBack": { "Type": "Task", "Resource": "arn:aws:states:::aws-sdk:ssm:startAutomationExecution.waitForTaskToken", "Parameters": { "DocumentName": "SfnRunCommandByInstanceIds", "Parameters": { "InstanceIds": ["i-1234567890abcdef0"], "taskToken.$": "States.Array($$.Task.Token)",`	AWS DevOps

Task	描述	所需技能
	"workingD irectory": ["/home/ssm- user/"], "Commands": ["echo \"This is a test running automation waitForTa skToken\" >> automatio n.log", "sleep 100"], "executio nTimeout": ["10800"], "delivery Timeout": ["30"], "shell": ["Shell"] } }, "End": true } } } 此代码调用运行手册来运行两个命令来演示对 Systems Manager Automation 的 `waitForTaskToken` 调用。 `shell`参数值（`Shell`或`PowerShell`）决定自动化文档是运行`AWS-RunShellScript` 还	

Task	描述	所需技能
	是AWS-RunPowerShellScript 。 该任务将 “这是测试运行自动化 waitForTask令牌” 写入/home/ssm-user/automation.log 文件，然后休眠 100 秒，然后使用任务令牌进行响应并释放工作流程中的下一个任务。 如果要改为调用 SfnRunCommandByTargets 运行手册，请将前述代码的 Parameters 部分替换为以下内容： `"Parameters": { "Targets": [ { "Key": "InstanceIds", "Values": [ "i-02573cafcfEXAMPLE", "i-0471e04240EXAMPLE" ] } ],`	

Task	描述	所需技能
更新状态机的 IAM 角色。	上一步会自动为状态机创建专用 IAM 角色。但是，它不授予调用运行手册的权限。通过添加以下权限来更新角色： `{ "Effect": "Allow", "Action": "ssm:StartAutomationExecution", "Resource": "*" }`	AWS DevOps
验证同步调用。	运行状态机以验证 Step Functions 和 Systems Manager Automation 之间的同步调用。 有关示例输出，请参阅其他信息部分。	AWS DevOps

相关资源

- 入门 AWS Step Functions（AWS Step Functions 开发者指南）
- 等待带有任务令牌的回调（AWS Step Functions 开发者指南，服务集成模式）
- send_task_success 和 send_task_failure API 调用（Boto3 文档）
- AWS Systems Manager 自动化（AWS Systems Manager 用户指南）

其他信息

实施详情

此模式提供了一个部署两个 Systems Manager 运行手册的 CloudFormation 模板：

- SfnRunCommandByInstanceIds使用实例运行AWS-RunShellScript或AWS-RunPowerShellScript命令 IDs。
- SfnRunCommandByTargets使用目标运行AWS-RunShellScript或AWS-RunPowerShellScript命令。

使用 Step Functions 中的.waitForTaskToken选项时，每个 runbook 都实现了四个步骤来实现同步调用。

步骤	操作	描述
1	Branch	检查shell参数值（Shell或PowerShell ），以决定是在 Linux 上运行AWS-RunShellScript 还是在 Window AWS-RunPowerShellScript s 上运行。
2	RunCommand_Shell 或RunCommand_PowerShell	接受多个输入并运行RunShellScript 或RunPowerShellScript 命令。有关更多信息，请查看 Systems Manager 控制台上RunCommand_Shell 或RunCommand_PowerShell 自动化文档的详细信息选项卡。
3	SendTaskFailure	在步骤 2 中止或取消时运行。它调用 Step Functions send_task_failure API，该 API 接受三个参数作为输入：状态机传递的令牌、失败错误和对失败原因的描述。
4	SendTaskSuccess	在步骤 2 成功时运行。它调用 Step Functions send_task

_success API，该 API 接受状态机传递的令牌作为输入。

运行手册参数

SfnRunCommandByInstanceIds运行手册：

参数名称	类型	必需或可选	描述
shell	字符串	必需	实例外壳AWS-RunShellScript 用于决定是在 Linux 上运行还是在 Window AWS-RunPowerShellScript s 上运行。
deliveryTimeout	整数	可选	等待命令传送到实例上的 SSM 代理的时间（以秒为单位）。此参数的最小值为 30（0.5 分钟），最大值为 2592000（720 小时）。
executionTimeout	字符串	可选	在被视为已失败前命令将运行的时间（单位：秒）。默认值为 3600（1 小时）。最长值为 172800（48 小时）。
workingDirectory	字符串	可选	实例上工作目录的路径。
Commands	StringList	必需	要运行的 Shell 脚本或命令。

InstanceIds	StringList	必需	您要在其中运行命令的实例。 IDs
taskToken	字符串	必需	用于回调响应的任务令牌。

SfnRunCommandByTargets运行手册：

名称	类型	必需或可选	描述
shell	字符串	必需	实例外壳AWS-RunShellScript 用于决定是在 Linux 上运行还是在 Window AWS-RunPowerShellScript s 上运行。
deliveryTimeout	整数	可选	等待命令传送到实例上的 SSM 代理的时间（以秒为单位）。此参数的最小值为 30（0.5 分钟），最大值为 2592000（720 小时）。
executionTimeout	整数	可选	在被视为已失败前命令将运行的时间（单位：秒）。默认值为 3600（1 小时）。最长值为 172800（48 小时）。
workingDirectory	字符串	可选	实例上工作目录的路径。
Commands	StringList	必需	要运行的 Shell 脚本或命令。

Targets	MapList	必需	一组搜索条件，使用您指定的键值对来识别实例。例如：[{"Key":"InstanceIds","Values":["i-02573cafcfEXAMPLE","i-0471e04240EXAMPLE"]}]
taskToken	字符串	必需	用于回调响应的任务令牌。

示例输出

下表提供了 Step 函数的示例输出。它显示步骤 5（TaskSubmitted）和步骤 6（TaskSucceeded）之间的总运行时间超过 100 秒。这表明步骤函数等待sleep 100命令完成后才移至工作流程中的下一个任务。

ID	类型	步骤	资源	运行时间 (ms)	Timestamp
1	ExecutionStarted		-	0	2022 年 3 月 11 日下午 02:50:34.303
2	TaskStateEntered	StartAutomationWaitForCallBack	-	40	2022 年 3 月 11 日下午 02:50:34.343
3	TaskScheduled	StartAutomationWaitForCallBack	-	40	2022 年 3 月 11 日下午 02:50:34.343

4	TaskStarted	StartAutomationWaitForCallBack	-	154	2022 年 3 月 11 日下午 02:50:34.457
5	TaskSubmitted	StartAutomationWaitForCallBack	-	657	2022 年 3 月 11 日下午 02:50:34.960
6	TaskSucceeded	StartAutomationWaitForCallBack	-	103835	2022 年 3 月 11 日下午 02:52:18.138
7	TaskStateExited	StartAutomationWaitForCallBack	-	103860	2022 年 3 月 11 日下午 02:52:18.163
8	ExecutionSucceeded		-	103897	2022 年 3 月 11 日下午 02:52:18.200

在 AWS Lambda 函数中使用 Python 并行读取 S3 对象

由 Eduardo Bortoluzzi (AWS) 创作

摘要

您可以使用此模式实时检索和汇总亚马逊简单存储服务 (Amazon S3) 存储桶中的文档列表。该模式提供了在 Amazon Web Services (AWS) 上并行读取 S3 存储桶中的对象的示例代码。该模式展示了如何使用 Python 使用 AWS Lambda 函数高效运行 I/O 绑定任务。

一家金融公司在交互式解决方案中使用这种模式来实时手动批准或拒绝相关的金融交易。金融交易文件存储在与市场相关的S3存储桶中。操作员从 S3 存储桶中选择了一系列文档，分析了解决方案计算的交易总价值，并决定批准或拒绝所选批次。

I/O 绑定任务支持多个线程。在此示例代码中，并行.futu res。 ThreadPoolExecutor可与 30 个并发线程一起使用，尽管 Lambda 函数最多支持 1,024 个线程（其中一个线程是您的主进程）。此限制是因为由于上下文切换和计算资源使用率过高，线程过多会造成延迟问题。您还需要增加最大池连接数，botocore以便所有线程都能同时执行 S3 对象下载。

示例代码在 S3 存储桶中使用一个包含 JSON 数据的 8.3 KB 对象。该对象被读取多次。在 Lambda 函数读取对象后，JSON 数据会被解码为 Python 对象。2024 年 12 月，运行此示例后的结果是，使用配置了 2,304 MB 内存的 Lambda 函数在 2.3 秒内处理了 1,000 次读取，在 27 秒内处理了 10,000 次读取。 AWS Lambda 支持从 128 MB 到 10,240 MB (10 GB) 的内存配置，但将 Lambdamemory 增加到 2,304 MB 以上无助于缩短运行此特定 I/O 绑定任务的时间。

AWS Lambda 电源调整工具用于测试不同的 Lambda 内存配置并验证任务的最佳 performance-to-cost 比率。有关测试结果，请参阅 “其他信息” 部分。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 熟练掌握 Python 开发

限制

- 一个 Lambda 函数最多可以有 1,024 个执行进程或线程。
- 新 AWS 账户 的 Lambda 内存限制为 3,008 MB。相应地调整 AWS Lambda 电源调整工具。有关更多信息，请参阅 “故障排除” 部分。

- Amazon S3 对每个分区前缀的限制为每秒 5,500 个 GET/HEAD 请求。

产品版本

- Python 3.9 或更高版本
- AWS Cloud Development Kit (AWS CDK) v2
- AWS Command Line Interface (AWS CLI) 版本 2
- AWS Lambda 功率调整 4.3.6（可选）

架构

目标技术堆栈

- AWS Lambda
- Amazon S3
- AWS Step Functions （如果部署了 AWS Lambda 电源调节）

目标架构

下图显示了一个 Lambda 函数，该函数可并行读取 S3 存储桶中的对象。该图还具有用于 AWS Lambda 功率调整工具的 Step Functions 工作流程，用于微调 Lambda 函数内存。这种微调有助于在成本和性能之间取得良好的平衡。

自动化和扩缩

Lambda 函数可在需要时快速扩展。为了避免在需求旺盛时出现 Amazon S3 的 503 个减速错误，我们建议对扩展设置一些限制。

工具

Amazon Web Services

- AWS Cloud Development Kit (AWS CDK) v2 是一个软件开发框架，可帮助您在代码中定义和配置 AWS Cloud 基础架构。创建示例基础架构是为了与之一起部署 AWS CDK。
- AWS Command Line InterfaceAWS CLI是一款开源工具，可帮助您 AWS 服务 通过命令行 shell 中的命令进行交互。在此模式中， AWS CLI 版本 2 用于上传示例 JSON 文件。

- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- 亚马逊简单存储服务 Amazon S3 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Step Functions是一项无服务器编排服务，可帮助您组合 AWS Lambda 功能和其他 AWS 服务来构建业务关键型应用程序。

其他工具

- Python 是通用的计算机编程语言。Python 版本 3.8 中引入了空闲工作线程的重用，此模式中的 Lambda 函数代码是为 Python 3.9 及更高版本创建的。

代码存储库

此模式的代码可在aws-lambda-parallel-download GitHub 存储库中找到。

最佳实践

- 此 AWS CDK 构造依赖于您的 AWS 账户用户权限来部署基础架构。如果您计划使用 Pipelin AWS CDK es 或跨账户部署，请参阅堆栈合成器。
- 此示例应用程序未在 S3 存储桶中启用访问日志。在生产代码中启用访问日志是一种最佳实践。

操作说明

准备开发环境

Task	描述	所需技能
检查已安装的 Python 版本。	这段代码专门在 Python 3.9 和 Python 3.13 上进行了测试，它应该可以在这两个版本之间的所有版本上运行。要检查您的 Python 版本，请在终端`python3 -V`中运行，并在需要时安装更新的版本。	云架构师

Task	描述	所需技能
	要验证是否已安装所需的模块，请运行`python3 -c "import pip, venv"`。没有错误消息表示模块已正确安装并且您已准备好运行此示例。	
安装 AWS CDK。	要安装（ AWS CDK 如果尚未安装），请按照入门中的说明进行操作 AWS CDK。要确认安装的 AWS CDK 版本是否为 2.0 或更高版本，请运行`cdk --version`。	云架构师
引导您的 环境。	要引导您的环境，如果尚未启动您的环境，请按照 Bootstrap 中的说明使用您的环境。 AWS CDK	云架构师

克隆示例存储库

Task	描述	所需技能
克隆存储库。	要克隆最新版本的存储库，请运行以下命令： `git clone --depth 1 --branch v1.2.0 \ git@github.com:aws-samples/aws-lambda-parallel-download.git`	云架构师
将工作目录更改为克隆的存储库。	运行以下命令：	云架构师

Task	描述	所需技能
	`cd aws-lambda-parallel-download`	
创建 Python 虚拟环境。	要创建 Python 虚拟环境，请运行以下命令： `python3 -m venv .venv`	云架构师
激活虚拟环境。	要激活虚拟环境，请运行以下命令： `source .venv/bin/activate`	云架构师
安装依赖项。	要安装 Python 依赖项，请运行以下`pip`命令： `pip install -r requirements.txt`	云架构师
浏览代码。	（可选）从 S3 存储桶下载对象的示例代码位于`resources/parallel.py` 。 基础架构代码位于该 `parallel_download` 文件夹中。	云架构师

部署和测试应用程序

Task	描述	所需技能
部署应用程序。	运行 `cdk deploy`。 写下 AWS CDK 输出：	云架构师

Task	描述	所需技能
	• ParallelDownloadStack.LambdaFunctionARN • ParallelDownloadStack.SampleS3BucketName • ParallelDownloadStack.StateMachineARN	
上传一个示例 JSON 文件。	存储库包含一个大约 9 KB 的 JSON 文件示例。要将文件上传到已创建堆栈的 S3 存储桶，请运行以下命令： `aws s3 cp sample.json s3://<ParallelDownloadStack.SampleS3BucketName>` <ParallelDownloadStack.SampleS3BucketName> 替换为 AWS CDK 输出中的相应值。	云架构师

Task	描述	所需技能
运行该应用程序。	要运行该应用程序，请执行以下操作： 1. 登录 AWS Management Console，导航到 Lambda 控制台，然后找到输出中包含了 ARN 的 Lambda 函数。 AWS CDK `ParallelDownloadStack.LambdaFunction` ARN 2. 在测试选项卡上，将事件 JSON 更改为以下内容： `{"objectKey": "sample.json"}` 3. 选择测试。 4. 要查看结果，请选择详细信息。详细信息将显示并行下载的统计信息、运行信息和日志。	云架构师
添加下载次数。	（可选）要运行 1,500 次获取对象调用，请在`Test`参数的事件 JSON 中使用以下 J SON： `{"repeat": 1500, "objectKey": "sample.json"}`	云架构师

可选：运行 AWS Lambda 功率调整

Task	描述	所需技能
运行 AWS Lambda 电源调整工具。	1. 登录控制台，然后导航到 Ste p Functions。 2. 使用输出中的 ARN 找到状态机。 AWS CDK `ParallelDownloadStack.StateMachineARN` 3. 选择 “开始执行”，然后粘贴以下 JSON： `{ "lambdaARN": "<ParallelDownloadStack.LambdaFunctionARN>", "num": 10, "strategy": "balanced", "payload": {"repeat": 2000, "objectKey": "sample.json"} }` 请记住`<ParallelDownloadStack.LambdaFunctionARN>` 用 AWS CDK 输出中的值进行替换。 运行结束时，结果将显示在 “执行输入和输出” 选项卡上。	云架构师

Task	描述	所需技能
在图表中查看 AWS Lambda 功率调整结果。	在 “执行输入和输出” 选项卡上，复制visualization 属性链接，然后将其粘贴到新的浏览器选项卡中。	云架构师

清理

Task	描述	所需技能
从 S3 存储桶中移除对象。	在销毁已部署的资源之前，请从 S3 存储桶中移除所有对象： `aws s3 rm s3://<ParallelDownloadStack.SampleS3BucketName> \ --recursive` 请记住<ParallelDownloadStack.SampleS3BucketName> 用 AWS CDK 输出中的值进行替换。	云架构师
摧毁资源。	要销毁为此试点创建的所有资源，请运行以下命令： `cdk destroy`	云架构师

故障排除

事务	解决方案
`'MemorySize' value failed to satisfy constraint: Member must have value less than or equal to 3008`	对于新账户，您的 Lambda 函数中可能无法配置超过 3,008 MB 的空间。要使用 AWS Lambda Power Tuning 进行测试，请在开始执行 Step Functions 时在输入 JSON 处添加以下属性： `"powerValues": [ 512, 1024, 1536, 2048, 2560, 3008 ]`

相关资源

- Python — 并行.futures。 ThreadPoolExecutor
- Lambda 配额 — 函数配置、部署和执行
- AWS CDK 在 Python 中使用
- 使用 AWS Lambda 功率调整功能进行性能分析

其他信息

代码

以下代码片段执行并行 I/O 处理：

```
with ThreadPoolExecutor(max_workers=MAX_WORKERS) as executor:
  for result in executor.map(a_function, (the_arguments)):
    ...
```

当线程可用时，会ThreadPoolExecutor重复使用它们。

测试和结果

这些测试是在2024年12月进行的。

第一次测试处理了 2,500 次对象读取，结果如下。

从 3,009 MB 开始，任何内存增加的处理时间都几乎保持不变，但是随着内存大小的增加，成本也随之增加。

另一项测试使用了256 MB的倍数并处理了10,000个对象读取的值，调查了1,536 MB到3,072 MB之间的内存范围，结果如下。

最佳 performance-to-cost比例是 2,304 MB 内存 Lambda 配置。

相比之下，读取 2,500 个对象的顺序过程花了 47 秒。使用 2,304 MB Lambda 配置的并行处理花了 7 秒钟，减少了 85%。

将遥测数据从发送到 AWS Lambda 以 OpenSearch 进行实时分析和可视化

由 Tabby Ward (AWS)、Guy Bachar (AWS) 和 David Kilzer (AWS) 创作

摘要

现代应用程序变得越来越分散和事件驱动，这增强了对实时监控和可观测性的需求。 AWS Lambda 是一种无服务器计算服务，在构建可扩展和事件驱动的架构方面起着至关重要的作用。但是，如果您仅依赖 Amazon Lo CloudWatch gs，Lambda 功能的监控和故障排除可能会很困难，这可能会带来延迟和有限的保留期。

为了应对这一挑战， AWS 推出了 Lambda Telemetry API，它使 Lambda 函数能够将遥测数据直接发送到第三方监控和可观察性工具。此 API 支持实时流式传输日志、指标和跟踪，并可全面及时地了解 Lambda 函数的性能和运行状况。

此模式解释了如何将 Lambda Telemetry API 与OpenSearch开源分布式搜索和分析引擎集成。 OpenSearch 为摄取、存储和分析大量数据提供了一个强大且可扩展的平台，这使其成为 Lambda 遥测数据的理想选择。具体而言，此模式演示了如何使用提供的 Lambda 扩展将用 Python 编写的 Lambda 函数中的日志直接发送到 OpenSearch 集群。 AWS该解决方案灵活且可自定义，因此您可以创建自己的 Lambda 扩展或更改示例源代码以根据需要更改输出格式。

该模式说明了如何设置和配置 Lambda Telemetry API 的集成 OpenSearch，并包括安全性、成本优化和可扩展性的最佳实践。目标是帮助您更深入地了解您的 Lambda 函数，并增强无服务器应用程序的整体可观察性。

> **Note**
>
> 这种模式侧重于将 Lambda 遥测 API 与托管集成。 OpenSearch但是，所讨论的原则和技术也适用于自我管理 OpenSearch 和 Elasticsearch。

先决条件和限制

在开始集成过程之前，请确保满足以下先决条件：

AWS 账户： AWS 账户 具有创建和管理以下 AWS 资源的相应权限的活跃用户：

- AWS Lambda
- AWS Identity and Access Management (IAM)

- Amazon OpenSearch 服务（如果您使用的是托管 OpenSearch 集群）

OpenSearch 集群：

- 您可以使用现有的自管理 OpenSearch 集群或托管服务，例如 OpenSearch 服务。
- 如果您使用的是 OpenSearch 服务，请按照服务文档中 Amazon OpenSearch 服务入门中的 OpenSearch 说明设置 OpenSearch 集群。
- 确保可通过您的 Lambda 函数访问 OpenSearch 集群，并配置了必要的安全设置，例如访问策略、加密和身份验证。
- 为 OpenSearch 集群配置必要的索引映射和设置，以提取 Lambda 遥测数据。有关更多信息，请参阅服务文档中的将流数据加载到 Amazon OpenSearch OpenSearch 服务。

网络连接：

- 确保您的 Lambda 函数具有访问集群所需的网络连接。 OpenSearch 有关如何配置虚拟私有云 (VPC) 设置的指导，请参阅 OpenSearch 服务文档中的在 VPC 内启动您的 Amazon OpenSearch 服务域。

IAM 角色和策略：

- 创建一个具有 Lambda 函数访问 OpenSearch 集群和访问存储在中的证书所需权限的 IAM 角色。 AWS Secrets Manager
- 将相应的 IAM 策略附加到该角色，例如`AWSLambdaBasicExecutionRole`策略和与之交互所需的任何其他权限 OpenSearch。
- 验证授予您的 Lambda 函数的 IAM 权限是否允许它向集群写入数据。 OpenSearch 有关管理 IAM 权限的信息，请参阅 Lambda 文档中的使用执行角色定义 Lambda 函数权限。

编程语言知识：

- 要理解和修改 Lambda 函数和 Lambda 扩展的示例代码，您需要具有 Python（或您选择的编程语言）的基本知识。

开发环境：

- 使用构建和部署 Lambda 函数和扩展所需的工具和依赖关系来设置本地开发环境。

AWS CLI 或 AWS Management Console：

- 安装并配置 AWS Command Line Interface (AWS CLI)，或者使用 AWS Management Console 具有相应凭据的，与所需的凭据进行交互 AWS 服务。

监控和记录：

- 熟悉监控和记录最佳实践 AWS，包括诸如 Amazon 之类的服务， CloudWatch 以及 AWS CloudTrail 用于监控和审计目的的实践。
- 查看您的 Lambda 函数的 CloudWatch 日志，找出与 Lambda 遥测 API 集成相关的任何错误或异常。有关疑难解答指南，请参阅 Lambda 遥测 API 文档。

架构

此模式使用 OpenSearch 服务来存储 Lambda 函数生成的日志和遥测数据。这种方法使您能够快速将日志直接流式传输到 OpenSearch 集群，从而减少与使用 CloudWatch 日志作为中介相关的延迟和成本。

> **Note**
>
> 您的 Lambda 扩展代码可以通过直接使用 OpenSearch API 或使用客户端库将遥测推送到 OpenSearch 服务。OpenSearch Lambda 扩展可以使用 OpenSearch API 支持的批量操作将遥测事件批量处理在一起，并在单个请求中将其发送到 OpenSearch 服务。

以下工作流程图说明了使用 OpenSearch 集群作为终端节点时 Lambda 函数的日志工作流程。

该架构包括以下组件：

- Lambda 函数：在执行过程中生成日志和遥测数据的无服务器函数。
- Lambda 扩展：一种基于 Python 的扩展，它使用 Lambda 遥测 API 直接与集群集成。 OpenSearch 在同一个执行环境中，此扩展与 Lambda 函数一起运行。
- Lambda 遥测 API：允许 Lambda 扩展程序将遥测数据（包括日志、指标和跟踪）直接发送到第三方监控和可观察性工具的 API。

- Amazon S OpenSearch ervice OpenSearch 集群：托管在上的托管集群 AWS。该集群负责接收、存储通过 Lambda 扩展从 Lambda 函数流出的日志数据并为其编制索引。

该工作流程包括以下步骤：

1. Lambda 函数被调用，并在其执行期间生成日志和遥测数据。
2. Lambda 扩展与函数一起运行，用于使用 Lambda 遥测 API 捕获日志和遥测数据。
3. Lambda 扩展与 OpenSearch 服务集群建立安全连接并实时流式传输日志数据。
4. OpenSearch 服务集群采集、索引和存储日志数据，以便通过使用 Kibana 或其他兼容应用程序等工具进行搜索、分析和可视化。

通过绕过 CloudWatch 日志并将日志数据直接发送到 OpenSearch 集群，此解决方案提供了以下好处：

- 实时日志流和分析，可加快故障排除速度并提高可观察性。
- 减少了延迟，并限制了与 CloudWatch 日志相关的潜在保留期限。
- 可以灵活地自定义 Lambda 扩展，也可以为特定的输出格式或其他处理创建自己的扩展。
- 与 S OpenSearch ervice 的搜索、分析和可视化功能集成，用于日志分析和监控。

Epics 部分提供了设置 Lambda 扩展、配置 Lambda 函数以及与服务集群集成的 step-by-step说明。OpenSearch 有关安全注意事项、成本优化策略以及监控和排除解决方案故障的提示，请参阅最佳实践部分。

工具

Amazon Web Services

- AWS Lambda 是一项计算服务，可使您无需调配或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。
- Amazon S OpenSearch er vice 是一 AWS 项由提供的完全托管的服务，可以轻松地在云中部署、操作和扩展 OpenSearch 集群。
- Lambda 扩展通过在函数旁边运行自定义代码来扩展您的 Lambda 函数的功能。您可以使用 Lambda 扩展将 Lambda 与各种监控、可观察性、安全和监管工具集成。
- AWS Lambda Telemetry API 使您能够使用扩展程序直接从 Lambda 捕获增强的监控和可观察性数据，并将其发送到您选择的目的地。

- AWS CloudFormation帮助您对 AWS 资源进行建模和设置，这样您就可以花更少的时间管理这些资源，而将更多的时间集中在应用程序上。

代码存储库

- AWS Lambda 扩展包括和 AWS 合作伙伴提供的演示 AWS 和示例项目，可帮助您开始构建自己的扩展。
- Lambda 遥测集成示例提供了 OpenSearch一个示例 Lambda 扩展，该扩展演示了如何将日志从 Lambda 函数发送到集群。 OpenSearch

其他工具

- OpenSearch是一个开源分布式搜索和分析引擎，为摄取、存储和分析大量数据提供了一个强大的平台。
- Kibana 是一款开源数据可视化和探索工具，您可以与之搭配使用。 OpenSearch请注意，可视化和分析的实现超出了这种模式的范围。有关更多信息，请参阅 Kibana 文档和其他资源。

最佳实践

将 Lambda 遥测 API 与集成时 OpenSearch，请考虑以下最佳实践。

安全和访问控制

- 安全通信：使用 HTTPS 加密您的 Lambda 函数与 OpenSearch 集群之间的所有通信。在 Lambda 扩展和配置中配置必要的 SSL/TLS 设置。 OpenSearch
- IAM 权限：
  - 扩展与 Lambda 函数在相同的执行环境中运行，因此它们继承了对文件系统、网络和环境变量等资源的相同访问级别。
  - 向您的 Lambda 函数授予访问 Lambda 遥测 API 和向集群写入数据所需的最低限度 IAM 权限。 OpenSearch 使用最小权限原则来限制权限范围。
- OpenSearch 访问控制：在 OpenSearch 集群中实施精细的访问控制，以限制对敏感数据的访问。使用中的内置安全功能，例如用户身份验证、基于角色的访问控制和索引级权限。 OpenSearch
- 可信扩展：始终仅从可信来源安装扩展。使用基础设施即代码 (IaC) 工具，例如简化 AWS CloudFormation 将相同扩展配置（包括 IAM 权限）附加到多个 Lambda 函数的过程。IaC 工具还提供以前使用的扩展和版本的审计记录。

- 敏感数据处理：构建扩展时，避免记录敏感数据。在出于审计目的记录或保留负载和元数据之前，请对其进行消毒。

成本优化

- 监控和警报：设置监控和警报机制，以跟踪 OpenSearch 从 Lambda 函数发送到的数据量。这将帮助您识别和解决任何潜在的成本超支。
- 数据保留：请仔细考虑中您的 Lambda 遥测数据的适当数据保留期。 OpenSearch更长的保留期会增加存储成本，因此请在可观测性需求与成本优化之间取得平衡。
- 压缩和索引：启用数据压缩并优化 OpenSearch 索引策略，以减少 Lambda 遥测数据的存储占用空间。
- 减少对日志的依赖 CloudWatch：通过将 Lambda Telemetry API 直接与集成 OpenSearch，您有可能减少对 CloudWatch 日志的依赖，从而节省成本。这是因为 Lambda Telemetry API 允许您直接向发送日志 OpenSearch，从而无需在中存储和处理数据。 CloudWatch

可扩展性和可靠性

- 异步处理：使用异步处理模式，例如亚马逊简单队列服务 (Amazon SQS) Simple Queue Service 或 Amazon Kinesis，将 Lambda 函数的执行与数据摄取分开。 OpenSearch 这有助于保持 Lambda 函数的响应能力，并提高系统的整体可靠性。
- OpenSearch 集群扩展：监控集 OpenSearch 群的性能和资源利用率，并根据需要向上或向下扩展以处理不断增加的 Lambda 遥测数据量。
- 故障转移和灾难恢复：为您的 OpenSearch 集群实施强大的灾难恢复策略，包括定期备份以及在出现故障时能够快速恢复数据。

可观察性和监测

- 仪表板和可视化效果：使用 Kibana 或其他仪表板工具创建自定义仪表板和可视化效果，根据中的遥测数据深入了解 Lambda 函数的性能和运行状况。 OpenSearch
- 警报和通知：设置警报和通知，以主动监控 Lambda 函数中的异常、错误或性能问题。将这些警报和通知与您现有的事件管理流程集成。
- 跟踪和关联：确保您的 Lambda 遥测数据包含相关的跟踪信息，例如请求 IDs 或关联 IDs，以实现分布式无服务器 end-to-end应用程序的可观察性和故障排除。

通过遵循这些最佳实践，您可以确保将 Lambda Telemet OpenSearch ry API 与 Lambda Telemetry API 的集成安全、经济实惠且可扩展，并为您的无服务器应用程序提供全面的可观察性。

操作说明

构建和部署 Lambda 扩展层

Task	描述	所需技能
下载源代码。	从扩展存储库下载示例AWS Lambda 扩展。	应用程序开发人员、云架构师
导航到 `python-example-telemetry-opensearch-extension` 文件夹。	您下载的AWS Lambda 扩展存储库包含多个用例和语言运行时的大量示例。导航到 python-example-telemetry-opensearch-e xtension文件夹，使用Python OpenSearch 扩展程序，该扩展程序会将日志发送到 OpenSearch该文件夹。	应用程序开发人员、云架构师
添加执行扩展端点的权限。	运行以下命令使扩展端点可执行： `chmod +x python-example-telemetry-opensearch-extension/extension.py`	应用程序开发人员、云架构师
在本地安装扩展依赖关系。	运行以下命令安装 Python 代码的本地依赖项： `pip3 install -r python-example-telemetry-opensearch-extension/requirements.txt -t ./python-example-telemetry-opensearch-extension/`	应用程序开发人员、云架构师

Task	描述	所需技能
	这些依赖项将与扩展代码一起安装。	

为扩展程序创建一个.zip 包，将其部署为一个层。	扩展名.zip 文件应包含一个名`extensions/` 为的根目录（扩展可执行文件所在的位置）和另一个名`python-example-telemetry-opensearch-extension/` 为的根目录（扩展的核心逻辑及其依赖项位于该目录中）。
	
	为扩展创建.zip 包：
	
	`chmod +x extensions/python-example-telemetry-opensearch-extension`
	`zip -r extension.zip extensions python-example-telemetry-opensearch-extension`
		应用程序开发人员、云架构师

将扩展程序部署为 Lambda 层。	使用扩展名.zip 文件和以下命令发布图层：
	
	`aws lambda publish-layer-version \`
	`--layer-name "python-example-telemetry-opensearch-extension" \`
	`--zip-file "fileb://extension.zip"`
		应用程序开发人员、云架构师

将扩展程序集成到您的函数中

Task	描述	所需技能
将层添加到函数。	1. 登录 AWS Management Console 并打开 AWS Lambda 控制台的 “功能” 页面。 2. 选择您的函数。 3. 在层下，选择添加层。 4. 在 “选择图层” 下，选择 “自定义图层” 作为图层来源，然后添加您的图层。 有关向 Lambda 函数添加层的更多信息，请参阅 Lambda 文档。	应用程序开发人员、云架构师
为函数设置环境变量。	在函数页面上，选择配置选项卡，然后将以下环境变量添加到您的函数中： • URL— 您的 OpenSearch 终端节点的 URI，您的日志将发送到该端点。 • AUTH_SECRET — 存储在中的 OpenSearch 凭证的 ARN。 AWS Secrets Manager它应存储为键值对并有两个键：username和。password • PLATFORM_INDEX FUNCTION_INDEX 、和 EXTENSION_INDEX — 用于存储遥测数据、函数日志和扩展日志	应用程序开发人员、云架构师

Task	描述	所需技能
	的索引的名称。确保他们遵守正确的命名标准。否则，您的索引将无法创建。 • DISPATCH_MIN_BATCH_SIZE — 要批处理的日志事件的数量。但是，当该功能关闭时，无论此设置如何，都将发送您的日志。	

添加日志语句并测试您的函数

Task	描述	所需技能

向您的函数添加日志语句。	使用内置日志机制之一或您选择的日志模块，将日志语句添加到函数中。
	以下是在 Python 中记录消息的示例：
	`print("Your Log Message Here")`
	`logger = logging.getLogger(__name__)`
	`logger.info("Test Info Log.")`
	`logger.error("Test Error Log.")`
		应用程序开发人员、云架构师

测试 函数。	1. 在函数页面上，选择测试选项卡。 2. 为您的函数创建测试事件并运行测试。有关更多信息，请参阅 Lambda 文档中的	应用程序开发人员、云架构师

Task	描述	所需技能
	在控制台中测试 Lambda 函数。 如果一切正常，你应该会看到“执行函数：成功”。	

查看您的登录信息 OpenSearch

Task	描述	所需技能
查询您的索引。	在中 OpenSearch，运行以下命令来查询您的索引： `SELECT * FROM index-name` 您的日志应显示在查询结果中。	云架构师

故障排除

事务	解决方案
连接问题	• 确认您的 Lambda 函数具有访问集群所需的网络连接。 OpenSearch 有关配置 VPC 设置的指南，请参阅OpenSearch 服务文档。 • 验证授予您的 Lambda 函数的 IAM 权限是否允许它向集群写入数据。 OpenSearch 查看 Lambda 文档，了解有关管理 IAM 权限的信息。
数据摄取错误	• 查看您的 Lambda 函数的 CloudWatch 日志，找出与 Lambda 遥测 API 集成相关的任

事务	解决方案
	何错误或异常。有关疑难解答指南，请参阅 Lambda 遥测 API 文档。 • 验证 OpenSearch 集群的配置是否正确，并且具有采集 Lambda 遥测数据所需的索引映射和设置。有关更多信息，请查阅OpenSearch 文档。

相关资源

- （存储库）的 Lambda 遥测集成示例 OpenSearch GitHub
- 使用 Lambda 扩展扩展 Lambda 函数（Lambda 文档）
- Lambda 遥测 API（Lambda 文档）
- 介绍 AWS Lambda 遥测 API（AWS 博客文章）
- 将@@ AWS Lambda 遥测 API 与 Prometheus 集成，以及（博客文章 OpenSearch）AWS

其他信息

更改日志结构

默认情况下，该扩展程序将日志作为嵌套文档发送 OpenSearch 到。这允许您执行嵌套查询来检索各个列的值。

如果默认日志输出不能满足您的特定需求，则可以通过修改提供的 Lambda 扩展的源代码对其进行自定义。 AWS AWS 鼓励客户调整输出以适应其业务需求。要更改日志输出，请在扩展程序源代码中的`telemetry_dispatcher.py`文件中找到该`dispatch_to_opensearch`函数，然后进行必要的更改。

为基于蜂窝的架构设置无服务器蜂窝路由器

由 Mian Tariq (AWS) 和 Ioannis Lioupras (AWS) 创建

摘要

作为全球基于蜂窝的应用程序系统的入口点，蜂窝路由器负责高效地将用户分配到相应的单元并为用户提供端点。蜂窝路由器处理诸如存储 user-to-cell映射、监控信元容量和在需要时请求新信元之类的功能。在潜在的中断期间，保持手机路由器的功能非常重要。

这种模式中的蜂窝路由器设计框架侧重于弹性、可扩展性和整体性能优化。该模式使用静态路由，即客户端在初始登录时缓存端点并直接与单元通信。这种解耦有助于在蜂窝路由器损坏期间确保基于蜂窝的应用程序不间断地运行，从而增强了系统的弹性。

此模式使用 AWS CloudFormation 模板来部署架构。有关模板部署的内容或使用部署相同配置的详细信息 AWS Management Console，请参阅 “其他信息” 部分。

> **Important**
>
> 此模式中显示的演示、代码和 AWS CloudFormation 模板仅用于解释目的。所提供的材料仅用于说明设计模式和帮助理解。演示和代码尚未准备就绪，不应用于任何现场制作活动。强烈建议不要尝试在生产环境中使用该代码或演示，风险自负。我们建议在生产环境中实施此模式或其任何组件之前，请咨询相应的专业人员并进行彻底的测试。

先决条件和限制

先决条件

- 有效的亚马逊 Web Services (AWS) 账户
- 最新版本的 AWS Command Line Interface (AWS CLI)
- 具有创建 AWS CloudFormation 堆栈、 AWS Lambda 函数和相关资源的必要权限的 AWS 证书

产品版本

- Python 3.12

架构

下图显示了蜂窝路由器的高级设计。

该图逐步完成以下工作流程：

1. 用户联系作为手机路由器 API 终端节点前端节点的 Amazon API Gateway。
2. Amazon Cognito 负责处理身份验证和授权。
3. 该 AWS Step Functions 工作流程由以下组件组成：
   - Orchestrator AWS Step Functions – `Orchestrator` 用于创建工作流程或状态机。该工作流程由蜂窝路由器 API 触发。根据`Orchestrator`资源路径执行 Lambda 函数。
   - D@@ ispatcher – Lamb `Dispatcher` da 函数为每个注册的新用户识别并分配一个静态单元。该函数搜索用户数量最少的单元格，将其分配给用户，然后返回端点。
   - 映射器-该`Mapper`操作处理由模板创建的 Amazon `RoutingDB` DynamoDB 数据库中的 user-to-cell映射。 AWS CloudFormation 触发后，该`Mapper`函数会为已分配的用户提供其终端节点。
   - Scaler ¿该`Scaler`功能可跟踪牢房占用率和可用容量。需要时，该`Scaler`函数可以通过亚马逊简单队列服务 (Amazon SQS) Simple Queue SQS 向配置和部署层发送请求，以请求新的单元格。
   - Validator ¿该`Validator`函数验证单元端点并检测任何潜在的问题。
4. `RoutingDB`存储单元信息和属性（API 端点 AWS 区域、状态、指标）。
5. 当信元的可用容量超过阈值时，蜂窝路由器会通过 Amazon SQS 请求配置和部署服务以创建新的单元。

创建新单元后，`RoutingDB`将从 “配置和部署” 层进行更新。但是，该过程超出了这种模式的范围。有关基于蜂窝的架构设计前提的概述以及有关此模式中使用的蜂窝路由器设计的详细信息，请参阅 “其他信息” 部分。

工具

AWS 服务

- Amazon API Gateway 可帮助您创建、发布、维护、监控和保护任何规模的 RES WebSocket APIs T、HTTP。
- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。

- Amazon Cognito 为您的 Web 和移动应用程序提供身份验证、授权和用户管理。
- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Storage Service (Amazon S3)是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon Simple Queue Service (Amazon SQS) 提供了一个安全、持久且可用的托管队列，它可帮助您集成和分离分布式软件系统与组件。
- AWS Step Functions是一项无服务器编排服务，可帮助您组合 Lambda 函数和其他函数 AWS 服务来构建业务关键型应用程序。

其他工具

- Python 是通用的计算机编程语言。

代码存储库

此模式的代码可在 GitHub Serverless-Cell- Router 存储库中找到。

最佳实践

有关构建基于单元的架构的最佳实践，请参阅以下 Well-Architected 指南 AWS ：

- 使用基于单元的架构缩小影响范围
- AWS Well-Architected Framework 可靠性支柱 REL1：0-BP04 使用隔板架构来限制影响范围

操作说明

准备源文件

Task	描述	所需技能
克隆示例代码存储库。	要将 Serverless-Cell-Router GitHub 存储库克隆到您的计算机，请使用以下命令：	开发人员

Task	描述	所需技能
	`git clone https://github.com/aws-samples/Serverless-Cell-Router/`	
设置 AWS CLI 临时证书。	使用您的 AWS CLI 凭据配置 AWS 账户。本演练使用 AWS IAM Identity Cen ter 命令行或编程访问选项提供的临时证书。这会将`AWS_ACCESS_KEY_ID` `AWS_SECRET_ACCESS_KEY` 、和`AWS_SESSION_TOKEN` AWS 环境变量设置为相应的凭据，以便与一起使用 AWS CLI。	开发人员
创建 S3 存储桶。	创建一个 S3 存储桶，用于存储和访问模板部署的 Serverless-Cell-Router Lambda 函数。AWS CloudFormation 要创建 S3 存储桶，请使用以下命令： `aws s3api create-bucket --bucket <bucket name> --region eu-central-1 --create-bucket-configuration LocationConstraint=eu-central-1`	开发人员

Task	描述	所需技能

创建.zip 文件。	为位于函数目录中的每个 Lambda 函数创建一个.zip 文件。这些.zip 文件将用于部署 Lambda 函数。在 Mac 上，使用以下zip命令：
	
	`zip -j mapper-scr.zip Functions/Mapper.py`
	`zip -j dispatcher-scr.zip Functions/Dispatcher.py`
	`zip -j scaler-scr.zip Functions/Scaler.py`
	`zip -j cp validator-scr.zip Functions/Validator.py`
	`zip -j dynamodbDummyData-scr.zip Functions/DynamodbDummyData.py`
		开发人员

Task	描述	所需技能

将.zip 文件复制到 S3 存储桶。	要将所有 Lambda 函数.zip 文件复制到 S3 存储桶，请使用以下命令：
	`aws s3 cp mapper-scr.zip s3://<bucket name>`
	`aws s3 cp dispatcher-scr.zip s3://<bucket name>`
	`aws s3 cp scaler-scr.zip s3://<bucket name>`
	`aws s3 cp validator-scr.zip s3://<bucket name>`
	`aws s3 cp dynamodbDummyData-scr.zip s3://<bucket name>`
		开发人员

创建堆 AWS CloudFormation 栈

Task	描述	所需技能

部署 AWS CloudFormation 模板。	要部署 AWS CloudFormation 模板，请运行以下 AWS CLI 命令：
	`aws cloudformation create-stack --stack-name serverless.cell-router \`
	`--template-body file://Serverless-Cell-Router-Stack-v10.yaml \`
	`--capabilities CAPABILITY_IAM \`
		开发人员

Task	描述	所需技能
	`--parameters ParameterKey=LambdaFunctionMapperS3KeyParameterSCR,ParameterValue=mapper-scr.zip \ ParameterKey=LambdaFunctionDispatcherS3KeyParameterSCR,ParameterValue=dispatcher-scr.zip \ ParameterKey=LambdaFunctionScalerS3KeyParameterSCR,ParameterValue=scaler-scr.zip \ ParameterKey=LambdaFunctionAddDynamoDBDummyItemsS3KeyParameterSCR,ParameterValue=dynamodbDummyData-scr.zip \ ParameterKey=LambdaFunctionsS3BucketParameterSCR,ParameterValue=<S3 bucket storing lambda zip files> \ ParameterKey=CognitoDomain,ParameterValue=<Cognito Domain Name> \ --region <enter your aws region id, e.g. "eu-central-1">`	

Task	描述	所需技能
查看进度。	登录 AWS Management Console，打开 AWS CloudFormation 控制台 https://console.aws.amazon.com/cloudformation/，然后查看堆栈开发进度。当状态为时 CREATE_COMPLETE ，表示堆栈已成功部署。	开发人员

评估和验证

Task	描述	所需技能
将单元格分配给用户。	要启动Orchestrator ，请运行以下 curl 命令： `curl -X POST \ -H "Authorization: Bearer {User id_token}" \ https://xxxxxx.execute-api.eu-central-1.amazonaws.com/Cell_Router_Development/cells` Orchestrator 触发Dispatcher 函数的执行。反过来Dispatcher ，会验证用户的存在。如果找到了用户，则会Dispatcher 返回关联的单元 ID 和终端节点 URLs。如果找不到用户，则会向用户Dispatcher 分配一个小区，并将小区 ID 发送给	开发人员

Task	描述	所需技能
	该Scaler函数，以评估分配的单元的剩余容量。 该Scaler函数的响应如下： "cellID : cell-0002 , endPoint_1 : https://xxxxx.execute-api.eu-north-1 .amazonaws.com/ , endPoint_2 : https://xxxxxxx.execute-api .eu-central-1.amaz onaws.com/"	

Task	描述	所需技能

检索用户单元格。	要使用Orchestrator 来执行Mapper函数，请运行以下命令：
	
	`curl -X POST \ -H "Authorization: Bearer {User id_token}" \ https://xxxxxxxxx.execute-api.eu-central-1.amazonaws.com/Cell_Router_Development/mapper`
	
	Orchestrator 搜索分配给用户的单元格并返回单元格 ID，并在以下响应 URLs 中返回：
	
	`"cellID : cell-0002 , endPoint_1 : https://xxxxx.execute-api.eu-north-1.amazonaws.com/ , endPoint_2 : https://xxxxxxx.execute-api.eu-central-1.amazonaws.com/"`
		开发人员

清理

Task	描述	所需技能
清除资源。	为避免在您的账户中产生额外费用，请执行以下操作：	应用程序开发人员

Task	描述	所需技能
	1. 清空您为 Lambda 函数创建的 S3 存储桶。 2. 删除 存储桶。 3. 删除 AWS CloudFormation 堆栈。	

相关资源

参考

- 使用可用区的静态稳定性
- AWS 故障隔离边界：静态稳定性

视频

Physalia：基于单元的架构将在亚马逊 EBS 上提供更高的可用性

https://www.youtube-nocookie.com/embed/6lknq？RZMFic 控件=0

其他信息

基于单元的架构设计前提

尽管这种模式侧重于蜂窝路由器，但了解整个环境非常重要。环境分为三个独立的层：

- 包含蜂窝路由器的路由层或薄层
- 细胞层，包括各种细胞
- 配置和部署层，用于配置单元并部署应用程序

即使存在影响其他层的损伤，每层也能维持其功能。 AWS 账户 用作故障隔离边界。

下图显示了高层次的图层。Cell 层和配置和部署层不在此模式的范围之内。

有关基于蜂窝的架构的更多信息，请参阅使用基于蜂窝的架构缩小影响范围：信元路由。

蜂窝路由器设计模式

蜂窝路由器是跨蜂窝的共享组件。为了减轻潜在的影响，路由层必须使用尽可能精简且可水平扩展的设计。作为系统的入口点，路由层仅包含高效地将用户分配到相应单元所需的组件。该层中的组件不参与单元的管理或创建。

这种模式使用静态路由，这意味着客户端在初次登录时缓存端点，然后与计算单元建立直接通信。启动客户端和蜂窝路由器之间的定期交互以确认当前状态或检索任何更新。这种有意的解耦可以让现有用户在手机路由器停机时不间断地运行，并且还能在系统内提供持续的功能和弹性。

在这种模式下，蜂窝路由器支持以下功能：

- 从 Provision 和 Deploy 层的单元数据库中检索单元格数据，并存储或更新本地数据库。
- 使用单元分配算法为应用程序的每个新注册用户分配一个单元。
- 将 user-to-cells映射存储在本地数据库中。
- 在用户分配期间检查单元的容量，并将自动售货机的事件引发到 Provision and Deploy 层以创建单元。
- 使用单元创建标准算法来提供此功能。
- 通过提供静态单元来响应新注册 URLs 的用户请求。这些 URLs 内容将在客户端上缓存，并附上生存时间 (TTL)。
- 通过提供新的或更新的 URL 来响应现有用户对无效 URL 的请求。

要进一步了解 AWS CloudFormation 模板设置的演示蜂窝路由器，请查看以下组件和步骤：

1. 设置和配置 Amazon Cognito 用户池。
2. 为蜂窝路由器设置和配置 API Gateway API。
3. 创建 DynamoDB 表。
4. 创建和配置 SQS 队列。
5. 实施`Orchestrator`.
6. 实现 Lambda 函数：`Dispatcher`、、`Scaler`、`Mapper`。`Validator`
7. 评估并验证。

前提是配置和部署层已经建立。它的实现细节超出了这个工件的范围。

由于这些组件是通过 AWS CloudFormation 模板设置和配置的，因此以下步骤将以描述性和高级的形式呈现。假设您具备完成设置和配置所需的 AWS 技能。

1。设置和配置 Amazon Cognito 用户池

登录并打开 Amazon Cognito 控制台，网址为。 AWS Management Consolehttps://console.aws.amazon.com/cognito/设置和配置名为的 Amazon Cognito 用户池`CellRouterPool`，该用户池包含应用程序集成、托管用户界面和必要的权限。

2。为蜂窝路由器设置和配置 API Gateway API

打开 API Gateway 控制台，网址为https://console.aws.amazon.com/apigateway/。使用与 Amazon Cognito 用户`CellRouter`池集成的 Amazon Cognito 授权机构设置和配置名为的 API。`CellRouterPool`实现以下元素：

- `CellRouter`API 资源，包括`POST`方法
- 与步骤 5 中实现的 Step Functions 工作流程集成
- 通过亚马逊 Cognito 授权机构进行授权
- 集成请求和响应映射
- 必要权限的分配

3。创建 DynamoDB 表

在上打开 DynamoDB 控制台，然后使用https://console.aws.amazon.com/dynamodb/以下配置创建名为的标准 DynamoDB 表：`tbl_router`

- 分区键 – `marketId`
- 排序键 `cellId`
- 容量模式 – 已预配置
- Point-in-time 恢复 (PITR)-关闭

在索引选项卡上，创建一个名为的全局二级索引`marketId-currentCapacity-index`。`ScalerLambda` 函数将使用索引对分配用户数最少的单元进行高效搜索。

使用以下属性创建表结构：

- `marketId`– 欧洲
- `cellId`– cell-0002
- `currentCapacity`– 2
- `endPoint_1`– <your endpoint for the first Region>

- endPoint_2– <your endpoint for the second Region>
- IsHealthy– 没错
- maxCapacity– 10
- regionCode_1 – eu-north-1
- regionCode_2 – eu-central-1
- userIds– <your email address>

4。创建和配置 SQS 队列

在上打开 Amazon SQS 控制台 https://console.aws.amazon.com/sqs/，然后创建一个名为 “CellProvisioning配置了亚马逊 SQS 密钥加密” 的标准 SQS 队列。

5。实现 Orchestrator

开发一个 Step Functions 工作流程作为路由Orchestrator器的工作流程。该工作流程可通过蜂窝路由器 API 调用。该工作流程根据资源路径执行指定的 Lambda 函数。将步进函数与适用于蜂窝路由器的 API Gateway API 集成CellRouter，并配置调用 Lambda 函数所需的权限。

图表显示了以下工作流程。选择状态会调用其中一个 Lambda 函数。如果 Lambda 函数成功运行，则工作流程结束。如果 Lambda 函数失败，则调用失败状态。

6。实现 Lambda 函数

实现Dispatcher、MapperScaler、和Validator函数。在演示中设置和配置每个函数时，请为该函数定义一个角色并分配在 DynamoDB 表上执行所需操作所需的权限。tbl_router此外，将每个功能集成到工作流程中Orchestrator。

调度器功能

该Dispatcher函数负责为每个新注册用户识别和分配一个静态单元。当新用户向全局应用程序注册时，请求将转到该Dispatcher函数。该函数使用预定义的评估标准来处理请求，例如：

1. 区域 – 在市场中选择用户所在的单元格。例如，如果用户正在从欧洲访问全球应用程序，请选择 AWS 区域 在欧洲使用的单元格。
2. 邻近度或延迟 – 选择离用户最近的蜂窝例如，如果用户从荷兰访问应用程序，则该函数会考虑使用法兰克福和爱尔兰的蜂窝。决定哪个小区最近，取决于诸如用户位置和单元区域之间的延迟之类的指标。对于此示例模式，信息是从配置和部署层静态提供的。

3. H ealth `Dispatcher` – 该函数根据提供的细胞状态检查所选单元格是否健康（Healthy = true 或 false）。
4. 容量 – 用户分布基于单元逻辑中最少的用户数，因此用户被分配到用户数量最少的单元。

> **Note**
>
> 提出这些标准只是为了解释这种示例模式。对于现实生活中的蜂窝路由器实现，您可以定义更精细且基于用例的标准。

`Orchestrator`调用 Dispatcher 函数将用户分配到单元格。在此演示函数中，市场价值是一个静态参数，定义为`europe`。

该`Dispatcher`函数评估单元格是否已分配给用户。如果已经分配了单元格，则该`Dispatcher`函数将返回单元格的端点。如果没有为用户分配任何单元格，则该函数将搜索用户数最少的单元格，将其分配给用户，然后返回端点。使用全局二级索引优化了单元格搜索查询的效率。

映射器函数

该`Mapper`函数负责监督数据库中 user-to-cell映射的存储和维护。为每个注册用户分配一个单元。每个单元都有两个不同的单元 URLs ——每个 AWS 区域各一个。这些端点充当托管在 API Gateway 上的 API 端点， URLs 充当全球应用程序的入站点。

当该`Mapper`函数收到来自客户端应用程序的请求时，它会对 DynamoDB `tbl_router` 表运行查询，以检索 user-to-cell与提供的电子邮件 ID 关联的映射。如果它找到了分配的单元格，则该`Mapper`函数会立即提供该单元格的两个单元格 URLs。该`Mapper`功能还会主动监控单元的更改 URLs，并启动通知或更新用户设置。

缩放器功能

该`Scaler`函数管理电池的剩余容量。对于每个新的用户注册请求，该`Scaler`函数都会评估该功能分配给用户的单元的`Dispatcher`可用容量。如果根据指定的评估标准，该单元已达到其预先确定的限制，则该函数会通过 Amazon SQS 队列向配置和部署层发起请求，请求配置和部署新单元。单元格的缩放可以根据一组评估标准来执行，例如：

1. 最大用户数 – 每个单元最多可以有 500 个用户。
2. 缓冲容量 – 每个单元的缓冲容量为 20%，这意味着每个单元可以随时分配给 400 个用户。剩余的 20% 的缓冲容量留给未来的用例和处理意外情况（例如，当单元创建和配置服务不可用时）。

3. 创建单元-一旦现有单元达到容量的 70%，就会触发请求以创建额外的单元。

> **Note**
>
> 提出这些标准只是为了解释这种示例模式。对于现实生活中的蜂窝路由器实现，您可以定义更精细且基于用例的标准。

演示Scaler代码由Dispatcher成功分配单元格给新注册的用户Orchestrator后执行。收到来自的小区ID后 ScalerDispatcher，将根据预定义的评估标准评估指定小区是否有足够的容量来容纳其他用户。如果手机容量不足，该Scaler函数会向 Amazon SQS 服务发送一条消息。此消息由配置和部署层内的服务检索，启动新单元的配置。

验证器函数

该Validator功能可识别并解决与手机访问有关的问题。当用户登录全局应用程序时，该应用程序会 URLs 从用户配置文件设置中检索单元格，并将用户请求路由到单元内两个分配的区域之一。如果无法访问， URLs 则应用程序可以向蜂窝路由器发送验证 URL 请求。手机路由器Orchestrator调用. Validator Validator启动验证过程。除其他检查外，验证可能包括以下内容：

- 将请求 URLs 中的单元格与 URLs 存储在数据库中的单元格进行交叉引用，以识别和处理潜在的更新
- 运行深度运行状况检查（例如，对单元终端节点的HTTP GET请求）

总而言之，该Validator功能会对客户应用程序请求做出响应，提供验证状态以及任何必需的补救步骤。

Validator旨在增强用户体验。假设这样的场景，即某些用户在访问全局应用程序时遇到困难，因为事件导致单元暂时不可用。该Validator函数可以提供指导性的补救步骤，而不是呈现一般性错误。这些步骤可能包括以下操作：

- 将事件通知用户。
- 提供服务可用之前的大致等待时间。
- 提供支持联系电话以获取更多信息。

该Validator函数的演示代码验证请求中用户提供的单元格是否与表 URLs 中存储的记录相匹配。tbl_router该Validator函数还会检查细胞是否健康。

通过 VPC 终端节点设置对 Amazon S3 存储桶的私有访问权限

由马丁·马里奇（AWS）、加布里埃尔·罗德里格斯·加西亚（AWS）、舒赫拉特·霍贾耶夫（AWS）、尼古拉斯·雅各布·贝尔（AWS）、Mohan Gowda Purushothama（AWS）和华金·里纳多（AWS）创作

摘要

在 Amazon Simple Storage Service (Amazon S3) 中， URLs 预签名允许您与目标用户共享任意大小的文件。默认情况下，预签名 URLs 的 Amazon S3 可以在到期时间窗口内从互联网访问，这使得它们使用起来很方便。但是，企业环境通常要求对预先签名 URLs 的 Amazon S3 的访问权限仅限于私有网络。

这种模式提供了一种无服务器解决方案，通过使用 URLs 来自私有网络的预签名与 S3 对象进行安全交互，无需遍历互联网。在该架构中，用户通过内部域名访问 Application Load Balancer。流量通过 Amazon API Gateway 和 S3 存储桶的虚拟私有云 (VPC) 终端节点在内部路由。该 AWS Lambda 函数生成预签名， URLs 以便通过私有 VPC 端点下载文件，这有助于增强敏感数据的安全和隐私。

先决条件和限制

先决条件

- 一种 VPC，包括部署在 AWS 账户 与公司网络相连的子网（例如，通过 AWS Direct Connect）。

限制

- S3 存储桶必须与域名同名，因此我们建议您查看 Amazon S3 存储桶命名规则。
- 此示例架构不包括对已部署基础架构的监控功能。如果您的用例需要监控，请考虑添加AWS 监控服务。
- 此示例架构不包括输入验证。如果您的用例需要输入验证并提高安全级别，请考虑使用 AWS WAF 来保护您的 API。
- 此示例架构不包括使用 Application Load Balancer 进行访问日志记录。如果您的用例需要访问日志记录，请考虑启用负载均衡器访问日志。

版本

- Python 版本 3.11 或更高版本
- Terraform 版本 1.6 或更高版本

架构

目标技术堆栈

目标技术堆栈中使用了以下 AWS 服务：

- Amazon S3 是用于安全上传、下载和存储文件的核心存储服务。
- Amazon API Gateway 公开了用于与 S3 存储桶交互的资源和终端节点。该服务在生成 URLs 用于下载或上传数据的预签名中起着作用。
- AWS Lambda生成 URLs 用于从 Amazon S3 下载文件的预签名。Lambda 函数由 API Gateway 调用。
- Amazon VPC 在 VPC 内部署资源以提供网络隔离。VPC 包括用于控制流量的子网和路由表。
- Application Load Balan cer 将传入流量路由到 API Gateway 或 S3 存储桶的 VPC 终端节点。它允许来自公司网络的用户在内部访问资源。
- Amazon S3 的 VPC 终端节点支持在 VPC 和 Amazon S3 中的资源之间进行直接、私密的通信，而无需通过公共互联网。
- AWS Identity and Access Management (IAM) 控制对 AWS 资源的访问权限。权限的设置是为了确保与 API 和其他服务的安全交互。

目标架构

该图阐释了以下内容：

1. 企业网络中的用户可以通过内部域名访问 Application Load Balancer。我们假设公司网络和中的内部网子网之间存在连接 AWS 账户 （例如，通过 AWS Direct Connect 连接）。
2. Application Load Balancer 将传入流量路由到 API Gateway URLs 以生成预签名以下载或上传到 Amazon S3 的数据，或者路由到 S3 存储桶的 VPC 终端节点。在这两种情况下，请求都是在内部路由的，不需要通过互联网。
3. API Gateway 公开了与 S3 存储桶交互的资源和终端节点。在此示例中，我们提供了一个用于从 S3 存储桶下载文件的终端节点，但也可以对其进行扩展以提供上传功能。
4. Lambda 函数使用应用程序负载均衡器的域名而不是公有 Amazon S3 域名生成用于从 Amazon S3 下载文件的预签名 URL。
5. 用户收到预签名 URL，然后使用它通过应用程序负载均衡器从 Amazon S3 下载文件。负载均衡器包含一条默认路由，用于将不适用于 API 的流量发送到 S3 存储桶的 VPC 终端节点。

6. VPC 终端节点将带有自定义域名的预签名 URL 路由到 S3 存储桶。S3 存储桶必须与域名同名。

自动化和扩缩

此模式使用 Terraform 将基础架构从代码存储库部署到. AWS 账户

工具

工具

- Python 是通用的计算机编程语言。
- Terraform 是一款基础设施即代码 (IaC) 工具 HashiCorp ，可帮助您创建和管理云和本地资源。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您通过命令行 shell 中的命令与 AWS 服务进行交互。

代码存储库

此模式的代码可在 private https://github.com/aws-samples/-s 3-vpce 的 GitHub 存储库中找到。

最佳实践

此模式的示例架构使用 IAM 权限来控制对 API 的访问权限。任何拥有有效 IAM 证书的人都可以调用 API。如果您的用例需要更复杂的授权模型，则可能需要使用不同的访问控制机制。

操作说明

将解决方案部署在 AWS 账户

Task	描述	所需技能
获取 AWS 凭证。	查看您的 AWS 凭证和您对账户的访问权限。有关说明，请参阅 AWS CLI 文档中的配置和凭据文件设置。	AWS DevOps，通用 AWS
克隆存储库。	克隆使用以下模式提供的 GitHub 存储库：	AWS DevOps，通用 AWS

Task	描述	所需技能
	`git clone https://github.com/aws-samples/private-s3-vpce`	
配置变量。	1. 在计算机上的 GitHub 存储库中，打开`terraform` 文件夹： `cd terraform` 2. 打开`example.tfvars` 文件并根据需要自定义参数。	AWS DevOps，通用 AWS
部署解决方案。	1. 在`terraform` 文件夹中，运行 Terraform 并传入您自定义的变量： `terraform apply -var-file="example.tfvars"` 2. 确认架构图中显示的资源已成功部署。	AWS DevOps，通用 AWS

测试解决方案

Task	描述	所需技能
创建测试文件。	将文件上传到 Amazon S3，为文件下载创建测试方案。您可以使用 Amazon S3 控制台或以下 AWS CLI 命令：	AWS DevOps，通用 AWS

Task	描述	所需技能
	`aws s3 cp /path/to/testfile s3://your-bucket-name/testfile`	
测试预签名 URL 功能。	1. 使用 a w scurl 向 Application Load Balancer 发送请求，要求为测试文件创建预签名 URL： `awscurl https://your-domain-name/api/get_url?key=testfile` 此步骤将根据您的凭证创建有效的签名，该签名将由 API Gateway 进行验证。 2. 解析您在上一步中收到的响应中的链接，然后打开预签名 URL 以下载文件。	AWS DevOps，通用 AWS
清理。	当不再需要资源时，请务必将其删除： `terraform destroy`	AWS DevOps，通用 AWS

故障排除

事务	解决方案
带有特殊字符（例如数字符号 (#)）的 S3 对象密钥名称会损坏 URL 参数并导致错误。	正确编码 URL 参数，并确保 S3 对象密钥名称符合 Amazon S3 指南。

相关资源

亚马逊 S3：

- 共享带有预签名的对象 URLs
- 使用存储桶策略控制来自 VPC 终端节点的访问

亚马逊 API Gateway：

- 在 API Gateway APIs 中使用私有的 VPC 终端节点策略

应用程序 Load Balancer：

- 使用 ALB、S3 和 PrivateLink（AWS 博客文章）托管内部 HTTPS 静态网站

使用 Amazon Bedroc AWS Step Functions k 对中的状态进行故障排除

由 Aniket Kurzadkar (AWS) 和 Sangam Kushwaha (AWS) 创作

摘要

AWS Step Functions 错误处理功能可以帮助你查看在工作流程状态下发生的错误，但要找到错误的根本原因并对其进行调试仍然是一项挑战。这种模式解决了这一挑战，并展示了 Amazon Bedrock 如何帮助您解决 Step Functions 状态下出现的错误。

Step Functions 提供工作流程编排，使开发人员可以更轻松地实现流程自动化。Step Functions 还提供了错误处理功能，具有以下优点：

- 开发人员可以创建更具弹性的应用程序，这些应用程序在出现问题时不会完全失败。
- 工作流程可以包含条件逻辑，以不同的方式处理不同类型的错误。
- 系统可以自动重试失败的操作，可能采用指数级退缩。
- 可以为错误场景定义替代执行路径，从而使工作流程能够调整并继续处理。

当 Step Functions 工作流程中出现错误时，此模式显示如何将错误消息和上下文发送到 Step Functions 支持的 Claude 3 等基础模型 (FM)。FM 可以分析错误，对其进行分类，并提出可能的补救措施。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 对工作流程AWS Step Functions 的基本了解
- 亚马逊 Bedrock API 连接

限制

- 你可以将这种模式的方法用于各种各样 AWS 服务。但是，根据随后由 Amazon Bedrock 评估的提示创建的提示 AWS Lambda ，结果可能会有所不同。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅按地区划分的 AWS 服务。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

架构

下图显示了此模式的工作流和体系结构组件。

下图显示了 Step Functions 状态机中错误处理和通知的自动工作流程：

1. 开发者开始执行状态机。
2. Step Functions 状态机开始处理其状态。有两种可能的结果：
   - (a) 如果所有状态都成功执行，则工作流程将直接转至 Amazon SNS 以获取电子邮件成功通知。
   - (b) 如果任何状态失败，工作流程将移至错误处理 Lambda 函数。
3. 如果出现错误，则会发生以下情况：
   - (a) Lambda 函数（错误处理程序）被触发。Lambda 函数从 Step Functions 状态机传递给它的事件数据中提取错误消息。然后，Lambda 函数根据此错误消息准备提示并将该提示发送到 Amazon Bedrock。该提示要求提供与遇到的特定错误相关的解决方案和建议。
   - (b) 托管生成人工智能模型的 Amazon Bedrock 负责处理输入提示。（此模式使用 Anthropic Claude 3 基础模型 (FM)，这是 FMs 亚马逊 Bedrock 支持的众多模型之一。）AI 模型分析错误背景。然后，模型会生成一个响应，其中可以包括错误发生原因的解释、解决错误的潜在解决方案以及避免将来犯同样错误的建议。

     Amazon Bedrock 返回其人工智能生成的对 Lambda 函数的响应。Lambda 函数处理响应，可能会对其进行格式化或提取关键信息。然后，Lambda 函数将响应发送到状态机输出。
4. 处理错误或成功执行后，工作流程以触发 Amazon SNS 发送电子邮件通知结束。

工具

AWS 服务

- Amazon Bedrock 是一项完全托管的服务，它通过统一的 API 提供来自领先的人工智能初创公司和亚马逊的高性能基础模型 (FMs) 供您使用。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。
- AWS Step Functions是一项无服务器编排服务，可帮助您组合 AWS Lambda 功能和其他功能 AWS 服务 来构建关键业务应用程序。

最佳实践

- 鉴于 Amazon Bedrock 是一种生成式 AI 模型，可以从经过训练的数据中学习，因此它还使用这些数据来训练和生成上下文。最佳做法是隐瞒任何可能导致数据泄露问题的私人信息。
- 尽管生成式人工智能可以提供宝贵的见解，但关键的错误处理决策仍应涉及人工监督，尤其是在生产环境中。

操作说明

为您的工作流程创建状态机

Task	描述	所需技能
创建一个 状态机。	要创建适合您的工作流程的状态机，请执行以下操作： 1. 登录并 AWS Management Console打开 AWS Step Functions 控制台。 2. 在左侧导航窗格中，选择状态机。 3. 选择创建状态机。 4. 根据您的用例选择模板，或者选择 “空白” 以根据您的要求创建模板。	AWS DevOps

创建 Lambda 函数

Task	描述	所需技能
创建一个 Lambda 函数。	要创建 Lambda 函数，请执行以下操作： 1. 在中 AWS Management Console，导航到 AWS Lambda 控制台。	AWS DevOps

Task	描述	所需技能
	2. 在左侧导航窗格中，选择函数，然后选择创建函数。 3. 在创建函数页面上，从选项中进行选择以创建函数。然后，在函数名称中输入名称，并从 Runtime 的下拉列表中选择相应的语言。 4. 选择 Create function (创建函数)。	

Task	描述	所需技能

在 Lambda 代码中设置所需的逻辑。	• 要使用连接到 Amazon Bedrock API 适用于 Python (Boto3) 的 AWS SDK，请使用以下代码。
	
	此代码为 Amazon Bedrock 设置客户端，准备必要的参数，然后向带有指定提示的 Claude 3 模型发送请求。
	
	这种模式调用了 Claude 3 模型。有关所有支持的基础模型（包括相关模型）的更多信息 IDs，请参阅 Amazon Bedrock 文档中的 Amazon Bedrock 中支持的基础模型。
	
	`client = boto3.client(`
	`service_name="bedrock-runtime", region_name="selected-region"`
	`)`
	
	`# Invoke Claude 3 with the text prompt`
	`model_id = "your-model-id" # Select your Model ID, Based on the Model Id, Change the body format`
	
	`try:`
	`response = client.invoke_model(`
	`modelId=model_id,`
		AWS DevOps

Task	描述	所需技能

	<pre>body=json
	.dumps(
	{
	"anthropic_version":
	"bedrock-2023-05-31",
	"max_tokens": 1024,
	"messages": [
	{
	"role": "user",
	"content"
	: [{"type": "text",
	"text": prompt}],
	}
	],
	}
	),
	)</pre>
	• （可选）将替换为占位 AWS 账户 IDs 符帐户 IDs。出于安全考虑，此方法可用于清理日志、错误消息或其他可能包含敏感账户信息的输出。
	
	以下代码将查找任何出现的用冒号括起的 12 位数字（这是 AWS 账户 IDs Amazon 资源名称 (ARNs) 和其他一些 AWS 标识符中的格式），并将其替换为占位	

Task	描述	所需技能

	符账户 ID。":123456789012:"
	
	`def replace_account_id(input_string):`
	`  # Use a regular expression to find the AWS account ID pattern`
	`  account_id_pattern = r'(:\d{12}:)'`
	`  # Replace the matched pattern with ":123456789012:"`
	`  modified_string = re.sub(account_id_pattern, ":123456789012:", input_string)`
	`  return modified_string`	

将 Step Functions 与 Lambda 集成

Task	描述	所需技能
设置 Lambda 以处理 Step Functions 中的错误。	要将 Step Functions 设置为在不中断工作流程的情况下处理错误，请执行以下操作： 1. 在 Step Functions 控制台中，导航到你之前创建的状态机。	AWS DevOps

Task	描述	所需技能
	2. 选择 “编辑”，然后选择要为其设置错误处理的服务，然后选择 “错误处理”。 3. 选择 “添加新捕手”，对于回退状态，选择 Lambda，然后选择您之前创建的 Lambda 函数。有关更多信息，请参阅 Step Functions 文档中的捕获错误。	

故障排除

事务	解决方案
Lambda 无法访问亚马逊 Bedrock API（未获得执行权限）	当 Lambda 角色无权访问 Amazon Bedrock API 时，就会发生此错误。要解决此问题，请为 Lambda 角色添加`AmazonBedrockFullAccess` 策略。有关更多信息，请参阅AmazonBedrockFullAccess《AWS 托管策略参考指南》。
Lambda 超时错误	有时，生成响应并将其发送回可能需要超过 30 秒，具体视提示而定。要解决此问题，请延长配置时间。有关更多信息，请参阅AWS Lambda 开发人员指南中的配置 Lambda 函数超时。

相关资源

- Amazon Bedrock
- 亚马逊 Bedrock API 访问权限
- 创建您的第一个 Lambda 函数
- 使用 Step Functions 开发工作流程
- AWS Step Functions

使用无服务器方法将 Amazon Web Services 串在一起

由 Aniket Braganza (AWS) 编写

摘要

此模式演示了一种可扩展的无服务器方法，通过将 Amazon Simple Storage Service (Amazon S3)、Amazon Simple Notification Service (Amazon SNS)、Amazon Simple Queue Service (Amazon SQS) 和 AWS Lambda 链接在一起来处理上传的文件。上传的文件示例用于演示目的。您可使用无服务器方法通过链接实现业务目标所需 Amazon Web Services 组合来完成其他任务。无服务器方法采用异步工作流，该工作流依赖于事件驱动的通知、弹性存储和函数即服务（FaaS）计算来处理请求。您可以使用无服务器方法进行扩展以满足需求，同时最大限度地降低成本。

> **Note**
>
> 通过无服务器方法将 AWS 服务链接到一起，有多种选择。例如，您可以使用将 Lambda 与 Amazon S3 相结合的方法，而不是 Amazon SNS 和 Amazon SQS。但是，此模式使用 Amazon SNS 和 Amazon SQS，因为此方法可以在事件通知期间将多个集成点添加到 Lambda 调用过程中，并扩展实施以在无服务器编排中包含多个侦听器，同时最大限度地减少处理开销。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 以编程方式访问 Amazon Web Services account。有关更多信息，请参阅：
  - AWS Cloud Development Kit (AWS CDK) 文档中的先决条件
  - AWS 命令行界面（AWS CLI）文档中的先决条件
- AWS CDK，已安装
- AWS CLI，已安装和配置
- Python 3.9

产品版本

- AWS CDK 2.x

- Python 3.9

架构

下图说明了链接的 AWS 服务如何使用户能够将文件上传到 S3 存储桶进行处理。

图表显示了以下工作流：

1. 用户将文件上传到 S3 存储桶。
2. 上传会启动一个 S3 事件，该事件将消息发布到 SNS 主题。此消息包含 S3 事件的详细信息。
3. 发布到 SNS 主题的消息将插入到 SQS 队列中，该队列订阅并接收该主题的通知。
4. Lambda 函数轮询 SQS 队列（作为其事件源）并等待消息处理。
5. 当 Lambda 函数从 SQS 队列接收消息时，它会处理这些消息并确认收到这些消息。
6. 如果消息未由 Lambda 处理，则该消息将返回到 SQS 队列并最终传输到 SQS 死信队列。

技术堆栈

- Amazon S3
- Amazon SNS
- Amazon SQS
- AWS Lambda

工具

Amazon Web Services

- Amazon Simple Storage Service(Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。
- Amazon Simple Queue Service (Amazon SQS) 提供了一个安全、持久且可用的托管队列，它可帮助您集成和分离分布式软件系统与组件。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。

其他工具

- AWS Cloud Development Kit (AWS CDK) 是与您的 AWS CDK 应用程序交互的主要工具。它执行您的应用程序，查询您定义的应用程序模型，并生成和部署由 AWS CDK 生成的 AWS CloudFormation 模板。
- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- Python 是一种高级解释型通用编程语言。

代码

此模式的代码可在 GitHub 链接 S3 到 SNS 到 SQS 再到 Lambda 存储库中找到。

操作说明

开发您的无服务器环境

Task	描述	所需技能
克隆存储库。	克隆存储库，然后导航到 `python/s3-sns-sqs-lambda-chain` 文件夹。	应用程序开发人员
设置虚拟环境。	1. 在 AWS CDK 中，运行 `python3 -m venv .venv` 命令。 2. 在 MacOS/Linux 上运行 `source .venv/bin/activate` 命令，或在 Windows 上运行 `.venv\Scripts\activate.bat` 。	应用程序开发人员
安装依赖项。	运行 `pip install -r requirements.txt` 命令。	应用程序开发人员

测试堆 CloudFormation 栈

Task	描述	所需技能
运行单元测试。	1. 运行 `pip install -r requirements-dev.txt` 命令。 2. **Important** （可选）运行`cdk synth --no-staging > template.yml` 命令以生成 CloudFormation 堆栈。您可以检查堆栈，但要避免生成暂存的资源和工件。 3. 运行 `pytest` 命令以运行所有单元测试。 4. （可选）运行 `pytest tests/unit/<test_filename>` 命令对特定文件运行测试。	应用程序开发人员、测试工程师

部署堆 CloudFormation 栈

Task	描述	所需技能
设置引导环境。	按照 AWS 文档中 Bootstrapping 中的说明在将要部署 CloudFormation 堆栈的每个 AWS 区域中引导环境，以便部署 AWS CDK。	应用程序开发人员、 DevOps 工程师、数据工程师

Task	描述	所需技能
	Note 此步骤要求您拥有具有编程访问权限的证书。	
部署 CloudFormation 堆栈。	运行 `cdk deploy` 命令构建堆栈并将其部署到 Amazon Web Services account。	应用程序开发人员、 DevOps 工程师、AWS DevOps

清理环境的资源

Task	描述	所需技能
删除 CloudFormation 堆栈并移除关联的资源。	要删除已创建的 CloudFormation 堆栈并移除所有关联的资源，请运行 de `run cdk` stroy 命令。	应用程序开发人员

更多模式

- 使用 Athena 访问、查询和联接 Amazon DynamoDB 表
- 在 Amazon DynamoDB 中聚合数据，以便在 Athena 中进行 ML 预测
- 使用 GitHub 操作在亚马逊自动 CodeGuru 审查 AWS CDK Python 应用程序
- 自动执行 AWS 资源评测
- 使用 AWS SAM 自动部署嵌套应用程序
- 在跨区域自动复制 Amazon RDS 实例 AWS 账户
- 使用 DynamoDB TTL 自动将项目归档到 Amazon S3
- 自动检测变化并为 monorepo 启动不同的 CodePipeline 管道 CodeCommit
- 使用 DevOps 实践和 AWS Cloud9 构建具有微服务的松散耦合架构
- 在 Amazon 服务中构建多租户无服务器架构 OpenSearch
- 在 Amazon Web Services Cloud 中构建高级大型机文件查看器
- 通过 Amazon Web Services 计算风险价值 (VaR)
- 跨不同 Amazon Web Services account 和 AWS 区域 复制 AWS Service Catalog 产品
- 自动为 Java 和 Python 项目创建动态 CI 管道
- 使用 CQRS 和事件溯源将整体分解为微服务
- 将基于 React 的单页应用程序部署到 Amazon S3 CloudFront
- 使用私有端点和应用程序负载均衡器在内部网站上部署 Amazon API Gateway API
- 部署和调试 Amazon EKS 集群
- 使用基础设施即代码，在 Amazon Web Services Cloud 上部署和管理无服务器数据湖
- 使用 Terraform 和 Amazon AWS Bedrock 部署 RAG 用例
- 使用 Amazon Bedrock 代理和知识库开发基于聊天的全自动助手
- 使用 RAG 和提示开发基于 AI 聊天的高级生成式 AI 助手 ReAct
- 使用 Step Functions 通过 IAM Access Analyzer 动态生成 IAM policy
- 确保在启动时启用 Amazon EMR 日志记录到 Amazon S3
- 估算按需容量的 DynamoDB 表成本
- 使用 Amazon Personalize 生成个性化和重新排名的推荐
- 使用 AWS Glue 作业和 Python 生成测试数据
- 从 SQL Server 迁移到 PostgreSQL 时，对 PII 数据实施 SHA1 哈希处理
- 使用 AWS Step Functions 实施无服务器 saga 模式

- 通过在多个 AWS 区域、账户中启用 Amazon DevOps Guru 并 OUs 使用 AWS CDK 来提高运营绩效
- 使用 Step Functions 和 Lambda 代理函数在 AWS 账户上启动 CodeBuild 项目
- 使用 AWS Glue 将 Apache Cassandra 工作负载迁移到亚马逊密钥空间
- 监控多个共享的 Amazon 系统映像的使用情况 AWS 账户
- 使用 AWS CDK 和 GitHub 操作工作流程优化多账户无服务器部署
- 使用 AWS Step Functions 编排 ETL 管道，包含验证、转换和分区
- 使用 Amazon Athena 使用 SQL 查询亚马逊 DynamoDB 表
- 使用 AWS Fargate 大规模运行事件驱动型和计划性工作负载
- 使用亚马逊通过 VPC 在 Amazon S3 存储桶中提供静态内容 CloudFront
- 使用自动化工作流程简化 Amazon Lex 机器人的开发和部署
- 使用 AWS Lambda 以六边形架构构建 Python 项目
- 将自然语言翻译成查询 DSL OpenSearch 和 Elasticsearch 查询
- 在多账户环境中关闭所有 Security Hub 成员账户的安全标准控件
- 跨账户将数据从 Amazon Redshift 集群卸载到亚马逊 S3
- 使用 AWS Fargate WaitCondition 挂钩构造协调资源依赖关系和任务执行
- 使用 Amazon Bedrock 代理通过基于文本的提示在 Amazon EKS 中自动创建访问入口控制

网络连接

主题

- 使用 AWS Transit Gateway 自动设置区域间对等互连
- 使用 AWS Transit Gateway 集中网络连接
- 使用 Applicati EnterpriseOne on Load Balancer 为 Oracle WebLogic JD Edwards 配置 HTTPS 加密
- 通过私有网络连接到 Application Migration Service 数据和控制面板
- 使用 AWS CloudFormation 自定义资源和 Amazon SNS 创建 Infoblox 对象
- 自定义 Amazon CloudWatch 提醒 AWS Network Firewall
- 使用 Terraform 在 AWS Wavelength 区域中部署资源
- 将 DNS 记录批量迁移至 Amazon Route 53 私有托管区
- 在 AWS 上从 F5 迁移到应用程序负载均衡器时修改 HTTP 标头
- 从多个私密访问中央 AWS 服务终端节点 VPCs
- 创建网络访问分析器在多个入站互联网访问中发现的报告 AWS 账户
- 在多账户 AWS 环境中为混合网络设置 DNS 解析
- 验证 ELB 负载均衡器是否需 TLS 终止
- 使用 Splunk 查看 AWS Network Firewall 日志和指标
- 更多模式

使用 AWS Transit Gateway 自动设置区域间对等互连

创建者：Ram Kandaswamy (AWS)

摘要

AWS Transit Gateway 通过中央枢纽连接虚拟私有云 (VPCs) 和本地网络。Transit Gateway 流量始终保持在全球亚马逊网络服务 (AWS) 主干上，不会穿越公共互联网，从而减少了威胁载体，例如常见的漏洞利用和分布式拒绝服务攻击。DDo

如果您需要在两个或多个 AWS 区域之间进行通信，则可以使用区域间 Transit Gateway 对等互连在不同区域公交网关之间建立对等连接。但是，使用 Transit Gateway 手动配置区域间对等互连可能是一个耗时的过程，需要很多步骤。此模式提供了自动流程，通过使用代码执行对等互连来删除这些手动步骤。如果您在多区域组织设置期间必须重复配置多个区域和 Amazon Web Services account，则可使用这种方法。

此模式使用的 AWS CloudFormation 堆栈包括 AWS Step Functions 工作流程、AWS Lambda 函数、AWS 身份和访问管理 (IAM) 角色以及亚马逊 CloudWatch 日志中的日志组。然后，您可以启动 Step Functions 执行并为中转网关创建区域间对等连接。要手动设置区域间对等互连，请参阅使用 AWS Transit Gateway VPCs 在不同 AWS 区域进行点对等。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 现有的 Amazon Simple Storage Service (Amazon S3) 存储桶
- 在请求者区域和接受者区域中创建和配置的中转网关。请求者区域是对等互连请求的发起地，接受者区域接受对等互连请求。有关更多信息，请参阅创建并接受 Amazon VPC 对等连接。
- VPCs，已在接受者和请求者区域安装和配置。有关创建 VPC 的步骤，请参阅 Amazon VPC 文档中的 Amazon VPC 入门的创建 VPC。
- VPCs 必须使用`addToTransitGateway`标签和`true`值。
- 您的安全组和网络访问控制列表 (ACLs) VPCs，根据您的要求进行配置。有关这方面的更多信息，请参阅 Amazon VPC 文档 ACLs中的您的 VPC 和网络的安全组。

AWS 区域和限制

- 只有某些 AWS 区域 支持区域内对等连接。有关支持区域间对等互连的区域的完整列表，请参阅 AWS Tran sit Gateway。 FAQs
- 在随附的示例代码中，假设请求者区域为 `us-east-2`，假设接受者区域为 `us-west-2`。如果要配置不同的区域，则必须在所有 Python 文件编辑这些值。要实现涉及两个以上区域的更复杂的设置，您可以更改 Step Function 以将区域作为参数传递给 Lambda 函数，并为每个组合运行该函数。

架构

图表显示了以下工作流：

1. 用户创建一个 AWS CloudFormation 堆栈。
2. AWS CloudFormation 创建了一台使用 Lambda 函数的 Step Functions 状态机。有关这方面的更多信息，请参阅 AWS Step Functions 文档中的创建使用 Lambda 的 Step Functions 状态机。
3. Step Functions 调用 Lambda 函数执行对等互连。
4. Lambda 函数在中转网关之间创建对等连接。
5. Step Functions 调用 Lambda 函数修改路由表。
6. Lambda 函数通过添加的无类域间路由 (CIDR) 块来修改路由表。 VPCs

Step Function 工作流

图表显示了以下工作流：

1. Step Functions 工作流调用 Lambda 函数以实现中转网关对等互连。
2. 计时器调用将等待一分钟。
3. 检索对等状态并将其发送到条件块。数据块负责循环。
4. 如未满足成功条件，则对工作流进行编码以进入计时器阶段。
5. 如果满足成功条件，则会调用 Lambda 函数修改路由表。在此次调用之后，Step Functions 工作流结束。

工具

- AWS CloudFormation — AWS CloudFormation 是一项可帮助您建模和设置 AWS 资源的服务。

- Amazon CloudWatch CloudWatch Logs — Logs 可帮助您集中管理您使用的所有系统、应用程序和 AWS 服务的日志。
- AWS Identity and Access Management (IAM) - IAM 是一项 Web 服务，可帮助您安全地控制对 Amazon Web Services 的访问。
- AWS Lambda – Lambda 在可用性高的计算基础设施上运行您的代码，执行计算资源的所有管理工作。
- AWS Step Functions - 借助 Step Functions，您可以轻松地将分布式应用程序组件作为可视化工作流中的一系列步骤进行协调。

操作说明

自动对等功能

Task	描述	所需技能
将附加文件上传到 S3 存储桶。	登录 AWS 管理控制台，打开 Amazon S3 控制台，然后将 `modify-transit-gateway-routes.zip` 、`peer-transit-gateway.zip` 、和 `get-transit-gateway-peering-status.zip` 文件（附件）上传到您的 S3 存储桶。	常规 AWS
创建 AWS CloudFormation 堆栈。	运行以下命令使用`transit-gateway-peering.json` 文件（附后）创建 AWS CloudFormation 堆栈： `aws cloudformation create-stack --stack-name myteststack --template-body file://sampletemplate.json`	DevOps 工程师

Task	描述	所需技能
	AWS CloudFormation 堆栈创建 Step Functions 工作流程、Lambda 函数、IAM 角色和 CloudWatch 日志组。 确保 AWS CloudFormation 模板引用了包含您之前上传的文件的 S3 存储桶。 **Note** 您也可以使用 AWS CloudFormation 控制台创建堆栈。有关这方面的更多信息，请参阅 AWS CloudFormation 文档中的在 AWS CloudFormation 控制台上创建堆栈。	
在 Step Functions 中开始新执行。	Step Functions 控制台会打开 New execution（新执行）页面。Step Functions 调用 Lambda 函数，并为中转网关创建对等连接。您不需要输入 JSON 文件。确认附件可用且连接类型是否为 对等连接。 有关这方面的更多信息，请参阅 AWS Step Functions 文档中的AWS Step Functions 入门中的开始新执行。	DevOps 工程师，通用 AWS

Task	描述	所需技能
验证路由表的路由。	在中转网关之间建立区域间对等互连。路由表将使用对等区域 VPC 的 IPv4 CIDR 区块范围进行更新。 打开 Amazon VPC 控制台，然后在路由表中选择与中转网关连接附件对应的关联选项卡。验证对等互连区域 VPC CIDR 区块范围。 有关详细步骤和说明，请参阅 Amazon VPC 文档中的 关联中转网关路由表。	网络管理员

相关资源

- Step Functions 的执行数量
- 中转网关对等连接挂载
- 使用 AWS Transit Gateway VPCs 在不同的 AWS 区域进行点对点
- 使用 AWS Transit Gateway 在 AWS 区域 VPCs 之间互连-演示（视频）

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS Transit Gateway 集中网络连接

由 Mydhili Palagummi (AWS) 和 Nikhil Marrapu (AWS) 编写

摘要

此模式描述了最简单的配置，在该配置中，AWS Transit Gateway 可用于将本地网络连接到 AWS 区域内多个 AWS 账户中的虚拟私有云 (VPCs)。使用此设置，您可以建立一个连接区域中多个 VPC 网络和本地网络的混合网络。这是通过使用运输网关和与本地网络的虚拟专用网络（VPN）连接来完成的。

先决条件和限制

先决条件

- 托管网络服务的账户，作为 AWS Organizations 组织的会员账户管理
- VPCs 在多个 AWS 账户中，没有重叠的无类域间路由 (CIDR) 块

限制

此模式不支持在特定网络 VPCs 或本地网络之间隔离流量。连接到运输网关的所有网络都将能够相互联系。若要隔离流量，您需要在公交网关上使用自定义路由表。此模式仅使用单个默认中转网关路由表（这是最简单的配置）来连接 VPCs 和本地网络。

架构

目标技术堆栈

- AWS Transit Gateway
- AWS Site-to-Site VPN
- VPC
- AWS Resource Access Manager (AWS RAM)

目标架构

工具

Amazon Web Services

- AWS Resource Access Manager (AWS RAM) 可帮助您从 AWS Organizations 在 Amazon Web Services account、组织单位或整个组织之间安全地共享资源。
- AWS Transit Gateway 是一个连接虚拟私有云 (VPCs) 和本地网络的中心枢纽。

操作说明

在网络服务账户创建中转网关

Task	描述	所需技能
创建中转网关。	在您要托管网络服务的 Amazon Web Services account 中，在目标 AWS 区域创建传输网关。有关说明，请参阅创建中转网关。请注意以下几点： • 选择默认路由表关联。 • 选择默认路由表传播。	网络管理员

将中转网关连接至本地网络

Task	描述	所需技能
为 VPN 连接设置客户网关设备。	客户网关设备连接在传输网关和您的本地网络之间的 Site-to-Site VPN 连接的本地端。有关更多信息，请参阅 AWS Site-to-Site VPN 文档中的您的客户网关设备。识别或启动支持的本地客户设备，并记录其公共IP地址。VPN 配置将在本操作说明的稍后部分完成。	网络管理员

Task	描述	所需技能
在网络服务账户中，创建到中转网关的 VPN 连接。	要设置连接，请为运输网关创建 VPN 附件。有关说明，请参阅公交网关 VPN 附件。	网络管理员
在本地网络中的客户网关设备上配置 VPN。	下载与传输网关关联的 Site-to-Site VPN 连接的配置文件，并在客户网关设备上配置 VPN 设置。有关说明，请参阅下载配置文件。	网络管理员

在网络服务账户中与其他 AWS 账户或您的组织共享运输网关

Task	描述	所需技能
在 AWS Organizations 管理账户，开启共享。	若要与您的组织或某些组织单位共享公交网关，请在 AWS Organizations 中开启共享。否则，您将需要分别分别为每个账户共享运输网关。有关说明，请参阅在 AWS Organizations 内启用资源共享。	AWS 系统管理员
在网络服务账户中创建中转网关资源共享。	要允许组织 VPCs 内的其他 AWS 账户连接到公交网关，请在网络服务账户中使用 AWS RAM 控制台共享公交网关资源。有关说明，请参阅创建资源共享。	AWS 系统管理员

Connect VPCs 连接到公交网关

Task	描述	所需技能
在个人账户中创建 VPC 附件。	在共享过境网关的账户，创建中转网关 VPC 附件。有关说明，请参阅创建到 VPC 的中转网关连接。	网络管理员
接受 VPC 附件请求。	在网络服务账户中，接受运输网关 VPC 附件请求。有关说明，请参阅接受共享附件。	网络管理员

配置路由

Task	描述	所需技能
在个人账户中配置路由 VPCs。	在每个单独的账户 VPC 中，将路由添加到本地网络和其他 VPC 网络，使用 Transit Gateway 作为目标。有关说明，请参阅在路由表中添加和删除路由。	网络管理员
在中转网关路由表中配置路由。	应传播来自 VPCs 和 VPN 连接的路由，并应出现在传输网关的默认路由表中。如果需要，可在传输网关默认路由表中创建任何静态路由（例如，静态 VPN 连接的静态路由）。有关说明，请参阅创建静态路由。	网络管理员
添加安全组和网络访问控制列表 (ACL) 规则。	对于 VPC 中的 EC2 实例和其他资源，请确保安全组规则和网络 ACL 规则允许本地网络之间的 VPCs 流量。有关说明，	网络管理员

Task	描述	所需技能
	请参阅使用安全组控制资源流量和在 ACL 中添加和删除规则。	

测试连接

Task	描述	所需技能
测试之间的连通性 VPCs。	确保网络 ACL 和安全组允许 Internet 控制消息协议 (ICMP) 流量，然后从 VPC 中的实例执行 ping 操作，也连接至中转网关的VPC。	网络管理员
测试 VPCs 与本地网络之间的连接。	确保网络 ACL 规则、安全组规则和所有防火墙允许 ICMP 流量，然后在本地网络和中的 EC2 实例之间执行 ping 操作。VPCs必须先从本地网络启动网络通信，才能使 VPN 连接处于 UP 状态。	网络管理员

相关资源

- 构建可扩展的安全多 VPC AWS 网络基础设施（AWS 白皮书）
- 使用共享资源（AWS RAM 文档）
- 使用中转网关（AWS Transit Gateway 文档）

使用 Applicati EnterpriseOne on Load Balancer 为 Oracle WebLogic JD Edwards 配置 HTTPS 加密

创建者：Thanigaivel Thirumalai (AWS)

摘要

此模式说明了如何在 Oracle JD Edwards 中为 Oracle 工作负载 EnterpriseOne 上的 SSL 卸载配置 HTTPS 加密。 WebLogic 这种方法对用户浏览器和负载均衡器之间的流量进行加密，从而减轻 EnterpriseOne 服务器的加密负担。

许多用户使用 AWS App lication Load Balancer 水平扩展 JA V EnterpriseOne A 虚拟机 (JVM) 层。负载均衡器充当客户端的单一联系点，并将传入流量分配给多个 JVMs客户端。或者，负载均衡器可以将流量分配到多个可用区并提高的可用性 EnterpriseOne。

此模式中描述的过程在浏览器和负载均衡器之间配置加密，而不是加密负载均衡器与之间的流量。EnterpriseOne JVMs这种方法被称为 SSL 分流。将 SSL 解密过程从 EnterpriseOne Web 或应用程序服务器转移到 Application Load Balancer 可以减轻应用程序端的负担。负载均衡器终止 SSL 后，未加密的流量将路由到 AWS 上的应用程序。

Oracle JD Edwards EnterpriseOne 是一款企业资源规划 (ERP) 解决方案，适用于制造、构造、分销、维修或管理产品或实物资产的组织。JD Edwards EnterpriseOne 支持各种硬件、操作系统和数据库平台。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 一个 AWS 身份和访问管理 (IAM) 角色，有权拨打 AWS 服务调用和管理 AWS 资源
- SSL 证书

产品版本

- 此模式已在 Oracle WebLogic 12c 中进行了测试，但您也可以使用其他版本。

架构

有多种方法可以执行 SSL 分流。此模式使用应用程序负载均衡器和 Oracle HTTP Server (OHS)，如下图所示。

下图显示了 JD Edwards EnterpriseOne、Application Load Balancer 和 Java 应用程序服务器 (JAS) JVM 布局。

工具

Amazon Web Services

- 应用程序负载均衡器将传入的应用程序流量分配到多个可用区域中的多个目标，例如亚马逊弹性计算云（Amazon EC2 实例）。
- AWS Certificate Manager (ACM) 帮助您创建、存储和续订公有及私有 SSL/TLS X.509 证书和密钥，这些证书和密钥可保护您的 AWS 网站和应用程序。
- Amazon Route 53 是一种可用性高、可扩展性强的 DNS Web 服务。

最佳实践

- 有关 ACM 最佳实践，请参阅 ACM 文档。

操作说明

设置 WebLogic 和 OHS

Task	描述	所需技能
安装与配置 Oracle 组件。	1. 按标准安装过程安装 Fusion Middleware Infrastructure。该程序可帮助您安装和配置 WebLogic 域。有关说明，请参阅 Oracle 文档。	JDE CNC，管理员 WebLogic

Task	描述	所需技能
	2. 按照标准安装进程安装 OHS。有关说明，请参阅 Oracle 文档。 3. 安装完成后，启动配置向导（`config.sh` 文件）以配置 OHS。 • 您可更新现有域或创建一个新域。此模式假定您正在更新现有域。 • 对于可用模板，请选择 Oracle Enterprise Manager-Restricted JRF 和 Oracle HTTP Server (Restricted JRF)。选择这些 Java 所需文件 (JRF) 选项，可消除与外部数据库的连接。 • 对于托管服务器、集群、服务器模板、一致性集群、计算机、向计算机分配服务器、虚拟目标和分区，接受默认配置值，然后选择下一步进入下一类别。 • 填写 OHS 实例的配置详细信息（例如，管理员主机和端口、监听地址和端口、服务器名称）（例如 `ohs1`）。	

Task	描述	所需技能
在域级别启用该 WebLogic 插件。	该 WebLogic 插件是负载平衡所必需的。要启用该插件，请执行以下操作： 1. 使用以下链接登录 WebLogic 管理控制台： `http://<WeblogicServer>:<Adminport>/console` 2. 选择锁定并编辑，然后选择配置、Web 应用程序。 3. 选择 “启用WebLogic 插件”（复选框或下拉选项）。 4. 选择保存并激活更改。	JDE CNC，管理员 WebLogic

Task	描述	所需技能
编辑配置文件。	该mod_wl_ohs.conf 文件将来自 OHS 的代理请求配置为。 WebLogic 1. 编辑此文件。它的位置： $ORACLE_HOME/user_projects/domains/ 例如： /home/oracle/Oracle/Middleware/Oracle_Home/user_projects/domains/base_domain/config/fmwconfig/components/OHS/instances/ohs1 2. 添加 WebLogic 主机 (WebLogicHost) 和端口 (WebLogicPort) 值（此模式假设本地主机和端口 8000。 ） 3. 添加 WLProxySSL 和 WLProxySSLPassThrough 值，如下所示： <VirtualHost *:8000> <Location /jde> WLSRequest On SetHandler weblogic-handler WebLogicHost localhost WebLogicPort 8000 WLProxySSL On	JDE CNC，管理员 WebLogic

Task	描述	所需技能

	<pre>WLProxySSLPassThrough
	 On
	</Location>
	</VirtualHost></pre>	

Task	描述	所需技能

使用 Enterprise Manager 启动 OHS。	1. 使用以下链接登录 Enterprise Manager Fusion Middleware：
	
	`http://<WeblogicServer>:<Adminport>/em/`
	2. 在目标导航中的 HTTP 服务器，选择 OHS 实例（例如 `ohs1`）。
	3. 选择关闭和启动以重启 OHS 实例。
	4. OHS 设置完成后，您可以使用带有端口 8000 的 EnterpriseOne HTTP 服务器主机名而不是服务器主机名来连接到 HTML 客户端。EnterpriseOne
	• 旧链接：`http://<Webserver>:80/jde/owhtml`
	• 新链接：`http://<HTTP server or web server>:8000/jde/owhtml`
	
	如果您使用的端口不是默认 Oracle HTTP 端口，请编辑 `httpd.conf` 文件以在两个位置为该端口添加侦听器：
	
	`#[Listen] OHS_LISTEN_PORT`
		JDE CNC，管理员 WebLogic

Task	描述	所需技能

	`Listen 8000`
	
	和：
	
	`#`
	`ServerName <Weblogic Server1>:8000`	

配置应用程序负载均衡器。

Task	描述	所需技能
设置组。	1. 为 HTTP 服务器端口 8000 创建目标组。 2. 使用相同端口注册目标组下的目标。 3. 检查目标的状态以确认它们是否健康。 4. 按需要配置运行状况检查设置。 有关详细说明，请参阅弹性负载均衡文档。	AWS 管理员
设置负载均衡器。	1. 创建具有默认属性和必要虚拟私有云（VPC）、安全组和子网的应用程序负载均衡器。有关说明，请参阅弹性负载均衡文档。 2. 为 HTTPS 443 添加侦听器条目，然后将其转发到上一步骤中创建的目标组。（有关说明，请参阅弹性负载	AWS 管理员

Task	描述	所需技能
	均衡文档。）HTTPS 侦听器需要 SSL 证书。您可从 ACM 中选择证书或上传证书。 3. 对于两个侦听器，请按照弹性负载均衡文档中的说明启用粘性。	
添加一条 Route 53 (DNS) 记录。	（可选）您可为子域名添加 Amazon Route 53 的 DNS 记录。这条记录将指向您的应用程序负载均衡器。有关说明，请参阅 Route 53 文档。	AWS 管理员

故障排除

事务	解决方案
HTTP 服务器未显示。	如果 HTTP 服务器未出现在 Enterprise Manager 控制台的目标导航列表，请执行以下步骤： 1. 在 “WebLogic 域” “管理” 下，选择 OHS 实例。 2. 选择创建以创建新的 OHS 实例。 3. 提供实例名称，然后选择确定以创建该实例。 创建实例并激活更改后，您将能够在目标导航面板看到 HTTP 服务器。

相关资源

AWS 文档

- 应用程序负载均衡器
- 使用公有托管区
- 使用私有托管区

Oracle 文档：

- Oracle WebLogic 服务器代理插件概述
- 使用基础架构安装程序安装 WebLogic 服务器
- 安装和配置 Oracle HTTP Server

通过私有网络连接到 Application Migration Service 数据和控制面板

由 Dipin Jain (AWS) 和 Mike Kuznetsov (AWS) 编写

摘要

此模式说明了如何使用接口 VPC 终端节点连接到安全私有网络上的 AWS Application Migration Service 数据平面和控制平面。

应用程序迁移服务是一种高度自动化 lift-and-shift（重新托管）的解决方案，可简化、加快应用程序迁移并降低向其迁移的成本。 AWS它使公司能够重新托管大量物理、虚拟或云服务器，而不会出现兼容性问题、性能中断或长时间的割接窗口。应用程序迁移服务可从中获得 AWS Management Console。这样可以与其他 AWS 服务公司（例如 Amazon CloudWatch 和 AWS Identity and Access Management (IAM)）进行无缝集成。 AWS CloudTrail

您可以使用应用程序迁移服务中的服务 AWS VPN AWS Direct Connect或 VPC 对等互连，通过私有连接从源数据中心连接到数据平面，即连接到用作目标 VPC 中数据复制暂存区域的子网。您还可以使用由提供支持的接口 VPC 终端节点通过私有网络 AWS PrivateLink 连接到应用程序迁移服务控制平面。

先决条件和限制

先决条件

- 暂存区域子网-在设置应用程序迁移服务之前，请创建一个子网，用作从源服务器复制到 AWS （即数据平面）的数据的暂存区域。首次访问 Application Migration Service 控制台时，必须在复制设置模板中指定此子网。您可以在“复制设置”模板中为特定源服务器覆盖此子网。尽管您可以使用自己的现有子网 AWS 账户，但我们建议您为此目的创建一个新的专用子网。
- 网络要求 — 由应用程序迁移服务在您的暂存区域子网中启动的复制服务器必须能够将数据发送到应用程序迁移服务 API 终端节点`https://mgn.<region>.amazonaws.com/`，其中`<region>`是 AWS 区域 您要复制到的代码（例如）。`https://mgn.us-east-1.amazonaws.com`下载应用程序迁移服务软件需要亚马逊简单存储服务 (Amazon S3) S URLs ervice 服务。
  - AWS 复制代理安装程序应有权访问您在应用程序迁移服务中使用的亚马逊简单存储服务 (Amazon S3) 存储桶 URL。 AWS 区域
  - 暂存区子网应有权访问 Amazon S3。
  - 安装了 AWS Replication Agent 的源服务器必须能够将数据发送到暂存区域子网中的复制服务器和位于的应用程序迁移服务 API 端点。`https://mgn.<region>.amazonaws.com/`

下表列出了所需端口。

源	目标位置	端口	有关更多信息，请参阅
源数据中心	亚马逊 S3 服务 URLs	443（TCP）	通过 TCP 端口 443 进行通信
源数据中心	AWS 区域应用程序迁移服务的特定控制台地址	443（TCP）	源服务器与 Application Migration Service 之间通过 TCP 端口 443 进行通信
源数据中心	暂存区子网	1500 (TCP)	源服务器与暂存区子网之间通过 TCP 端口 1500 进行通信
暂存区子网	AWS 区域应用程序迁移服务的特定控制台地址	443（TCP）	暂存区子网与 Application Migration Service 之间通过 TCP 端口 443 进行通信
暂存区子网	亚马逊 S3 服务 URLs	443（TCP）	通过 TCP 端口 443 进行通信
暂存区子网	子网的亚马逊弹性计算云 (Amazon EC2) 终端节点 AWS 区域	443（TCP）	通过 TCP 端口 443 进行通信

限制

应用程序迁移服务目前并非在所有操作系统中 AWS 区域 都可用。

- 支持的 AWS 区域
- 支持的操作系统

架构

下图展示了典型迁移的网络架构。有关此架构的更多信息，请参阅 Application Migration Service 文档和 Application Migration Service 架构和网络架构视频。

以下详细视图显示了暂存区 VPC 中用于连接 Amazon S3 和 Application Migration Service 的接口 VPC 端点的配置。

工具

- AWS Application Migration Service简化、加快并降低在上重新托管应用程序的成本。 AWS
- 接口 VPC 终端节点使您 AWS PrivateLink 无需互联网网关、NAT 设备、VPN 连接或 AWS Direct Connect 连接即可连接到由其提供支持的服务。VPC 中的实例无需公有 IP 地址便可与服务中的资源通信。VPC 和其他服务之间的通信不会离开 Amazon 网络。

操作说明

为应用程序迁移服务、Amazon 和 Amazon EC2 S3 创建终端节点

Task	描述	所需技能
为 Application Migration Service 配置接口端点。	源数据中心和暂存区 VPC 通过您在目标暂存区 VPC 中创建的接口端点以私有方式连接到 Application Migration Service 控制面板。要创建端点： 1. 打开亚马逊虚拟私有云（亚马逊 VPC）控制台。 2. 在导航窗格中，选择端点、创建端点。 3. 对于 Service category（服务类别），选择 AWS 服务。 4. 对于服务名称，输入 `com.amazonaws.<region>.mgn` 。对于类型，选择接口。	迁移主管

Task	描述	所需技能
	5. 对于 VPC，选择要在其中创建端点的目标暂存区 VPC。 6. 对于子网，选择要在其中创建端点网络接口的子网。 7. 要为接口端点启用私有 DNS，请在其他设置部分中选择启用 DNS 名称。 8. 选择允许通过 TCP 443 从暂存区 VPC 子网进入的安全组。 9. 选择创建端点。 有关更多信息，请参阅 Amazon VPC 文档中的 AWS 服务 使用接口 VPC 终端节点访问和。	
为 Amazon 配置接口终端节点 EC2。	暂存区域 VPC 通过您在目标暂存区域 VPC 中创建的接口终端节点私下连接到 Amazon EC2 API。若要创建端点，请按照上一篇文章中提供的说明进行操作。 • 对于服务名称，输入 `com.amazonaws.<region>.ec2` 。对于类型，选择接口。 • 安全组必须允许来自暂存区 VPC 子网的入站 HTTPS 流量通过端口 443。 • 在其他设置部分中，选择启用 DNS 名称。	迁移主管

Task	描述	所需技能
配置 Amazon S3 的接口端点。	源数据中心和暂存区 VPC 通过您在目标暂存区 VPC 中创建的接口端点以私有方式连接到 Amazon S3 API。若要创建端点，请按照第一篇文章提供中的说明进行操作。 • 对于服务名称，输入 `com.amazonaws.<region>.s3` 。对于类型，选择接口。 • VPC 安全组必须允许来自暂存区 VPC 子网的入站 HTTPS 流量通过端口 443。 • 在其他设置部分中，清除启用 DNS 名称。Amazon S3 接口端点不支持私有 DNS 名称。 **Note** 您之所以使用接口终端节点，是因为网关终端节点连接无法扩展到 VPC 之外。（有关详细信息，请参阅AWS PrivateLink 文档。）	迁移主管

Task	描述	所需技能
配置 Amazon S3 网关端点。	在配置阶段，复制服务器必须连接到 S3 存储桶才能下载 AWS 复制服务器的软件更新。但是，Amazon S3 接口端点不支持私有 DNS 名称，并且无法向复制服务器提供 Amazon S3 端点 DNS 名称。 要缓解此问题，请在暂存区子网所属的 VPC 中创建一个 Amazon S3 网关端点，并使用相关路由更新暂存子网的路由表。有关更多信息，请参阅 AWS PrivateLink 文档中的创建网关终端节点。	云管理员
配置本地 DNS 以解析端点的私有 DNS 名称。	应用程序迁移服务和 Amazon 的接口终端节点 EC2 具有可在 VPC 中解析的私有 DNS 名称。但是，您还需要配置本地服务器以解析这些接口端点的私有 DNS 名称。 配置这些服务器有多种方法。在这种模式中，我们通过将本地 DNS 查询转发到暂存区域 VPC 中的 Amazon Route 53 Resolver 入站终端节点来测试此功能。有关更多信息，请参阅 Route 53 文档中的解析 VPCs 与您的网络之间的 DNS 查询。	迁移工程师

通过私有链接连接到 Application Migration Service 控制面板

Task	描述	所需技能
使用安装 AWS 复制代理 AWS PrivateLink。	1. 将 AWS 复制代理下载到目标区域的私有 S3 存储桶。 2. 登录待迁移的源服务器。AWS 复制代理安装程序需要通过网络访问应用程序迁移服务和 Amazon S3 终端节点。由于您的本地网络不对应用程序迁移服务和 Amazon S3 公共终端节点开放，因此您必须使用在前面步骤中创建的接口终端节点的帮助下安装代理 AWS PrivateLink。 以下是 Linux 的示例： 1. 使用以下命令下载代理： `wget -O ./aws-replication-installer-init.py \ https://aws-application-migration-service-<aws_region>.bucket.<s3-endpoint-DNS-name>/latest/linux/aws-replication-installer-init.py` 例如，如果 Amazon S3 接口终端节点的 DNS 名称 AWS 区域 为`us-west-1` ，`vpce-009c`	迁移工程师

Task	描述	所需技能
	8b07adb052a11-qgf8q50y.s3.us-west-1.vpce.amazonaws.com 而为，则应使用以下命令： `wget -O ./aws-replication-installer-init.py \ https://aws-application-migration-service-us-west-1.bucket.vpce-009c8b07adb052a11-qgf8q50y.s3.us-west-1.vpce.amazonaws.com/latest/linux/aws-replication-installer-init.py` **Note** bucket是您必须在 Amazon S3 接口终端节点 DNS 名称之前添加的静态关键字。有关更多信息，请参阅 Amazon S3 文档。 2. 安装代理： • 如果您在为 Application Migration Service 创建接口端点时选择了启用 DNS 名称，请运行以下命令：	

Task	描述	所需技能

	<pre>sudo python3 aws-replication-installer-init.py \
	--region <aws_region> \
	--aws-access-key-id <access-key> \
	--aws-secret-access-key <secret-key> \
	--no-prompt \
	--s3-endpoint <s3-endpoint-DNS-name></pre>
	• 如果您在为 Application Migration Service 创建接口端点时未选择启用 DNS 名称，请运行以下命令：
	<pre>sudo python3 aws-replication-installer-init.py \
	--region <aws_region> \
	--aws-access-key-id <access-key> \
	--aws-secret-access-key <secret-key> \
	--no-prompt \
	--s3-endpoint <s3-endpoint-DNS-name> \
	--endpoint <mgn-endpoint-DNS-name></pre>	

Task	描述	所需技能
	有关更多信息，请参阅应用程序迁移服务文档中的 AWS Replication Agent 安装说明。 与应用程序迁移服务建立连接并安装 AWS 复制代理后，请按照应用程序迁移服务文档中的说明将源服务器迁移到目标 VPC 和子网。	

相关资源

Application Migration Service 文档

- 概念
- 迁移工作流
- 快速入门指南
- 常见问题解答
- 故障排除

其他资源

- 使用 VPC 接口终端节点在多账户架构中重新托管您的应用程序（AWS 规范性指导指南） AWS
- AWS Application Migration Service — 技术简介（AWS 培训和认证演练）
- AWS Application Migration Service 架构和网络架构（视频）

其他信息

对 Linux 服务器上的AWS 复制代理安装进行故障排除

如果您在 Amazon Linux 服务器上收到 gcc 错误，请配置软件包存储库并使用以下命令：

```
## sudo yum groupinstall "Development Tools"
```

使用 AWS CloudFormation 自定义资源和 Amazon SNS 创建 Infoblox 对象

由 Tim Sutton (AWS) 创建

摘要

注意： AWS Cloud9 不再向新客户开放。的现有客户 AWS Cloud9 可以继续照常使用该服务。了解更多

Infoblox 域名系统（DNS）、动态主机配置协议（DHCP）和 IP 地址管理（Infoblox DDI）使您能够集中并高效控制复杂的混合环境。借助 Infoblox DDI，除了使用相同的设备管理本地和 Amazon Web Services （AWS） 云上的 DNS 外，您还可以在一个权威 IP 地址管理器（IPAM）数据库中发现和记录所有网络资产。

此模式描述了如何使用 AWS CloudFormation 自定义资源通过调用 Infoblox WAPI 来创建 Infoblox 对象（例如 DNS 记录或 IPAM 对象）。有关 Infoblox WAPI 的详细信息，请参阅 Infoblox 文档中的 WAPI 文档。

通过使用此模式的方法，除了删除创建记录和预置网络的手动流程外，您还可以获得 AWS 和本地环境的 DNS 记录和 IPAM 配置的统一视图。您可以将此模式的方法用于以下用例：

- 在创建亚马逊弹性计算云 (Amazon EC2) 实例后添加 A 记录
- 创建应用程序负载均衡器后添加 CNAME 记录
- 创建虚拟私有云（VPC）后添加网络对象
- 提供下一个网络范围并使用该范围创建子网

您还可以扩展此模式并使用其他 Infoblox 设备功能，例如添加不同的 DNS 记录类型或配置 Infoblox vDiscovery。

该模式使用的 hub-and-spoke设计是，中心需要连接到 AWS 云上或本地的 Infoblox 设备，并使用 AWS Lambda 调用 Infoblox API。分支位于 AWS Organizations 中同一组织中的相同或不同账户中，并使用 AWS CloudFormation 自定义资源调用 Lambda 函数。

先决条件和限制

先决条件

- 现有的 Infoblox 设备或网格，安装在 AWS Cloud和/或本地，并配置了可以管理 IPAM 和 DNS 操作的管理员用户。有关此内容的详细信息，请参阅 Infoblox 文档中的关于管理员账户。

- 要在 Infoblox 设备上添加记录的现有 DNS 权威区域。有关此内容的更多信息，请参阅 Infoblox 文档中的配置权威区域。
- AWS Organizations 中的两个活动 Amazon Web Services account。一个账户是中心账户，另一个账户是分支账户。
- 中心账户和分支账户必须位于同一个 AWS 区域。
- 中心账户的 VPC 必须连接到 Infoblox 设备；例如，通过使用 AWS Transit Gateway 或 VPC 对等连接。
- AWS 无服务器应用程序模型 (AWS SAM) Model，在本地安装并使用 AWS Cloud9 或 AWS 进行配置。 CloudShell
- `Infoblox-Hub.zip` 和 `ClientTest.yaml` 文件（附加），下载到包含 AWS SAM 的本地环境。

限制

- AWS CloudFormation 自定义资源的服务令牌必须来自创建堆栈的同一区域。我们建议您在每个区域中使用中心账户，而不是在一个区域中创建 Amazon Simple Notification Service（Amazon SNS）主题，然后在另一个区域调用 Lambda 函数。

产品版本

- Infoblox WAPI 版本 2.7

架构

下图演示了此模式的工作流程。

该图显示了此模式解决方案的以下组件：

1. AWS CloudFormation 自定义资源允许您在创建、更新或删除堆栈时 AWS CloudFormation 运行的模板中编写自定义配置逻辑。当您创建堆栈时，AWS CloudFormation 会向由 EC2 实例上运行的应用程序监控的 SNS 主题发送`create`请求。
2. 来自 AWS CloudFormation 自定义资源的 Amazon SNS 通知通过特定的 AWS 密钥管理服务 (AWS KMS) 密钥进行加密，并且仅限组织中的账户访问 Organizations。SNS 主题启动调用 Infoblox WAPI API 的 Lambda 资源。

3. Amazon SNS 调用以下 Lambda 函数，这些函数将 Infoblox WAPI 网址、用户名和密码 AWS Secrets Manager 亚马逊资源名称 () 作为环境变量：ARNs
   - `dnsapi.lambda_handler`— 从 AWS CloudFormation 自定义资源中接收`DNSName`、`DNSType`和`DNSValue`值，并使用这些值创建 DNS A 记录和别名记录。
   - `ipaddr.lambda_handler`— 从 AWS CloudFormation 自定义资源接收`VPCCIDR`、`TypeSubnetPrefix`、、和`Network Name`值，并使用这些值将网络数据添加到 Infoblox IPAM 数据库中，或者为自定义资源提供下一个可用于创建新子网的可用网络。
   - `describeprefixes.lambda_handler` – 使用 `"com.amazonaws."+Region+".s3"` 筛选条件调用 `describe_managed_prefix_lists` AWS API 以检索所需 `prefix ID`。

   > **Important**
   >
   > 这些 Lambda 函数是用 Python 编写的，彼此相似，但调用方式不同。 APIs

4. 您可以将 Infoblox 网格部署为物理、虚拟或基于云的网络设备。 它可以在本地部署，也可以使用一系列虚拟机管理程序作为虚拟设备部署，包括 VMware ESXi微软 Hyper-V、Linux KVM 和 Xen。您还可以使用亚马逊机器映像（AMI）在 Amazon Web Services Cloud 上部署 Infoblox 网格。
5. 该图显示了 Infoblox 网格的混合解决方案，该解决方案为 Amazon Web Services Cloud 和本地资源提供 DNS 和 IPAM。

技术堆栈

- AWS CloudFormation
- IAM
- AWS KMS
- AWS Lambda
- AWS SAM
- AWS Secrets Manager
- Amazon SNS
- Amazon VPC

工具

- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。

- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥，以帮助保护您的数据。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Organizations 是一项账户管理服务，使您可将多个 Amazon Web Services account 整合到您所创建的组织中并进行集中管理。
- AWS Secrets Manager 帮助您将代码中的硬编码凭证（包括密码）替换为对 Secrets Manager 的 API 调用，以便以编程方式检索密钥。
- AWS Serverless Application Model（AWS SAM）是一个开源框架，帮助您在 Amazon Web Services Cloud 中构建无服务器应用程序。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。

代码

您可以使用`ClientTest.yaml`示例 AWS CloudFormation 模板（附后）来测试 Infoblox 中心。您可以自定义 AWS CloudFormation 模板以包含下表中的自定义资源。

使用 Infoblox 分支自定义资源创建 A 记录	返回值：
	
	`infobloxref` – Infoblox 参考资料
	
	示例资源：
	
	`ARECORDCustomResource:`
	
	`Type: "Custom::InfobloxAPI"`
	
	`Properties:`
	
	`ServiceToken: !Sub arn:aws:sns:${AWS::Region}:${HubAccountID}:RunInfobloxDNSFunction`

```
    DNSName: 'arecordtest.compa
ny.com'

    DNSType: 'ARecord'

    DNSValue: '10.0.0.1'
```

使用 Infoblox 分支自定义资源创建 CNAME 记录

返回值：

`infobloxref` – Infoblox 参考资料

示例资源：

```
CNAMECustomResource:

  Type: "Custom::InfobloxAPI"

  Properties:

    ServiceToken: !Sub arn:aws:sns:
${AWS::Region}:${HubAccountID}:Ru
nInfoblox

    DNSFunction

    DNSName: 'cnametest.company.com'

    DNSType: 'cname'

    DNSValue: 'aws.amazon.com'
```

使用 Infoblox 分支自定义资源创建网络对象

返回值：

`infobloxref` – Infoblox 参考资料

`network` – 网络范围（与 VPCCIDR 相同）

示例资源：

```
VPCCustomResource:

  Type: 'Custom::InfobloxAPI'

  Properties:

    ServiceToken: !Sub  arn:aws:sns:
${AWS::Region}:${HubAccountID}:Ru
nInfobloxNextSubnetFunction

    VPCCIDR: !Ref VpcCIDR

    Type: VPC

    NetworkName: My-VPC
```

使用 Infoblox 分支自定义资源检索下一个可用子网

返回值：

`infobloxref` – Infoblox 参考资料

`network` – 子网的网络范围

示例资源：

```
Subnet1CustomResource:

  Type: 'Custom::InfobloxAPI'

  DependsOn: VPCCustomResource

  Properties:

    ServiceToken: !Sub  arn:aws:sns:
${AWS::Region}:${HubAccountID}:Ru
nInfobloxNextSubnetFunction

    VPCCIDR: !Ref VpcCIDR

    Type: Subnet

    SubnetPrefix: !Ref SubnetPrefix

NetworkName: My-Subnet
```

操作说明

创建和配置中心账户的 VPC

Task	描述	所需技能
创建与 Infoblox 设备连接的 VPC。	登录您的中心账户的 AWS 管理控制台，并按照 AWS Quick Start 中的 Amazon Web Services Cloud 上的 Amazon VPC 快速入门参考部署中的步骤创建 VPC。	网络管理员、系统管理员

Task	描述	所需技能
	Important VPC 必须与 Infoblox 设备具有 HTTPS 连接，我们建议您使用私有子网进行此连接。	
（可选）为私有子网创建 VPC 端点。	VPC 端点为您的私有子网提供与公共服务的连接。需要以下端点： • 亚马逊简单存储服务 (Amazon S3) 的网关终端节点，允许 Lambda 与 AWS 通信 CloudFormation • Secrets Manager 的接口端点，用于启用与 Secrets Manager 的连接 • AWS KMS 的接口端点，用于加密 SNS 主题和 Secrets Manager 密钥 有关为私有子网创建端点的更多信息，请参阅 Amazon VPC 文档中的 VPC 端点。	网络管理员、系统管理员

部署 Infoblox 中心

Task	描述	所需技能
构建 AWS SAM 模板。	1. 在包含 AWS SAM 的环境中运行 `unzip Infoblox-Hub.zip` 命令。	开发人员、系统管理员

Task	描述	所需技能
	2. 运行 `cd Hub/` 命令将目录更改为 Hub 目录。 3. 运行 `sam build` 命令以处理 AWS SAM 模板文件、应用程序代码以及任何特定于语言的文件和依赖项。`sam build` 命令还按照以下故事预期的格式和位置复制构建构件。	

Task	描述	所需技能

部署 AWS SAM 模板。	该`sam deploy`命令获取所需的参数并将其保存到`samconfig.toml` 文件中，将 AWS CloudFormation 模板和 Lambda 函数存储在 S3 存储桶中，然后将 AWS CloudFormation 模板部署到您的中心账户。
	
	以下示例代码演示如何部署 AWS SAM 模板：
	
	`$ sam deploy --guided`
	
	`Configuring SAM deploy`
	`==========`
	`Looking for config file [samconfig.toml] : Found`
	`Reading default arguments : Success`
	`Setting default arguments for 'sam deploy'`
	`=========`
	`Stack Name [Infoblox-Hub]:`
	`AWS Region [eu-west-1]:`
	`Parameter InfobloxUsername:`
	`Parameter InfobloxPassword:`
	`Parameter InfobloxIPAddress [xxx.xxx.xx.xxx]:`
		开发人员、系统管理员

Task	描述	所需技能
	Parameter AWSOrganisationID [o-xxxxxxxxx]: Parameter VPCID [vpc-xxxxxxxxx]: Parameter VPCCIDR [xxx.xxx.xxx.xxx/16]: Parameter VPCSubnetID1 [subnet-xxx]: Parameter VPCSubnetID2 [subnet-xxx]: Parameter VPCSubnetID3 [subnet-xxx]: Parameter VPCSubnetID4 []: #Shows you resources changes to be deployed and require a 'Y' to initiate deploy Confirm changes before deploy [Y/n]: y #SAM needs permission to be able to create roles to connect to the resources in your template Allow SAM CLI IAM role creation [Y/n]: n Capabilities [['CAPABILITY_NAMED_IAM']]: Save arguments to configuration file [Y/n]: y SAM configuration file [samconfig.toml]:	

Task	描述	所需技能
	SAM configura tion environment [default]: **Important** 您每次都必须使用该--guided选项，因为 Infoblox 登录凭据未存储在文件中。samconfig.toml	

相关资源

- 开始 WAPIs 使用 Postman（Infoblox 博客）
- 使用 BYOL 模型为 AWS 预置 vNIOS（Infoblox 文档）
- quickstart-aws-vpc（GitHub 存储库）
- describe_managed_prefix_lists（适用于 Python 的 AWS SDK 文档）

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

自定义 Amazon CloudWatch 提醒 AWS Network Firewall

由 Jason Owens (AWS) 创建

摘要

该模式可帮助您自定义由生成的Amazon CloudWatch 提醒 AWS Network Firewall。您可以使用预定义的规则或创建自定义规则来确定警报的消息、元数据和严重性。然后，您可以根据这些提醒采取行动，或者由其他亚马逊服务（例如亚马逊）自动回复 EventBridge。

在此模式中，您将生成与 Suricata 兼容的防火墙规则。Suricata 是一个开源威胁检测引擎。您首先创建简单的规则，然后对其进行测试以确认 CloudWatch 警报已生成并记录在案。成功测试规则后，您可以修改它们以定义自定义消息、元数据和严重性，然后再次测试以确认更新。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- AWS Command Line Interface (AWS CLI) 已在你的 Linux、macOS 或 Windows 工作站上安装和配置。有关更多信息，请参阅安装或更新 AWS CLI的最新版本。
- AWS Network Firewall 已安装并配置为使用 CloudWatch 日志。有关更多信息，请参阅记录来自的网络流量 AWS Network Firewall。
- 位于受网络防火墙保护的虚拟私有云 (VPC EC2) 私有子网中的亚马逊弹性计算云 (Amazon) 实例。

产品版本

- 对于的版本 1 AWS CLI，请使用 1.18.180 或更高版本。对于的版本 2 AWS CLI，请使用 2.1.2 或更高版本。
- Suricata 版本 5.0.2 中的 classification.config 文件。有关此配置文件的副本，请参阅其他信息部分。

架构

架构图显示了以下工作流程：

1. 私有子网中的 Amazon EC2 实例使用 c url 或 W get 发出请求。

2. Network Firewall 处理流量并生成警报。
3. Network Firewall 将记录的警报发送到 CloudWatch 日志。

工具

AWS 服务

- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和运行的应用程序 AWS 的指标。
- Amazon CloudWatch Lo gs 可帮助您集中所有系统和应用程序的日志， AWS 服务 这样您就可以监控它们并安全地将其存档。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行外壳中的命令进行交互。
- AWS Network Firewall是一项针对虚拟私有云 (VPCs) 中的状态托管网络防火墙以及入侵检测和防御服务 ()。 AWS Cloud

其他工具

- curl 是一个开源命令行工具和库。
- GNU Wget 是一个免费的命令行工具。

操作说明

创建防火墙规则和规则组

Task	描述	所需技能
创建规则。	1. 在文本编辑器中，创建要添加到防火墙的规则列表。每个规则必须位于单独的行上。`classtype` 参数中的值来自默认的 Suricata 分类配置文件。有关完整的配置文件内容，请参阅其他信息部分。以下是规则的两个示例。	AWS 系统管理员、网络管理员

Task	描述	所需技能

	<pre>alert http any
	 any -> any any
	 (content:"badstuff
	"; classtype:misc-
	activity; sid:3;
	 rev:1;)
	alert http any any
	 -> any any (content:
	"morebadstuff";
	 classtype:bad-unkn
	own; sid:4; rev:1;)</pre> 2. 将规则保存在名为 `custom.rules` 的文件中。	

Task	描述	所需技能

创建规则组。	在中 AWS CLI，输入以下命令。这将创建规则组。
	
	`# aws network-firewall create-rule-group \ --rule-group-name custom --type STATEFUL \ --capacity 10 --rules file://custom.rules \ --tags Key=environment,Value=development`
	
	下面是一个示例输出。记下 RuleGroupArn ，在后面的步骤中需要用到它。
	
	`{ "UpdateToken": "4f998d72-973c-490a-bed2-fc3460547e23", "RuleGroupResponse": { "RuleGroupArn": "arn:aws:network-firewall:us-east-2:1234567890:stateful-rulegroup/custom", "RuleGroupName": "custom", "RuleGroupId": "238a8259-9eaf-48bb-90af-5e690cf8c48b", "Type": "STATEFUL", "Capacity": 10, "RuleGroupStatus": "ACTIVE",`
		AWS 系统管理员

Task	描述	所需技能

	<pre>"Tags": [
	{
	"Key": "environment",
	"Value": "development"
	}
	]
	}</pre>	

更新防火墙策略

Task	描述	所需技能

获取防火墙策略的 ARN。	在中 AWS CLI，输入以下命令。这将返回防火墙策略的 Amazon 资源名称（ARN）。记录 ARN 以供稍后在此模式中使用。
	<pre># aws network-firewall describe-firewall \
	--firewall-name aws-network-firewall-anfw \
	--query 'Firewall.FirewallPolicyArn'</pre>
	以下是此命令返回的示例 ARN。
	<pre>"arn:aws:network-firewall:us-east-2:1234567890:firewall-policy/firewall-policy-anfw"</pre>
		AWS 系统管理员

Task	描述	所需技能

更新防火墙策略。	在文本编辑器中，复制并粘贴以下代码。将 <RuleGroupArn> 替换为在上一个长篇故事中记录的值。将该文件保存为 firewall-policy-anfw.json 。
	
	`{ "StatelessDefaultActions": [ "aws:forward_to_sfe" ], "StatelessFragmentDefaultActions": [ "aws:forward_to_sfe" ], "StatefulRuleGroupReferences": [ { "ResourceArn": "<RuleGroupArn>" } ] }`
	
	在中输入以下命令 AWS CLI。此命令需要更新令牌才能添加新规则。该令牌用于确认自您上次检索策略以来该策略未发生更改。
	
	`UPDATETOKEN=(`aws network-firewall describe-firewall-policy \`
		AWS 系统管理员

Task	描述	所需技能
	`--firewall-policy-name firewall-policy-anfw \ --output text --query UpdateToken`) `aws network-firewall update-firewall-policy \ --update-token $UPDATETOKEN \ --firewall-policy-name firewall-policy-anfw \ --firewall-policy file://firewall-policy-anfw.json`	

Task	描述	所需技能

确认策略更新。	（可选）如果您想确认已添加规则并查看策略格式，请在中输入以下命令 AWS CLI。
	
	`# aws network-firewall describe-firewall-policy \ --firewall-policy-name firewall-policy-anfw \ --query FirewallPolicy`
	
	下面是一个示例输出。
	
	`{ "StatelessDefaultActions": [ "aws:forward_to_sfe" ], "StatelessFragmentDefaultActions": [ "aws:forward_to_sfe" ], "StatefulRuleGroupReferences": [ { "ResourceArn": "arn:aws:network-firewall:us-east-2:1234567890:stateful-rulegroup/custom" } ] }`
		AWS 系统管理员

测试警报功能

Task	描述	所需技能

生成用于测试的警报。	1. 登录到防火墙子网中的测试工作站。
	2. 输入应生成警报的命令。例如，您可以使用 `wget` 或 `curl`。
	`wget -U "badstuff" http://www.amazon.com -o /dev/null`
	`curl -A "morebadstuff" http://www.amazon.com -o /dev/null`
		AWS 系统管理员

验证警报是否已记录。	1. 打开 CloudWatch 管理控制台。 2. 导航到正确的日志组和流。有关更多信息，请参阅查看发送到日志的 CloudWatch 日志数据（CloudWatch 日志文档）。 3. 确认记录的事件类似于以下示例。这些示例仅显示警报的相关部分。 示例 1 `"alert": { "action": "allowed", "signature_id": 3, "rev": 1,`	AWS 系统管理员

Task	描述	所需技能

	<pre>"signatur
	e": "",
	"category
	": "Misc activity",
	"severity
	": 3
	}</pre>
	示例 2
	<pre>"alert": {
	"action":
	"allowed",
	"signatur
	e_id": 4,
	"rev": 1,
	"signatur
	e": "",
	"category
	": "Potentially Bad
	Traffic",
	"severity
	": 2
	}</pre>	

更新防火墙规则和规则组

Task	描述	所需技能

更新防火墙规则。	1. 在文本编辑器中，打开 `custom.rules` 文件。
	2. 将第一条规则更改为与以下内容类似。此规则必须在文件中的一行中输入。
	<pre>alert http any any ->
	any any (msg:"Watch</pre>
		AWS 系统管理员

Task	描述	所需技能
	out - Bad Stuff!!"; content:"badstuff" ; classtype:misc-activity; priority: 2; sid:3; rev:2; metadata:custom-field-2 Danger!, custom-field More Info;) 这会对规则进行以下更改： • 添加 msg（Suricata 网站）字符串，该字符串提供有关签名或警报的文本信息。在生成的警报中，这会映射到签名。 • 将 misc-activity 的默认优先级（Suricata 网站）从 3 调整为 2。有关各种 classtypes 的默认值，请参阅其他信息部分。 • 将自定义元数据（Suricata 网站）添加到警报。这是添加到签名中的附加信息。建议使用键值对。 • 将 rev（Suricata 网站）从 1 更改为 2。这代表签名的版本。	

Task	描述	所需技能

更新规则组。	在中 AWS CLI，运行以下命令。使用防火墙策略的 ARN。这些命令获取更新令牌，并使用规则更改更新规则组。
	
	`# UPDATETOKEN=(`aws network-firewall \ describe-rule-group \ --rule-group-arn arn:aws:network-firewall:us-east-2:123457890:stateful-rulegroup/custom \ --output text --query UpdateToken`)`
	
	`# aws network-firewall update-rule-group \ --rule-group-arn arn:aws:network-firewall:us-east-2:1234567890:stateful-rulegroup/custom \ --rules file://custom.rules \ --update-token $UPDATETOKEN`
	
	下面是一个示例输出。
	
	`{ "UpdateToken": "7536939f-6a1d-414c-96d1-bb28110996ed", "RuleGroupResponse": { "RuleGroupArn": "arn:aws:network-f`
		AWS 系统管理员

Task	描述	所需技能

	<pre>irewall:us-east-2:
	1234567890:stateful-
	rulegroup/custom",
	"RuleGrou
	pName": "custom",
	"RuleGroupId":
	"238a8259-9eaf-48b
	b-90af-5e690cf8c48b",
	"Type":
	"STATEFUL",
	"Capacity": 10,
	"RuleGrou
	pStatus": "ACTIVE",
	"Tags": [
	{
	"Key":
	"environment",
	"Value":
	"development"
	}
	]
	}
	}</pre>	

测试更新的警报功能

Task	描述	所需技能

生成用于测试的警报。	1. 登录到防火墙子网中的测试工作站。
	2. 输入应生成警报的命令。例如，您可以使用 `curl`。
	<pre>curl -A "badstuff"
	http://www.amazon.
	com -o /dev/null</pre>
		AWS 系统管理员

Task	描述	所需技能
验证警报是否已更改。	1. 打开 CloudWatch 管理控制台。 2. 导航到正确的日志组和流。 3. 确认记录的事件类似于以下示例。该示例仅显示警报的相关部分。 `"alert": { "action": "allowed", "signature_id": 3, "rev": 2, "signature": "Watch out - Bad Stuff!!", "category": "Misc activity", "severity": 2, "metadata": { "custom-field": [ "More Info" ], "custom-field-2": [ "Danger!" ] } }`	AWS 系统管理员

相关资源

参考

- 从 AWS Network Firewall Slack 频道发送提醒（AWS 规范性指导）

- AWS 使用 Suricata 扩展威胁防御能力（AWS 博客文章）
- AWS Network Firewall（AWS 博客文章）的部署模型
- Suricata 元密钥（Suricata 文档）

教程和视频

- AWS Network Firewall 工作坊

其他信息

以下是 Suricata 5.0.2 中的分类配置文件。创建防火墙规则时将使用这些分类。

```
# config classification:shortname,short description,priority

config classification: not-suspicious,Not Suspicious Traffic,3
config classification: unknown,Unknown Traffic,3
config classification: bad-unknown,Potentially Bad Traffic, 2
config classification: attempted-recon,Attempted Information Leak,2
config classification: successful-recon-limited,Information Leak,2
config classification: successful-recon-largescale,Large Scale Information Leak,2
config classification: attempted-dos,Attempted Denial of Service,2
config classification: successful-dos,Denial of Service,2
config classification: attempted-user,Attempted User Privilege Gain,1
config classification: unsuccessful-user,Unsuccessful User Privilege Gain,1
config classification: successful-user,Successful User Privilege Gain,1
config classification: attempted-admin,Attempted Administrator Privilege Gain,1
config classification: successful-admin,Successful Administrator Privilege Gain,1

# NEW CLASSIFICATIONS
config classification: rpc-portmap-decode,Decode of an RPC Query,2
config classification: shellcode-detect,Executable code was detected,1
config classification: string-detect,A suspicious string was detected,3
config classification: suspicious-filename-detect,A suspicious filename was detected,2
config classification: suspicious-login,An attempted login using a suspicious username
 was detected,2
config classification: system-call-detect,A system call was detected,2
config classification: tcp-connection,A TCP connection was detected,4
config classification: trojan-activity,A Network Trojan was detected, 1
config classification: unusual-client-port-connection,A client was using an unusual
 port,2
config classification: network-scan,Detection of a Network Scan,3
```

```
config classification: denial-of-service,Detection of a Denial of Service Attack,2
config classification: non-standard-protocol,Detection of a non-standard protocol or
 event,2
config classification: protocol-command-decode,Generic Protocol Command Decode,3
config classification: web-application-activity,access to a potentially vulnerable web
 application,2
config classification: web-application-attack,Web Application Attack,1
config classification: misc-activity,Misc activity,3
config classification: misc-attack,Misc Attack,2
config classification: icmp-event,Generic ICMP event,3
config classification: inappropriate-content,Inappropriate Content was Detected,1
config classification: policy-violation,Potential Corporate Privacy Violation,1
config classification: default-login-attempt,Attempt to login by a default username and
 password,2

# Update
config classification: targeted-activity,Targeted Malicious Activity was Detected,1
config classification: exploit-kit,Exploit Kit Activity Detected,1
config classification: external-ip-check,Device Retrieving External IP Address
 Detected,2
config classification: domain-c2,Domain Observed Used for C2 Detected,1
config classification: pup-activity,Possibly Unwanted Program Detected,2
config classification: credential-theft,Successful Credential Theft Detected,1
config classification: social-engineering,Possible Social Engineering Attempted,2
config classification: coin-mining,Crypto Currency Mining Activity Detected,2
config classification: command-and-control,Malware Command and Control Activity
 Detected,1
```

使用 Terraform 在 AWS Wavelength 区域中部署资源

由 Zahoor Chaudhrey (AWS) 和 Luca Iannario (AWS) 创作

摘要

AWS Wavelength帮助您构建针对多接入边缘计算 (MEC) 应用程序进行了优化的基础架构。W@@ avelength Zones 是在通信服务提供商 (CSP) 的 5G 网络中嵌入 AWS 计算和存储服务的 AWS 基础设施部署。来自 5G 设备的应用流量无需离开电信网络即可到达在 Wavelength Zones 中运行的应用服务器。以下内容便于通过 Wavelength 进行网络连接：

- 虚拟私有云 (VPCs) — VPCs AWS 账户 可以扩展到跨多个可用区，包括 Wavelength 区域。亚马逊弹性计算云 (Amazon EC2) 实例和相关服务作为您的区域 VPC 的一部分出现。 VPCs 在亚马逊 Virtual Private Cloud（亚马逊 VPC）中创建和管理。
- 运营商网关 — 运营商网关支持从 Wavelength 区域的子网连接到 CSP 网络、互联网或 AWS 区域通过 CSP 的网络。运营商网关有两个用途。它允许来自特定位置的 CSP 网络的入站流量，并允许电信网络和互联网的出站流量。

此模式及其关联的 Terraform 代码可帮助您在波长区域中启动资源，例如亚马逊 EC2 实例、亚马逊弹性区块存储 (Amazon EBS) Block Store 卷 VPCs、子网和运营商网关。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 集成开发环境 (IDE)
- 选择进入目标 Wavelength 区域
- AWS Command Line Interface (AWS CLI)，已安装并配置
- Terraform 版本 1.8.4 或更高版本，已安装（Terraform 文档）
- Terraform P AWS rovider 版本 5.32.1 或更高版本，已配置（Terraform 文档）
- Git，已安装 (GitHub)
- 创建 Amazon VPC、Wavelength 和亚马逊资源的@@ 权限 EC2

限制

并非所有区域都 AWS 区域 支持 Wavelength Zones。有关更多信息，请参阅 Waveleng th 文档中的可用波长区域。

架构

下图显示了如何在 Wavelength 区域中创建子网和 AWS 资源。 VPCs 在 Wavelength 区域中包含子网的可以连接到运营商网关。运营商网关允许您连接到以下资源：

- 电信运营商网络上的 4G/LTE 和 5G 设备。
- 为特定的 Wavelength Zone 合作伙伴提供固定无线接入。有关更多信息，请参阅多路访问 AWS Wavelength。
- 公共互联网资源的出站流量。

工具

AWS 服务

- Amazon Virtual Private Cloud（亚马逊 VPC）可帮助您将 AWS 资源启动到您定义的虚拟网络中。该虚拟网络类似于您在数据中心中运行的传统网络，并具有使用 AWS的可扩展基础设施的优势。
- AWS Wavelength将 AWS Cloud 基础设施扩展到电信提供商的 5G 网络。这可以帮助您构建能够为移动设备和最终用户提供超低延迟的应用程序。

其他工具

- Terraform 是一款基础设施即代码 (IaC) 工具 HashiCorp ，可帮助您创建和管理云和本地资源。

代码存储库

此模式的代码可在使用 Terraform GitHub 创建 AWS Wavelength 基础架构存储库中找到。Terraform 代码部署了以下基础架构和资源：

- 一个 VPC
- A Wavelength Zone
- 波长（Wavelength）区域中的公共子网
- 波长（Wavelength Zone）中的运营商网关

- Wavelength 区域中的亚马逊 EC2 实例

最佳实践

- 在部署之前，请确认您使用的是最新版本的 Terraform 和. AWS CLI
- 使用持续集成和持续交付 (CI/CD) 管道部署 IaC。有关更多信息，请参阅博客上的 C AWS I/CD Pipeline 中管理 Terraform 状态文件的最佳实践。 AWS

操作说明

预调配基础设施

Task	描述	所需技能
克隆存储库。	输入以下命令，将 “使用 Terraform 创建 AWS Wavelength 基础架构” 存储库克隆到您的环境中。 `git clone git@github.com:aws-samples/terraform-wavelength-infrastructure.git`	DevOps 工程师
更新变量。	1. 导航到克隆的存储库。 `cd terraform-wavelength-infrastructure` 2. 使用任何文本编辑器，在根目录中创建一个名为 terraform.tfv ars 的文件。 3. 创建以下变量并输入其值： • `region = <enter Region name>`	DevOps 工程师，Terraform

Task	描述	所需技能
	• `vpc_cidr = <enter CIDR block used by VPC>` • `wavelength_subnet_cidr = <enter CIDR block for the subnet in the Wavelength Zone>` • `availabilityzone_wavelength = <enter Wavelength Zone name>` 4. 保存 terraform .tfvars 文件。	
初始化配置。	输入以下命令来初始化工作目录。 `terraform init`	DevOps 工程师，Terraform
预览 Terraform 计划。	输入以下命令，将目标状态与 AWS 环境的当前状态进行比较。此命令生成将要配置的资源的预览。 `terraform plan`	DevOps 工程师，Terraform

Task	描述	所需技能
验证并部署。	1. 查看 Terraform 计划中的配置更改，并确认要实现这些更改。 2. 输入以下命令以应用计划并创建基础架构。 `terraform apply` 3. 输入`yes`继续。Terraform 创建了在配置文件中声明的架构。有关架构的更多信息，请参阅此模式的 “目标架构” 部分。	DevOps 工程师，Terraform

验证并清理

Task	描述	所需技能
验证基础架构部署。	1. 如果您的公有子网中还没有 Amazon EC2 实例 AWS 区域，请创建一个。有关说明，请参阅启动你的 Linux 实例或启动你的 Windows 实例。您将使用此实例来测试从 Wavelength 区域 AWS 区域 到波长区域的连接。 2. 测试从中的实例到 Wavelen AWS 区域 gth 区域中的实例的连接。有关说明，请参阅 Wav elength 文档中的测试连接。	AWS DevOps， DevOps 工程师

Task	描述	所需技能
（可选）清理 基础架构。	如果您需要删除 Terraform 配置的所有资源，请执行以下操作： 1. 输入以下命令。 `terraform destroy` 2. 输入 `yes`以确认。	DevOps 工程师，Terraform

故障排除

事务	解决方案
与中的 Amazon EC2 实例的连接 AWS 区域。	请参阅对连接您的 Linux 实例进行故障排除或对连接到 Windows 实例进行故障排除。
连接 Wavelength 区域中的 Amazon EC2 实例。	参见排除我在 Wavelength 区域启动的 EC2 实例的 SSH 或 RDP 连接问题。
Wavelength 区域的容量。	参见 W avelength 区域的配额和注意事项。
从运营商网络到的移动或运营商连接 AWS 区域.	1. 验证运营商网关是否处于运行状态。执行以下操作： a. 打开 Amazon VPC 控制台。 b. 在导航窗格中，选择您的 VPCs。 c. 选择包含 Wavelength 区域的 VPC。 d. 在详细信息窗格中，对于运营商网关，确认已附加该值。 2. 验证连接到 Wavelength 区域中实例的所有弹性 IP 地址是否正常运行。执行以下操作： a. 打开 Amazon EC2 控制台。 b. 在导航窗格中，选择实例。 c. 在 Wavelength 区域中选择实例。

事务	解决方案
	d. 选择网络选项卡。 e. 确认 elastic network 接口已连接弹性 IP 地址。 3. 请联系运营商网络支持团队。

相关资源

- 什么是 AWS Wavelength？
- 如何 AWS Wavelength 运作
- 韧性在 AWS Wavelength

将 DNS 记录批量迁移至 Amazon Route 53 私有托管区

创建者：Ram Kandaswamy (AWS)

摘要

网络工程师和云管理员需要一种高效且简单的方法将域名系统（DNS）记录添加到 Amazon Route 53 中的私有托管区。使用手动方法将 Microsoft Excel 工作表中的条目复制到 Route 53 控制台中的适当位置非常繁琐且容易出错。此模式描述了一种自动化方法，可减少添加多个记录所需时间和精力。它还提供了一组可重复的步骤来创建多个托管区域。

这种模式使用亚马逊简单存储服务 (Amazon S3) Service 来存储记录。为了有效地处理数据，该模式使用 JSON 格式，因为它简单且能够支持 Python 字典（`dict` 数据类型）。

> **Note**
>
> 如果可以从系统生成区域文件，请考虑改用 R oute 53 导入功能。

先决条件和限制

先决条件

- 包含私有托管区记录的 Excel 工作表
- 熟悉不同类型的 DNS 记录，例如 A 记录、域名授权指针 (NAPTR) 记录和 SRV 记录（请参阅支持的 DNS 记录类型）
- 熟悉 Python 语言及其库

限制

- 该模式并未广泛覆盖所有用例场景。例如，change_resource_record_sets 调用并未使用 API 的所有可用属性。
- 在 Excel 工作表中，假定每行中的值是唯一的。每个完全限定域名 (FQDN) 的多个值应出现在同一行。如果情况并非如此，您应该修改此模式中提供的代码以执行必要的串联。
- 该模式使用适用于 Python 的 Amazon SDK (Boto3) 直接调用 Route 53 服务。您可以增强代码以使用`create_stack`和`update_stack`命令的 AWS CloudFormation 封装器，并使用 JSON 值填充模板资源。

架构

技术堆栈

- Route 53 私有托管区，用于路由流量
- Amazon S3，用于存储输出 JSON 文件

工作流由以下步骤组成，如上图所示，并在操作说明部分中进行了讨论：

1. 将包含记录集信息的 Excel 工作表上传至 S3 存储桶。
2. 创建并运行 Python 脚本，将 Excel 数据转换为 JSON 格式。
3. 从 S3 存储桶读取记录并清理数据。
4. 在您的私有托管区中创建记录集。

工具

- Route 53 — Amazon Route 53 是高度可用且可扩展的 DNS 网络服务，用于处理域注册、DNS 路由和运行状况检查。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项对象存储服务。您可以通过 Amazon S3 随时在 Web 上的任何位置存储和检索的任意大小的数据。

操作说明

为自动化操作准备数据

Task	描述	所需技能
创建一个 Excel 文件作为记录。	使用从当前系统导出的记录创建一个 Excel 工作表，其中包含记录所需列，例如完全限定域名 (FQDN)、记录类型、生存时间 (TTL) 和值。对于 NAPTR 和 SRV 记录，该值是多个属性的组合，因此请使用	数据工程师、Excel 技能

Task	描述	所需技能
	Excel `concat` 方法来组合这些属性。 FqdnN Recor e 值 TTL somet .exam org A 1.1.1.1 900	

验证工作环境。	在你的 IDE 中，创建一个 Python 文件，将 Excel 输入工作表转换为 JSON 格式。（你也可以使用亚马逊 SageMaker 笔记本来处理 Python 代码，而不是 IDE。）
	
	确认您使用的 Python 版本是 3.7 或更高版本。
	
	`python3 --version`
	
	安装 pandas 程序包。
	
	`pip3 install pandas --user`
		常规 AWS

Task	描述	所需技能

将 Excel 工作表数据转换为 JSON。	创建一个包含以下代码的 Python 文件，以将 Excel 转换为 JSON。
	
	`import pandas as pd`
	`data=pd.read_excel('./Book1.xls')`
	`data.to_json(path_or_buf='my.json', orient='records')`
	
	其中 Book1 是 Excel 工作表的名称，`my.json` 是输出 JSON 文件的名称。
		数据工程师，Python 技能

将 JSON 文件上传到 S3 存储桶。	将 `my.json`文件上传到 S3 存储桶。有关更多信息，请参阅 Amazon S3 文档中的创建存储桶。	应用程序开发人员

插入记录

Task	描述	所需技能

创建私有托管区	使用 create_hosted_zone API 及以下 Python 示例代码创建私有托管区。将 `hostedZoneName` 、`vpcRegion` 和 `vpcId` 参数值替换为您自己的值。
	
	`import boto3`
	`import random`
	`hostedZoneName ="xxx"`
	`vpcRegion = "us-east-1"`
		云架构师、网络管理员、Python 技能

Task	描述	所需技能

	<pre>vpcId="vpc-xxxx"
	route53_client =
	 boto3.client('rout
	e53')
	response = route53_c
	lient.create_hoste
	d_zone(
	        Name= hostedZon
	eName,
	        VPC={
	            'VPCRegion:
	 vpcRegion,
	            'VPCId':
	 vpcId
	        },
	        CallerRef
	erence=str(random.
	random()*100000),
	        HostedZon
	eConfig={
	            'Comment'
	: "private hosted zone
	 created by automatio
	n",
	            'PrivateZ
	one': True
	        }
	    )
	 print(response)</pre>
	您也可以使用诸如 AWS 之类的基础设施即代码 (IaC) 工具，CloudFormation 将这些步骤替换为使用适当资源和属性的堆栈创建堆栈的模板。	

Task	描述	所需技能
从 Amazon S3 以字典形式检索详细信息。	使用以下代码从 S3 存储桶中读取数据并获取 Python 字典形式的 JSON 值。 `fileobj = s3_client .get_object( Bucket=bucket_name, Key='my.json' ) filedata = fileobj['Body'].read() contents = filedata.decode('utf-8') json_content=json.loads(contents) print(json_content )` 其中 `json_content` 包含 Python 字典。	应用程序开发人员，Python 技能

Task	描述	所需技能

清理空格和 Unicode 字符数据值。	作为确保数据正确性的安全措施，请使用以下代码对 `json_content` 中的值执行剥离操作。此代码删除每个字符串前面和末尾的空格字符。它还使用 `replace` 方法来删除硬的（不间断的）空格（`\xa0` 字符）。
	
	`for item in json_content:`
	`  fqn_name = unicodedata.normalize("NFKD",item["FqdnName"].replace("u'", "'").replace('\xa0', '').strip())`
	`  rec_type = item["RecordType"].replace('\xa0', '').strip()`
	`  res_rec = {`
	`  'Value': item["Value"].replace('\xa0', '').strip()`
	`  }`
		应用程序开发人员，Python 技能

Task	描述	所需技能
插入记录。	使用以下代码作为上一个 for 循环的一部分。 `change_response = route53_client.change_resource_record_sets( HostedZoneId="xxxxxxxx", ChangeBatch={ 'Comment': 'Created by automation', 'Changes': [ { 'Action': 'UPSERT', 'ResourceRecordSet': { 'Name': fqn_name, 'Type': rec_type, 'TTL': item["TTL"], 'ResourceRecords': res_rec } } ] } )` xxxxxxx 是本操作说明第一步所述的托管区 ID。	应用程序开发人员，Python 技能

相关资源

参考

- 通过导入区域文件创建记录（Amazon Route 53 文档）
- create_hosted_zone 方法（Boto3 文档）
- change_resource_record_sets 方法（Boto3 文档）

教程和视频

- Python 教程（Python 文档）
- 使用 Amazon Route 53 进行域名系统设计（YouTube 视频，AWS 在线技术讲座）

在 AWS 上从 F5 迁移到应用程序负载均衡器时修改 HTTP 标头

由 Sachin Trivedi (AWS) 创建

摘要

当您将使用 F5 负载均衡器的应用程序迁移到 Amazon Web Services (AWS) 并希望在 AWS 上使用应用程序负载均衡器时，迁移用于标头修改的 F5 规则是一个常见问题。Application Load Balancer 不支持修改标头，但你可以使用亚马逊 CloudFront 作为内容分发网络 (CDN)，使用 Lambda @Edge 来修改标头。

此模式描述了所需的集成，并提供了使用 AWS CloudFront 和 Lambda @Edge 修改标头的示例代码。

先决条件和限制

先决条件

- 使用 F5 负载均衡器的本地应用程序，其配置使用 `if, else` 替换 HTTP 标头值。有关此配置的详细信息，请参阅 F5 产品文档中的 HTTP::header。

限制

- 此模式适用于 F5 负载均衡器标头自定义。对于其他第三方负载均衡器，请查看负载均衡器文档以获取支持信息。
- 用于 Lambda@Edge 的 Lambda 函数必须位于美国东部(弗吉尼亚州北部)区域。

架构

下图显示了 AWS 上的架构，包括 CDN 与其他 AWS 组件之间的集成流程。

工具

Amazon Web Services

- 应用程序负载均衡器 - 应用程序负载均衡器是一项 AWS 完全托管的负载均衡服务，在开放系统互连 (OSI) 模型的第七层运行。它可以在多个目标之间平衡流量，并支持基于 HTTP 标头和方法、查询字符串以及基于主机或基于路径的路由的高级路由请求。

- 亚马逊 CloudFront — Amazon CloudFront 是一项网络服务，可加快向用户分发静态和动态网页内容（例如.html、.css、.js 和图像文件）的速度。 CloudFront 通过名为边缘位置的全球数据中心网络提供内容，以降低延迟并提高性能。
- Lambda @Edge – Lambda @Edge 是 AWS Lambda 的扩展，它允许你运行函数来自定义所交付的内容。 CloudFront 您可以在美国东部（弗吉尼亚北部）区域创作函数，然后将该函数与 CloudFront 发行版相关联，以便在不预配置或管理服务器的情况下自动在全球范围内复制您的代码。如此便可减少延迟并改善用户体验。

代码

以下示例代码提供了修改 CloudFront 响应标头的蓝图。按照操作说明部分中的说明部署代码。

```
exports.handler = async (event, context) => {
    const response = event.Records[0].cf.response;
    const headers = response.headers;


    const headerNameSrc = 'content-security-policy';
    const headerNameValue = '*.xyz.com';


    if (headers[headerNameSrc.toLowerCase()]) {
        headers[headerNameSrc.toLowerCase()] = [{
            key: headerNameSrc,
            value: headerNameValue,
        }];
        console.log(`Response header "${headerNameSrc}" was set to ` +
                    `"${headers[headerNameSrc.toLowerCase()][0].value}"`);
    }
    else {
            headers[headerNameSrc.toLowerCase()] = [{
            key: headerNameSrc,
            value: headerNameValue,
            }];
    }
    return response;
};
```

操作说明

创建 CDN 分配

Task	描述	所需技能
创建 CloudFront Web 分发。	在此步骤中，您将创建一个 CloudFront 分发，以告知您要从 CloudFront 哪里交付内容，以及有关如何跟踪和管理内容交付的详细信息。 要使用控制台创建分配，请登录 AWS 管理控制台，打开CloudFront 控制台，然后按照CloudFront 文档中的步骤操作。	云管理员

创建并部署 Lambda@Edge 函数

Task	描述	所需技能
创建并部署 Lambda@Edge 函数。	您可以使用修改 CloudFront 响应标头的蓝图来创建 Lambda @Edge 函数。（其他蓝图可用于不同的用例；有关更多信息，请参阅文档中的CloudFront Lambda @Edge 示例函数。） 若要创建 Lambda@Edge 函数： 1. 登录 AWS 管理控制台并打开 AWS Lambda 控制台，网址为。https://console.aws.amazon.com/lambda/	AWS 管理员

Task	描述	所需技能
	2. 请确保您位于美国东部（弗吉尼亚北部）地区。CloudFront 蓝图仅在该地区可用。 3. 选择 Create function (创建函数)。 4. 选择使用蓝图，然后在蓝图搜索字段中输入 cloudfront。 5. 选择cloudfront-modify-response-header蓝图，然后选择配置。 6. 在基本信息页面上，输入以下信息： a. 输入函数名称。 b. 对于执行角色，请选择从 AWS 策略模板创建新角色。 c. 将一个 AWS 身份验证与访问管理(IAM)角色与该选项相关联。 7. 选择创建函数。 8. 在页面的设计器部分中，选择您的函数名称。 9. 在函数代码部分中，将模板代码替换为之前在代码部分中此模式中提供的示例代码。 10.在示例代码中，将 `xyz.com` 替换为您的域名。 11选择保存。	

Task	描述	所需技能
部署 Lambda@Edge 函数。	按照亚马逊 CloudFront 文档中教程：创建简单的 Lambda @Edge 函数的第 4 步中的说明配置 CloudFront 触发器并部署该函数。	AWS 管理员

相关资源

CloudFront 文档

- 自定义源的请求和响应行为
- 使用分配
- Lambda@Edge 示例函数
- 使用 Lambda@Edge 在边缘进行自定义
- 教程：创建简单的 Lambda@Edge 函数

从多个私密访问中央 AWS 服务终端节点 VPCs

由 Martin Guenthner (AWS) 和 Samuel Gordon (AWS) 编写

摘要

您的环境的安全与合规要求可能会规定，Amazon Web Services (AWS) 服务或端点的流量不得通过公共互联网。这种模式是专为拓扑设计的解决方案，在这种hub-and-spoke拓扑中，中央集线器 VPC 连接到多个分布式分支 VPCs。在此解决方案中，您可以使用 AWS PrivateLink 在中心账户中为 AWS 服务创建接口 VPC 终端节点。然后，您可以使用传输网关和分布式域名系统 (DNS) 规则将请求解析到连接的终端节点的私有 IP 地址 VPCs。

此模式描述了如何使用 AWS Transit Gateway、入站亚马逊 Route 53 Resolver 终端节点和共享的 Route 53 转发规则来解析来自所连接资源的 DNS 查询。 VPCs您可在中心账户中创建端点、中转网关、解析程序和转发规则。然后，您可以使用 AWS Resource Access Manager (AWS RAM) 与分支共享传输网关和转发规则 VPCs。提供的 AWS CloudFormation 模板可帮助您在中心 VPC 和分支中部署和配置资源 VPCs。

先决条件和限制

先决条件

- 一个中心账户和一个或多个分支账户，在 AWS Organizations 的同一组织中进行管理。有关更多信息，请参阅创建并管理组织。
- AWS Resource Access Manager (AWS RAM) 在 AWS Organizations 中配置为可信服务。有关更多信息，请参阅 将 AWS Organizations 与其他 Amazon Web Services 结合使用。
- 必须在中心和分支中启用 DNS 解析 VPCs。有关更多信息，请参阅 VPC 的 DNS 属性（Amazon Virtual Private Cloud 文档）。

限制

- 这种模式将同一 AWS 区域中的中心账户和分支账户连接起来。对于多区域部署，您必须对每个区域重复该模式。
- AWS 服务必须 PrivateLink 作为接口 VPC 终端节点与集成。有关完整列表，请参阅与 AWS 集成的 AWS 服务 PrivateLink（PrivateLink 文档）。
- 无法保证可用区域的亲和性。例如，来自可用区 A 的查询可能会使用可用区 B 的 IP 地址进行响应。
- 与 VPC 终端节点关联的 elastic network interface 限制为每秒 10,000 次查询。

架构

目标技术堆栈

- 中心 Amazon Web Services account 中的中心 VPC
- 在分支 AWS 账户 VPCs 中进行一次或多次讲话
- 中心账户中的一个或多个接口 VPC 端点
- 中心账户中的入站与出站 Route 53 解析程序
- 部署在中心账户中并与分支账户共享的 Route 53 解析程序转发规则
- 部署在中心账户中并与分支账户共享的中转网关
- 连接中心和辐条的 AWS Transit Gateway VPCs

目标架构

下图显示了此解决方案的示例架构。在此架构中，中心账户中的 Route 53 解析程序转发规则与其他架构组件具有以下关系：

1. 转发规则通过使用 AWS RAM 与分支 VPC 共享。
2. 转发规则与中心 VPC 中的出站解析程序关联。
3. 转发规则针对中心 VPC 中的入站解析程序。

下图显示了通过示例架构的流量流：

1. 分支 VPC 中的资源（例如亚马逊弹性计算云 (Amazon EC2) 实例）向发出 DNS 请求`<service>.<region>.amazonaws.com`。该请求由分支 Amazon DNS 解析程序接收。
2. 从中心账户共享并与分支 VPC 关联的 Route 53 转发规则将拦截请求。
3. 在中心 VPC 中，出站解析程序使用转发规则将请求转发到入站解析程序。
4. 入站解析程序使用中心 VPC Amazon DNS 解析程序将 `<service>.<region>.amazonaws.com` 的 IP 地址解析为 VPC 端点的私有 IP 地址。如果不存在 VPC 端点，它将解析为公共 IP 地址。

工具

AWS 工具和服务

- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Resource Access Manager (AWS RAM) 可帮助您跨 Amazon Web Services account 安全共享资源，以减少运营开销，提供可见性和可审计性。
- Amazon Route 53 是一种可用性高、可扩展性强的域名系统（DNS）Web 服务。
- AWS Systems Manager 可帮助您管理在 Amazon Web Services Cloud 中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。
- AWS Transit Gateway 是一个连接 VPCs 本地网络的中心枢纽。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。

其他工具和服务

- nslookup 是用于查询 DNS 记录的命令行工具。在这种模式中，您可使用此工具来测试解决方案。

代码存储库

此模式的代码可在 GitHubvpc-endpoint-sharing存储库中找到。此模式提供两个 AWS CloudFormation 模板：

- 用于在中心账户中部署以下资源模板：
  - `rSecurityGroupEndpoints` — 控制对 VPC 端点的访问权限的安全组。
  - `rSecurityGroupResolvers` — 控制对 Route 53 Resolver 的访问权限的安全组。
  - `rKMSEndpoint`、`rSSMMessagesEndpoint`、`rSSMEndpoint` 和 `rEC2MessagesEndpoint` — 中心账户中的接口 VPC 端点示例。您的使用案例自定义此端点。

  - rInboundResolver — 一款 Route 53 解析程序，用于解析针对中心 Amazon DNS 解析程序的 DNS 查询。
  - rOutboundResolver — 将查询转发给入站解析程序的出站 Route 53 解析程序。
  - rAWSApiResolverRule— 与所有分支 VPCs共享的 Route 53 解析器转发规则。
  - rRamShareAWSResolverRule— 允许分支 VPCs 使用rAWSApiResolverRule转发规则的 AWS RAM 共享。
  - * rVPC — VPC 用于对共享服务进行建模。
  - * rSubnet1 — 用于存放中心资源私有子网。
  - * rRouteTable1 — 中心 VPC 路由表。
  - * rRouteTableAssociation1 — 对于中心 VPC 中的 rRouteTable1 路由表，指私有子网的关联。
  - * rRouteSpoke — 从中心 VPC 至分支 VPC 的路由。
  - * rTgw — 与所有分支共享的传输网关 VPCs。
  - * rTgwAttach — 允许中心 VPC 将流量路由到 rTgw 中转网关的附件。
  - * rTgwShare — 允许分支账户使用 rTgw 中转网关的 AWS RAM 共享。
- 用于在分支账户中部署以下资源的模板：
  - rAWSApiResolverRuleAssociation — 允许分支 VPC 使用中心账户中的共享转发规则的关联。
  - * rVPC — 分支 VPC。
  - * rSubnet1, rSubnet2, rSubnet3 — 每个可用区的子网，用于存放分支私有资源。
  - * rTgwAttach — 允许分支 VPC 将流量路由到 rTgw 中转网关的附件。
  - * rRouteTable1 — 分支 VPC 路由表。
  - * rRouteEndpoints — 从分支 VPC 中的资源至中转网关的路由。
  - * rRouteTableAssociation1/2/3 — 对于分支 VPC 中的 rRouteTable1 路由表，指私有子网的关联。
  - * rInstanceRole — 用于测试解决方案的 IAM 角色。
  - * rInstancePolicy — 用于测试解决方案的 IAM policy。
  - * rInstanceSg — 用于测试解决方案的安全组。
  - * rInstanceProfile — 用于测试解决方案的 IAM 实例配置文件。
  - * rInstance — 预先配置为通过 AWS Systems Manager 访问的 EC2 实例。使用此实例测试解决方案。

* 这些资源支持示例架构，在现有登录区中实现此模式时可能不需要这些资源。

操作说明

准备 CloudFormation 模板

Task	描述	所需技能
克隆代码存储库。	1. 在命令行界面中，将工作目录更改为要存储示例文件的位置。 2. 输入以下命令： `git clone https://github.com/aws-samples/vpc-endpoint-sharing.git`	网络管理员、云架构师
修改模板。	1. 在克隆的存储库中，打开 hub.yml 和 spoke.yml 文件。 2. 查看由这些模板创建的资源，并按您的环境需要调整模板。有关完整列表，请参阅工具的代码库部分。如果您的账户已经拥有其中一些资源，请将其从 CloudFormation 模板中删除。有关更多信息，请参阅使用模板（CloudFormation 文档）。 3. 保存并关闭 hub.yml 和 spoke.yml 文件。	网络管理员、云架构师

在目标账户中部署资源

Task	描述	所需技能
部署中心资源。	使用 hub.yml 模板创建堆栈。CloudFormation 当出现提示时，为模板中的参数提供值。有关更多信息，请参阅创建堆栈（CloudFormation 文档）。	云架构师、网络管理员
部署分支资源。	使用 spoke.yml 模板创建一个堆栈。 CloudFormation 当出现提示时，为模板中的参数提供值。有关更多信息，请参阅创建堆栈（CloudFormation 文档）。	云架构师、网络管理员

测试解决方案

Task	描述	所需技能
测试对 Amazon Web Services 的私有 DNS 查询。	1. 使用会话管理器连接到`rInstance` EC2 实例，这是 AWS Systems Manager 的一项功能。有关更多信息，请参阅使用会话管理器连接您的 Linux 实例（Amazon EC2 文档）。 2. 对于中心账户中具有 VPC 端点的 Amazon Web Services，`nslookup` 请使用确认已返回入站 Route 53 解析程序的私有 IP 地址。 下面是使用 `nslookup` 访问 Amazon Systems Manager 端点的示例。	网络管理员

Task	描述	所需技能

	`nslookup ssm.<region>.amazonaws.com`
	3. 在AWS 命令行界面（AWS CLI）中，输入可帮助您确认更改不会影响服务功能的命令。有关命令的列表，请参阅 AWS CLI 命令参考。
	例如以下命令应该返回 Amazon Systems Manager 文档列表。
	`aws ssm list-documents`	

Task	描述	所需技能

测试对 Amazon Web Services 的公共 DNS 查询。	1. 对于中心账户中没有 VPC 端点的 Amazon Web Services，请使用 `nslookup` 确认已返回公有 IP 地址。下面是使用 `nslookup` 访问 Amazon Simple Notification Service(Amazon SNS) 端点的示例。
	`nslookup sns.<region>.amazonaws.com`
	2. 在 AWS CLI 中，输入可帮助您确认更改不影响服务功能的命令。有关命令的列表，请参阅 AWS CLI 命令参考。
	例如，如果中心账户中存在任何 Amazon SNS 主题，则以下命令应返回主题列表。
	`aws sns list-topics`
		网络管理员

相关资源

- 构建可扩展的安全多 VPC AWS 网络基础设施（AWS 白皮书）
- 使用共享资源（AWS RAM 文档）
- 使用中转网关（AWS Transit Gateway 文档）

创建网络访问分析器在多个入站互联网访问中发现的报告 AWS 账户

由 Mike Virgilio (AWS) 创建

摘要

无意中入站 Internet 访问 AWS 资源可能会给组织的数据外围带来风险。Network Access Analyzer 是一项亚马逊虚拟私有云（亚马逊 VPC）功能，可帮助您识别对亚马逊 Web Services（AWS）上资源的意外网络访问。您可以使用网络访问分析器指定网络访问要求并识别不满足指定要求的潜在网络路径。您可以使用网络访问分析器执行以下操作：

1. 确定可通过互联网网关访问互联网的 AWS 资源。
2. 验证您的虚拟私有云 (VPCs) 是否进行了适当的分段，例如隔离生产和开发环境以及分离事务性工作负载。

Network Access Analyzer 分析 end-to-end网络可访问性条件，而不仅仅是分析单个组件。为了确定资源是否可以访问互联网，Network Access Analyzer 会评估互联网网关、VPC 路由表、网络访问控制列表 (ACLs)、弹性网络接口上的公有 IP 地址和安全组。如果这些组件中的任何一个阻止互联网访问，则网络访问分析器不会生成调查发现。例如，如果 Amazon Elastic Compute Cloud (Amazon EC2) 实例有一个开放的安全组，允许来自的流量，`0/0`但该实例位于无法从任何互联网网关路由的私有子网中，则网络访问分析器将不会生成结果。这提供了高保真结果，以便您可以识别真正可从互联网访问的资源。

运行网络访问分析器时，使用网络访问作用域来指定网络访问要求。此解决方案可识别 互联网网关和弹性网络接口之间的网络路径。在这种模式下，您可以在组织 AWS 账户 中集中部署解决方案，由其管理 AWS Organizations，并分析组织中任何 AWS 区域帐户中的所有帐户。

该解决方案的设计考虑了以下内容：

- 这些 AWS CloudFormation 模板减少了在这种模式下部署 AWS 资源所需的工作量。
- 您可以在部署时调整 CloudFormation 模板和 naa-script.sh 脚本中的参数，以便根据您的环境对其进行自定义。
- Bash 脚本自动并行预配和分析多个账户的网络访问范围。
- Python 脚本处理结果，提取数据，然后合并结果。您可以选择查看 CSV 格式或格式的 Network Access Analyzer 发现结果的合并报告 AWS Security Hub。CSV 报告的示例可在此模式的其他信息部分找到。
- 您可以修正结果，也可以通过将结果添加到 naa-exclusions.csv 文件来将其从将来的分析中排除。

先决条件和限制

先决条件

- AWS 账户 用于托管安全服务和工具，作为中组织的成员帐户进行管理 AWS Organizations。在此模式中，此账户称为安全账户。
- 在安全账户中，必须具有具有出站互联网访问权限的私有子网。有关说明，请参阅 Amazon VPC 文档中的创建子网。您可以使用 NAT 网关或接口 VPC 端点建立互联网访问。
- 对 AWS Organizations 管理账户或拥有委托管理员权限的账户的访问权限 CloudFormation。有关说明，请参阅 CloudFormation 文档中的注册委托管理员。
- 在 AWS Organizations 和之间启用可信访问 CloudFormation。有关说明，请参阅 CloudFormation 文档 AWS Organizations中的使用启用可信访问。
- 如果您要将调查结果上传到 Security Hub，则必须在账户和亚马逊 EC2 实例的配置 AWS 区域 位置中启用 Security Hub。有关更多信息，请参阅设置 AWS Security Hub。

限制

- 由于网络访问分析器功能的限制，当前不会分析跨账户网络路径。
- 目标 AWS 账户 必须作为一个组织在中进行管理 AWS Organizations。如果你不使用 AWS Organizations，你可以为你的环境更新 naa-execrole.yaml CloudFormation 模板和 naa-script.sh 脚本。相反，您可以提供要在其中运行脚本的 AWS 账户 IDs 和区域的列表。
- 该 CloudFormation 模板旨在将 Amazon EC2 实例部署到具有出站互联网访问权限的私有子网中。AWS Systems Manager 代理（SSM 代理）需要出站访问才能访问 Systems Manager 服务端点，并且您需要出站访问权限才能克隆代码存储库和安装依赖项。如果您想使用公有子网，则必须修改 naa-resources.yaml 模板以将弹性 IP 地址与亚马逊实例相关联。 EC2

架构

目标架构

选项 1：访问 Amazon S3 存储桶中的结果

此图显示以下流程：

1. 如果您手动运行解决方案，则用户使用会话管理器对 Amazon EC2 实例进行身份验证，然后运行 naa-script.sh 脚本。此 Shell 脚本执行步骤 2-7。

如果自动运行解决方案，则 naa-script.sh 脚本将按照在 cron 表达式中定义的计划自动启动。此 Shell 脚本执行步骤 2-7。有关详细信息，请参阅本节末尾的自动化和扩缩。

2. 亚马逊 EC2 实例从亚马逊 S3 存储桶下载最新的 naa-exception.csv 文件。稍后在 Python 脚本处理排除项时，将在此过程中使用此文件。
3. Amazon EC2 实例担任 NAAEC2Role AWS Identity and Access Management (IAM) 角色，该角色授予访问 Amazon S3 存储桶和在组织中其他账户中担任 NAAExecRole IAM 角色的权限。
4. Amazon EC2 实例在组织的管理账户中担任 NAAExecRole IAM 角色，并生成组织中的账户列表。
5. Amazon EC2 实例在组织的成员账户（架构图中称为工作负载账户）中担任 NAAExecRole IAM 角色，并在每个账户中执行安全评估。调查结果以 JSON 文件形式存储在亚马逊 EC2 实例上。
6. Amazon EC2 实例使用 Python 脚本来处理 JSON 文件、提取数据字段并创建 CSV 报告。
7. 亚马逊 EC2 实例将 CSV 文件上传到亚马逊 S3 存储桶。
8. Amazon EventBridge 规则会检测文件上传，并使用 Amazon SNS 主题发送一封电子邮件，通知用户报告已完成。
9. 用户从 Amazon S3 存储桶中下载 CSV 文件。用户将结果导入 Excel 模板，并查看结果。

选项 2：访问中的调查结果 AWS Security Hub

该图显示了以下过程：

1. 如果您手动运行解决方案，则用户使用会话管理器对 Amazon EC2 实例进行身份验证，然后运行 naa-script.sh 脚本。此 Shell 脚本执行步骤 2-7。

   如果自动运行解决方案，则 naa-script.sh 脚本将按照在 cron 表达式中定义的计划自动启动。此 Shell 脚本执行步骤 2-7。有关详细信息，请参阅本节末尾的自动化和扩缩。
2. 亚马逊 EC2 实例从亚马逊 S3 存储桶下载最新的 naa-exception.csv 文件。稍后在 Python 脚本处理排除项时，将在此过程中使用此文件。
3. Amazon EC2 实例担任 NAAEC2Role IAM 角色，该角色授予访问 Amazon S3 存储桶和在组织中其他账户中担任 NAAExecRole IAM 角色的权限。
4. Amazon EC2 实例在组织的管理账户中担任 NAAExecRole IAM 角色，并生成组织中的账户列表。
5. Amazon EC2 实例在组织的成员账户（架构图中称为工作负载账户）中担任 NAAExecRole IAM 角色，并在每个账户中执行安全评估。调查结果以 JSON 文件形式存储在亚马逊 EC2 实例上。
6. 亚马逊 EC2 实例使用 Python 脚本来处理 JSON 文件并提取数据字段以导入 Security Hub。

7. Amazon EC2 实例将网络访问分析器的结果导入到 Security Hub。
8. Amazon EventBridge 规则会检测导入，并使用 Amazon SNS 主题发送一封电子邮件，通知用户该过程已完成。
9. 用户在 Security Hub 中查看结果。

自动化和扩缩

您可以计划此解决方案，以便按自定义计划自动运行 naa-script.sh 脚本。要设置自定义计划，请在 naa-resources.yaml CloudFormation 模板中修改参数。`CronScheduleExpression`例如，默认值 `0 0 * * 0` 在每个星期日的午夜运行解决方案。如果值为 `0 0 * 1-12 0`，则该解决方案将在每个月第一个星期日的午夜运行。有关使用 cron 表达式的更多信息，请参阅 Systems Manager 文档中的 Cron 和 rate 表达式。

如果要在部署 `NAA-Resources` 堆栈后调整计划，可以在 `/etc/cron.d/naa-schedule` 中手动编辑 cron 计划。

工具

AWS 服务

- 亚马逊弹性计算云 (Amazon EC2) 在中提供可扩展的计算容量 AWS Cloud。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如， AWS Lambda 函数、使用 API 目的地的 HTTP 调用端点或其他 AWS 账户目的地的事件总线。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Organizations是一项账户管理服务，可帮助您将多个账户整合 AWS 账户 到一个由您创建和集中管理的组织中。
- AWS Security Hub提供了您的安全状态的全面视图 AWS。它还可以帮助您根据安全行业标准和最佳实践检查您的 AWS 环境。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

- AWS Systems Manager 可帮助您管理在 AWS Cloud中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。此模式使用 Session Manager，这是 Systems Manager 的一项功能。

代码存储库

此模式的代码可在 GitHub Network Access Analyzer 多账户分析存储库中找到。代码存储库包含以下文件：

- naa-script.sh — 此 bash 脚本用于启动并行分析多个 AWS 账户网络访问分析器分析。根据 naa-resources.yaml CloudFormation 模板中的定义，此脚本将自动部署到亚马逊实例上的文件夹`/usr/local/naa`。 EC2
- naa-resources.yaml — 您可以使用此 CloudFormation 模板在组织的安全账户中创建堆栈。此模板部署了该账户所需所有资源，以支持此解决方案。此堆栈必须在 naa-execrole.yaml 模板之前部署。

> **Note**
>
> 如果此堆栈被删除并重新部署，则必须重新构建`NAAExecRole`堆栈集，以便重建 IAM 角色之间的跨账户依赖关系。

- naa-execrole.yaml — 您可以使用此 CloudFormation 模板创建堆栈集，该堆栈集可在组织中的所有账户（包括管理账户）中部署 `NAAExecRole` IAM 角色。
- naa-processfindings.py - naa-script.sh 脚本会自动调用此 Python 脚本来处理网络访问分析器 JSON 输出，排除 naa-exclusions.csv 文件中任何已知良好的资源，然后生成合并结果的 CSV 文件或将结果导入 Security Hub。

操作说明

准备部署

Task	描述	所需技能
克隆代码存储库。	1. 在命令行界面中，将工作目录更改为要存储示例文件的位置。 2. 输入以下命令：	AWS DevOps

Task	描述	所需技能
	`git clone https://github.com/aws-samples/network-access-analyzer-multi-account-analysis.git`	
查看模板	1. 在克隆的存储库中，打开 naa-resources.yaml 和 naa-execrole.yaml 文件。 2. 查看由这些模板创建的资源，并按您的环境需要调整模板。有关更多信息，请参阅 CloudFormation 文档中的使用模板。 3. 保存并关闭 naa-resources.yaml 和 naa-execrole.yaml 文件。	AWS DevOps

创建堆 CloudFormation 栈

Task	描述	所需技能
在安全账户中预置资源。	使用 naa-resources.yaml 模板，您可以创建一个 CloudFormation 堆栈，用于在安全账户中部署所有必需的资源。有关说明，请参阅 CloudFormation 文档中的创建堆栈。部署此模板时应注意以下几点：	AWS DevOps

Task	描述	所需技能
	1. 在指定模板页面上，选择模板已准备就绪，然后上传 naa-resources.yaml 文件。 2. 在指定堆栈详细信息页面上的堆栈名称框中，输入 `NAA-Resources` 。 3. 在参数部分中，输入以下内容： • `VPCId` - 在账户中选择一个 VPC。 • `SubnetId` -- 选择可以访问 Internet 的私有子网。 **Note** 如果您选择公有子网，则可能不会为 Amazon EC2 实例分配公有 IP 地址，因为默认情况下， CloudFormation 模板不会预配置和附加弹性 IP 地址。 • `InstanceType` - 保留默认实例类型。 • `InstanceImageId` - 保留默认值。 • `KeyPairName` - 如果您使用 SSH 进行访问，请指定现有密钥对的名称。 • `PermittedSSHInbound` - 如果您使用 SSH	

Task	描述	所需技能
	进行访问，请指定允许的 CIDR 块。如果不使用 SSH，请保留默认值 `127.0.0.1` 。 • `BucketName` – 默认值为 `naa-<accountID>-<region>` 。您可以根据需要进行修改。如果指定自定义值，则账户 ID 和区域会自动追加到指定值。 • `EmailAddress` - 在分析完成后指定 Amazon SNS 通知的电子邮件地址。 **Note** 分析完成前必须确认 Amazon SNS 订阅配置，否则将不会发送通知。 • `NAAEC2Role` - 保留默认值，除非您的命名约定要求此 IAM 角色使用不同的名称。 • `NAAExecRole` - 保留默认值，除非在部署 naa-execrole.yaml 时使用其他名称 • `Parallelism` - 指定要执行的并行评测的数量。	

Task	描述	所需技能
	• Regions— 指定 AWS 区域 要分析的。 • ScopeNameValue - 指定将分配给范围的标记。该标签用于确定网络访问范围。 • ExclusionFile - 指定排除文件名。此文件中的条目将从结果中排除。 • FindingsToCSV - 指定是否应将结果输出到 CSV。接受的值为 true 和 false • FindingsToSecurityHub - 指定是否应将结果导入 Security Hub。接受的值为 true 和 false • EmailNotificationsForSecurityHub - 指定将结果导入 Security Hub 是否应生成电子邮件通知。接受的值为 true 和 false • ScheduledAnalysis - 如果希望解决方案按计划自动运行，请输入 true，然后在 CronScheduleExpression 参数中自定义计划。如果不想自动运行解决方案，请输入 false。 • CronScheduleExpression - 如果自动运行解	

Task	描述	所需技能
	决方案，请输入 cron 表达式来定义计划。有关更多信息，请参阅此模式的架构部分中的自动化和扩缩。 1. 在 “查看” 页面上，选择 “以下资源需要能力：[AWS::IAM::Role]”，然后选择 “创建堆栈”。 2. 成功创建堆栈后，在 CloudFormation 控制台的输出选项卡上，复制 `NAAEC2Role` Amazon 资源名称 (ARN)。您稍后在部署 naa-execrole.yaml 文件时将会使用此 ARN。	

Task	描述	所需技能
在成员账户中预置 IAM 角色。	在 AWS Organizations 管理账户或具有委托管理员权限的账户中 CloudFormation，使用 naa-execrole.yaml 模板创建堆栈集。 CloudFormation 堆栈集在组织中的所有成员账户中部署 `NAAExecRole` IAM 角色。有关说明，请参阅 CloudFormation 文档中的使用服务管理权限创建堆栈集。部署此模板时应注意以下几点： 1. 在准备模板下，选择模板已准备就绪，然后上传 naa-execrole.yaml 文件。 2. 在 “指定 StackSet 详细信息” 页面上，为堆栈集命名`NAA-ExecRole` 。 3. 在参数部分中，输入以下内容： • `AuthorizedARN` - 输入您在创建 `NAA-Resources` 堆栈时复制的 `NAAEC2Role` ARN。 • `NAARoleName` - 保留默认值 `NAAExecRole` ，除非在部署 naa-resources.yaml 文件时使用了其他名称。 4. 在权限下方，选择服务托管权限。	AWS DevOps

Task	描述	所需技能
	5. 在设置部署选项页面，在部署目标下，选择部署到组织并接受所有默认选项。 **Note** 如果要将堆栈同时部署到所有成员账户，请将最大并发账户数和容错率设置为较高的值，例如100。 6. 在部署区域下，选择部署网络访问分析器的 Amazon EC2 实例的区域。由于 IAM 资源是全局资源而不是区域资源，因此这会在所有活动区域中部署 IAM 角色。 7. 在“审阅”页面上，选择“我确认 AWS CloudFormation 可能会使用自定义名称创建 IAM 资源”，然后选择“创建” StackSet。 8. 监控堆栈实例选项卡（用于单个账户状态）和操作选项卡（用于整体状态）以确定部署何时完成。	

Task	描述	所需技能
在管理账户中预置 IAM 角色。	使用 naa-execrole.yaml 模板，您可以创建一个 CloudFormation 堆栈，用于在组织的管理账户中部署 `NAAExecRole` IAM 角色。您之前创建的堆栈集不在管理账户中部署 IAM 角色。有关说明，请参阅 CloudFormation 文档中的创建堆栈。部署此模板时应注意以下几点： 1. 在指定模板页面上，选择模板已准备就绪，然后上传 naa-execrole.yaml 文件。 2. 在指定堆栈详细信息页面上的堆栈名称框中，输入 `NAA-ExecRole` 。 3. 在参数部分中，输入以下内容： • `AuthorizedARN` - 输入您在创建 `NAA-Resources` 堆栈时复制的 `NAAEC2Role` ARN。 • `NAARoleName` - 保留默认值 `NAAExecRole` ，除非在部署 naa-resources.yaml 文件时使用了其他名称。 4. 在 “查看” 页面上，选择 “以下资源需要能力：[AWS::IAM::Role]”，然后选择 “创建堆栈”。	AWS DevOps

执行分析

Task	描述	所需技能

自定义 shell 脚本。	1. 登录到组织中的安全账户。
	2. 使用会话管理器，连接到您之前配置的网络访问分析器的 Amazon EC2 实例。有关说明，请参阅使用会话管理器连接至您的 Linux 实例。如果您无法连接，请参阅此模式的故障排除部分。
	3. 输入以下命令打开 naa-script.sh 文件进行编辑：
	`sudo -i`
	`cd /usr/local/naa`
	`vi naa-script.sh`
	4. 根据您的环境需要，查看和修改此脚本中的可调参数和变量。有关自定义选项的更多信息，请参阅脚本开头的注释。
	例如，您可以修改脚本以指定要扫描的 AWS 账户 IDs 或，也可以引用包含这些参数的外部文件 AWS 区域 ，而不是从管理账户获取组织中所有成员账户的列表。
	5. 保存并关闭 naa-script.sh 文件。
		AWS DevOps

分析目标账户。	1. 输入以下命令。这将运行 naa-script.sh 脚本： `sudo -i`	AWS DevOps

Task	描述	所需技能

	<pre>cd /usr/local/naa
	screen
	./naa-script.sh</pre>
	请注意以下几点：
	• screen 命令允许脚本在连接超时或您失去控制台访问权限时继续运行。
	• 扫描开始后，可以通过按 Ctrl+A D 强制分离屏幕。屏幕将分离，您可以在分析进行时关闭实例连接。
	• 要恢复已分离的会话，请连接至实例，输入 sudo -i，然后输入 screen -r。
	2. 监控输出中是否存在任何错误，以确保脚本正常工作。有关示例查询，请参阅此模式的其他信息部分。
	3. 等待分析完成。如果您配置了电子邮件通知，则当结果上传到 Amazon S3 存储桶或导入到 Security Hub 时，您会收到一封电子邮件。	

Task	描述	所需技能
选项 1-从 Amazon S3 存储桶中检索结果。	1. 从 `naa-<accountID>-<region>` 桶下载 CSV 文件。有关说明，请参阅 Amazon S3 文档中的下载对象。 2. 从 Amazon S3 存储桶中删除 CSV 文件。这是成本优化的最佳实践。有关说明，请参阅 Amazon S3 文档中的删除对象。	AWS DevOps
选项 2 - 在 Security Hub 中查看结果。	1. 打开 S ecurity Hub 控制台。 2. 从导航窗格中选择调查发现。 3. 查看网络访问分析器的调查发现。有关说明，请参阅 Security Hub 文档中的查看调查发现列表和详细信息。 **Note** 您可以通过添加以筛选器开头的标题并输入来搜索结果`Network Access Analyzer`。	AWS DevOps

修正和排除结果

Task	描述	所需技能
纠正发现的问题。	纠正您想要解决的任何问题。有关如何围绕 AWS 身份、资源和网络创建边界的更多信息和最佳实践，请参阅在 AWS（AWS 白皮书）上构建数据边界。	AWS DevOps
排除具有已知良好网络路径的资源。	如果网络访问分析器生成应可从互联网访问的资源的调查发现，则可以将这些资源添加到排除列表中。下次运行网络访问分析器时，它不会为该资源生成调查发现。 1. 导航到 `/usr/local/naa`，然后打开 naa-script.sh 脚本。记下 `S3_EXCLUSION_FILE` 变量的值。 2. 如果 `S3_EXCLUSION_FILE` 变量的值为 `true`，请从 `naa-<accountID>-<region>` 桶下载 naa-exclusions.csv 文件。有关说明，请参阅 Amazon S3 文档中的下载对象。 如果 `S3_EXCLUSION_FILE` 变量的值为 `false`，请导航到 `/usr/local/naa`，然后打开 naa-exclusions.csv 文件。	AWS DevOps

Task	描述	所需技能

	Note
	如果S3_EXCLUSION_FILE 变量的值为false，则脚本使用排除文件的本地版本。如果您稍后将该值更改为true，则脚本会使用 Amazon S3 存储桶中的文件覆盖本地版本。
	3. 在 naa-exclusions.csv 文件中，输入要排除的资源。在每行中输入一个资源，并使用以下格式。
	`<resource_id>,<secgroup_id>,<sgrule_cidr>,<sgrule_portrange>,<sgrule_protocol>`
	以下为资源示例。
	`eni-1111aaaaa2222bbbb,sg-3333ccccc4444dddd,0.0.0.0/0,80 to 80,tcp`
	4. 保存并关闭 naa-exclusions.csv 文件。
	5. 如果您从 Amazon S3 存储桶下载了 naa-exclusions.csv 文件，请上传新	

Task	描述	所需技能
	版本。有关说明，请参阅 Amazon S3 文档中的上传对象。	

（可选）更新 naa-script.sh 脚本

Task	描述	所需技能

更新 naa-script.sh 脚本。	如果要将 naa-script.sh 脚本更新到存储库中的最新版本，请执行以下操作：
	1. 使用会话管理器连接到 Amazon EC2 实例。有关说明，请参阅使用会话管理器连接至您的 Linux 实例。
	2. 输入以下命令：
	`sudo -i`
	3. 导航到 naa-script.sh 脚本目录：
	`cd /usr/local/naa`
	4. 输入以下命令来存储本地脚本，以便您可以将自定义更改合并到最新版本中：
	`git stash`
	5. 输入以下命令以获取脚本的最新版本：
	`git pull`
		AWS DevOps

Task	描述	所需技能
	6. 输入以下命令将自定义脚本与最新版本的脚本合并： `git stash pop`	

(可选)清除

Task	描述	所需技能
删除所有已部署资源。	您可以将资源保留在账户中。 如果要取消预配所有资源，请执行以下操作： 1. 删除管理账户中预置的 `NAA-ExecRole` 堆栈。有关说明，请参阅 CloudFormation 文档中的删除堆栈。 2. 删除在组织的管理账户或委托管理员账户中预置的 `NAA-ExecRole` 堆栈集。有关说明，请参阅 CloudFormation 文档中的删除堆栈集。 3. 删除 `naa-<accountID>-<region>` Amazon S3 存储桶中的所有对象。有关说明，请参阅 Amazon S3 文档中的删除对象。 4. 删除安全账户中预置的 `NAA-Resources` 堆栈。有关说明，请参阅 CloudFormation 文档中的删除堆栈。	AWS DevOps

故障排除

事务	解决方案
无法使用会话管理器连接到 Amazon EC2 实例。	SSM 代理必须能够与 Systems Manager 端点通信。执行以下操作： 1. 验证部署 Amazon EC2 实例的子网是否可以访问互联网。 2. 重启亚马逊 EC2 实例。
部署堆栈集时，CloudFormation 控制台会提示您这样做`Enable trusted access with AWS Organizations to use service-managed permissions` 。	这表示未在 AWS Organizations 和之间启用可信访问 CloudFormation。部署服务托管堆栈集需要可信访问权限。选择该按钮以启用受信任的访问。有关更多信息，请参阅 CloudFormation 文档中的启用可信访问。

相关资源

- 新增 — Amazon VPC 网络访问分析器（AWS 博客文章）
- AWS re: inForce 2022-在 AWS (NIS202) 上验证有效的网络访问控制（视频）
- 演示 - 使用网络访问分析器进行组织范围的 Internet 入口数据路径分析(视频)

其他信息

控制台输出示例

以下示例显示了生成目标账户列表和分析目标账户的输出。

```
[root@ip-10-10-43-82 naa]# ./naa-script.sh
download: s3://naa-<account ID>-us-east-1/naa-exclusions.csv to ./naa-exclusions.csv

AWS Management Account: <Management account ID>

AWS Accounts being processed...
<Account ID 1> <Account ID 2> <Account ID 3>

Assessing AWS Account: <Account ID 1>, using Role: NAAExecRole
```

```
Assessing AWS Account: <Account ID 2>, using Role: NAAExecRole
Assessing AWS Account: <Account ID 3>, using Role: NAAExecRole
Processing account: <Account ID 1> / Region: us-east-1
Account: <Account ID 1> / Region: us-east-1 - Detecting Network Analyzer scope...
Processing account: <Account ID 2> / Region: us-east-1
Account: <Account ID 2> / Region: us-east-1 - Detecting Network Analyzer scope...
Processing account: <Account ID 3> / Region: us-east-1
Account: <Account ID 3> / Region: us-east-1 - Detecting Network Analyzer scope...
Account: <Account ID 1> / Region: us-east-1 - Network Access Analyzer scope detected.
Account: <Account ID 1> / Region: us-east-1 - Continuing analyses with Scope ID.
 Accounts with many resources may take up to one hour
Account: <Account ID 2> / Region: us-east-1 - Network Access Analyzer scope detected.
Account: <Account ID 2> / Region: us-east-1 - Continuing analyses with Scope ID.
 Accounts with many resources may take up to one hour
Account: <Account ID 3> / Region: us-east-1 - Network Access Analyzer scope detected.
Account: <Account ID 3> / Region: us-east-1 - Continuing analyses with Scope ID.
 Accounts with many resources may take up to one hour
```

CSV 报告示例

下图是 CSV 输出的示例。

在多账户 AWS 环境中为混合网络设置 DNS 解析

由 Anvesh Koganti (AWS) 创作

摘要

此模式为在包含多个 Amazon Web Services (AWS) 账户的混合网络环境中设置 DNS 解析提供了一个全面的解决方案。它支持通过 Amazon Route 53 Resolver 端点在本地网络和 AWS 环境之间进行双向 DNS 解析。该模式提供了两种在多账户、集中式架构中启用 DNS 解析的解决方案：

- 基本设置不使用 Route 53 配置文件。它有助于优化复杂性较低的中小型部署的成本。
- 增强型设置使用 Route 53 配置文件来简化操作。它最适合大型或更复杂的 DNS 部署。

> **Note**
>
> 在实施之前，请查看 “限制” 部分，了解服务限制和配额。在做出决定时，请考虑管理开销、成本、运营复杂性和团队专业知识等因素。

先决条件和限制

先决条件

- 跨共享服务和工作负载 AWS 账户部署亚马逊虚拟私有云（Amazon VPC）的多账户环境（最好遵循账户结构 AWS 最佳实践通过 AWS Control Tower 进行设置）。
- 您的本地网络与 AWS 环境之间现有的混合连接（AWS Direct Connect 或 AWS Site-to-Site VPN）。
- Amazon VPC 对等互连 AWS Transit Gateway，或用于第 3 层网络之间 VPCs连接的 AWS Cloud WAN。（应用程序流量需要此连接。 不需要 DNS 解析才能起作用。 DNS 解析的运行独立于两者之间的网络连接 VPCs。）
- 在本地环境中运行的 DNS 服务器。

限制

- Route 53 Resolver 端点、规则和配置文件是区域结构，可能需要 AWS 区域 对全球组织进行多个复制。

- 有关 Route 53 解析器、私有托管区域和配置文件服务配额的完整列表，请参阅 Route 53 文档中的配额。

架构

目标技术堆栈

- 路由 53 出站和入站终端节点
- Route 53 条件转发的解析器规则
- AWS Resource Access Manager (AWS RAM)
- Route 53 私有托管区域

目标架构

出站和入站终端节点

下图显示了从本地 AWS 到内部的 DNS 解析流程。这是用于出站分辨率的连接设置，其中域名托管在内部。以下是设置过程的高级概述。有关详细信息，请参阅操作说明部分。

1. 在共享服务 VPC 中部署出站 Route 53 解析器终端节点。
2. 在共享服务帐户中为本地托管的域创建 Route 53 解析器规则（转发规则）。
3. VPCs 在托管需要解析本地托管域名的资源的其他账户中共享规则并将其关联。根据您的用例，可以通过不同的方式完成此操作，如本节后面所述。

设置连接后，出站解析所涉及的步骤如下：

1. 亚马逊弹性计算云 (Amazon EC2) 实例通过 VPC+2 地址向 VPC 的 Route 53 解析器发送 DNS 解析请求。`db.onprem.example.com`
2. Route 53 Resolver 检查解析器规则，并使用出站终端节点将请求转发到本地 DNS 服务器 IPs 。
3. 出站终端节点将请求转发到本地 DNS IPs。流量通过共享服务 VPC 和本地数据中心之间已建立的混合网络连接传输。
4. 本地 DNS 服务器回复出站终端节点，然后出站终端节点将响应转发回 VPC 的 Route 53 解析器。解析器将响应返回给 EC2 实例。

下图显示了从本地环境到的 DNS 解析流程 AWS。这是托管域名的入站分辨率的连接设置 AWS。以下是设置过程的高级概述。有关详细信息，请参阅操作说明部分。

1. 在共享服务 VPC 中部署入站解析器终端节点。
2. 在共享服务账户中创建私有托管区域（集中式方法）。
3. 将私有托管区域与共享服务 VPC 关联。共享这些区域并将其与跨账户关联以进行 VPC-to-VPC DN VPCs S 解析。根据您的用例，可以通过不同的方式完成此操作，如本节后面所述。

设置连接后，入站解析所涉及的步骤如下：

1. 本地资源向本地 DNS 服务器发送 DNS 解析请求。`ec2.prod.aws.example.com`
2. 本地 DNS 服务器通过混合网络连接将请求转发到共享服务 VPC 中的入站解析器终端节点。
3. 入站解析器终端节点在 VPC Route 53 解析器的帮助下，在关联的私有托管区域中查找请求并获取相应的 IP 地址。
4. 这些 IP 地址被发送回本地 DNS 服务器，后者会将响应返回到本地资源。

此配置使本地资源能够通过入站端点将查询路由到相应的 AWS 私有托管区域，从而解析私有域名。在此架构中，私有托管区域集中在共享服务 VPC 中，允许单个团队进行中央 DNS 管理。这些区域可以与许多区域相关联 VPCs ，以解决 VPC-to-VPC DNS 解析用例。或者，您可能希望将 DNS 域的所有权和管理委托给每个人 AWS 账户。在这种情况下，每个账户都管理自己的私有托管区域，并将每个区域与中央共享服务 VPC 关联起来，以实现与本地环境的统一解决方案。这种去中心化的方法超出了这种模式的范围。有关更多信息，请参阅跨多个账户扩展 DNS 管理以及 VPCs Amazon VPC 的混合云 DNS 选项白皮书。

使用解析器终端节点建立基本的 DNS 解析流程时，您需要确定如何管理跨您的解析器规则和私有托管区域的共享和关联。 AWS 账户您可以通过两种方式来解决这个问题：通过使用 AWS RAM 共享解析器规则和直接私有托管区域关联进行自我管理共享（如基本设置部分所述），或者通过 Route 53 配置文件（如增强设置部分所述）。选择取决于贵组织的 DNS 管理偏好和操作要求。以下架构图说明了一个扩展的环境，该环境包括多个 VPCs 跨不同账户的环境，这代表了典型的企业部署。

基本设置

在基本设置中，在多账户 AWS 环境中实现混合 DNS 解析时使用 AWS RAM 共享解析器转发规则和私有托管区域关联来管理本地和 AWS 资源之间的 DNS 查询。此方法使用连接到您的本地网络的共享服务 VPC 中的集中式 Route 53 解析器终端节点来高效处理入站和出站 DNS 解析。

- 要进行出站解析，需要在 Shared Services 账户中创建解析器转发规则，然后使用 AWS RAM与其他 AWS 账户 人共享。此共享仅限于同一区域内的账户。然后，目标账户可以将这些规则与其关联起来， VPCs 并使这些规则中的资源 VPCs 能够解析本地域名。
- 为了进行入站解析，私有托管区域是在共享服务账户中创建的，并与共享服务 VPC 关联。然后，可以使用 Rout VPCs e 53 API 或 AWS Command Line Interface (AWS CLI) 将这些区域关联到其他账户中。 AWS SDKs然后，关联中的资源 VPCs 可以解析私有托管区域中定义的 DNS 记录，从而在您的 AWS 环境中创建统一的 DNS 视图。

下图显示了此基本设置中的 DNS 解析流程。

当您在有限的规模上使用 DNS 基础设施时，此设置效果很好。但是，随着环境的增长，管理起来可能会变得困难。随着规模的扩展，管理私有托管区域和解析器规则的共享和关联方式的运营开 VPCs 销会显著增加。此外，服务配额（例如每个私有托管区域的 300 个 VPC 关联限制）可能会成为大规模部署的限制因素。增强的设置解决了这些难题。

增强的设置

Route 53 配置文件为管理跨多个混合网络中的 DNS 解析提供了简化的解决方案 AWS 账户。您无需单独管理私有托管区域和解析器规则，而是可以将 DNS 配置分组到一个容器中，便于在一个区域中的多个账户之间共享 VPCs 和应用。此设置在共享服务 VPC 中维护集中式解析器终端节点架构，同时显著简化了 DNS 配置的管理。

下图显示了增强设置中的 DNS 解析流程。

Route 53 配置文件允许您将私有托管区域关联、解析器转发规则和 DNS 防火墙规则打包到一个可共享的单元中。您可以在 Shared Services 帐户中创建个人资料并使用将其与成员帐户共享 AWS RAM。共享配置文件并将其应用于目标时 VPCs，服务会自动处理所有必要的关联和配置。这显著降低了 DNS 管理的运营开销，并为不断增长的环境提供了出色的可扩展性。

自动化和扩缩

使用 Terraform 等 AWS CloudFormation 基础设施即代码 (IaC) 工具自动配置和管理 Route 53 Resolver 端点、规则、私有托管区域和配置文件。将 DNS 配置与持续集成和持续交付 (CI/CD) 管道集成，以实现一致性、可重复性和快速更新。

工具

AWS 服务

- AWS Resource Access Manager (AWS RAM) 可帮助您安全地共享资源， AWS 账户 从而减少运营开销并提供可见性和可审计性。
- Amazon Route 53 Resolver以递归方式响应来自 AWS 资源的 DNS 查询，默认情况下全部 VPCs可用。您可以创建 Resolver 端点和条件转发规则，以解析本地数据中心和您的数据中心之间的 DNS 命名空间。 VPCs
- Amazon Route 53 私有托管区域是一个容器，其中包含有关您希望 Route 53 如何响应域及其子域名的 DNS 查询的信息。
- Amazon Route 53 配置文件允许您以简化的方式在许多 VPCs 不同 AWS 账户 版本中应用和管理与 DNS 相关的 Route 53 配置。

最佳实践

本节提供了优化 Route 53 Resolver 的一些最佳实践。这些是 Route 53 最佳实践的一部分。有关完整列表，请参阅 Amazon Route 53 的最佳实践。

避免使用解析器端点进行循环配置

- 设计您的 DNS 架构，通过仔细规划 VPC 关联来防止递归路由。当 VPC 托管入站终端节点时，请避免将其与可能创建循环引用的解析器规则相关联。
- 当您在账户之间共享 DNS 资源以保持干净的路由路径时，请 AWS RAM 策略性地使用。

有关更多信息，请参阅 Route 53 文档中的避免使用解析器端点进行环路配置。

缩放解析器端点

- 对于需要高每秒查询数 (QPS) 的环境，请注意，终端节点中每个 ENI 的 QPS 限制为 10,000。 ENIs 可以向终端节点添加更多内容以扩展 DNS QPS。
- Amazon CloudWatch 提供`InboundQueryVolume`和`OutboundQueryVolume`指标（参见CloudWatch 文档）。我们建议您设置监控规则，在阈值超过特定值（例如，10,000 QPS 的 80%）时提醒您。
- 为解析器端点配置状态安全组规则，以防止连接跟踪限制在高流量流量期间导致 DNS 查询受限。要详细了解安全组中连接跟踪的工作原理，请参阅亚马逊 EC2 文档中的亚马逊 EC2 安全组连接跟踪。

有关更多信息，请参阅 Route 53 文档中的解析器端点扩展。

为解析器端点提供高可用性

- 创建入站端点，采用至少两个可用区中的 IP 地址以实现冗余。
- 配置额外的网络接口，以确保维护期间或流量激增期间的可用性。

有关更多信息，请参阅 Route 53 文档中的解析器终端节点的高可用性。

操作说明

部署 Route 53 解析器端点

Task	描述	所需技能
部署入站终端节点。	Route 53 Resolver 使用入站端点接收来自本地 DNS 解析程序的 DNS 查询。有关说明，请参阅 Route 53 文档 VPCs 中的将入站 DNS 查询转发给您。记下入站端点 IP 地址。	AWS 管理员、云管理员
部署出站终端节点。	Route 53 解析程序使用出站端点向本地 DNS 解析程序发送 DNS 查询。有关说明，请参阅 Route 53 文档中的将出站 DNS 查询转发到您的网络。记下输出端点 ID。	AWS 管理员、云管理员

配置和共享 Route 53 私有托管区域

Task	描述	所需技能
为托管的域创建私有托管区域 AWS。	此区域保存 AWS托管域（例如`prod.aws.example.com`）中应从本地环境解析的资源的 DNS 记录。有关说明，	AWS 管理员、云管理员

Task	描述	所需技能
	请参阅 Route 53 文档中的创建私有托管区。 创建私有托管区域时，必须将 VPC 与同一账户拥有的托管区域关联。为此选择共享服务 VPC。	
基本设置：将私有托管区域与其他 VPCs 账户相关联。	如果你使用的是基本设置（参见 “架构” 部分）： 要使成员账户中的资源 VPCs 能够解析此私有托管区域中的 DNS 记录，您必须将您的资源 VPCs 与托管区域关联。您必须授权关联，然后以编程方式建立关联。有关说明，请参阅 Rou te 53 文档 AWS 账户中的将 Amazon VPC 与您创建的私有托管区域关联到其他区域。	AWS 管理员、云管理员

Task	描述	所需技能
增强设置：配置和共享 Route 53 配置文件。	如果您使用的是增强设置（请参阅 “架构” 部分）： 1. 创建 Route 53 配置文件并将相关的私有托管区域与之关联。有关说明，请参阅 Route 53 文档中的创建 Route 53 配置文件。 2. 用于 AWS RAM 与成员帐户共享个人资料，然后将共享的个人资料与目标相关联 VPCs。有关说明，请参阅 Route 53 文档 VPCs中的共享 R oute 53 配置文件和将 Route 53 配置文件关联到。 **Note** 根据组织的结构和 DNS 要求，您可能需要为不同的账户或工作负载创建和管理多个配置文件。	AWS 管理员、云管理员

配置和共享 Route 53 解析器转发规则

Task	描述	所需技能
为本地托管的域名创建转发规则。	此规则将指示 Route 53 解析器将针对本地域（例如`onprem.example.com`）的任何 DNS 查询转发到本地 DNS 解析器。要创建此规	AWS 管理员、云管理员

Task	描述	所需技能
	则，您需要本地 DNS 解析器的 IP 地址和出站终端节点 ID。有关说明，请参阅 Route 53 文档中的创建转发规则。	
基本设置：共享转发规则并将其与您在其他账户 VPCs 中的账户关联。	如果你使用的是基本设置： 要使转发规则生效，您必须将该规则与您在其他账户 VPCs 中共享和关联。然后，Route 53 解析器在解析域名时会考虑该规则。有关说明，请参阅 Route 53 文档中的与其他人共享解析器规则 AWS 账户 和使用共享规则和将转发规则与 VPC 关联。	AWS 管理员、云管理员

Task	描述	所需技能
增强设置：配置和共享 Route 53 配置文件。	如果您使用的是增强设置： 1. 如果您已经在前面的步骤中创建了 Route 53 配置文件，则可以使用相同的配置文件。如果不是，请创建 Route 53 配置文件并将相关的解析器转发规则与其关联起来。有关说明，请参阅 Route 53 文档中的创建 Route 53 配置文件。 2. 用于 AWS RAM 与成员帐户共享个人资料，然后将共享的个人资料与目标关联 VPCs。有关说明，请参阅 Route 53 文档 VPCs中的共享 R oute 53 配置文件和将 Route 53 配置文件关联到。 **Note** 根据组织的结构和 DNS 要求，您可能需要为不同的账户或工作负载创建和管理多个配置文件。	AWS 管理员、云管理员

配置本地 DNS 解析器以进行集成 AWS

Task	描述	所需技能
在本地 DNS 解析器中配置条件转发。	要 AWS 从本地环境发送 DNS 查询以进行解析，您必须在	网络管理员

Task	描述	所需技能
	本地 DNS 解析器中配置条件转发以指向入站终端节点 IP 地址。这指示 DNS 解析器将 AWS托管域（例如`prod.aws.example.com`）的所有 DNS 查询转发到入站终端节点 IP 地址，由 Route 53 解析器进行解析。	

在混合环境中验证 end-to-end DNS 解析

Task	描述	所需技能
测试从本地环境 AWS 到本地环境的 DNS 解析。	从 VPC 中与其关联的转发规则的实例中，对本地托管域执行 DNS 查询（例如，对于`db.onprem.example.com`）。	网络管理员
测试从本地环境到的 DNS 解析 AWS。	在本地服务器上，对 AWS托管域执行 DNS 解析（例如，对于`ec2.prod.aws.example.com`）。	网络管理员

相关资源

- 亚马逊 VPC 的混合云 DNS 选项（AWS 白皮书）
- 使用私有托管区域（Route 53 文档）
- R@@ oute 53 解析器入门（Route 53 文档）
- 使用 Route 53 解析器简化多账户环境中的 DNS 管理（AWS 博客文章）
- 使用带有多个 VPCs 和 AWS 账户（AWS 博客文章）的 Amazon Route 53 配置文件统一 DNS 管理
- 将您的多账户 DNS 环境迁移到 Amazon Route 53 配置文件（AWS 博客文章）
- 将 Amazon Route 53 配置文件用于可扩展的多账户 AWS 环境（AWS 博客文章）

验证 ELB 负载均衡器是否需 TLS 终止

由 Priyanka Chaudhary (AWS) 编写

摘要

在亚马逊网络服务 (AWS) 云上，Elastic Load Balancing (ELB) 会自动将传入的应用程序流量分配到多个目标，例如亚马逊弹性计算云 (Amazon EC2) 实例、容器、IP 地址和 AWS Lambda 函数。负载均衡器使用侦听器定义负载均衡器用来接受来自用户的流量的端口和协议。应用程序负载均衡器在应用程序层做出路由决策并使用. HTTP/HTTPS protocols. Classic Load Balancers make routing decisions at either the transport layer, by using TCP or Secure Sockets Layer (SSL) protocols, or at the application layer, by using HTTP/HTTPS

此模式提供了安全控制，用于检查应用程序负载均衡器和经典负载均衡器的多种事件类型。调用该函数时，AWS Lambda 会检查事件，并确保负载均衡器合规。

该函数通过以下 API 调用启动 Amazon Events CloudWatch 事件：CreateLoadBalancerCreateLoadBalancerListenersDeleteLoadBalancerListeners、、CreateLoadBalance和ModifyListener。当事件检测到其中一个事件时 APIs，它会调用运行 Python 脚本的 AWS Lambda。Python 脚本会进行评估以查看侦听器是否包含 SSL 凭证，以及所应用的策略是否使用传输层安全性协议（TLS）。如果确定 SSL 策略不是 TLS，则该函数会向用户发送包含相关信息的 Amazon Simple Notification Service (Amazon SNS) 通知。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account

限制

- 若非对负载均衡器侦听器进行了更新，否则此安全控制不会检查现有的负载均衡器。
- 这种安全控制为区域性的。您必须将其部署在要监控的每个 AWS 区域。

架构

目标架构

自动化和扩缩

- 如果您使用的是 AWS Organ izations，则可以使用 AWS Cloudformation StackSets 在要监控的多个账户中部署此模板。

工具

Amazon Web Services

- AWS CloudFormation — AWS CloudFormation 可帮助您建模和设置 AWS 资源，快速一致地配置这些资源，并在资源的整个生命周期中对其进行管理。您可以使用模板来描述资源及其依赖关系，然后将它们作为堆栈一起启动和配置，而不必单独管理资源。
- Amazon CloudWatch Events — Amazon CloudWatch Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。
- AWS Lambda – AWS Lambda 是一项计算服务，支持无需预置或管理服务器即可运行代码。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项高度可扩展的对象存储服务，可用于各种存储解决方案，包括网站、移动应用程序、备份和数据湖。
- Amazon SNS — Amazon Simple Notification Service (Amazon SNS) 可协调和管理发布者和客户端之间消息的传送或发送，包括 Web 服务器和电子邮件地址。订阅用户接收所有发布至他们所订阅主题的消息，并且一个主题的所有订阅用户收到的消息都相同。

代码

此模式包括以下附件：

- `ELBRequirestlstermination.zip` — 用于安全控制的 Lambda 代码。
- `ELBRequirestlstermination.yml`— 用于设置事件和 Lambda 函数的 CloudFormation 模板。

操作说明

设置 S3 存储桶

Task	描述	所需技能
定义 S3 存储桶。	在 Amazon S3 控制台，选择或创建 S3 存储桶来托管	云架构师

Task	描述	所需技能
	Lambda 代码 .zip 文件。此 S3 存储桶必须与要评估的负载均衡器位于同一 AWS 区域 中。S3 存储桶名称是全局唯一的，并且命名空间由所有 Amazon Web Services account 共享。S3 存储桶名称不得包含前导斜杠。	
上传 Lambda 代码。	将附件部分中提供的 Lambda 代码（`ELBRequirestlstermination.zip` 文件）上传至 S3 存储桶。	云架构师

部署 CloudFormation 模板

Task	描述	所需技能
启动 AWS CloudFormation 模板。	在与 S3 存储桶相同的 AWS 区域中打开 AWS CloudFormation 控制台，然后部署所附的模板`ELBRequirestlstermination.yml` 。有关部署 AWS CloudFormation 模板的更多信息，请参阅 CloudFormation 文档中的在 AWS CloudFormation 控制台上创建堆栈。	云架构师
填写模板中的参数。	启动模板时，系统将会提示输入以下信息： • S3 存储桶：指定您在首个操作说明中创建或选择的存	云架构师

Task	描述	所需技能
	储桶。这是您上传所附的 Lambda 代码（`ELBRequirestlstermination.zip` 文件）的地方。 • S3 密钥：指定 Lambda .zip 文件在您的 S3 存储桶中的位置(例如 `ELBRequirestlstermination.zip` 或 `controls/ELBRequirestlstermination.zip`)。切勿纳入前导斜字符。 • 通知电子邮件：提供有效的电子邮件地址以接收 Amazon SNS 通知。 • Lambda 日志级别： 指定 Lambda 函数的日志记录级别和频率。使用信息记录有关进度的详细信息消息，使用错误记录仍然允许部署继续的错误事件，使用警告记录潜在的有害情况。	

确认订阅

Task	描述	所需技能
确认订阅。	CloudFormation 模板成功部署后，它会向您提供的电子邮件地址发送一封订阅电子邮件。您必须确认此电子邮件订阅，才能开始接收违规通知。	云架构师

相关资源

- 在 AWS CloudFormation 控制台上创建堆栈（AWS CloudFormation 文档）
- 什么是 AWS Lambda？(AWS Lambda 文档)
- 什么是经典负载均衡器？(ELB 文档)
- 什么是应用程序负载均衡器？（ELB 文档）

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 Splunk 查看 AWS Network Firewall 日志和指标

由 Ivo Pinto 创作

摘要

许多组织使用 Splunk Enter prise 作为来自不同来源的日志和指标的集中聚合和可视化工具。 此模式可帮助您配置 Splunk，使其使用适用于 AWS 的 Splunk 插件从亚马逊 CloudWatch 日志中获取 AWS Network Firew all 日志和指标。

为此，您需要创建一个只读的 AWS Identity and Access Management (IAM) 角色。适用于 AWS 的 Splunk 附加组件使用此角色进行访问 CloudWatch。您可以配置 AWS 的 Splunk 插件以从中 CloudWatch获取指标和日志。最后，您可以在 Splunk 中根据检索到的日志数据和指标创建可视化效果。

先决条件和限制

先决条件

- 一个 Splunk 账户
- Splunk Enterprise 实例，版本 8.2.2 或更高版本
- 一个有效的 Amazon Web Services account
- Network Firewall，设置并配置为向日志发送 CloudWatch 日志

限制

- Splunk Enterprise 必须作为亚马逊弹性计算云 (Amazon EC2) 实例的集群部署在 AWS 云中。
- AWS 中国区域不支持使用自动发现 EC2 的 Amazon IAM 角色收集数据。

架构

该图阐释了以下内容：

1. Network Firewall 将日志发布到 CloudWatch 日志。
2. Splunk Enterprise 从中检索指标和日志。 CloudWatch

为了在此架构中填充示例指标和日志，工作负载会生成通过 Network Firewall 端点进入互联网的流量。这是通过使用路由表来实现的。 尽管此模式使用单个 Amazon EC2 实例作为工作负载，但只要将 Network Firewall 配置为向 CloudWatch 日志发送日志，这种模式就可以应用于任何架构。

该架构还使用另一个虚拟私有云 (VPC) 中的 Splunk Enterprise 实例。但是，Splunk 实例可以位于其他位置，例如与工作负载位于同一个 VPC 中，前提是它可以到达。 CloudWatch APIs

工具

Amazon Web Services

- Amazon CloudWatch Lo gs 可帮助您集中管理来自所有系统、应用程序和 AWS 服务的日志，以便您可以监控它们并安全地将其存档。
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- AWS Network Fire wall 是一项有状态的托管式网络防火墙以及入侵检测和防御服务，适用于 AWS 云 VPCs 中。

其他工具

- Splunk 可帮助您监控、可视化并分析日志数据。

操作说明

创建 IAM 角色

Task	描述	所需技能
创建 IAM policy。	按照使用 JSON 编辑器创建策略中的说明创建授予 CloudWatch 日志数据和 CloudWatch 指标只读访问权限的 IAM 策略。将下面的 策略粘贴到 JSON 编辑器中。 `{ "Statement": [ {`	AWS 管理员

Task	描述	所需技能
	"Action": ["cloudwatch:List*", "cloudwatch:Get*", "network-firewall:List*", "logs:Describe*", "logs:Get*", "logs:List*", "logs:StartQuery", "logs:StopQuery", "logs:TestMetricFilter", "logs:FilterLogEvents", "network-firewall:Describe*"], "Effect": "Allow", "Resource": "*" }], "Version": "2012-10-17" }	

Task	描述	所需技能
创建新的 IAM 角色。	按照创建角色中的说明向 AWS 服务委派权限，创建适用于 AWS 的 Splunk 附加组件访问 CloudWatch的 IAM 角色。对于权限策略，请选择您之前创建的策略。	AWS 管理员
为 Splunk 集群中的 EC2 实例分配 IAM 角色。	1. 打开 Amazon EC2 控制台，网址为https://console.aws.amazon.com/ec2/。 2. 在导航窗格中，选择实例。 3. 在 Splunk 集群中选择 EC2 实例。 4. 选择 “操作”、“安全”，然后选择 “修改 IAM 角色”。 5. 选择您之前创建的 IAM 角色，然后选择保存。	AWS 管理员

安装适用于 AWS 的 Splunk 附加组件

Task	描述	所需技能
安装附加组件。	1. 在 Splunk 控制面板中，导航到 Splunk 应用程序。 2. 搜索适用于亚马逊 Web Services 的 Splunk 附加组件。 3. 选择安装。 4. 提供您的 Splunk 凭证。	Splunk 管理员
配置 AWS 证书。	1. 在 Splunk 控制面板中，导航到适用于 AWS 的 Splunk 附加组件。	Splunk 管理员

Task	描述	所需技能
	2. 选择配置。 3. 在自动发现的 IAM 角色列中，选择您之前创建的 IAM 角色。 有关更多信息，请参阅 Splunk 文档中的在您的 Splunk 平台实例中查找 IAM 角色。	

将 Splunk 的访问权限配置为 CloudWatch

Task	描述	所需技能
配置从日志中检索 Network Firewall 日 CloudWatch 志。	1. 在 Splunk 控制面板中，导航到适用于 AWS 的 Splunk 附加组件。 2. 选择 “输入”。 3. 选择 “创建新输入”。 4. 在列表中，选择 “自定义数据类型”，然后选择 “CloudWatch 日志”。 5. 为您的 Network Firewall 日志提供名称、AW S 账户、AWS 区域和日志组。 6. 选择保存。 默认情况下，Splunk 每 10 分钟提取一次日志数据。这是高级设置下的可配置参数。有关更多信息，请参阅 Splunk 文档中的使用 Splunk Web 配置 CloudWatch 日志输入。	Splunk 管理员

Task	描述	所需技能
配置从中检索 Network Firewall 指标 CloudWatch。	1. 在 Splunk 控制面板中，导航到适用于 AWS 的 Splunk 附加组件。 2. 选择 “输入”。 3. 选择 “创建新输入”。 4. 在列表中，选择CloudWatch。 5. 为您的 Network Firewall 指标提供名称、AW S 账户和 AWS 区域。 6. 在 “指标配置” 旁边，选择 “在高级模式下编辑”。 7. （可选）删除所有预配置的命名空间。 8. 选择 “添加命名空间”，然后将其命名为 AWS/NetworkFirewall。 9. 在维度值中，添加以下内容。 `[{"AvailabilityZone":[".*"],"Engine":[".*"],"FirewallName":[".*"]}]` 10对于指标，选择全部。 11对于指标统计信息，选择总和。 12选择确定。 13选择保存。 默认情况下，Splunk 每 5 分钟提取一次指标数据。这是高	Splunk 管理员

Task	描述	所需技能
	级设置下的可配置参数。有关更多信息，请参阅 Splunk 文档中的使用 Splunk Web 配置 CloudWatch 输入。	

使用查询创建 Splunk 可视化效果

Task	描述	所需技能
查看排名靠前的源 IP 地址。	1. 在 Splunk 控制面板中，导航到“搜索和报告”。 2. 在“在此处输入搜索”框中，输入以下内容。 `sourcetype="aws:cloudwatchlogs" | top event.src_ip` 此查询按降序显示流量最多的源 IP 地址表。 3. 对于图形表示，请选择可视化。	Splunk 管理员
查看数据包统计信息。	1. 在 Splunk 控制面板中，导航到“搜索和报告”。 2. 在“在此处输入搜索”框中，输入以下内容。 `sourcetype="aws:cloudwatch"| timechart sum(Sum) by metric_name` 此查询显示ReceivedPackets 每分钟	Splunk 管理员

Task	描述	所需技能
	指标DroppedPackets PassedPackets 、和的表。 3. 对于图形表示，请选择可视化。	
查看最常用的源端口。	1. 在 Splunk 控制面板中，导航到 “搜索和报告”。 2. 在 “在此处输入搜索” 框中，输入以下内容。 `sourcetype="aws:cloudwatchlogs" | top event.dest_port` 此查询按降序显示流量最大的源端口表。 3. 对于图形表示，请选择可视化。	Splunk 管理员

相关资源

AWS 文档

- 创建向 AWS 服务委派权限的角色（IAM 文档）
- 创建 IAM policy（IAM 文档）
- 在 AWS Network Firewall 中进行日志记录和监控（网络防火墙文档）
- AWS Network Firewall 的路由表配置（网络防火墙文档）

AWS Blog 文章

- AWS Network Firewall 部署模型

Amazon Web Services Marketplace

- Splunk Enterprise Amazon 机器映像 (AMI)

更多模式

- 使用会话管理器和 Amazon Instance Connect 访问堡垒主 EC2 机
- 使用 AWS Fargate PrivateLink、AWS 和网络负载均衡器在 Amazon ECS 上私下访问容器应用程序
- 使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon ECS 上私密访问容器应用程序
- 使用本地微软 Active Direct AWS Managed Microsoft AD ory 集中解析 DNS
- 检查和的安全组入口规则中是否有单主机网络条目 IPv4 IPv6
- 使用 AWS Amplify、Angular 和模块联合为微前端创建门户
- 使用 AWS Network Firewall 和 AWS Transit Gateway 部署防火墙
- 使用私有端点和应用程序负载均衡器在内部网站上部署 Amazon API Gateway API
- 使用以下方法为公共子网部署基于侦探属性的访问控制 AWS Config
- 为公有子网部署基于属性的预防性访问控制
- 在 Amazon RDS 中为 PostgreSQL 数据库实例启用加密连接
- 使用 AWS Transit Gateway Connect 扩展 VRFs 到 AWS
- 将 F5 BIG-IP 工作负载迁移至 Amazon Web Services Cloud 上的 F5 BIG-IP VE
- 启用 Amazon EKS 自动模式时迁移 NGINX 入口控制器
- 在非工作负载子网的多账户 VPC 设计中保留可路由的 IP 空间
- 使用服务控制策略阻止账户级别的互联网访问
- 将警报从 AWS Network Firewall 发送到 Slack 通道
- 使用亚马逊通过 VPC 在 Amazon S3 存储桶中提供静态内容 CloudFront
- 使用 AWS 弹性灾难恢复为 Oracle JD Edwar EnterpriseOne ds 设置灾难恢复
- 使用 BMC Discovery 查询提取迁移数据以进行迁移规划
- 使用 Network Firewall 从服务器名称指示中捕获出站流量的 DNS 域名

内容分发

主题

- 使用 AWS Firewall Manager 和 Amazon Data Firehose 将 AWS WAF 日志发送到 Splunk
- 使用亚马逊通过 VPC 在 Amazon S3 存储桶中提供静态内容 CloudFront
- 更多模式

使用 AWS Firewall Manager 和 Amazon Data Firehose 将 AWS WAF 日志发送到 Splunk

由 Michael Friedenthal (AWS)、Aman Kaur Gandhi (AWS) 和 JJ Johnson (AWS) 创建

摘要

一直以来，有两种方法可将数据移入 Splunk：推送架构或提取架构。提取架构通过重试为交付数据提供保障，但它需要 Splunk 中的专用资源来轮询数据。因轮询之故，提取架构通常不是实时的。推送架构通常具有更低的延迟性、更高的可扩展性，并且可以降低操作复杂性和成本。但是，其无法保证交付，通常需要代理。

Splunk 与 Amazon Data Firehose 集成，通过 HTTP 事件收集器 (HEC) 向 Splunk 提供实时流式传输数据。此集成带来了推送架构和提取架构的优势，确保通过重试进行数据传输，近实时，延迟低、复杂性低。HEC 通过 HTTP 或 HTTPS 快速高效地将数据直接发送到 Splunk。 HECs 是基于令牌的，因此无需在应用程序或支持文件中对凭据进行硬编码。

在 AWS Firewall Manager 策略中，您可以为所有账户中的所有 AWS WAF Web ACL 流量配置日志记录，然后可以使用 Firehose 传输流将该日志数据发送到 Splunk 进行监控、可视化和分析。此解决方案具有以下优势：

- 集中管理和记录所有账户中的 AWS WAF Web ACL 流量
- Splunk 与单一集成 AWS 账户
- 可扩展性
- 日志数据的近实时传送
- 成本已经无服务器解决方案优化，因此您无需为未使用的资源付费。

先决条件和限制

先决条件

- 属于组织 AWS 账户 成员的活跃分子 AWS Organizations。
- 要使用 Firehose 启用日志功能，您必须具有以下权限：
  - `iam:CreateServiceLinkedRole`
  - `firehose:ListDeliveryStreams`
  - `wafv2:PutLoggingConfiguration`

- AWS WAF 并且 ACLs 必须对其网络进行配置。有关说明，请参阅入门 AWS WAF。
- AWS Firewall Manager 必须进行设置。有关说明，请参阅AWS Firewall Manager 先决条件。
- AWS WAF 必须配置的 Firewall Manager 安全策略。有关说明，请参阅AWS Firewall ManagerAWS WAF 策略入门。
- Splunk 必须使用可通过 Firehose 访问的公共 HTTP 端点进行设置。

限制

- AWS 账户 必须在中的单个组织中进行管理 AWS Organizations。
- Web ACL 必须与传输流位于同一区域。如果您要为亚马逊捕获日志 CloudFront，请在美国东部（弗吉尼亚北部）地区创建 Firehose 传送流。`us-east-1`
- 适用于 Firehose 的 Splunk 插件可用于付费 Splunk 云部署、分布式 Splunk Enterprise 部署和单实例 Splunk Enterprise 部署。该附加组件不支持免费试用 Splunk Cloud 部署。

架构

目标技术堆栈

- Firewall Manager
- Firehose
- Amazon Simple Storage Service（Amazon S3）
- AWS WAF
- Splunk

目标架构

下图显示了如何使用防火墙管理器集中记录所有 AWS WAF 数据并通过 Firehose 将其发送到 Splunk。

1. AWS WAF 网络 ACLs 向 Firewall Manager 发送防火墙日志数据。
2. Firewall Manager 将日志数据发送到 Firehose。
3. Firehose 传输流将日志数据转发到 Splunk 和 S3 存储桶。当 Firehose 传输流出现错误时，S3 存储桶可充当备份。

自动化和扩缩

该解决方案旨在扩展和适应组织 ALCs 内的所有 AWS WAF Web。您可以将所有网络配置 ACLs 为使用相同的 Firehose 实例。但是，如果您想设置和使用多个 Firehose 实例，则可以。

工具

AWS 服务

- AWS Firewall Manager是一项安全管理服务，可帮助您在中的账户和应用程序中集中配置和管理防火墙规则 AWS Organizations。
- Amazon Data Firehose 可帮助您将实时流数据传输到受支持的第三方服务提供商（例如 Splunk）拥有的其他 AWS 服务自定义 HTTP 终端节点和 HTTP 终端节点。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS WAF是一种 Web 应用程序防火墙，可帮助您监控转发到受保护的 Web 应用程序资源的 HTTP 和 HTTPS 请求。

其他工具

- Splunk 可帮助您监控、可视化并分析日志数据。

操作说明

配置 Splunk

Task	描述	所需技能
安装适用于 Splunk 的应用程序。 AWS	1. 登录您的 Splunk heavy 转发服务器。默认 URL 为 `http://<IP address>:8000` 。 2. 在左侧导航栏中应用程序旁，选择齿轮按钮。 3. 选择浏览更多应用程序。 4. 搜索 AWS。	安全管理员、Splunk 管理员

Task	描述	所需技能
	5. 对于适用于 Splunk 的应用程序 AWS，请选择安装。 6. 输入 Splunk.com 登录凭证，接受条款和条件，然后选择登录并安装。 7. 选择完成。	
安装的附加组件 AWS WAF。	重复前面的说明安装适用于 Splunk AWS 的 Web 应用程序防火墙插件。	安全管理员、Splunk 管理员

Task	描述	所需技能
安装并配置 Firehose 的 Splunk 插件。	1. 安装并配置 Firehose 的 Splunk 插件。作为安装和配置的一部分，如果您的 Splunk 平台有必要，您可以设置 HTTP 事件收集器，并准备将日志数据发送至索引器的基础架构。请参阅与您的 Splunk 部署对应的说明： • Splunk Cloud 部署（Splunk 文档） • 分布式 Splunk Enterprise 部署（Splunk 文档） • 单实例 Splunk Enterprise 部署（Splunk 文档） **Important** 安装并配置 Splunk 插件后，停止此过程。请勿继续执行配置 Firehose 以向 Splunk 平台发送数据的说明。 2. 记下 HTTP 事件收集器令牌与 HTTP 端点。您稍后配置传送流时需要使用该值。	安全管理员、Splunk 管理员

创建 Firehose 传送流

Task	描述	所需技能
授予 Firehose 访问 Splunk 目的地的权限。	配置访问策略，允许 Firehose 访问 Splunk 目标并将日志数据备份到 S3 存储桶。有关更多信息，请参阅授予 Firehose 访问 Splunk 目标的权限。	安全管理员
创建 Firehose 传送流。	在您管理网页 ACLs 的同一个账户中 AWS WAF，在 Firehose 中创建传送流。创建传输流时，您需要拥有 IAM 角色。Firehose 担任该 IAM 角色并获得对指定 S3 存储桶的访问权限。有关说明，请参阅创建传输流。请注意以下几点： • 传输流名称必须以 `aws-waf-logs-` 开头。 • 对于源，请选择直接 PUT。 • 对于 S3 备份模式，请选择备份所有事件，然后选择现有存储桶或创建新存储桶。 • 对于目的地，请按照 Firehose 文档中为目的地选择 Splunk 中的说明进行操作。有关 Splunk 终端节点和终端节点类型的值的信息，请参阅 Splunk 文档中的配置 Amazon Data Firehos e。 对您在 HTTP 事件收集器中配置的每个令牌重复此过程。	安全管理员

Task	描述	所需技能
测试传输流。	测试传输流以验证配置是否正确。有关说明，请参阅 Firehose 文档中的使用 Splunk 作为目标进行测试。	安全管理员

将 Firewall Manager 配置为记录数据

Task	描述	所需技能
配置 Firewall Manager 策略。	必须将 Firewall Manager 策略配置为启用日志记录并将日志转发到正确的 Firehose 传输流。有关更多信息和说明，请参阅为 AWS WAF 策略配置日志记录。	安全管理员

相关资源

AWS resources

- 记录 Web ACL 流量（AWS WAF 文档）
- 为 AWS WAF 策略配置日志记录（AWS WAF 文档）
- 教程：使用亚马逊数据 Firehose 向 Splunk 发送 VPC 流日志（Firehos e 文档）
- 如何使用 Amazon Data Firehose 将 VPC 流日志推送到 Splunk？（AWS 知识中心）
- 使用 Amazon Data Firehose 支持向 Splunk 摄取数据（博客文章）AWS

Splunk 文档

- 适用于亚马逊 Data Firehose 的 Splunk 附加组件

使用亚马逊通过 VPC 在 Amazon S3 存储桶中提供静态内容 CloudFront

创建者：Angel Emmanuel Hernandez Cebrian

摘要

当您提供托管在亚马逊网络服务 (AWS) 上的静态内容时，推荐的方法是使用亚马逊简单存储服务 (S3) Simple Service 存储桶作为来源，然后使用 CloudFront 亚马逊来分发内容。该解决方案有两个主要优点：便于在边缘位置缓存静态内容，以及能够为 CloudFront 分发定义 Web 访问控制列表 (Web ACLs)，这有助于您以最小的配置和管理开销保护对内容的请求。

然而，标准推荐方法有一个常见的架构限制。在某些环境中，您希望部署在虚拟私有云（VPC）中的虚拟防火墙设备检查所有内容，包括静态内容。标准方法不通过 VPC 路由流量进行检查。此模式提供了另一种架构解决方案。您仍然使用 CloudFront 分配在 S3 存储桶中提供静态内容，但是流量是使用 Application Load Balancer 通过 VPC 路由的。然后，AWS Lambda 函数从 S3 存储桶检索并返回内容。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- S3 存储桶中托管的静态网站内容。

限制

- 这种模式中的资源必须位于单个 AWS 区域 ，但可以在不同的 Amazon Web Services account 中进行预调配。
- 限制分别适用于 Lambda 函数可以接收和发送的最大请求与响应大小。有关更多信息，请参阅 Lambda 函数作为目标中的限制（弹性负载均衡文档）。
- 使用此方法时，在性能、可扩展性、安全性和成本效益之间找到良好的均衡非常重要。尽管 Lambda 具有很高的可扩展性，但如果并发 Lambda 调用数量超过最大限额，某些请求会受到限制。有关更多信息，请参阅 Lambda 限额（Lambda 文档）。在使用 Lambda 时，您还需考虑定价。要最大限度地减少 Lambda 调用，请确保正确定义分配的缓存。 CloudFront 有关更多信息，请参阅优化缓存和可用性（CloudFront 文档）。

架构

目标技术堆栈

- CloudFront
- Amazon Virtual Private Cloud（Amazon VPC）
- 应用程序负载均衡器
- Lambda
- Amazon S3

目标架构

下图显示了当您需要使用 CloudFront 通过 VPC 从 S3 存储桶提供静态内容时建议的架构。

1. 客户端请求 CloudFront 分发 URL 以获取 S3 存储桶中的特定网站文件。
2. CloudFront 将请求发送到 AWS WAF。AWS WAF 使用 ACLs 应用于分配的 Web 来筛选请求。CloudFront 如果确定请求有效，则流程继续。如果请求被确定为无效，客户端会收到 403 错误。
3. CloudFront 检查其内部缓存。如果存在与传入请求匹配的有效密钥，则关联的值将作为响应发送回客户端。否则，流程将继续。
4. CloudFront 将请求转发到指定的 Application Load Balancer 的 URL。
5. 应用程序负载均衡器有一个与基于 Lambda 函数目标组关联的侦听器。应用程序负载均衡器调用 Lambda 函数。
6. Lambda 函数连接至 S3 存储桶，对其执行 `GetObject` 操作，然后将内容作为响应返回。

自动化和扩缩

要使用此方法自动部署静态内容，请创建 CI/CD 管线来更新托管网站的 Amazon S3 存储桶。

Lambda 函数会在服务的限额和限制范围内自动扩缩以处理并发请求。有关更多信息，请参阅 Lambda 函数扩展和 Lambda 限额（Lambda 文档）。对于其他 AWS 服务和功能，例如 CloudFront 和 Application Load Balancer，AWS 会自动对其进行扩展。

工具

- Amazon 通过全球数据中心网络交付您的网页内容，从而降低延迟并提高性能，从而 CloudFront加快网络内容的分发。
- 弹性负载均衡（ELB）将传入的应用程序或网络流量分发到多个目标。在这种模式中，您可使用通过弹性负载均衡预调配的应用程序负载均衡器，将流量引导到 Lambda 函数。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Storage Service (Amazon S3)是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。

操作说明

用于通过 VPC 提供 CloudFront 来自 Amazon S3 的静态内容

Task	描述	所需技能
创建 VPC。	创建一个 VPC 以托管在此模式中部署的资源，例如应用程序负载均衡器和 Lambda 函数。 有关说明，请参阅创建 VPC（Amazon VPC 文档）。	云架构师
创建 AWS WAF web ACL。	创建 AWS WAF web ACL。稍后在此模式中，您将此 Web ACL 应用于 CloudFront 分发。有关说明，请参阅创建 Web ACL（AWS WAF 文档）。	云架构师
创建 Lambda 函数。	创建 Lambda 函数，将 S3 存储桶中托管的静态内容以网站形式提供。使用此模式的其他信息部分中提供的代码。自定	常规 AWS

Task	描述	所需技能
	义代码，以识别您的目标 S3 存储桶。	
上传 Lambda 函数。	输入以下命令将 Lambda 函数代码上传至 Lambda 中的 .zip 文件存档中。 `aws lambda update-function-code \ --function-name \ --zip-file fileb://lambda-alb-s3-website.zip`	常规 AWS
创建应用程序负载均衡器。	创建指向 Lambda 函数的面向互联网的应用程序负载均衡器。有关说明，请参阅为 Lambda 函数创建目标组（弹性负载均衡文档）。要实现高可用性配置，请创建应用程序负载均衡器并将其连接至不同可用区中的私有子网。	云架构师

Task	描述	所需技能
创建分 CloudFront 配。	创建指向您创建的 Application Load Balancer 的分 CloudFront 配。 1. 登录 AWS 管理控制台并在 https://console.aws.amazon.com/cloudfront/v3/ hom CloudFront e 中打开控制台。 2. 选择 Create Distribution（创建分配）。 3. 在创建分配向导的第一页上，在 Web 部分中选择开始使用。 4. 为您的分发指定设置。有关更多信息，请参阅您创建或更新分配时指定的值。请注意以下几点： a. 将应用程序负载均衡器设置为源。 b. 在分发设置中，选择要通过 AWS WAF 应用的现有网站 ACLs 。有关更多信息，请参阅 AWS WAF Web ACL。 5. 保存您的更改。 6. CloudFront 创建分配后，分配的 “状态” 列的值将从InProgress变为 “已部署”。如果您选择启用该分配，其状态变为已部署后就已就绪，可以处理请求了。	云架构师

相关资源

AWS 文档

- 优化缓存和可用性（CloudFront 文档）
- Lambda 函数作为目标（弹性负载均衡文档）
- Lambda 限额（Lambda 文档）

Amazon Web Services 网站

- 应用程序负载均衡器
- Lambda
- CloudFront
- Amazon S3
- AWS WAF
- Amazon VPC

其他信息

代码

下面的 Lambda函数示例是用 Node.js 编写而成。此 Lambda 函数充当 Web 服务器，对包含网站资源的 S3 存储桶执行 `GetObject` 操作。

```
/**

 * This is an AWS Lambda function created for demonstration purposes.

 * It retrieves static assets from a defined Amazon S3 bucket.

 * To make the content available through a URL, use an Application Load Balancer with a
 Lambda integration.
 *
 * Set the S3_BUCKET environment variable in the Lambda function definition.
 */

var AWS = require('aws-sdk');
```

```
exports.handler = function(event, context, callback) {

    var bucket = process.env.S3_BUCKET;
    var key = event.path.replace('/', '');

    if (key == '') {
        key = 'index.html';
    }

    // Fetch from S3
    var s3 = new AWS.S3();
    return s3.getObject({Bucket: bucket, Key: key},
       function(err, data) {

            if (err) {
                return err;
            }

            var isBase64Encoded = false;
            var encoding = 'utf8';

            if (data.ContentType.indexOf('image/') > -1) {
                isBase64Encoded = true;
                encoding = 'base64'
            }

            var resp = {
                statusCode: 200,
                headers: {
                    'Content-Type': data.ContentType,
                },
                body: new Buffer(data.Body).toString(encoding),
                isBase64Encoded: isBase64Encoded
            };

            callback(null, resp);
        }
    );
};
```

更多模式

- 查看亚马逊 CloudFront 分配的访问日志、HTTPS 和 TLS 版本
- 在 Amazon EKS 集群上部署基于 gRPC 的应用程序并使用应用程序负载均衡器访问它
- 为公有子网部署基于属性的预防性访问控制
- 使用 Terraform 在 AWS Wavelength 区域中部署资源
- 使用 Terraform 为 AWS WAF 解决方案部署安全自动化
- 为基于蜂窝的架构设置无服务器蜂窝路由器
- 使用 Amazon Q Developer 作为编码助手来提高工作效率
- 使用 Splunk 查看 AWS Network Firewall 日志和指标

数据库和存储

主题

- 数据库
- 存储和备份

数据库

主题

- 使用链接服务器从亚马逊 EC2 上的微软 SQL Server 访问本地微软 SQL Server 表
- 使用只读副本 PeopleSoft 在 Amazon RDS Custom 上将 HA 添加到 Oracle
- 评测将 SQL Server 数据库迁移至 MongoDB Atlas on AWS 的查询性能
- 使用 IaC 原则自动部署 Amazon Aurora 全球数据库
- 使用 AWS Lambda 任务计划程序在亚马逊 EC2 上运行的 SQL Server Express 版中自动执行数据库任务
- 使用 DR Orchestrator 框架自动执行跨区域故障转移和故障恢复
- 在跨区域自动复制 Amazon RDS 实例 AWS 账户
- 使用 Systems Manager 自动备份 SAP HANA 数据库和 EventBridge
- 使用 Python 应用程序为亚马逊 DynamoDB 自动生成 PynamoDB 模型和 CRUD 函数
- 通过使用 Cloud Custodian 来阻止对 Amazon RDS 的公有访问
- 配置对 Amazon DynamoDB 的跨账户访问
- 在 AWS 上的 SQL Server 的“始终打开”可用性组中配置只读路由
- 在 pgAdmin 中使用 SSH 隧道进行连接
- 将 JSON Oracle 查询转换至 PostgreSQL 数据库 SQL
- 使用跨账户复制 Amazon DynamoDB 表 AWS Backup
- 使用自定义实施，跨账户复制 Amazon DynamoDB 表
- 为 Amazon RDS 和 Amazon Aurora 创建详细的成本和使用情况报告
- 使用 Aurora PostgreSQL 中的自定义端点模拟 Oracle RAC 工作负载
- 在 Amazon RDS 中为 PostgreSQL 数据库实例启用加密连接
- 加密现有 Amazon RDS for PostgreSQL 数据库实例
- 在启动时强制对 Amazon RDS 数据库执行自动标记
- 估算按需容量的 DynamoDB 表成本
- 估算 Amazon DynamoDB 表的存储成本
- 使用 AWR 报告估计 Oracle 数据库的 Amazon RDS 引擎大小
- 使用 AWS DMS 将 Amazon RDS for SQL Server 表导出至 S3 存储桶
- 在 Aurora PostgreSQL 中处理动态 SQL 语句中的匿名块
- 在 Aurora PostgreSQL 兼容中处理重载的 Oracle 函数

- 帮助强制执行 DynamoDB 标签
- 通过 AWS DMS 和 Amazon Aurora 实施跨区域灾难恢复
- 将含有 100 多个参数的 Oracle 函数和过程迁移到 PostgreSQL
- 将 Amazon RDS for Oracle 数据库实例迁移到使用 AMS 的其他账户
- 将 Oracle OUT 绑定变量迁移到 PostgreSQL 数据库
- 使用具有相同主机名的 SAP HSR 将 SAP HANA 迁移至 AWS
- 使用分布式可用性组将 SQL Server 迁移至 AWS
- 使用和 AWS DMS 从 Oracle 8i 或 9i 迁移到适用于 Oracle 的 Amazon RD SharePlex S
- 监控 Amazon Aurora 以查找未加密的实例
- 使用亚马逊监控 Oracle GoldenGate 日志 CloudWatch
- 从 Oracle Database Enterprise Edition 更换平台到 Amazon RDS for Oracle 上的 Standard Edition 2。
- 使用 Precission Connect 将大型机数据库复制到 AWS
- 使用 Lambda 和 Secrets Manager 计划适用于 Amazon RDS for PostgreSQL 和 Aurora PostgreSQL 的任务
- 使用本地 SMTP 服务器和数据库邮件发送 Amazon RDS for SQL Server 数据库实例通知
- 在 IBM Db2 on AWS 上为 SAP 设置灾难恢复
- 使用 Terraform 为数据库迁移设置 CI/CD 管道
- 使用有效备用数据库为 Amazon RDS Custom 上的 Oracle 电子商务套件设置 HA/DR 架构
- EC2 使用 GTID 在亚马逊 RDS for MySQL 和亚马逊上的 MySQL 之间设置数据复制
- 在 Amazon RDS 上为 Oracle PeopleSoft 应用程序过渡角色适用于 Oracle 定制
- 跨账户将数据从 Amazon Redshift 集群卸载到亚马逊 S3
- 按照工作负载分类的数据库迁移模式
- 更多模式

使用链接服务器从亚马逊 EC2 上的微软 SQL Server 访问本地微软 SQL Server 表

由 Tirumala Dasari (AWS) 和 Eduardo Valentim (AWS) 创建

摘要

此模式描述了如何使用链接服务器从运行或托管在亚马逊弹性计算云 (亚马逊 EC2) Windows 或 Linux 实例上的微软 SQL Server 数据库访问在微软 Windows 上运行的本地微软 SQL Server 数据库表。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 在亚马逊 EC2 Linux AMI（亚马逊机器映像）上运行微软 SQL Server 的亚马逊
- 本地微软 SQL Server (Windows) 服务器和 Windows 或 Linux EC2 实例之间的 AWS Direct Connect

产品版本

- SQL Server 2016 或更高版本

架构

源技术堆栈

- 在 Windows 上运行的本地 Microsoft SQL Server 数据库
- 亚马逊 EC2 的微软 SQL Server 在 Windows AMI 或 Linux AMI 上运行

目标技术堆栈

- 亚马逊在亚马逊 EC2 Linux AMI 上运行微软 SQL Server
- 亚马逊 EC2 搭载微软 SQL Server 在 Windows AMI 上运行

源数据库和目标数据库架构

工具

- Microsoft SQL Server Management Studio (SSMS) 是用于管理 SQL Server 基础结构的集成环境。它提供了用户界面和一组工具，其中包含与 SQL Server 交互的丰富脚本编辑器。

操作说明

在 Windows SQL Server 中将 SQL Server 的身份验证模式更改为 Windows

Task	描述	所需技能
通过 SSMS 连接到 Windows SQL Server。		数据库管理员
从 Windows SQL Server 实例的上下文（右键单击）菜单中将身份验证模式更改为 SQL Server 中的 Windows。		数据库管理员

重新启动 Windows MSSQL 服务

Task	描述	所需技能
重新启动 SQL 服务。	1. 在 SSMS 对象资源管理器中，选择 SQL Server 实例。 2. 打开上下文（右键单击）菜单。 3. 选择重新启动。	数据库管理员

在 Windows SQL Server 中创建新登录名并选择要访问的数据库

Task	描述	所需技能
在“安全”选项卡中，打开“登录”的上下文（右键单击）菜单，然后选择新的登录名。		数据库管理员
在“常规”选项卡中，选择“SQL Server 身份验证”，输入用户名，输入密码，然后确认密码并清除用于在下次登录时更改密码的选项。		数据库管理员
在“服务器角色”选项卡中，选择“公共”。		数据库管理员
在“用户映射”选项卡中，选择要访问的数据库和架构，然后突出显示该数据库以选择数据库角色。	选择“public”和“db_datareader”以访问数据库表中的数据。	数据库管理员
选择“确定”以创建用户。		数据库管理员

将 Windows SQL Server IP 添加到 Linux SQL Server 主机文件

Task	描述	所需技能
通过终端窗口连接到 Linux SQL Server 框。		数据库管理员
打开the /etc/hosts文件并使用 SQL Server 添加 Windows 计算机的 IP 地址。		数据库管理员
保存主机文件。		数据库管理员

在 Linux SQL Server 上创建链接服务器

Task	描述	所需技能
使用存储过程 master.sys.sp_addlinkedserver 和 master.dbo.sp_addlinkedsrvlogin 创建链接服务器。	有关使用这些存储过程的详细信息，请参阅其他信息部分。	数据库管理员、开发人员

验证在 SSMS 中创建的链接服务器和数据库

Task	描述	所需技能
在 SSMS 的 Linux SQL Server 中，转到“链接服务器”并刷新。		数据库管理员
在左窗格中展开创建的链接服务器和目录。	您将看到选定的 SQL Server 数据库以及表和视图。	数据库管理员

验证是否可以访问 Windows SQL Server 数据库表

Task	描述	所需技能
在 SSMS 查询窗口中，运行查询：“select top 3 * from [sqllin].dms_sample_win.dbo.mlb_data”。	请注意，FROM 子句使用由四部分组成的语法：computer.database.schema.table（例如，从 [sqllin] .master.sys.databases 中选择名称 “数据库”SQL2 ）。在我们的示例中，我们在主机文件 SQL2 中为创建了别名，因此您无需在方括号之间输入实际的 NetBIOS 名称。如果您使用实际的 NetBIOS 名称，请注意，AWS 默认使用 Win-xxxx 等 NetBIOS 名称，并且 SQL	数据库管理员、开发人员

Task	描述	所需技能
	Server 要求使用方括号作为带破折号的名称。	

相关资源

- Linux 上的 SQL Server 发行说明

其他信息

使用存储过程创建链接服务器

SSMS 不支持为 Linux SQL Server 创建链接服务器，因此必须使用以下存储过程来创建它们：

```
EXEC master.sys.sp_addlinkedserver @server= N'SQLLIN' , @srvproduct= N'SQL Server'
EXEC master.dbo.sp_addlinkedsrvlogin
 @rmtsrvname=N'SQLLIN',@useself=N'False',@locallogin=NULL,@rmtuser=N'username',@rmtpassword='Te
```

注 1：在存储过程 `master.dbo.sp_addlinkedsrvlogin` 中输入您之前在 Windows SQL Server 中创建的登录凭据。

注 2：`@server` 名称 `SQLLIN` 和主机文件条目名称 `172.12.12.4 SQLLIN` 应相同。

您可以使用此过程为以下方案创建链接服务器：

- Linux SQL Server 通过链接服务器到 Windows SQL Server（在此模式中指定）
- Windows SQL Server 通过链接服务器到 Linux SQL Server
- Linux SQL Server 通过链接服务器连接到另一个 Linux SQL Server

使用只读副本 PeopleSoft 在 Amazon RDS Custom 上将 HA 添加到 Oracle

创建者：sampath kathirvel (AWS)

摘要

要在亚马逊网络服务 (AWS) 上运行 Oracle PeopleSoft 企业资源规划 (ERP) 解决方案，您可以使用亚马逊关系数据库服务 (Amazon RDS) 或 Amazon RDS Custom for Oracle，后者支持需要访问底层操作系统和数据库环境的传统、自定义和打包应用程序。有关规划迁移时需要考虑的关键因素，请参阅 AWS Prescriptive Guidance 中的 Oracle 数据库迁移策略。

在撰写本文时，RDS Custom for Oracle 不支持多可用区选项，该选项可作为使用存储复制的Amazon RDS for Oracle 的 HA 解决方案使用。相反，此模式通过使用备用数据库来创建和维护主数据库的物理副本来实现 HA。该模式侧重于使用 Oracle Data Guard 设置只读副本在 Amazon RDS Custom 上运行带高可用性的 PeopleSoft 应用程序数据库的步骤。

此模式还会将只读副本更改为只读模式。将只读副本置于只读模式可带来其他好处：

- 从主数据库卸载只读工作负载
- 通过使用 Oracle Active Data Guard 功能从备用数据库中检索正常运行的块，从而自动修复损坏的块
- 使用 Far Sync 功能使远程备用数据库保持同步，而不会产生与长距离重做日志传输相关的性能开销。

在只读模式下使用副本需要 Oracle Active Data Guard 选项，这需要额外付费，因为它是 Oracle Database Enterprise Edition 单独许可的功能。

先决条件和限制

先决条件

- Amazon RDS 定制版上的现有 PeopleSoft 应用程序。如果您没有应用程序，请参阅 “将 Oracle 迁移 PeopleSoft 到 Amazon RDS Custom” 模式。
- 单个 PeopleSoft 应用程序层。但是，您可以调整此模式以使用多个应用程序层。
- Amazon RDS Custom 配置了至少 8 GB 的交换空间。
- Oracle Active Data Guard 数据库许可证，用于将只读副本转换为只读模式，并使用它来将报告任务卸载到备用数据库。有关详细信息，请参阅 Oracle 技术商业价目表。

限制

- RDS Custom for Oracle 的一般限制和不支持的配置
- 适用于 Oracle 的 Amazon RDS Custom 只读副本的相关限制

产品版本

- 有关 Amazon RDS Custom 支持的 Oracle 数据库版本，请参阅适用于 Oracle 的 RDS Custom。
- 有关 Amazon RDS Custom 支持的 Oracle 数据库实例类，请参阅适用于 Oracle 的 RDS Custom 支持的数据库实例类。

架构

目标技术堆栈

- 适用于 Oracle 的 Amazon RDS Custom
- AWS Secrets Manager
- Oracle Active Data Guard
- 甲骨文 PeopleSoft 应用程序

目标架构

下图显示了 Amazon RDS Custom 数据库实例和 Amazon RDS Custom 只读副本。只读副本使用 Oracle Active Data Guard 复制到另一个可用区。您还可以使用只读副本卸载主数据库上的读取流量并用于报告目的。

有关 PeopleSoft 在 AWS 上使用 Oracle 的代表性架构，请参阅在 AWS 上设置高可用 PeopleSoft 架构。

工具

Amazon Web Services

- 适用于 Oracle 的 Amazon RDS Custom 是一项托管式数据库服务，适用于需要访问底层操作系统和数据库环境的旧版、自定义和打包应用程序。
- AWS Secrets Manager 帮助您将代码中的硬编码凭证（包括密码）替换为对 Secrets Manager 的 API 调用，以便以编程方式检索密钥。在这种模式中，您可以从 Secrets Manager 中使用密钥名称

`do-not-delete-rds-custom-+<<RDS Resource ID>>+-dg` 检索 RDS_DATAGUARD 的数据库用户密码。

其他工具

- Oracle Data Guard 可帮助您创建、维护、管理和监控备用数据库。

最佳实践

若要实现零数据丢失（RPO=0）目标，请使用 `MaxAvailability` Data Guard 保护模式，并使用重做传输 `SYNC+NOAFFIRM` 设置以获得更好的性能。有关选择数据库保护模式的详细信息，请参阅其他信息部分。

操作说明

创建只读副本

Task	描述	所需技能
创建只读副本。	要创建 Amazon RDS Custom 数据库实例的只读副本，请按照 Amazon RDS 文档中的说明操作，并使用您创建的 Amazon RDS Custom 数据库实例（请参阅先决条件部分）作为源数据库。 默认情况下，Amazon RDS Custom 只读副本创建为物理备用副本，并处于已装载状态。这样做是为了确保遵守 Oracle Active Data Guard 许可。 此模式包含用于设置多租户容器数据库 (CDB) 或非 CDB 实例的代码。	数据库管理员

将 Oracle 数据卫士保护模式更改为 MaxAvailability

Task	描述	所需技能
访问主数据库上的 Data Guard 代理配置。	在此示例中，Amazon RDS Custom 只读副本对于非 CDB 实例为 `RDS_CUSTOM_ORCL_D` ，对于 CDB 实例为 `RDS_CUSTOM_RDSCDB_B` 。非 CDB 的数据库是 `orcl_a`（主数据库）和 `orcl_d`（备用数据库）。CDB 的数据库名称为 `rdscdb_a`（主数据库）和 `rdscdb_b`（备用数据库）。 您可以直接或通过主数据库连接到 RDS Custom 只读副本。您可以在位于 `$ORACLE_HOME/network/admin` 目录中的 `tnsnames.ora` 文件中找到数据库的网络服务名称。RDS Custom for Oracle 会自动为您的主数据库和只读副本填充这些条目。 `RDS_DATAGUARD` 用户的密码存储在 AWS Secrets Manager 中，密钥名称为 `do-not-delete-rds-custom-+<<RDS Resource ID>>+-dg`。有关如何使用从 Secrets Manager 检索到的 SSH（安全 Shell）密钥连接到 RDS 自定义实例的更多信息，请参阅使用 SSH 连接到 RDS 自定义数据库实例。	数据库管理员

Task	描述	所需技能

	要通过 Data Guard 命令行 (dgmgrl) 访问 Oracle Data Guard 代理配置，请使用以下代码。
	
	非 CDB
	
	`$ dgmgrl RDS_DATAGUARD@RDS_CUSTOM_ORCL_D`
	`DGMGRL for Linux: Release 19.0.0.0.0 - Production on Fri Sep 30 22:44:49 2022`
	`Version 19.10.0.0.0`
	`Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved.`
	`Welcome to DGMGRL, type "help" for information.`
	`Password:`
	`Connected to "ORCL_D"`
	`Connected as SYSDG.`
	`DGMGRL>`
	`DGMGRL> show database orcl_d`
	`Database - orcl_d`
	`Role: PHYSICAL STANDBY`
	`Intended State: APPLY-ON`
	`Transport Lag: 0 seconds (computed 0 seconds ago)`
	`Apply Lag: 0 seconds (computed 0 seconds ago)`
	`Average Apply Rate: 11.00 KByte/s`
	`Instance(s):`	

Task	描述	所需技能
	ORCL SUCCESS DGMGRL> CDB -bash-4.2$ dgmgrl C##RDS_DATAGUARD@R DS_CUSTOM_RDSCDB_B DGMGRL for Linux: Release 19.0.0.0.0 - Production on Wed Jan 11 20:24:11 2023 Version 19.16.0.0.0 Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved. Welcome to DGMGRL, type "help" for informati on. Password: Connected to "RDSCDB_B " Connected as SYSDG. DGMGRL> DGMGRL> show database rdscdb_b Database - rdscdb_b Role: PHYSICAL STANDBY Intended State: APPLY-ON Transport Lag: 0 seconds (computed 1 second ago) Apply Lag: 0 seconds (computed 1 second ago) Average Apply Rate: 2.00 KByte/s	

Task	描述	所需技能
	Real Time Query: OFF Instance(s): RDSCDB Database Status: SUCCESS DGMGRL>	

Task	描述	所需技能

通过从主节点连接到 DGMGRL 来更改日志传输设置。	将日志传输模式更改为 FastSync，对应于重做传输设置 SYNC+NOAFFIRM 。若要确保在角色切换后具有有效的设置，请同时更改主数据库和备用数据库的设置。
	
	非 CDB
	
	`DGMGRL>`
	`DGMGRL> edit database orcl_d set property logxptmode=fastsync;`
	`Property "logxptmode" updated`
	`DGMGRL> show database orcl_d LogXptMode;`
	`LogXptMode = 'fastsync'`
	`DGMGRL> edit database orcl_a set property logxptmode=fastsync;`
	`Property "logxptmode" updated`
	`DGMGRL> show database orcl_a logxptmode;`
	`LogXptMode = 'fastsync'`
	`DGMGRL>`
	
	CDB
	
	`DGMGRL> edit database rdscdb_b set property logxptmode=fastsync;DGMGRL> edit database rdscdb_b set property logxptmode=fastsync;`
		数据库管理员

Task	描述	所需技能
	Property "logxptmode" updated DGMGRL> show database rdscdb_b LogXptMode; LogXptMode = 'fastsync' DGMGRL> edit database rdscdb_a set property logxptmode=fastsync; Property "logxptmode" updated DGMGRL> show database rdscdb_a logxptmode; LogXptMode = 'fastsync' DGMGRL>	

Task	描述	所需技能

将保护模式更改为 MaxAvailability。	通过从主节点连接到 DGMGRL，将保护模式更改为 `MaxAvailability` 。
	
	非 CDB
	
	`DGMGRL> edit configuration set protection mode as maxavailability;`
	`Succeeded.`
	`DGMGRL> show configuration;`
	`Configuration - rds_dg`
	`Protection Mode: MaxAvailability`
	`Members:`
	`orcl_a - Primary database`
	`orcl_d - Physical standby database`
	`Fast-Start Failover: Disabled`
	`Configuration Status:`
	`SUCCESS (status updated 38 seconds ago)`
	`DGMGRL>`
	
	CDB
	
	`DGMGRL> show configuration`
	`Configuration - rds_dg`
	`Protection Mode: MaxAvailability`
	`Members:`
	`rdscdb_a - Primary database`
	`rdscdb_b - Physical standby database`
		数据库管理员

Task	描述	所需技能
	Fast-Start Failover: Disabled Configuration Status: SUCCESS (status updated 57 seconds ago) DGMGRL>	

将复制副本状态从附加更改为只读，并启用重做应用

Task	描述	所需技能
停止对备用数据库进行重做应用。	默认情况下，只读副本是在 MOUNT 模式下创建的。要以只读模式打开它，首先需要通过从主节点或备用节点连接到 DGMGRL 来关闭重做应用。 非 CDB DGMGRL> show database orcl_dDGMGRL> show database orcl_d Database - orcl_d Role: PHYSICAL STANDBY Intended State: APPLY- ON Transport Lag: 0 seconds (computed 1 second ago) Apply Lag: 0 seconds (computed 1 second ago) Average Apply Rate: 11.00 KByte/s Real Time Query: OFF Instance(s): ORCL Database Status: SUCCESS	数据库管理员

Task	描述	所需技能

	DGMGRL> edit database
	 orcl_d set state=app
	ly-off;
	Succeeded.
	DGMGRL> show database
	 orcl_d
	Database - orcl_d
	Role: PHYSICAL STANDBY
	Intended State: APPLY-
	OFF
	Transport Lag: 0
	 seconds (computed 1
	 second ago)
	Apply Lag: 42 seconds
	 (computed 1 second ago)
	Average Apply Rate:
	 (unknown)
	Real Time Query: OFF
	Instance(s):
	ORCL
	Database Status:
	SUCCESS
	DGMGRL>
	
	CDB
	
	DGMGRL> show configura
	tionDGMGRL> show
	 configuration
	Configuration - rds_dg
	  Protection Mode:
	 MaxAvailability
	  Members:
	  rdscdb_a - Primary
	 database
	    rdscdb_b - Physical
	 standby database
	Fast-Start Failover:
	 Disabled
	Configuration Status:	

Task	描述	所需技能
	SUCCESS (status updated 57 seconds ago) DGMGRL> show database rdscdb_b; Database - rdscdb_b Role: PHYSICAL STANDBY Intended State: APPLY-ON Transport Lag: 0 seconds (computed 1 second ago) Apply Lag: 0 seconds (computed 1 second ago) Average Apply Rate: 2.00 KByte/s Real Time Query: OFF Instance(s): RDSCDB Database Status: SUCCESS DGMGRL> edit database rdscdb_b set state=apply-off; Succeeded. DGMGRL> show database rdscdb_b; Database - rdscdb_b Role: PHYSICAL STANDBY Intended State: APPLY-OFF Transport Lag: 0 seconds (computed 1 second ago) Apply Lag: 0 seconds (computed 1 second ago) Average Apply Rate: (unknown)	

Task	描述	所需技能
	Real Time Query: OFF Instance(s): RDSCDB Database Status: SUCCESS	

Task	描述	所需技能

以只读模式打开只读副本实例。	使用 TNS 条目连接到备用数据库，然后通过从主节点或备用节点连接到备用数据库，以只读模式打开该数据库。
	
	非 CDB
	
	`$ sqlplus RDS_DATAGUARD@RDS_CUSTOM_ORCL_D as sysdg`
	`-bash-4.2$ sqlplus RDS_DATAGUARD@RDS_CUSTOM_ORCL_D as sysdg`
	`SQL*Plus: Release 19.0.0.0.0 - Production on Fri Sep 30 23:00:14 2022`
	`Version 19.10.0.0.0`
	`Copyright (c) 1982, 2020, Oracle. All rights reserved.`
	`Enter password:`
	`Last Successful login time: Fri Sep 30 2022 22:48:27 +00:00`
	`Connected to:`
	`Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production`
	`Version 19.10.0.0.0`
	`SQL> select open_mode from v$database;`
	`OPEN_MODE`
	`--------------------`
	`MOUNTED`
	`SQL> alter database open read only;`
	`Database altered.`
		数据库管理员

Task	描述	所需技能
	SQL> select open_mode from v$database; OPEN_MODE -------------------- READ ONLY SQL>	
	CDB -bash-4.2$ sqlplus C##RDS_DATAGUARD@R DS_CUSTOM_RDSCDB_B as sysdg SQL*Plus: Release 19.0.0.0.0 - Productio n on Wed Jan 11 21:14:07 2023 Version 19.16.0.0.0 Copyright (c) 1982, 2022, Oracle. All rights reserved. Enter password: Last Successful login time: Wed Jan 11 2023 21:12:05 +00:00 Connected to: Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production Version 19.16.0.0.0 SQL> select name,open _mode from v$database; NAME OPEN_MODE --------- --------- ---------- RDSCDB MOUNTED SQL> alter database open read only; Database altered.	

Task	描述	所需技能
	SQL> select name,open _mode from v$database; NAME OPEN_MODE --------- --------- ----------- RDSCDB READ ONLY SQL>	

Task	描述	所需技能

在只读副本实例上激活重做应用。	在主节点或备用节点上使用 DGMGR L 在只读副本实例上激活重做应用。
	
	非 CDB
	
	`$ dgmgrl RDS_DATAG UARD@RDS_CUSTOM_OR CL_D`
	`DGMGRL for Linux: Release 19.0.0.0.0 - Production on Fri Sep 30 23:02:16 2022`
	`Version 19.10.0.0.0`
	`Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved.`
	`Welcome to DGMGRL, type "help" for informati on.`
	`Password:`
	`Connected to "ORCL_D"`
	`Connected as SYSDG.`
	`DGMGRL>`
	`edit database orcl_d set state=apply-on;`
	`DGMGRL> edit database orcl_d set state=app ly-on;`
	`Succeeded.`
	`DGMGRL> show database orcl_d`
	`Database - orcl_d`
	`Role: PHYSICAL STANDBY`
	`Intended State: APPLY-ON`
	`Transport Lag: 0 seconds (computed 0 seconds ago)`
		数据库管理员

Task	描述	所需技能
	Apply Lag: 0 seconds (computed 0 seconds ago) Average Apply Rate: 496.00 KByte/s Real Time Query: ON Instance(s): ORCL Database Status: SUCCESS DGMGRL> CDB -bash-4.2$ dgmgrl C##RDS_DATAGUARD@RDS_CUSTOM_RDSCDB_B -bash-4.2$ dgmgrl C##RDS_DATAGUARD@RDS_CUSTOM_RDSCDB_B DGMGRL for Linux: Release 19.0.0.0.0 - Production on Wed Jan 11 21:21:11 2023 Version 19.16.0.0.0 Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved. Welcome to DGMGRL, type "help" for information. Password: Connected to "RDSCDB_B" Connected as SYSDG. DGMGRL> edit database rdscdb_b set state=apply-on; Succeeded.	

Task	描述	所需技能
	DGMGRL> show database rdscdb_b Database - rdscdb_b Role: PHYSICAL STANDBY Intended State: APPLY-ON Transport Lag: 0 seconds (computed 0 seconds ago) Apply Lag: 0 seconds (computed 0 seconds ago) Average Apply Rate: 35.00 KByte/s Real Time Query: ON Instance(s): RDSCDB Database Status: SUCCESS DGMGRL> show database rdscdb_b Database - rdscdb_b Role: PHYSICAL STANDBY Intended State: APPLY-ON Transport Lag: 0 seconds (computed 1 second ago) Apply Lag: 0 seconds (computed 1 second ago) Average Apply Rate: 16.00 KByte/s Real Time Query: ON Instance(s): RDSCDB Database Status: SUCCESS DGMGRL>	

相关的资源

- 将 Amazon RDS 配置为 Oracle PeopleSoft 数据库（AWS 白皮书）
- Oracle Data Guard 代理指南（Oracle 参考文档）
- Data Guard 概念和管理（Oracle 参考文档）

其他信息

选择数据库保护模式

Oracle Data Guard 提供三种保护模式，可根据您的可用性、保护和性能要求配置 Data Guard 环境。下表对这三种模式进行了汇总。

保护模式	重做传输设置	描述
最佳性能	ASYNC	对于主数据库上发生的事务，重做数据将异步传输并写入备用数据库重做日志。因此，对性能的影响微乎其微。 由于异步日志传送，MaxPerformance 无法提供 RPO=0。
最大程度的保护	SYNC+AFFIRM	对于主数据库上的事务，在确认事务之前，重做数据将同步传输并写入磁盘上的备用数据库重做日志。如果备用数据库不可用，则主数据库将自行关闭，以确保事务受到保护。
最大可用性	SYNC+AFFIRM	这类似于 MaxProtection 模式，除非未从备用数据库收到确认。在这种情况下，它会像在 MaxPerformance 模式下一样运行，以保持主数据库的可用性，直到它能够再次

	将其重做流写入同步备用数据库。
SYNC+NOAFFIRM	对于主数据库上的事务，重做将同步传输到备用数据库，并且主数据库仅等待确认已在备用数据库上收到重做，而不是等待重做已写入备用磁盘。此模式（也称为 FastSync）可以提供性能优势，但代价是在多个同时发生故障的特殊情况下可能会丢失数据。

RDS Custom for Oracle 中的只读副本是使用最高性能保护模式创建的，这也是 Oracle Data Guard 的默认保护模式。最高性能模式对主数据库的性能影响最小，这可以帮助您满足以秒为单位的恢复点目标（RPO）要求。

要努力实现零数据丢失（RPO=0）目标，可以将 Oracle Data Guard 保护模式自定义为 MaxAvailability，并将 SYNC+NOAFFIRM 作为重做传输的设置，以获得更好的性能。由于只有在将相应的重做向量成功传输到备用数据库后，才会确认主数据库上的提交，因此主实例和副本之间的网络延迟对于提交敏感型工作负载至关重要。建议对工作负载执行负载测试，以评测将只读副本自定义为在 MaxAvailability 模式下运行时对性能的影响。

与在不同的可用区中部署只读副本相比，将只读副本部署在与主数据库相同的可用区中可提供更低的网络延迟。但是，在同一可用区中部署主副本和只读副本可能无法满足您的 HA 要求，因为在极少数情况下，如果可用区不可用，主实例和只读副本实例都会受到影响。

评测将 SQL Server 数据库迁移至 MongoDB Atlas on AWS 的查询性能

由 Battulga Purevragchaa (AWS)、Krishnakumar Sathyanarayana（美国公司）和 Babu S PeerIslands rinivasan（MongoDB）创作

摘要

这种模式为使用近乎真实的数据加载 MongoDB 以及评测尽可能接近生产场景的 MongoDB 查询性能提供指导。该评测提供的信息可帮助您规划从关系数据库迁移至MongoDB 的操作。该模式使用PeerIslands 测试数据生成器和性能分析器来测试查询性能。

这种模式对于 Microsoft SQL Server 迁移至 MongoDB 特别有用，因为执行架构转换以及将数据从当前 SQL Server 实例加载到 MongoDB 可能非常复杂。相反，在开始实际迁移之前，您可以将近乎真实的数据加载至 MongoDB，了解 MongoDB 的性能并微调架构设计。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 熟悉 MongoDB Atlas
- 目标 MongoDB 架构
- 典型查询模式

限制

- 数据加载时间和性能受到 MongoDB 集群实例大小限制。我们建议您选择适用于生产的实例，以了解实际性能。
- PeerIslands 测试数据生成器和性能分析器目前仅支持在线数据加载和查询。尚不支持离线批处理（例如，使用 Spark 连接器将数据加载至 MongoDB）。
- PeerIslands 测试数据生成器和性能分析器支持集合中的字段关系。它不支持跨集合关系。

产品版本

- 此模式同时支持 MongoDB Atlas 和 MongoDB Enterprise Advanced。

架构

目标技术堆栈

- MongoDB Atlas 或 MongoDB Enterprise Advanced

架构

PeerIslands 测试数据生成器和性能分析器使用 Java 和 Angular 构建，并将其生成的数据存储在亚马逊 Elastic Block Store (Amazon EBS) Elastic Block Store 上。该工具包含两个工作流：测试数据生成与性能测试。

- 在测试数据生成中，您需要创建模板，该模板是必须生成的数据模型的 JSON 表示形式。创建模板后，您可按照负载生成配置的定义在目标集合中生成数据。
- 在性能测试中，您可创建配置文件。配置文件是一种多阶段测试方案，您可在其中配置创建、读取、更新和删除 (CRUD) 操作、聚合管道、每个操作的权重以及每个阶段的持续时间。创建配置文件后，您可根据配置对目标数据库运行性能测试。

PeerIslands 测试数据生成器和性能分析器将其数据存储在 Amazon EBS 上，因此您可以使用任何 MongoDB 支持的连接机制（包括对等互连、允许列表和私有终端节点）将 Amazon EBS 连接到 MongoDB。默认情况下，该工具不包含操作组件；但是，如果需要，可以将其配置为适用于 Prometheus 的亚马逊托管服务、亚马逊托管 Grafana、Amazon 和 AWS Secrets Manag CloudWatch er。

工具

- PeerIslands 测试数据生成器和性能分析器包括两个组件。测试数据生成器组件可帮您根据 MongoDB 架构生成高度针对客户的真实数据。该工具完全由用户界面驱动，包含丰富的数据库，可用于在 MongoDB 上快速生成数十亿条记录。该工具还提供在 MongoDB 架构中实现字段间关系的功能。Performance Analyzer 组件可帮助您生成高度针对客户的查询和聚合，并在 MongoDB 上执行真实的性能测试。您可以使用 Performance Analyzer 通过丰富的负载配置文件和针对特定用例的参数化查询来测试 MongoDB 性能。

最佳实践

请参阅以下资源：

- MongoDB 架构设计最佳实践（MongoDB 开发人员网站）
- 在 AWS 上部署 MongoDB Atlas 的最佳实践（Mongo DB 网站）
- 使用 AWS 将应用程序安全地连接到 MongoDB Atlas 数据平面 PrivateLink（AWS 博客文章）
- MongoDB 性能最佳实践指南（MongoDB 网站）

操作说明

了解您的源数据

Task	描述	所需技能
了解当前 SQL Server 源数据库占用空间。	了解当前的 SQL Server 占用空间。这可以通过对数据库 `INFORMATION` 架构运行查询来实现。确定表数量和每个表的大小。分析每个表关联的索引。有关 SQL 分析的更多信息，请参阅 PeerIslands 网站上的博客文章 SQL2Mongo：数据迁移之旅。	数据库管理员
了解源架构。	确定表架构和数据的业务表示形式（例如邮政编码、名称和货币）。使用现有的实体关系 (ER) 图或者从现有数据库生成 ER 图。有关更多信息，请参阅 PeerIslands 网站上的博客文章《SQL2Mongo：数据迁移之旅》。	数据库管理员
了解查询模式。	记录您使用的前 10 个 SQL 查询。你可以使用数据库中可用的 performance_schema.events_statements_summary_by_digest 表了解热门查询。有关更多信息，请参阅 PeerIslands 网站上的博客	数据库管理员

Task	描述	所需技能
	文章《SQL2Mongo：数据迁移之旅》。	
了解 SLA 承诺。	记录数据库操作的目标服务级别协议 (SLAs)。常见衡量标准包括查询延迟和每秒查询次数。这些措施及其目标通常可在非功能性需求 (NFR) 文档中查找。	数据库管理员

定义 MongoDB 架构

Task	描述	所需技能
定义目标架构。	为目标 MongoDB 架构定义多种选项。有关更多信息，请参阅 MongoDB Atlas 文档中的 Schemas。根据表格关系考虑最佳实践与设计模式。有关详细信息，请参阅 MongoDB 文档中的数据模型示例和模式。	MongoDB 工程师
定义目标查询模式。	定义 MongoDB 查询和聚合管道。这些查询等同于您为 SQL Server 工作负载所捕获的热门查询。要了解如何构造 MongoDB 聚合管道，请参阅 MongoDB 文档。	MongoDB 工程师
定义 MongoDB 实例类型。	确定您计划用于测试的实例大小。有关指南，请参阅 MongoDB 文档。	MongoDB 工程师

准备目标数据库

Task	描述	所需技能
设置 MongoDB Atlas 集群。	要在 AWS 上设置 MongoDB 集群，请按照 MongoDB 文档中的说明进行操作。	MongoDB 工程师
在目标数据库中创建用户。	按照 MongoDB 文档中的说明配置 MongoDB Atlas 集群以实现访问和网络安全。	MongoDB 工程师
在 AWS 中创建相应的角色，并为 Atlas 配置基于角色的访问控制。	如有需要，请按照 MongoDB 文档中的说明设置其他用户。通过 AWS 角色配置身份验证与授权。	MongoDB 工程师
设置 Compass for MongoDB Atlas 访问。	设置 MongoDB Compass GUI 实用程序以便于导航和访问。	MongoDB 工程师

使用测试数据生成器设置基本负荷

Task	描述	所需技能
安装测试数据生成器。	在您的环境中安装PeerIsland 测试数据生成器。	MongoDB 工程师
配置测试数据生成器，以生成相应的数据。	使用数据库为 MongoDB 架构中的所有字段生成特定数据，从而创建模板。有关更多信息，请参阅 MongoDB 数据生成器和性能。分析器视频。	MongoDB 工程师
水平扩展测试数据生成器，以生成所需负载。	使用您创建的模板，通过配置所需并行度开始针对目标集合生成负载。确定生成必要数据的时间范围与规模。	MongoDB 工程师

Task	描述	所需技能
验证 MongoDB Atlas 中的负载。	检查加载至 MongoDB Atlas 中的数据。	MongoDB 工程师
在 MongoDB 上生成所需索引。	根据查询模式定义所需索引 有关最佳实践，请参阅 MongoDB 文档。	MongoDB 工程师

执行性能测试

Task	描述	所需技能
在性能分析器设置负载配置文件。	通过配置特定的查询及其相应的权重、测试运行的持续时间和阶段，在 Performance Analyzer 中创建性能测试的配置文件。有关更多信息，请参阅 MongoDB 数据生成器和性能。分析器视频。	MongoDB 工程师
运行性能测试。	使用您创建的性能测试配置文件，通过配置所需并行度开始对目标集合进行测试。水平扩展性能测试工具，以对 MongoDB Atlas 运行查询。	MongoDB 工程师
记录测试结果。	记录查询 P95、P99 延迟。	MongoDB 工程师
调整架构与查询模式。	修改索引和查询模式，以解决任何性能问题。	MongoDB 工程师

关闭项目

Task	描述	所需技能
关闭临时 AWS 资源。	删除用于测试数据生成器与性能分析器的所有临时资源。	AWS 管理员
更新性能测试结果。	了解 MongoDB 的查询性能，并将其与您的查询性能进行比较。 SLAs如有必要，可微调 MongoDB 架构并重新运行该过程。	MongoDB 工程师
结束项目。	关闭项目并提供反馈。	MongoDB 工程师

相关资源

- GitHub 存储库：s3toAtlas
- 架构：MongoDB 架构设计
- 聚合管道：MongoDB 聚合管道
- MongoDB Atlas 大小调整：大小调整等级选择
- 视频：MongoDB 数据生成器和性能。分析器
- 参考文献：MongoDB 文档
- 教程：MongoDB 开发人员指南、MongoDB Jumpstart
- Amazon Web Services Marketplace：·Amazon Web Services Marketplace 上的 MongoDB Atlas
- AWS 合作伙伴解决方案： AWS Reference Deployment 中的 MongoDB Atlas

其他资源：

- SQL 分析
- MongoDB 开发人员社区论坛
- MongoDB 性能微调问题
- 使用 Atlas 和 Redshift 执行运营分析
- 使用 MongoDB Atlas 和 AWS Elastic Beanstalk 实施应用程序现代化

使用 IaC 原则自动部署 Amazon Aurora 全球数据库

由 Ishwar Chauthaiwale (AWS)、ANKIT JAIN (AWS) 和 Ramu Jagini (AWS) 创作

摘要

对于在 Amazon Aurora 全球数据库上运行关键工作负载的组织来说，管理数据库更新、迁移或扩展工作可能具有挑战性。确保这些操作在零停机时间内无缝执行，对于维护服务可用性和避免用户中断至关重要。

blue/green deployment strategy offers a solution to this challenge by allowing you to run two identical environments concurrently: blue (the current environment) and green (the new environment). A blue/green策略使您能够以最小的风险和停机时间实施更改、执行测试和在环境之间切换流量。

此模式可帮助您使用基础设施即代码 (IaC) 原则自动执行 Aurora 全球数据库的蓝/绿部署流程。它使用AWS CloudFormationAWS Lambda、和 Amazon Route 53 来简化蓝/绿部署。为了提高可靠性，它使用全局事务标识符 (GTIDs) 进行复制。与二进制日志 (binlog) 复制相比，基于 GTID 的复制在环境之间提供了更好的数据一致性和故障转移功能。

> **Note**
>
> 此模式假设您使用的是兼容 Aurora MySQL 的版本全局数据库集群。如果你改用与 Aurora PostgreSQL 兼容的命令，请使用与 MySQL 命令相当的 PostgreSQL 命令。

按照此模式中的步骤操作，您可以：

- 配置绿色 Aurora 全球数据库：使用 CloudFormation 模板可以创建反映现有蓝色环境的绿色环境。
- 设置基于 GTID 的复制：您可以配置 GTID 复制以保持蓝色和绿色环境的同步。
- 无缝切换流量：完全同步后，您可以使用 Route 53 和 Lambda 自动将流量从蓝色环境切换到绿色环境。
- 完成部署：验证绿色环境作为主数据库是否可以完全运行，然后停止复制并清理所有临时资源。

这种模式中的方法具有以下好处：

- 减少关键数据库更新或迁移期间的停机时间：自动化可确保环境之间的平稳过渡，同时最大限度地减少服务中断。

- 支持快速回滚：如果流量切换到绿色环境后出现问题，您可以快速恢复到蓝色环境并保持服务连续性。
- 增强测试和验证：可以在不影响实时蓝色环境的情况下对绿色环境进行全面测试，从而降低生产中出现错误的可能性。
- 确保数据一致性：基于 GTID 的复制可使蓝绿环境保持同步，从而防止迁移期间出现数据丢失或不一致的情况。
- 保持业务连续性：自动化蓝/绿部署有助于在更新或迁移期间保持服务可用性，从而避免长时间中断和经济损失。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- 与 Aurora MySQL 兼容的源全局数据库集群（蓝色环境）。全球数据库为高可用性和灾难恢复提供了多区域配置。有关设置全局数据库集群的说明，请参阅 Aurora 文档。
- 在源集群上启用了@@ 基于 GTID 的复制。

限制

- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额页面，然后选择服务的链接。

产品版本

- 兼容 Aurora MySQL 的 8.0 或更高版本

架构

该图阐释了以下内容：

- 全局数据库设置：Aurora 全球数据库集群策略性地部署在两个集群上 AWS 区域。此配置支持地理分布和区域冗余，以增强灾难恢复能力。

- 主区域到辅助区域复制：逻辑复制机制可确保从主区域到辅助区域的数据无缝同步。这种复制可以保持数据一致性，同时最大限度地减少跨地理距离的延迟。
- 集群之间基于 GTID 的复制：基于 GTID 的复制可保持蓝色主集群和绿色主集群之间的事务一致性和有序的数据流，并确保可靠的数据同步。
- 蓝色主群集到辅助复制：逻辑复制在蓝色主群集与其辅助群集之间建立了强大的数据管道。这种复制可实现连续的数据同步和高可用性。
- Route 53 DNS 配置：Route 53 托管区域记录管理所有蓝色和绿色集群数据库端点的 DNS 解析。此配置可提供无缝的端点映射，并在故障转移场景中实现高效的流量路由。

工具

Amazon Web Services

- Amazon Aurora 是与 MySQL 和 PostgreSQL 兼容的完全托管式的云端关系数据库引擎。
- AWS CloudFormation帮助您对 AWS 资源进行建模和设置，这样您就可以花更少的时间管理这些资源，而将更多的时间集中在运行的应用程序上 AWS。您可以创建一个描述所需所有 AWS 资源的模板，并 CloudFormation 负责为您配置和配置这些资源。
- AWS Lambda是一项计算服务，支持在不预置或管理服务器的情况下运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。
- Amazon Route 53 是一种可用性高、可扩展性强的 DNS Web 服务。

最佳实践

我们建议您仔细阅读 AWS 文档，以加深对 Route 53 中的蓝/绿部署策略、基于 GTID 的复制和加权路由策略的理解。这些知识对于有效实施和管理数据库迁移、确保数据一致性以及优化流量路由至关重要。通过全面了解这些 AWS 功能和最佳实践，您将能够更好地处理未来的更新，最大限度地减少停机时间并维护弹性和安全的数据库环境。

有关使用此模式 AWS 服务 的指南，请参阅以下 AWS 文档：

- 亚马逊 Aurora MySQL 的最佳实践
- AWS CloudFormation 最佳实践
- 使用 AWS Lambda 函数的最佳实践
- 亚马逊 Route 53 最佳实践

操作说明

创造绿色环境

Task	描述	所需技能
从蓝色集群创建快照备份。	在蓝/绿部署中，绿色环境代表当前（蓝色）数据库环境的全新、相同版本。在切换生产流量之前，您可以使用绿色环境来安全地测试更新、验证更改并确保稳定性。它充当实施数据库变更的舞台，同时最大限度地减少对实时环境的干扰。 要创建绿色环境，请先在兼容 Aurora MySQL 的全局数据库中创建主（蓝色）集群的快照。这张快照是创建绿色环境的基础。 要创建快照，请执行以下操作： 1. 登录 AWS Management Console 并打开亚马逊关系数据库服务 (Amazon RDS) 控制台。 2. 选择您的主（蓝色）集群。 3. 选择操作，拍摄快照。 4. 为快照提供名称（例如）`blue-green-demo`，然后启动备份过程。 或者，您可以使用 AWS Command Line Interface (AWS CLI) 来创建快照：	数据库管理员

Task	描述	所需技能
	`aws rds create-db-cluster-snapshot --db-cluster-snapshot-identifier blue-green-demo --db-cluster-identifier ex-global-cluster --region eu-west-1` 在继续下一步之前，请确保快照成功完成。	
为您的全局数据库及其资源生成 CloudFormation 模板。	CloudFormation IaC 生成器可帮助您利用现有 AWS 资源生成 CloudFormation 模板。使用此功能为现有 Aurora MySQL 兼容的全局数据库及其相关资源创建 CloudFormation 模板。此模板配置子网组、安全组、参数组和其他设置。 1. 按照 CloudFormation 文档中的说明导航到该工具并将其连接到您的 AWS 环境。 2. 选择您的 Aurora 全球数据库和相关资源以生成模板。	数据库管理员

Task	描述	所需技能
修改绿色环境的 CloudFormation 模板。	自定义 CloudFormation 模板以反映绿色环境的设置。这包括更新资源名称和标识符，以确保绿色环境独立于蓝色群集运行。 1. 更新`DBClusterIdentifier` 和`DBInstanceIdentifier` 属性以表示绿色环境。 2. 修改其他资源名称（例如子网组和安全组），以避免与现有的蓝色环境发生冲突。 3. 按照 Aurora 文档中所述，通过配置正确的参数在模板中启用基于 GTID 的复制。 4. 更改`SnapshotIdentifier` 属性以指定 AWS 区域、您的账户 ID 和上一步中的快照名称： `SnapshotIdentifier: arn:aws:rds:<region>:<account-id>:snapshot:<snapshot-name>` **Note** 如果您使用`SnapshotIdentifier` 属性恢复数据库集群，请避免	数据库管理员

Task	描述	所需技能
	指定诸如GlobalClusterIdentifier MasterUsername 、或之类的属性MasterUserPassword 。	

Task	描述	所需技能
部署 CloudFormation 堆栈为绿色环境创建资源。	在此步骤中，您将部署自定义 CloudFormation 模板来创建绿色环境的资源。 要部署 CloudFormation 堆栈，请执行以下操作： 1. 打开 AWS CloudFormation 管理控制台。 2. 在右上角，选择创建堆栈，使用新资源（标准）。 3. 上传修改后的 CloudFormation 模板或指定模板网址。选择下一步。 4. 输入堆栈名称（例如）`GreenClusterStack`，并提供任何必要的参数（例如，`GreenClusterIdentifier`）。选择下一步。 5. 根据需要配置其他堆栈选项，然后选中复选框以确认 CloudFormation 可能会创建 AWS Identity and Access Management (IAM) 资源。选择下一步。 6. 查看堆栈详细信息。 7. 选择提交。 CloudFormation 启动创建绿色环境资源的过程。此过程可能需要几分钟才能完成。	数据库管理员

Task	描述	所需技能
验证 CloudFormation 堆栈和资源。	CloudFormation 堆栈部署完成后，您需要验证绿色环境是否已成功创建： 1. 在 CloudFormation 堆栈的输出部分，检查数据库集群和数据库实例的终端节点以验证设置是否正确。 2. 打开 Amazon RDS 控制台并确认新的 Aurora 数据库集群（绿色环境）是否可用。 3. 确保所有关联资源（例如子网和安全组）均已创建并链接到绿色环境。 验证后，您的绿色环境已准备好进行进一步设置，包括从蓝色环境进行复制。	数据库管理员

配置基于 GTID 的复制

Task	描述	所需技能
验证蓝色集群上的 GTID 设置。	GTIDs 为在蓝色和绿色环境之间复制数据提供了一种高度可靠的方法。基于 GTID 的复制通过为蓝色环境中的每笔交易分配唯一标识符，提供了一种弹性、简化的方法。与传统的二进制日志复制相比，此方法可确保环境之间的数据同步是无缝的、一致的，并且更易于管理。	数据库管理员

Task	描述	所需技能
	在配置复制之前，您需要确保在蓝色集群上正确启用基于 GTID 的复制。此步骤可确保蓝色环境中的每笔交易都得到唯一跟踪，并且可以在绿色环境中复制。 要确认已启用 GTID，请执行以下操作： 1. 在 Amazon RDS 控制台上，查看分配给蓝色集群的参数组。 2. 验证是否设置了以下参数： • `gtid-mode = ON` • `enforce_gtid_consistency = ON` 这些设置允许在蓝色环境中对所有未来交易进行 GTID 跟踪。确认这些设置后，就可以开始设置复制了。	

Task	描述	所需技能
创建复制用户。	要将数据从蓝色环境复制到绿色环境，您需要在蓝色群集上创建一个专用的复制用户。该用户将负责管理复制过程。 要设置复制用户，请执行以下操作： 1. 使用 MySQL 客户端连接到蓝色集群。 2. 运行以下命令来创建复制用户： `CREATE USER 'repl_user'@'%' IDENTIFIED BY 'repl_password'; GRANT REPLICATION SLAVE ON . TO 'repl_user'@'%'; FLUSH PRIVILEGES;` 此用户现在具有在两个环境之间复制数据的必要权限。	数据库管理员

Task	描述	所需技能
在绿色集群上配置基于 GTID 的复制。	下一步是为基于 GTID 的复制配置绿色集群。此设置可确保绿色环境将持续镜像蓝色环境中发生的所有交易。 要配置绿色群集，请执行以下操作： 1. 使用 MySQL 客户端连接到绿色集群。 2. 运行以下命令来配置复制： `CHANGE MASTER TO MASTER_HOST='blue-cluster-endpoint', MASTER_USER='repl_user', MASTER_PASSWORD='repl_password', MASTER_AUTO_POSITION=1;` 其中： • `blue-cluster-endpoint` 替换为蓝色集群的终端节点。 • 该`MASTER_AUTO_POSITION=1` 设置指示 MySQL 使用基于 GTID 的复制。它会自动定位绿色集群以复制蓝色集群的交易，而无需手动跟踪日志和位置。	数据库管理员

Task	描述	所需技能
在绿色集群上开始复制。	现在，您可以开始复制过程了。在绿色集群上，运行以下命令： `START SLAVE;` 这使绿色环境能够开始同步数据，并从蓝色环境接收和应用事务。	数据库管理员
验证复制过程。	要验证绿色环境是否准确地复制了蓝色群集中的数据，请执行以下操作： 1. 在绿色集群上运行以下命令以检查复制状态： `SHOW SLAVE STATUS\G;` 2. 查看输出以验证以下内容： • `Slave_IO_Running = Yes` • `Slave_SQL_Running = Yes` • `Retrieved_Gtid_Set` 和的`Executed_Gtid_Set` 值 up-to-date 与蓝色群集同步。 • `Last_Error` 现场没有复制错误。 如果所有指标都正确，则基于 GTID 的复制运行平稳，绿色环境与蓝色环境完全同步。	数据库管理员

将流量从蓝色集群切换到绿色集群

Task	描述	所需技能
配置 Route 53 加权路由策略。	验证蓝色和绿色环境之间的数据一致性后，您可以将流量从蓝色集群切换到绿色集群。这种过渡应该是平稳的，应最大限度地减少停机时间，并确保应用程序数据库的完整性。为了满足这些要求，您可以使用 Route 53 进行 DNS 路由，使用 Lambda 自动切换流量。此外，定义明确的回滚计划可确保您在出现任何问题时可以恢复到蓝色集群。 第一步是在 Route 53 中配置加权路由。加权路由允许您控制蓝色和绿色集群之间的流量分布，并逐渐将流量从一个环境转移到另一个环境。 要配置加权路由： 1. 打开 Route 53 控制台并选择您的托管区域。 2. 为数据库创建两条 DNS 记录 (CNAMEs)：一条记录用于蓝色集群，一条记录用于绿色集群。 3. 分配初始权重： • 为绿色集群设置较低的初始权重（例如 5%），以发送一小部分流量进行测试。	AWS DevOps

Task	描述	所需技能
	• 为蓝色集群设置更高的权重（例如 95%），这样它就可以保留大部分流量。 此配置允许您在完全切换之前执行逐步过渡，从而降低风险并支持实时测试。 有关加权路由策略的更多信息，请参阅 R oute 53 文档。	

Task	描述	所需技能

部署 Lambda 函数以监控复制延迟。	为确保绿色环境与蓝色环境完全同步，请部署一个 Lambda 函数来监控集群之间的复制延迟。此函数可以检查复制状态，特别是 Seconds_Behind_Master 指标，以确定绿色集群是否已准备好处理所有流量。
	
	以下是您可以使用的示例 Lambda 函数：
	
	`import boto3`
	
	`def check_replication_lag(event, context):`
	`client = boto3.client('rds')`
	`response = client.describe_db_instances(DBInstanceIdentifier='green-cluster-instance')`
	`replication_status = response['DBInstances'][0]['ReadReplicaDBInstanceIdentifiers']`
	`if replication_status:`
	`lag = replication_status[0]['ReplicationLag']`
	`return lag`
	`return -1`
		AWS DevOps

Task	描述	所需技能
	此函数检查复制延迟并返回值。如果延迟为零，则绿色群集与蓝色群集完全同步。	

Task	描述	所需技能

使用 Lambda 自动调整 DNS 权重。	当复制延迟达到零时，是时候将所有流量切换到绿色集群了。您可以使用另一个 Lambda 函数自动执行此过渡，该函数可调整 Route 53 中的 DNS 权重，将 100% 的流量引导到绿色集群。
	
	以下是自动切换流量的 Lambda 函数的示例：
	
	`import boto3`
	
	`def switch_traffic(eve nt, context):`
	`route53 = boto3.cli ent('route53')`
	`lag = check_rep lication_lag(event, context)`
	`if lag == 0:`
	`response = route53.change_res ource_record_sets(`
	`HostedZon eId='YOUR_HOSTED_Z ONE_ID',`
	`ChangeBat ch={`
	`'Changes': [`
	`{`
	`'Action': 'UPSERT',`
	`'ResourceRecordSet': {`
	`'Name': 'db.examp le.com',
		AWS DevOps

Task	描述	所需技能

	<pre>'Type': 'CNAME',
	
	'SetIdentifier': 'GreenCluster',
	
	'Weight': 100,
	
	'TTL': 60,
	
	'ResourceRecords': [{'Value': 'green-cluster-endpoint'}]
	
	}
	},
	{
	
	'Action': 'UPSERT',
	
	'ResourceRecordSet': {
	
	'Name': 'db.example.com',
	
	'Type': 'CNAME',
	
	'SetIdentifier': 'BlueCluster',
	
	'Weight': 0,
	
	'TTL': 60,
	
	'ResourceRecords': [{'Value': 'blue-cluster-endpoint'}]
	
	}
	}
	]</pre>	

Task	描述	所需技能

	<pre> }
	        )
	        return response</pre>
	此功能检查复制延迟，并在延迟为零时更新 Route 53 DNS 权重，以将流量完全切换到绿色集群。
	
	Note
	在转换过程中，如果蓝色集群遇到大量写入流量，请考虑在切换期间暂停写入操作。这样可以确保复制赶上，并防止蓝群集和绿色群集之间的数据不一致。	

Task	描述	所需技能
验证流量开关。	Lambda 函数调整 DNS 权重后，您应验证所有流量是否都定向到绿色集群以及切换是否成功。 要验证： 1. 监控 Route 53 的 DNS 记录，确认流量正被定向到绿色集群。有关更多信息，请参阅 Route 53 文档。 2. 通过确认绿色环境为用户提供服务来检查应用程序性能。 3. 验证数据库连接以确认绿色集群正在处理所有数据库请求。 4. 监控 Amazon CloudWatch 指标，以了解任何延迟、复制延迟或性能下降的迹象。有关更多信息，请参阅 Aurora 文档。 如果一切按预期运行，则流量切换已完成。	AWS DevOps

Task	描述	所需技能
如果您遇到任何问题，请回滚更改。	制定回滚计划至关重要，以防流量切换后出现任何问题。以下是必要时快速恢复到蓝色集群的方法： 1. 恢复 Route 53 中的 DNS 权重：使用相同的 Lambda 函数或手动调整 Route 53 的 DNS 权重，将 100% 的流量引导回蓝色集群。 2. 监控应用程序性能：立即监控应用程序日志、CloudWatch 指标和数据库性能，以确认切换回蓝色环境已解决问题。 3. 识别并解决问题：在尝试再次切换流量之前，请调查并解决绿色集群的所有问题。 通过实施此回滚计划，您可以确保在出现任何意外问题时将对用户的干扰降至最低。	AWS DevOps

验证并停止基于 GTID 的复制

Task	描述	所需技能
在绿色集群上停止基于 GTID 的复制。	将流量从蓝色环境切换到绿色环境后，应验证过渡是否成功，并确保绿色集群按预期运行。此外，必须停止蓝色和绿色集群之间基于 GTID 的复制，因为绿色环境现在用作主数据库。完成这些任务可确保	数据库管理员

Task	描述	所需技能

	您的环境安全、简化并针对持续运营进行优化。
	
	要停止复制，请执行以下操作
	
	1. 使用 MySQL 客户端连接到绿色集群。
	2. 运行以下 SQL 命令以停止绿色集群上的复制过程：
	
	`STOP SLAVE;`
	
	3. （可选）如果需要，您可以重置复制配置以清除所有剩余的复制设置：
	
	`RESET SLAVE ALL;`
	
	停止复制后，绿色集群将完全独立，并作为工作负载的主数据库环境运行。	

Task	描述	所需技能
清理资源。	清理从蓝色群集迁移到绿色群集期间创建的任何临时或未使用的资源，可确保您的环境保持优化、安全且经济实惠。清理工作包括调整安全设置、进行最终备份以及停用不必要的资源。 要清理资源，请执行以下操作： 1. 更新安全组：配置与蓝色和绿色集群关联的安全组以反映新的主环境（绿色）。如果不再需要蓝色环境，请限制其访问权限，并验证绿色集群的安全设置是否遵循最佳实践。 2. 对绿色集群进行最终备份：迁移完成后，拍摄绿色集群的最终快照作为备份。如有必要，您可以使用此快照在将来恢复环境。 `aws rds create-db-snapshot --db-instance-identifier green-cluster-instance --db-snapshot-identifier green-cluster-final-snapshot` 3. 查看和移除临时资源：查看迁移期间创建的所有临时资源，例如临时安全组、快照或其他配置。删除不再需要	AWS DevOps

Task	描述	所需技能
	的资源，以避免不必要的开支。例如，如果不再需要蓝色群集，请将其删除： `aws rds delete-db-cluster --db-cluster-identifier blue-cluster-identifier --skip-final-snapshot` 清理资源有助于维护安全和简化的环境，降低成本，并确保只剩下必要的基础架构。	

相关资源

AWS CloudFormation:

- AWS CloudFormation 用户指南
- AWS CloudFormation 最佳实践
- 使用 iaC 生成器从现有资源生成模板
- 将@@ 整个应用程序导入 AWS CloudFormation（AWS 博客文章）

亚马逊 Aurora：

- Amazon Aurora 用户指南
- 管理亚马逊 Aurora 数据库集群

蓝/绿部署策略：

- 亚马逊 Aurora 蓝/绿部署概述

基于 GTID 的复制：

- 使用基于 GTID 的复制（亚马逊 RDS 文档）

AWS Lambda:

- AWS Lambda 开发人员指南
- 使用 AWS Lambda 函数的最佳实践

亚马逊 53 号公路：

- Amazon Route 53 开发人员指南
- 加权路由

MySQL 客户端工具：

- pyMyS

使用 AWS Lambda 任务计划程序在亚马逊 EC2 上运行的 SQL Server Express 版中自动执行数据库任务

由 Subhani Shaik (AWS) 编写

摘要

此模式演示了如何在 SQL Server Express 版（SQL Server 的免费版本）中计划和管理数据库任务。但是，SQL Server Express 版缺少通常处理自动数据库操作的 SQL Server 代理服务。此模式说明了在亚马逊弹性计算云 (Amazon) 实例上运行的 SQL Server Express 版中，如何使用任务计划程序和 Lambda 作为自动执行数据库任务的替代方案。 EC2

任务计划程序是一个内置的 Windows 系统实用程序，便于自动执行例行任务。它提供了一种安排和管理自动化操作的机制，无需在重复的流程中进行手动干预。 AWS Lambda是一种无服务器计算服务，可自动运行代码以响应事件，而无需您管理底层基础架构。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 使用亚马逊虚拟私有云 (Amazon VPC) 创建的虚拟私有云 (VPC)
- 装有 Windows 服务器的亚马逊 EC2 实例
- 通过 Windows Server 连接到亚马逊实例的亚马逊 Elastic Block Store (A EC2 mazon EBS) 卷
- SQL 服务器精简版二进制文件

限制

- 有关 SQL Server Express 版功能限制的信息，请访问微软网站。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅按地区划分的AWS 服务。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

产品版本

- 带有 SQL Server Express 版本的 SQL S

架构

下图显示了安装了 SQL Server Express 版的情况下运行的亚马逊 EC2 实例。该实例可通过远程桌面协议 (RDP) 客户端或从中 AWS Systems Manager Session Manager访问。 AWS Key Management Service (AWS KMS) 处理 Amazon EBS 卷的数据加密以确保 data-at-rest安全。基础设施还包括 AWS Identity and Access Management (IAM)，它提供访问控制和管理执行 Lambda 函数的权限。亚马逊简单存储服务 (Amazon S3) Service 存储 Lambda 函数。

工具

AWS 服务

- Amazon Elastic Block Store (Amazon EBS) 提供用于亚马逊实例的块级存储卷。 EC2
- 亚马逊弹性计算云 (Amazon EC2) 在中提供可扩展的计算容量 AWS Cloud。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥以帮助保护您的数据。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Storage Service (Amazon S3)是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Systems Manager Session Manager是一个完全托管的 AWS Systems Manager 工具。使用会话管理器，您可以管理您的 Amazon EC2 实例、边缘设备、本地服务器和虚拟机 (VMs)。
- Amazon Virtual Private Cloud（亚马逊 VPC）可帮助您将 AWS 资源启动到您定义的虚拟网络中。该虚拟网络类似于您在数据中心中运行的传统网络，并具有使用 AWS的可扩展基础设施的优势。

其他工具

- Microsoft SQL Server Management Studio (SSMS) 是一款用于管理 SQL Server 的工具，包括访问、配置和管理 SQL Server 组件。
- Python 是通用的计算机编程语言。您可以使用它在上构建应用程序、自动执行任务和开发服务AWS Cloud。
- 任务计划程序是一个 Microsoft 工具，你可以用它在电脑上自动安排例行任务。

最佳实践

- Amazon 的最佳实践 EC2
- 在亚马逊上部署微软 SQL Server 的最佳实践 EC2
- 使用 AWS Lambda 函数的最佳实践
- IAM 安全最佳实操

操作说明

创建亚马逊 EC2 实例并安装 SQL Server Express 版

Task	描述	所需技能
部署 Amazon EC2 实例。	要创建亚马逊 EC2 实例，请打开亚马逊 EC2 控制台，网址为，https://console.aws.amazon.com/ec2/然后从可用于 Windows 服务器的实例列表中选择一个亚马逊系统映像 (AMI)。 有关更多信息，请参阅 AWS 文档中的启动 Amazon EC2 实例。	DBA、AWS DevOps
安装 SQL Server 快捷版。	要安装 SQL Server Express 版，请完成以下步骤： 1. 要连接到您的 Amazon EC2 实例，请选择一个选项： • 选项 A-使用远程桌面协议 (RDP)。有关说明，请参阅 AWS 文档中的使用 RDP 客户端连接您的 Windows 实例。 • 选项B —使用Amazon EC2 控制台和 AWS	DBA、AWS DevOps

Task	描述	所需技能
	Systems Manager Session Manager. 有关说明，请参阅 AWS 文档中的使用会话管理器连接您的 Amazon EC2 实例。 2. 要下载所需的 SQL Server Express 版本，请访问微软网站上的 SQL Server 下载页面。 3. 要安装 SQL Server Express 版，请按照微软网站上规划 SQL Server 安装中的说明进行操作。	

创建自动数据库维护任务

Task	描述	所需技能
确定例行任务。	确定要自动执行的例行任务。例如，以下任务符合自动化的条件： • 数据库备份（完整、差异和事务日志） • 索引维护和重组 • 统计数据更新 • 特定于应用程序的操作 • 数据清理或存档	数据库管理员
准备 SQL 脚本。	要准备 SQL 脚本，请执行以下操作： 1. 为每项维护任务创建 SQL 查询。以下是用于执行特定	数据库管理员

Task	描述	所需技能
	数据库备份的 T-SQL 查询示例：`Backup Database <Database_Name> To Disk='C:\Backups\Database_Name.bak'` 2. 将脚本文件另存为`<File Name>.sql` 。然后，将脚本保存到 Amazon EC2 实例或网络文件共享上服务器本地驱动器上可访问的位置。	
配置访问权限。	要配置访问权限，请执行以下操作： 1. 设置适当的文件系统权限。有关说明，请参阅 Microsoft 网站上的“为数据库引擎访问权限配置文件系统权限”。 2. 检查 SQL Server 服务帐户是否具有必要的访问权限。有关说明，请参阅微软网站上的“配置 Windows 服务帐户和权限”。 3. 验证远程共享的网络连接。有关更多信息，请参阅 AWS 文档中的使用文件共享访问数据。	数据库管理员

使用任务计划程序自动执行任务

Task	描述	所需技能
创建批处理文件。	• 要创建批处理文件，请使用文本编辑器键入以下命	AWS DevOps、DBA

Task	描述	所需技能

	令。用您自己的值替换参数username和password。然后将文件另存为 <Name>.bat 。
	
	`sqlcmd -S servername -U username -P password -i <T-SQL query path.sql>`
	
	• 要为 SQL 任务创建批处理文件，请使用文本编辑器并键入以下命令。用您自己的值替换参数 ServerName DatabaseName username、、和password。然后将文件另存为<Name>.bat 。
	
	`@echo off`
	`sqlcmd -S [ServerName] -d [DatabaseName] -U username -P password -i "PathToSQLScript\Script.sql" -o "PathToOutput\Output.txt"`	

Task	描述	所需技能
在 “任务计划程序” 中创建任务。	要在任务计划程序中创建任务，请使用以下步骤： 1. 要打开 “任务计划程序”，请在 Windows 搜索中键入 taskschd.msc。 2. 选择 “操作” 菜单，然后选择 “创建基本任务”。 3. 在 “名称” 中，提供任务的名称，然后选择 “下一步”。 4. 在 “触发器” 中，选择任务启动时间的选项，然后选择 “下一步”。 5. 提供任务的 “开始” 和 “重复” 信息，然后选择 “下一步”。 6. 在 “操作” 部分，选择 “启动程序”，然后选择 “下一步”。 7. 对于 “程序/脚本”，指定您在上一个任务中创建的批处理文件的路径，然后选择 “下一步”。 8. 选择完成。 要手动运行任务，请右键单击新创建的任务，然后选择运行。	数据库管理员

Task	描述	所需技能
查看任务状态。	要在 “任务计划程序” 中查看任务的状态，请按以下步骤操作： 1. 在 “任务计划程序” 中，转到 “任务计划程序库”，其中显示了所有任务。 2. 要查看您之前创建的任务的状态，请选择该任务，然后转到 “历史记录” 选项卡。	DBA、AWS DevOps

使用自动执行任务 AWS Lambda

Task	描述	所需技能
实施解决方案。	要实现此模式的解决方案，请使用以下步骤： 1. 创建一个 Lambda 函数。有关说明，请参阅文档中的创建您的第一个 Lambda 函数。 AWS 2. 安排 Lambda 函数。有关说明，请参阅文档中的按计划调用 Lambda 函数。 AWS 3. 运行 T-SQL 查询。有关更多信息，请参阅文档中的教程：使用 Lambda 函数访问 Amazon RDS 数据库。 AWS 本教程介绍了如何通过 Lambda 函数连接 Amazon RDS 数据库以运行 SQL 查询	AWS DevOps， DevOps 工程师

故障排除

事务	解决方案
Lambda 问题	有关使用时可能遇到的错误和问题的帮助 AWS Lambda，请参阅文档中的 Lambda 问题疑难解答。 AWS

相关资源

- Amazon EC2 实例类型
- AWS Lambda 文档
- AWS Lambda 定价
- 开发人员任务计划程序（微软网站）

使用 DR Orchestrator 框架自动执行跨区域故障转移和故障恢复

由 Jitendra Kumar (AWS)、Oliver Francis (AWS) 和 Pavithra Balasubramanian (AWS) 创作

摘要

此模式描述了如何使用 DR Orchestrator Fram ework 来编排和自动执行容易出错的手动步骤，以便在 Amazon Web Services () 区域中执行灾难恢复。AWS该模式涵盖以下数据库：

- 适用于 MySQL 的亚马逊关系数据库服务（亚马逊 RDS）、适用于 PostgreSQL 的亚马逊 RDS 或适用于 MariaDB 的 Amazon RDS
- 亚马逊 Aurora MySQL 兼容版或亚马逊 Aurora PostgreSQL 兼容版（使用集中式文件）
- 亚马逊 ElastiCache （Redis OSS）

为了演示 DR Orchestrator Framework 的功能，您需要创建两个数据库实例或集群。主节点在中 AWS 区域 `us-east-1`，次要在中`us-west-2`。要创建这些资源，您可以使用 aws-cross-region-dr-databases GitHub 存储库`App-Stack`文件夹中的 AWS CloudFormation 模板。

先决条件和限制

一般先决条件

- 灾难恢复 Orchestrator 框架同时部署在主服务器和辅助设备中 AWS 区域
- 两个 Amazon 简单存储服务存储桶
- 具有两个子网和一个 AWS 安全组的虚拟私有云 (VPC)

特定于引擎的先决条件

- 亚马逊 Aurora — Aur ora 全球数据库中必须至少有一个 Aurora 全球数据库在两个数据库中可用 AWS 区域。您可以`us-east-1`用作主要区域，也可以`us-west-2`用作辅助区域。
- 亚马逊 ElastiCache （Redis OSS）— ElastiCache 全球数据存储必须一分为二。 AWS 区域您可以`use us-east-1`作为主要区域，也可以`us-west-2`用作辅助区域。

亚马逊 RDS 限制

- DR Orchestrator Framework 在进行故障转移或故障恢复之前不会检查复制延迟。必须手动检查复制延迟。

- 此解决方案已使用带有一个只读副本的主数据库实例进行了测试。如果要使用多个只读副本，请在生产环境中实施解决方案之前对其进行全面测试。

Aurora 的限制

- 功能可用性和支持因每个数据库引擎的特定版本和不同版本而异 AWS 区域。有关跨区域复制功能和区域可用性的更多信息，请参阅跨区域只读副本。
- Aurora 全局数据库对支持的 Aurora 数据库实例类和最大数量有特定的配置要求 AWS 区域。有关更多信息，请参阅 Amazon Aurora 全球数据库的配置要求。
- 此解决方案已使用带有一个只读副本的主数据库实例进行了测试。如果要使用多个只读副本，请在生产环境中实施解决方案之前对其进行全面测试。

ElastiCache 局限性

- 有关全球数据存储的区域可用性和 ElastiCache 配置要求的信息，请参阅 ElastiCache 文档中的先决条件和限制。

Amazon RDS p 产品版本

Amazon RDS 支持以下引擎版本：

- MySQL — Amazon RDS 支持运行以下版本的 MySQL 的数据库实例：MyS QL 8.0 和 MySQL 5.7
- Postgre@@ SQL — 有关适用于 PostgreSQL 的亚马逊 RDS 支持的版本的信息，请参阅可用的 PostgreSQL 数据库版本。
- MariaDB — 亚马逊 RDS 支持运行以下版本的 MariaDB 的数据库实例：
  - MariaDB 10.11
  - MariaDB 10.6
  - MariaDB 10.5

Aurora 产品版本

- Amazon Aurora 全球数据库切换需要兼容 Aurora MySQL，兼容 MySQL 5.7，版本 2.09.1 及更高版本

  有关更多信息，请参阅 Amazon Aurora 全球数据库的限制。

ElastiCache (Redis OSS) 产品版本

亚马逊 ElastiCache （Redis OSS）支持以下 Redis 版本：

- Redis 7.1（加强版）
- Redis 7.0（加强版）
- Redis 6.2（加强版）
- Redis 6.0（加强版）
- Redis 5.0.6（加强版）

有关更多信息，请参阅支持的 ElastiCache (Redis OSS) 版本。

架构

亚马逊 RDS 架构

Amazon RDS 架构包括以下资源：

- 在主区域 (us-east-1) 中创建的主要 Amazon RDS 数据库实例，客户端具有读/写访问权限
- 在次要区域 (us-west-2) 中创建的 Amazon RDS 只读副本，客户端具有只读访问权限
- DR Orchestrator 框架部署在主区域和次要区域

此图显示以下内容：

1. 主实例和辅助实例之间的异步复制
2. 主区域内客户端的读/写访问权限
3. 次要区域中客户端的只读访问权限

Aurora 架构

亚马逊 Aurora 架构包括以下资源：

- 在主区域 (us-east-1) 中创建的带有主动写入器终端节点的主要 Aurora 数据库集群
- 在辅助区域 (us-west-2) 中创建的 Aurora 数据库集群，其终端节点处于非活动状态
- DR Orchestrator 框架部署在主区域和次要区域

此图显示以下内容：

1. 主群集和辅助群集之间的异步复制
2. 带有主动写入器终端节点的主数据库集群
3. 带有非活动写入器终端节点的辅助数据库集群

ElastiCache (Redis OSS) 架构

亚马逊 ElastiCache （Redis OSS）架构包括以下资源：

- 由两个集群创建的 ElastiCache （Redis OSS）全球数据存储：
  1. 主区域中的主集群 (`us-east-1`)
  2. 辅助区域中的辅助群集 (`us-west-2`)
- 两个集群之间使用 TLS 1.2 加密的 Amazon 跨区域链接
- DR Orchestrator 框架已部署在主区域和次要区域

自动化和扩缩

DR Orchestrator Framework 具有可扩展性，并行支持多个 AWS 数据库的故障转移或故障恢复。

您可以使用以下负载代码对账户中的多个 AWS 数据库进行故障切换。在此示例中，三个 AWS 数据库（两个全球数据库，例如兼容 Aurora MySQL 或兼容 Aurora PostgreSQL 的数据库，以及一个 Amazon RDS for MySQL 实例）故障转移到灾难恢复区域：

```
{
  "StatePayload": [
    {
      "layer": 1,
      "resources": [
        {
          "resourceType": "PlannedFailoverAurora",
          "resourceName": "Switchover (planned failover) of Amazon Aurora global
 databases (MySQL)",
          "parameters": {
            "GlobalClusterIdentifier": "!Import dr-globaldb-cluster-mysql-global-
identifier",
```

```
            "DBClusterIdentifier": "!Import dr-globaldb-cluster-mysql-cluster-
identifier"
          }
        },
        {
          "resourceType": "PlannedFailoverAurora",
          "resourceName": "Switchover (planned failover) of Amazon Aurora global
 databases (PostgreSQL)",
          "parameters": {
            "GlobalClusterIdentifier": "!Import dr-globaldb-cluster-postgres-global-
identifier",
            "DBClusterIdentifier": "!Import dr-globaldb-cluster-postgres-cluster-
identifier"
          }
        },
        {
          "resourceType": "PromoteRDSReadReplica",
          "resourceName": "Promote RDS for MySQL Read Replica",
          "parameters": {
            "RDSInstanceIdentifier": "!Import rds-mysql-instance-identifier",
            "TargetClusterIdentifier": "!Import rds-mysql-instance-global-arn"
          }
        }
      ]
    }
  ]
}
```

工具

AWS 服务

- Amazon Aurora 是与 MySQL 和 PostgreSQL 兼容的完全托管式的云端关系数据库引擎。
- Amazon ElastiCache 可帮助您在中设置、管理和扩展分布式内存缓存环境。 AWS Cloud这种模式使用亚马逊 ElastiCache （Redis OSS）。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。在这种模式中，使用 Lambda 函数 AWS Step Functions 来执行这些步骤。
- Amazon Relational Database Service（Amazon RDS）可帮助您在中设置、操作和扩展关系数据库 AWS Cloud。此模式支持适用于 MySQL 的亚马逊 RDS、适用于 PostgreSQL 的亚马逊 RDS 和适用于 MariaDB 的亚马逊 RDS。

- 适用于 Python (Boto3) 的 AWS SDK帮助您将 Python 应用程序、库或脚本与集成 AWS 服务。在这种模式中，Boto3 APIs 用于与数据库实例或全局数据库通信。
- AWS Step Functions是一项无服务器编排服务，可帮助您组合 AWS Lambda 功能和其他功能 AWS 服务 来构建关键业务应用程序。在这种模式中，Step Functions 状态机用于协调和运行数据库实例或全局数据库的跨区域故障转移和故障恢复。

代码存储库

此模式的代码可在上的 aws-cross-region-dr GitHub-databases 存储库中找到。

操作说明

安装 DR Orchestrator 框架

Task	描述	所需技能
克隆 GitHub 存储库。	要克隆存储库，请运行以下命令： `git clone https://github.com/aws-samples/aws-cross-region-dr-databases.git`	AWS DevOps，AWS 管理员

将 Lambda 函数代码打包到.zip 文件存档中。	为 Lambda 函数创建存档文件以包含 DR Orchestrator 框架依赖项：
	
	`cd <YOUR-LOCAL-GIT-FOLDER>/DR-Orchestration-artifacts`
	
	`bash scripts/deploy-orchestrator-sh.sh`
		AWS 管理员

创建 S3 存储桶。	需要使用 S3 存储桶来存储 DR Orchestrator Framework 以及您的最新配置。创建两个 S3 存储桶，一个在主区域 (us-	AWS 管理员

Task	描述	所需技能
	east-1 ），一个在辅助区域 (us-west-2 ）： • dr-orchestrator-xxxxxx-us-east-1 • dr-orchestrator-xxxxxx-us-west-2 替换为xxxxxx随机值以使存储桶名称独一无二。	
创建子网和安全组。	在主区域 (us-east-1 ）和次要区域 (us-west-2 ）中，为在您的 VPC 中部署 Lambda 函数创建两个子网和一个安全组： • subnet-XXXXXXX • subnet-YYYYYYY • sg-XXXXXXXXXXX	AWS 管理员

Task	描述	所需技能
更新 DR Orchestrator 参数文件。	在该<YOUR-LOCAL-GIT-FOLDER>/DR-Orchestration-artifacts/cloudformation 文件夹中，更新以下 DR Orchestrator 参数文件： • Orchestrator-Deployer-parameters-us-east-1.json • Orchestrator-Deployer-parameters-us-west-2.json 使用以下参数值，用资源名称替换x和y： [{ "ParameterKey": "TemplateStoreS3BucketName", "ParameterValue": "dr-orchestrator-xxxxxx-us-east-1" }, { "ParameterKey": "TemplateVPCId", "ParameterValue": "vpc-xxxxxx" }, { "ParameterKey": "TemplateLambdaSubnetID1",	AWS 管理员

Task	描述	所需技能

	"Paramete
	rValue": "subnet-x
	xxxxx"
	},
	{
	"ParameterKey":
	"TemplateLambdaSub
	netID2",
	"Paramete
	rValue": "subnet-y
	yyyyy"
	},
	{
	"ParameterKey":
	"TemplateLambdaSec
	urityGroupID",
	"Paramete
	rValue": "sg-xxxxx
	xxxxx"
	}
	]	

Task	描述	所需技能
将 DR Orchestrator 框架代码上传到 S3 存储桶。	S3 存储桶中的代码比本地目录中的代码更安全。将`DR-Orchestration-artifacts` 目录（包括所有文件和子文件夹）上传到 S3 存储桶。 要上传代码，请执行以下操作： 1. 登录到 AWS Management Console。 2. 导航到 Amazon S3 控制台。 3. 选择`dr-orchestrator-xxxxxx-us-east-1` bucket。 4. 选择 “上传”，然后选择 “添加文件夹”。 5. 选择文件`DR-Orchestration-artifacts` 夹。 6. 选择上传。 7. 选择`dr-orchestrator-xxxxxx-us-west-2` 存储桶。 8. 重复步骤 4—7。	AWS 管理员

Task	描述	所需技能

在主区域部署 DR Orchestrator 框架。	要在主区域 (`us-east-1`) 中部署 DR Orchestrator 框架，请运行以下命令：
	
	`cd <YOUR-LOCAL-GIT-FOLDER>/DR-Orchestration-artifacts/cloudformation`
	
	`aws cloudformation deploy \`
	`--region us-east-1 \`
	`--stack-name dr-orchestrator \`
	`--template-file Orchestrator-Deployer.yaml \`
	`--parameter-overrides file://Orchestrator-Deployer-parameters-us-east-1.json \`
	`--capabilities CAPABILITY_AUTO_EXPAND CAPABILITY_NAMED_IAM CAPABILITY_IAM \`
	`--disable-rollback`
		AWS 管理员

Task	描述	所需技能

在辅助区域部署 DR Orchestrator 框架。	在辅助区域 (`us-west-2`) 中，运行以下命令：
	
	`cd <YOUR-LOCAL-GIT-FOLDER>/DR-Orchestration-artifacts/cloudformation`
	
	`aws cloudformation deploy \`
	`--region us-west-2 \`
	`--stack-name dr-orchestrator \`
	`--template-file Orchestrator-Deployer.yaml \`
	`--parameter-overrides file://Orchestrator-Deployer-parameters-us-west-2.json \`
	`--capabilities CAPABILITY_AUTO_EXPAND CAPABILITY_NAMED_IAM CAPABILITY_IAM \`
	`--disable-rollback`
		AWS 管理员

Task	描述	所需技能
验证部署。	如果 AWS CloudFormation 命令成功运行，它将返回以下输出： `Successfully created/ updated stack - dr-orchestrator` 或者，您可以导航到 AWS CloudFormation 控制台并验证`dr-orchestrator` 堆栈的状态。	AWS 管理员

创建数据库实例或集群

Task	描述	所需技能
创建数据库子网和安全组。	在您的 VPC 中，在主 (`us-east-1`) 和辅助 (`us-west-2`) 区域中为数据库实例或全局数据库创建两个子网和一个安全组： • `subnet-XXXXXX` • `subnet-XXXXXX` • `sg-XXXXXXXXXX`	AWS 管理员
更新主数据库实例或集群的参数文件。	在`<YOUR LOCAL GIT FOLDER>/App-Stack` 文件夹中，更新主区域的参数文件。 Amazon RDS	AWS 管理员

Task	描述	所需技能
	在RDS-MySQL-parameter-us-east-1.json 文件中SubnetIds ，DBSecurityGroup 使用您创建的资源的名称更新和： `{ "Parameters": { "SubnetIds": "subnet-xxxxxx,subnet-xxxxxx", "DBSecurityGroup": "sg-xxxxxxxxxx", "MySqlGlobalIdentifier":"rds-mysql-instance", "InitialDatabaseName": "mysqldb", "DBPortNumber": "3789", "PrimaryRegion": "us-east-1", "SecondaryRegion": "us-west-2", "KMSKeyAliasName": "rds/rds-mysql-instance-KmsKeyId" } }`  Amazon Aurora  在Aurora-MySQL-parameter-us-east-1.json 文件中SubnetIds ，DBSecurityGroup 使用您创建的资源的名称更新和：  `{`	

Task	描述	所需技能
	"Parameters": { "SubnetIds": "subnet1-xxxxxx,subnet2-xxxxxx", "DBSecurityGroup": "sg-xxxxxxxxxx", "GlobalClusterIdentifier":"dr-globaldb-cluster-mysql", "DBClusterName":"dbcluster-01", "SourceDBClusterName":"dbcluster-02", "DBPortNumber": "3787", "DBInstanceClass": "db.r5.large", "InitialDatabaseName": "sampledb", "PrimaryRegion": "us-east-1", "SecondaryRegion": "us-west-2", "KMSKeyAliasName": "rds/dr-globaldb-cluster-mysql-KmsKeyId" } } 亚马逊 ElastiCache（Redis OSS） 在`ElastiCache-parameter-us-east-1.json` 文件中`SubnetIds` ，`DBSecurityGroup` 使用您创建的资源的名称更新和。 {	

Task	描述	所需技能
	"Parameters": { "CacheNodeType": "cache.m5.large", "DBSecurityGroup": "sg-xxxxxxxxxx", "SubnetIds": "subnet-xxxxxx,subnet-xxxxxx", "EngineVersion": "5.0.6", "GlobalReplicationGroupIdSuffix": "demo-redis-global-datastore", "NumReplicas": "1", "NumShards": "1", "ReplicationGroupId": "demo-redis-cluster", "DBPortNumber": "3788", "TransitEncryption": "true", "KMSKeyAliasName": "elasticache/demo-redis-global-datastore-KmsKeyId", "PrimaryRegion": "us-east-1", "SecondaryRegion": "us-west-2" } }	

Task	描述	所需技能

在主区域部署您的数据库实例或集群。	要在主区域 (us-east-1) 中部署您的实例或集群，请根据您的数据库引擎运行以下命令。
	
	Amazon RDS
	
	`cd <YOUR-LOCAL-GIT-FOLDER>/App-Stack`
	
	`aws cloudformation deploy \`
	`--region us-east-1 \`
	`--stack-name rds-mysql-app-stack \`
	`--template-file RDS-MySQL-Primary.yaml \`
	`--parameter-overrides file://RDS-MySQL-parameter-us-east-1.json \`
	`--capabilities CAPABILITY_AUTO_EXPAND CAPABILITY_NAMED_IAM CAPABILITY_IAM \`
	`--disable-rollback`
	
	Amazon Aurora
	
	`cd <YOUR-LOCAL-GIT-FOLDER>/App-Stack`
	
	`aws cloudformation deploy \`
	`--region us-east-1 \`
	`--stack-name aurora-mysql-app-stack \`
	`--template-file Aurora-MySQL-Primary.yaml \`
		AWS 管理员

Task	描述	所需技能
	--parameter-overrides file://Aurora-MySQL-parameter-us-east-1.json \ --capabilities CAPABILITY_AUTO_EXPAND CAPABILITY_NAMED_IAM CAPABILITY_IAM \ --disable-rollback 亚马逊 ElastiCache（Redis OSS） cd <YOUR-LOCAL-GIT-FOLDER>/App-Stack aws cloudformation deploy \ --region us-east-1 --stack-name elasticache-ds-app-stack \ --template-file ElastiCache-Primary.yaml \ --parameter-overrides file://ElastiCache-parameter-us-east-1.json \ --capabilities CAPABILITY_AUTO_EXPAND CAPABILITY_NAMED_IAM CAPABILITY_IAM \ --disable-rollback 验证 AWS CloudFormation 资源是否成功部署。	

Task	描述	所需技能
更新辅助数据库实例或集群的参数文件。	在<YOUR LOCAL GIT FOLDER>/App-Stack 文件夹中，更新辅助区域的参数文件。 Amazon RDS 在RDS-MySQL-parameter-us-west-2.json 文件中SubnetIDs ，DBSecurityGroup 使用您创建的资源的名称更新和。PrimaryRegionKMSKeyArn 使用MySQLKmsKeyId 从主数据库实例 AWS CloudFormation 堆栈的 O ut puts 部分中获取的值更新： `{ "Parameters": { "SubnetIds": "subnet-aaaaaaaaa, subnet-bbbbbbbbb", "DBSecurityGroup": "sg-ccccccccc", "MySqlGlobalIdentifier":"rds-mysql-instance", "InitialDatabaseName": "mysqldb", "DBPortNumber": "3789", "PrimaryRegion": "us-east-1", "SecondaryRegion": "us-west-2", "KMSKeyAliasName": "rds/rds-mysql-instance-KmsKeyId",`	AWS 管理员

Task	描述	所需技能

	<pre> "PrimaryRegionKMSKeyArn":"arn:aws:kms:us-east-1:xxxxxxxxx:key/mrk-xxxxxxxxxxxxxxxxx"
	  }
	}</pre>
	Amazon Aurora
	
	在Aurora-MySQL-parameter-us-west-2.json 文件中，DBSecurityGroup 使用您创建的资源的名称更新SubnetIDs 和。PrimaryRegionKMSKeyArn 使用AuroraKmsKeyId 从主数据库实例 AWS CloudFormation 堆栈的 O ut puts 部分中获取的值更新：
	
	<pre>{
	  "Parameters": {
	    "SubnetIds": "subnet1-aaaaaaaaa,subnet2-bbbbbbbbb",
	    "DBSecurityGroup": "sg-ccccccccc",
	    "GlobalClusterIdentifier":"dr-globaldb-cluster-mysql",
	    "DBClusterName":"dbcluster-01",
	    "SourceDBClusterName":"dbcluster-02",
	    "DBPortNumber": "3787",</pre>	

Task	描述	所需技能
	<pre>"DBInstanceClass": "db.r5.large", "InitialDatabaseName": "sampledb", "PrimaryRegion": "us-east-1", "SecondaryRegion": "us-west-2", "KMSKeyAliasName": "rds/dr-globaldb-cluster-mysql-KmsKeyId" } }</pre> 亚马逊 ElastiCache（Redis OSS） 在ElastiCache-parameter-us-west-2.json 文件中SubnetIDs ，DBSecurityGroup 使用您创建的资源的名称更新和。PrimaryRegionKMSKeyArn 使用ElastiCacheKmsKeyId 从主数据库实例 AWS CloudFormation 堆栈的 O ut puts 部分中获取的值更新： <pre>{ "Parameters": { "CacheNodeType": "cache.m5.large", "DBSecurityGroup": "sg-ccccccccc", "SubnetIds": "subnet-aaaaaaaaa, subnet-bbbbbbbbb",</pre>	

Task	描述	所需技能
	"EngineVersion": "5.0.6", "GlobalReplicationGroupIdSuffix": "demo-redis-global-datastore", "NumReplicas": "1", "NumShards": "1", "ReplicationGroupId": "demo-redis-cluster", "DBPortNumber": "3788", "TransitEncryption": "true", "KMSKeyAliasName": "elasticache/demo-redis-global-datastore-KmsKeyId", "PrimaryRegion": "us-east-1", "SecondaryRegion": "us-west-2" } }	

Task	描述	所需技能

在辅助区域部署您的数据库实例或集群。	根据您的数据库引擎运行以下命令。
	
	Amazon RDS
	
	`cd <YOUR-LOCAL-GIT-FOLDER>/App-Stack`
	
	`aws cloudformation deploy \`
	`--region us-west-2 \`
	`--stack-name rds-mysql-app-stack \`
	`--template-file RDS-MySQL-DR.yaml \`
	`--parameter-overrides file://RDS-MySQL-parameter-us-west-2.json \`
	`--capabilities CAPABILITY_AUTO_EXPAND CAPABILITY_NAMED_IAM CAPABILITY_IAM \`
	`--disable-rollback`
	
	Amazon Aurora
	
	`cd <YOUR-LOCAL-GIT-FOLDER>/App-Stack`
	
	`aws cloudformation deploy \`
	`--region us-west-2 \`
	`--stack-name aurora-mysql-app-stack \`
	`--template-file Aurora-MySQL-DR.yaml \`
	`--parameter-overrides file://Aurora-MySQ`
		AWS 管理员

Task	描述	所需技能
	L-parameter-us-wes t-2.json \ --capabilities CAPABILITY_AUTO_EX PAND CAPABILIT Y_NAMED_IAM CAPABILIT Y_IAM \ --disable-rollback 亚马逊 ElastiCache（Redis OSS） cd <YOUR-LOCAL-GIT-FO LDER>/App-Stack aws cloudformation deploy \ --region us-west-2 \ --stack-name elasticac he-ds-app-stack \ --template-file ElastiCache-DR.yaml \ --parameter-overrides file://ElastiCache -parameter-us-west -2.json \ --capabilities CAPABILITY_AUTO_EX PAND CAPABILIT Y_NAMED_IAM CAPABILIT Y_IAM \ --disable-rollback 验证 AWS CloudFormation 资源是否成功部署。	

相关资源

- 数据库的灾难恢复策略 AWS（AWS 规范性指导策略）

- 在 AWS（AWS 规范性指导指南）上自动执行关系数据库灾难恢复解决方案
- 使用 Amazon Aurora Global Database
- 跨 AWS 区域 使用全局数据存储进行复制
- 在 AWS（AWS 规范性指导指南）上自动执行关系数据库灾难恢复解决方案

在跨区域自动复制 Amazon RDS 实例 AWS 账户

由 Parag Nagwekar (AWS) 和 Arun Chandapillai (AWS) 编写

摘要

此模式向您展示了如何使用 AWS Step Functions 和自动执行跨不同的复制、跟踪和回滚亚马逊关系数据库服务 (Amazon RDS) 数据库实例 AWS 账户 的过程。 AWS Lambda您可以使用此自动化来执行 RDS 数据库实例的大规模复制，而不会影响性能或运营开销 - 无论您的组织规模如何。您还可以使用这种模式来帮助您的组织遵守强制性的数据治理策略或合规性要求，这些策略或合规性要求要求在不同的 AWS 账户 和之间复制和冗余数据 AWS 区域。大规模 Amazon RDS 数据的跨账户复制是低效且容易出错的手动过程，可能成本高昂且耗时，但此模式中的自动化可以帮助您安全、有效且高效地实现跨账户复制。

先决条件和限制

先决条件

- 二 AWS 账户
- 一个 RDS 数据库实例，在源中启动并正在运行 AWS 账户
- 目标中 RDS 数据库实例的子网组 AWS 账户
- 在源中创建 AWS 账户 并与目标账户共享的 AWS Key Management Service (AWS KMS) 密钥（有关策略详细信息的更多信息，请参阅此模式的其他信息部分。）
- 目标 AWS KMS key 中的一个 AWS 账户 ，用于加密目标账户中的数据库

限制

- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额页面，然后选择服务的链接。

产品版本

- Python 3.9（使用 AWS Lambda）
- PostgreSQL 11.3、13.x 和 14.x

架构

技术堆栈

- Amazon Relational Database Service(Amazon RDS)
- Amazon Simple Notification Service (Amazon SNS)
- AWS Key Management Service (AWS KMS)
- AWS Lambda
- AWS Secrets Manager
- AWS Step Functions

目标架构

下图显示了一种架构，用于使用 Step Functions 编排 RDS 数据库实例从源账户（账户 A）到目标账户（账户 B）的定时按需复制。

在源账户（图中的账户 A）中，Step Functions 状态机执行以下操作：

1. 从账户 A 的 RDS 数据库实例创建快照
2. 使用 AWS KMS key 来自账户 A 复制和加密快照。为确保传输过程中的加密，无论数据库实例是否加密，都将对快照进行加密。
3. 通过授予账户 B 对快照的访问权限，与账户 B 共享数据库快照。
4. 向 SNS 主题推送通知，然后 SNS 主题调用账户 B 中的 Lambda 函数。

在目标账户（图中的账户 B）中，Lambda 函数运行 Step Functions 状态机来编排以下内容：

1. 将共享快照从账户 A 复制到账户 B，同时使用 AWS KMS key 来自账户 A 先解密数据，然后使用账户 B AWS KMS key 中的账户对数据进行加密。
2. 从 Secrets Manager 中读取密钥，以捕获当前数据库实例的名称。
3. 使用新名称和 Amazon RDS 的默认 AWS KMS key 名称从快照中恢复数据库实例。
4. 读取新数据库的端点并使用新的数据库端点更新 Secrets Manager 中的密钥，然后为以前的数据库实例添加标签，以便日后将其删除。
5. 保留数据库最新 N 个实例，并删除所有其他实例。

工具

AWS 服务

- Amazon Relational Database Service（Amazon RDS）可帮助您在中设置、操作和扩展关系数据库 AWS Cloud。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。
- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥以帮助保护您的数据。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- 适用于 Python (Boto3) 的 AWS SDK是一个软件开发套件，可帮助您将 Python 应用程序、库或脚本与集成 AWS 服务。
- AWS Secrets Manager 有助于您通过对 Secrets Manager 的 API 调用来替换代码中的硬编码凭证（包括密码），以编程方式检索密钥。
- AWS Step Functions是一项无服务器编排服务，可帮助您组合 Lambda 函数和其他函数 AWS 服务来构建业务关键型应用程序。

代码存储库

此模式的代码可在 GitHub 跨账户 RDS 复制存储库中找到。

操作说明

只需单击一下即可自动复制 RDS 数据库实例 AWS 账户

Task	描述	所需技能
在源账户中部署 CloudFormation 堆栈。	1. 登录源账户（账户 A）并打开CloudFormation 控制台。AWS Management Console 2. 在导航窗格中，选择 Stacks（堆栈）。	云管理员、云架构师

Task	描述	所需技能
	3. 选择创建堆栈，然后选择使用现有的资源（导入资源）。 4. 在识别资源页面，选择下一步。 5. 在 Select Template（选择模板）页面上，选择 Upload a template（上传模板）。 6. 选择 “选择文件”，从 “GitHub 跨账户 RDS 复制” 存储库中选择`Cloudformation-SourceAccountRDS.yaml` 文件，然后选择 “下一步”。 7. 在堆栈名称中，输入资源堆栈的名称。 8. 在 Parameters（参数）部分中，指定在堆栈模板中定义的以下参数： • 对于 DestinationAccountNumber，输入您的目标 RDS 数据库实例的账号。 • 对于 KeyName，请输入您的 AWS KMS key。 • 对于 ScheduleExpression，输入 cron 表达式（默认值为每天上午 12:00）。 • 在源中DBIdentifier，输入源数据库的名称。 • 在源DBSnapshot名称中，输入快照的名称或接受默认名称。 9. 选择下一步。	

Task	描述	所需技能
	10.在配置堆栈选项页面上，保留默认设置，然后选择下一步。 11.查看集合配置并选择 Submit（提交）。 12.选择堆栈资源选项卡，然后记下 SNS 主题的 Amazon 资源名称（ARN）。	

Task	描述	所需技能
在目标账户中部署 CloudFormation 堆栈。	1. 登录目标账户（账户 B）并打开CloudFormation 控制台。 AWS Management Console 2. 在导航窗格中，选择 Stacks（堆栈）。 3. 选择创建堆栈，然后选择使用现有的资源（导入资源）。 4. 在识别资源页面，选择下一步。 5. 在 Select Template（选择模板）页面上，选择 Upload a template（上传模板）。 6. 选择文件，从 GitHub 跨账户 RDS 复制存储库中选择`Cloudformation-DestinationAccountRDS.yaml` 文件，然后选择下一步。 7. 在堆栈名称中，输入资源堆栈的名称。 8. 在 Parameters（参数）部分中，指定在堆栈模板中定义的以下参数： • 对于 DatabaseName，输入数据库的名称。 • 在引擎，输入与源数据库匹配的数据库引擎类型。 • 对于 DBInstanceClass，输入首选数据库实例类型或接受默认值。	云架构师、 DevOps 工程师、云管理员

Task	描述	所需技能
	• 在子网组，请输入现有 VPC 子网组。有关创建子网组的说明，请参阅 Amazon RDS 文档中的步骤 2：创建数据库子网组。 • 对于 SecretName，请输入路径和密钥名称，或者接受默认值。 • 在 SGID，请输入目标集群的安全组 ID。 • 对于 KMSKey，请输入目标账户中 KMS 密钥的 ARN。 • 对于 NoOfOlderInstances，输入要为回滚保留的 RDS 数据库实例的旧副本数量。 9. 选择下一步。 10. 在配置堆栈选项页面上，保留默认设置，然后选择下一步。 11. 查看集合配置并选择 Submit（提交）。 12. 选择堆栈的资源选项卡，记下其物理 ID 和 `InvokeStepFunction` ARN。	

Task	描述	所需技能
验证目标账户中是否创建了 RDS 数据库实例。	1. 登录 AWS Management Console 并打开 Amazon RDS 控制台。 2. 在导航窗格中，选择数据库，然后验证新 RDS 数据库实例是否出现在新集群下。	云管理员、云架构师、DevOps 工程师

Task	描述	所需技能

将 Lambda 函数订阅至 SNS 主题。	您必须运行以下 AWS Command Line Interface (AWS CLI) 命令才能将目标账户（账户 B）中的 Lambda 函数订阅到源账户（账户 A）中的 SNS 主题。
	
	在账户 A 中，运行以下命令：
	
	`aws sns add-permission \ --label lambda-access --aws-account-id <DestinationAccount> \ --topic-arn <Arn of SNSTopic > \ --action-name Subscribe ListSubscriptionsB yTopic`
	
	在账户 B 中，运行以下命令：
	
	`aws lambda add-permi ssion \ --function-name <Name of InvokeStepFunction > \ --source-arn <Arn of SNSTopic > \ --statement-id function-with-sns \ --action lambda:In vokeFunction \ --principal sns.amazo naws.com`
	
	在账户 B 中，运行以下命令：
		云管理员、云架构师、数据库管理员

Task	描述	所需技能
	`aws sns subscribe \ --protocol "lambda" \ --topic-arn <Arn of SNSTopic> \ --notification-endpoint <Arn of InvokeStepFunction>`	

Task	描述	所需技能
将源账户中的 RDS 数据库实例与目标账户同步。	通过在源账户中启动 Step Functions 状态机，启动按需数据库复制。 1. 打开 Step Functions 控制台 2. 在导航窗格中，选择状态管理器。 3. 选择状态机。 4. 在执行选项卡，选择您的函数，然后选择开始执行以启动工作流。 **Note** 已有计划程序可以帮助您按计划自动运行复制，但默认情况下，该计划程序处于关闭状态。您可以在目标账户 CloudFormation 堆栈的“资源”选项卡中找到计划程序的 Amazon CloudWatch 规则名称。有关如何修改 CloudWatch 事件规则的说明，请参阅 CloudWatch 文档中的删除或禁用 CloudWatch 事件规则。	云架构师、 DevOps 工程师、云管理员

Task	描述	所需技能
需要时，可以将数据库回滚至之前的任何副本。	1. 打开 Secrets Manager 控制台。 2. 从密钥列表中，选择您之前使用 CloudFormation 模板创建的密钥。您的应用程序使用该密钥来访问目标集群中的数据库。 3. 在秘密详细信息页面上的秘密值部分中，选择检索秘密值，然后选择编辑。 4. 输入数据库端点详细信息。	云管理员、数据库管理员、工程师 DevOps

相关资源

- 跨区域只读副本（Amazon RDS 文档）
- 蓝/绿部署（Amazon RDS 文档）

其他信息

您可以使用以下示例策略来 AWS KMS key 共享您的信息 AWS 账户。

```
{
    "Version": "2012-10-17",
    "Id": "cross-account-rds-kms-key",
    "Statement": [
        {
            "Sid": "Enable user permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::<SourceAccount>:root"
            },
            "Action": "kms:*",
            "Resource": "*"
        },
        {
            "Sid": "Allow administration of the key",
```

```
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::<DestinationAccount>:root"
            },
            "Action": [
                "kms:Create*",
                "kms:Describe*",
                "kms:Enable*",
                "kms:List*",
                "kms:Put*",
                "kms:Update*",
                "kms:Revoke*",
                "kms:Disable*",
                "kms:Get*",
                "kms:Delete*",
                "kms:ScheduleKeyDeletion",
                "kms:CancelKeyDeletion"
            ],
            "Resource": "*"
        },
        {
            "Sid": "Allow use of the key",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::<DestinationAccount>:root",
                    "arn:aws:iam::<SourceAccount>:root"
                ]
            },
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "kms:DescribeKey",
                "kms:CreateGrant"
            ],
            "Resource": "*"
        }
    ]
}
```

使用 Systems Manager 自动备份 SAP HANA 数据库和 EventBridge

创建者：Ambarish Satarkar（AWS）和 Gaurav Rath（AWS）

摘要

此模式描述了如何使用 AWS Systems Manager、亚马逊 EventBridge、亚马逊简单存储服务 (Amazon S3) Service 和适用于 SAP HANA 的 AWS Backint Agent 自动备份 SAP HANA 数据库。

此模式提供了一种使用 `BACKUP DATA` 命令的基于 Shell 脚本的方法，无需在多个系统中维护每个操作系统（OS）实例的脚本和作业配置。

> **Note**
>
> 截至2023年4月，AWS Backup宣布在亚马逊弹性计算云（亚马逊 EC2）上支持SAP HANA数据库。有关更多信息，请参阅亚马逊 EC2 实例上的 SAP HANA 数据库备份。

根据贵组织的需求，您可以使用 AWS Backup 服务自动备份您的 SAP HANA 数据库，也可以使用这种模式。

先决条件和限制

先决条件

- 在为 Systems Manager 配置的托管亚马逊弹性计算云 (Amazon EC2) 实例上，支持版本处于运行状态的现有 SAP HANA 实例
- 已安装 2.3.274.0 或更高版本的 Systems Manager Agent（SSM Agent）
- 未启用公开访问权限的 S3 存储桶
- 一个名为 `SYSTEM` 的 `hdbuserstore` 密钥
- AWS Identity and Access Management（IAM）角色用于自动化运行手册按计划运行
- 并且 `AmazonSSMManagedInstanceCore` 和 `ssm:StartAutomationExecution` 策略附加到 Systems Manager 自动化服务角色。

限制

- 适用于 SAP HANA 的 AWS Backint 代理不支持重复数据删除。

- 适用于 SAP HANA 的 AWS Backint 代理不支持数据压缩。

产品版本

在以下操作系统上支持 AWS Backint 代理：

- SUSE Linux Enterprise Server
- SUSE Linux Enterprise Server for SAP
- Red Hat Enterprise Linux for SAP

AWS Backint 代理支持以下数据库：

- SAP HANA 1.0 SP12（单节点和多节点）
- SAP HANA 2.0 及更高版本（单节点和多节点）

架构

目标技术堆栈

- AWS Backint 代理
- Amazon S3
- AWS Systems Manager
- Amazon EventBridge
- SAP HANA

目标架构

下图显示了安装 AWS Backint Agent、S3 存储桶和 Systems Manager EventBridge 的安装脚本，它们使用命令文档来安排定期备份。

自动化和扩缩

- 使用 Systems Manager 自动化运行手册可以安装多个 AWS Backint 代理。
- 根据目标选择，每次运行 Systems Manager 运行手册都可以扩展到 n 个 SAP HANA 实例。
- EventBridge 可以自动执行 SAP HANA 备份。

工具

- 适用于 SAP HANA 的 AWS Backint 代理是一款独立的应用程序，它与现有工作流程集成，可将 SAP HANA 数据库备份到您在配置文件中指定的 S3 存储桶。AWS Backint 代理支持 SAP HANA 数据库的完整、增量和差异备份。AWS Backint 代理在 SAP HANA 数据库服务器上运行，在该服务器上，备份和目录从 SAP HANA 数据库传输到 AWS Backint 代理中。
- Amazon EventBridge 是一项无服务器事件总线服务，可用于将应用程序与来自各种来源的数据连接起来。 EventBridge 将来自您的应用程序、软件即服务 (SaaS) 应用程序和 AWS 服务的实时数据流传输到目标，例如 AWS Lambda 函数、使用 API 目标的 HTTP 调用终端节点或其他账户中的事件总线。
- Amazon Simple Storage Service (Amazon S3) 是一项对象存储服务。您可以通过 Amazon S3 随时在 Web 上的任何位置存储和检索的任意大小的数据。
- AWS Systems Manager 可帮助您查看和控制您在 AWS 上的基础设施。通过使用 Systems Manager 控制台，您可以查看来自多个 Amazon Web Services 的操作数据并在 AWS 资源之间自动执行操作任务。

代码

此模式的代码可在aws-backint-automated-backup GitHub 存储库中找到。

操作说明

创建 hdbuserstore 密钥系统

Task	描述	所需技能
创建 hdbuserstore 密钥。	1. 导航到 `/usr/sap/<SID>/HDB<Inst No>/exe`。 2. 运行以下命令，以 XX 作为 SAP HANA 数据库实例号。 `hdbuserstore -i set SYSTEM <hostname>:3XX13@SYSTEMDB SYSTEM`	AWS 管理员、SAP HANA 管理员

Task	描述	所需技能
	例如，对于具有实例编号 00 的 SAP HANA 主机 saphanadb ，请运行以下命令。 `hdbuserstore -i set SYSTEM saphanadb :30013@SYSTEMDB SYSTEM`	

安装 AWS Backint 代理

Task	描述	所需技能
安装 AWS Backint 代理。	按照 AWS Backint 代理文档中安装和配置适用于 SAP HANA 的 AWS Backint 代理中的说明进行操作。	AWS 管理员、SAP HANA 管理员

创建 Systems Manager 命令文档

Task	描述	所需技能
创建 Systems Manager 命令文档。	1. 登录 AWS 管理控制台，并打开 AWS Systems Manager 控制台。 2. 选择文档，然后选择我拥有。 3. 确认并确保您与 SAP HANA 数据库位于同一 AWS 区域。 4. 选择创建文档、命令或会话来创建您的文档。	AWS 管理员、SAP HANA 管理员

Task	描述	所需技能
	5. 使用不带空格的唯一描述性名称（例如，SAP HANA-Backup）。 6. 确保将文档类型设置为命令文档。 7. 在内容标题下方，有一些示例代码。请务必选择 JSON 代码类型，并将该代码替换为GitHub存储库中HDB_Backup_SSM_Document.json 文件中的代码。 8. 选择创建文档。 9. 在我拥有部分查看您的文档。	

定期安排备份

Task	描述	所需技能
使用 Amazon 安排定期备份 EventBridge。	1. 打开 Amazon EventBridge 控制台，选择规则，然后选择创建规则。 2. 在定义规则详细信息屏幕上，输入规则的唯一名称和描述，然后使用默认的事件总线。 3. 在规则类型下方，选择计划，然后选择下一步。 4. 然后在定义计划屏幕上，根据所需频率选择相应的计划模式和 cron 或 rate 表达式。	AWS 管理员、SAP HANA 管理员

Task	描述	所需技能
	5. 在选择目标屏幕上，对于目标类型，选择 Amazon Web Services。在选择目标下方，选择 Systems Manager 运行命令。 6. 选择您之前创建的文档。 7. 在目标键和目标值下方，提供实例 ID。您可以使用标签名称和标签值来添加多个实例。 8. 在配置自动化参数下方，为增量备份或差异备份选择常量。如果要进行完整备份，请选择无参数。 9. 选择创建新角色或者使用现有角色。如果您使用现有角色，请确保该角色具有调用目标所需策略。 10.保留默认的其他设置，然后选择下一步。 11.配置标签屏幕是可选的。选择下一步。 12.在查看并创建屏幕上，查看规则设置，然后选择创建。规则应成功创建。 您可以从 S3 存储桶路径验证备份是否成功。 `s3:/<your_bucket_name>/<target folder>/<SID>/usr/sap/<SID>`	

Task	描述	所需技能
	`/SYS/global/hdb/backint/DB_<SID>/` 您还可以验证 SAP HANA 备份目录中的备份。	

相关资源

- 适用于 SAP HANA 的 AWS Backint 代理
- 安装和配置适用于 SAP HANA 的 AWS Backint 代理

使用 Python 应用程序为亚马逊 DynamoDB 自动生成 PynamoDB 模型和 CRUD 函数

由 Vijit Vashishtha (AWS)、Dheeraj Alimchandani (AWS) 和 Dhananjay Karanjkar (AWS) 创作

摘要

为了高效执行 Amazon DynamoDB 数据库操作，通常需要实体以及创建、读取、更新和删除 (CRUD) 操作函数。Pynamodb 是一个基于 Python 的接口，支持 Python 3。它还提供了诸如支持 Amazon DynamoDB 事务、自动属性值序列化和反序列化以及与常见 Python 框架（例如 Flask 和 Django）的兼容性等功能。这种模式提供了一个简化 PynamoDB 模型和 CRUD 操作函数的自动创建的库，从而帮助使用 Python 和 DynamoDB 的开发人员。虽然它可以为数据库表生成基本的 CRUD 函数，但它也可以对 PynamoDB 模型和 Amazon DynamoDB 表中的 CRUD 函数进行逆向工程。此模式旨在通过使用基于 Python 的应用程序来简化数据库操作。

以下是此解决方案的主要特点：

- PynamoDB 模型的 JSON 架构 — 通过导入 JSO N 架构文件在 Python 中自动生成 PynamoDB 模型。
- CRUD 函数生成 — 自动生成函数以对 DynamoDB 表执行 CRUD 操作。
- 来自 DynamoDB 的逆向工程 — 使用 PynamoDB 对象关系映射 (ORM) 对现有亚马逊 DynamoDB 表的 PynamoDB 模型和 CRUD 函数进行逆向工程。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- Python 3.8 或更高版本，已下载并安装
- Jinja2 版本 3.1.2 或更高版本，已下载并安装
- 要为其生成 ORM 的亚马逊 DynamoDB 表
- AWS Command Line Interface (AWS CLI)，已安装并配置
- PynamoDB 版本 5.4.1 或更高版本，已安装

架构

目标技术堆栈

- json 脚本
- Python 应用程序
- pynamodb 模型
- 亚马逊 DynamoDB 数据库实例

目标架构

1. 您创建了一个输入 JSON 架构文件。此 JSON 架构文件表示您要从中创建 PynamoDB 模型的相应的 DynamoDB 表的属性以及 CRUD 函数。它包含以下三个重要密钥：
   - `name`— 目标 DynamoDB 表的名称。
   - `region`— 桌子的托管 AWS 区域 位置
   - `attributes`— 作为目标表一部分的属性，例如分区键（也称为哈希属性）、排序键、本地二级索引、全局二级索引以及任何非键属性。当应用程序直接从目标表中获取关键属性时，此工具期望输入架构仅提供非键属性。有关如何在 JSON 架构文件中指定属性的示例，请参阅此模式的其他信息部分。
2. 运行 Python 应用程序并提供 JSON 架构文件作为输入。
3. Python 应用程序会读取 JSON 架构文件。
4. Python 应用程序连接到 DynamoDB 表以派生架构和数据类型。应用程序运行 desc ribe_table 操作并获取表的键和索引属性。
5. Python 应用程序结合了 JSON 架构文件和 DynamoDB 表中的属性。它使用 Jinja 模板引擎生成 PynamoDB 模型和相应的 CRUD 函数。
6. 您可以访问 PynamoDB 模型来对 DynamoDB 表执行 CRUD 操作。

工具

AWS 服务

- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。

其他工具

- Jinja 是一个可扩展的模板引擎，可将模板编译成经过优化的 Python 代码。这种模式使用 Jinja 通过在模板中嵌入占位符和逻辑来生成动态内容。
- PynamoDB 是亚马逊 DynamoDB 的基于 Python 的界面。
- Python 是通用的计算机编程语言。

代码存储库

此模式的代码可在 GitHub 自动生成 Pynamodb 模型和 CRUD 函数存储库中找到。存储库分为两个主要部分：控制器包和模板。

控制器包

控制器 Python 包包含帮助生成 PynamoDB 模型和 CRUD 函数的主要应用程序逻辑。其中包含以下内容：

- `input_json_validator.py`— 此 Python 脚本验证输入 JSON 架构文件并创建包含目标 DynamoDB 表列表和每个表所需属性的 Python 对象。
- `dynamo_connection.py`— 此脚本建立与 DynamoDB 表的连接，并使用`describe_table`该操作提取创建 PynamoDB 模型所需的属性。
- `generate_model.py`— 此脚本包含一个 Python 类`GenerateModel`，该类基于输入 JSON 架构文件和操作创建 PynamoDB 模型。`describe_table`
- `generate_crud.py`— 对于 JSON 架构文件中定义的 DynamoDB 表，此脚本使用`GenerateCrud`操作创建 Python 类。

模板

此 Python 目录包含以下 Jinja 模板：

- `model.jinja`— 这个 Jinja 模板包含用于生成 Pynamodb 模型脚本的模板表达式。
- `crud.jinja`— 此 Jinja 模板包含用于生成 CRUD 函数脚本的模板表达式。

操作说明

设置环境

Task	描述	所需技能
克隆存储库。	输入以下命令来克隆自动生成 Pynamodb 模型和 CRUD 函数存储库。 `git clone https://github.com/aws-samples/amazon-reverse-engineer-dynamodb.git`	应用程序开发人员

设置 Python 环境。	1. 导航到克隆存储库中的顶级目录。
	`cd amazon-reverse-engineer-dynamodb`
	2. 输入以下命令安装所需的库和软件包。
	`pip install -r requirements.txt`
		应用程序开发人员

生成 pynamoDB 模型和 CRUD 函数

Task	描述	所需技能
修改 JSON 架构文件。	1. 导航到克隆存储库中的顶级目录。 `cd amazon-reverse-engineer-dynamodb`	应用程序开发人员

Task	描述	所需技能
	2. 在您的首选编辑器中打开该`test.json` 文件。您可以使用此文件作为参考来创建自己的 JSON 架构文件，也可以更新此文件中的值以匹配您的环境。 3. 修改目标 DynamoDB 表的名称 AWS 区域、和属性值。 **Note** 如果您定义的表不存在于 JSON 架构文件中，则此解决方案不会为该表生成模型或 CRUD 函数。 4. 保存并关闭 `test.json` 文件。我们建议您使用新名称保存此文件。	
运行 Python 应用程序。	输入以下命令生成 Pynamodb 模型和 CRUD 函数，`<input_schema.json>` 其中是您的 JSON 架构文件的名称。 `python main.py --file <input_schema.json>`	应用程序开发人员

验证 pynamoDB 模型和 CRUD 函数

Task	描述	所需技能
验证生成的 PynamoDB 模型。	1. 在克隆存储库的顶级目录中，输入以下命令以导航到`models`存储库。 `cd models` 2. 默认情况下，此解决方案会命名 Pynamodb 模型文件。`demo_model.py` 验证此文件是否存在。	应用程序开发人员
验证生成的 CRUD 函数。	1. 在克隆存储库的顶级目录中，输入以下命令以导航到`crud`存储库。 `cd crud` 2. 默认情况下，此解决方案会命名脚本`demo_crud.py` 。验证此文件是否存在。 3. 使用`demo_crud.py` 文件中的 Python 类对目标 DynamoDB 表执行 CRUD 操作。确认操作成功完成。	应用程序开发人员

相关资源

- 亚马逊 DynamoDB 的核心组件（DynamoDB 文档）
- 使用@@ 二级索引改善数据访问（DynamoDB 文档）

其他信息

JSON 架构文件的示例属性

```
[
{
"name": "test_table",
"region": "ap-south-1",
"attributes": [
{
"name": "id",
"type": "UnicodeAttribute"
},
{
"name": "name",
"type": "UnicodeAttribute"
},
{
"name": "age",
"type": "NumberAttribute"
}
]
}
]
```

通过使用 Cloud Custodian 来阻止对 Amazon RDS 的公有访问

创建者：abhay kumar (AWS) 和 Dwarika Patra (AWS)

摘要

许多组织在多个云供应商上运行其工作负载和服务。在这些混合云环境中，除了各个云提供商提供的安全性外，云基础设施还需要严格的云治理。Amazon Relational Database Service (Amazon RDS) 等云数据库是一项重要的服务，必须对其进行监控以防存在任何访问和权限漏洞。尽管您可通过配置安全组来限制对 Amazon RDS 数据库的访问，但您可添加第二层保护来禁止诸如公有访问之类的操作。阻止公众访问有助于您遵守《通用数据保护条例》(GDPR)、《健康保险便携性和责任法案》(HIPAA)、美国国家标准与技术研究院 (NIST) 和支付卡行业数据安全标准 (PCI DSS) 的合规性。

Cloud Custodian 是一个开源规则引擎，可用于对亚马逊 RDS 等亚马逊 Web Services (AWS) 资源实施访问限制。借助 Cloud Custodian，您可设置规则，根据定义的安全和合规标准对环境进行验证。您可使用 Cloud Custodian 来管理您的云环境，帮助确保遵守安全政策、标签政策、未使用资源的垃圾回收和成本管理。借助 Cloud Custodian，您可使用单一界面在混合云环境中实施治理。例如，你可以使用云托管人界面与 AWS Microsoft Azure 进行交互，从而减少使用 AWS 安全组和 Azure 策略等 AWS Config机制的工作量。

此模式提供了有关如何使用云托管 AWS 来强制限制对 Amazon RDS 实例的公开访问权限的说明。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 密钥对
- AWS Lambda 已安装

架构

下图显示了云托管人将策略部署到并 AWS CloudTrail 启动`CreateDBInstance`事件 AWS Lambda，以及在 Amazon RDS 上将 Lambda 函数`PubliclyAccessible`设置为 false。

工具

AWS 服务

- AWS CloudTrail帮助您审计您的治理、合规和运营风险 AWS 账户。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行 shell 中的命令进行交互。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Relational Database Service（Amazon RDS）可帮助您在中设置、操作和扩展关系数据库 AWS Cloud。

其他工具

- Cloud Custodian 将大多数组织用于管理其公共云账户的多种工具和脚本统一到一个开源工具中。它使用无状态规则引擎来定义和实施策略，为云基础设施提供指标、结构化输出和详细报告。它与无服务器运行时系统紧密集成，以低运营开销提供实时修复和响应。

操作说明

设置 AWS CLI

Task	描述	所需技能
安装 AWS CLI。	要安装 AWS CLI，请按照AWS 文档中的说明进行操作。	AWS 管理员

设置 AWS 凭据。	配置 AWS CLI 用于与之交互的设置 AWS，包括 AWS 区域 和您要使用的输出格式。
	`$>aws configure`
	`AWS Access Key ID [None]: <your_access_key_id>`
	`AWS Secret Access Key [None]: <your_secret_access_key>`
	`Default region name [None]:`
		AWS 管理员

Task	描述	所需技能
	`Default output format [None]:` 有关更多信息，请参阅 AWS 文档。	
创建一个 IAM 角色。	要创建具有 Lambda 执行角色的 IAM 角色，请运行以下命令。 `aws iam create-role --role-name lambda-ex --assume-role-policy-document '{"Version": "2012-10-17","Statement": [{ "Effect": "Allow", "Principal": {"Service": "lambda.amazonaws.com"}, "Action": "sts:AssumeRole"}]}`	AWS DevOps

设置 Cloud Custodian

Task	描述	所需技能
安装 Cloud Custodian。	要为您的操作系统和环境安装 Cloud Custodian，请按照 Cloud Custodian 文档中的说明进行操作。	DevOps 工程师
检查 Cloud Custodian 架构。	要查看您可针对其运行策略的 Amazon RDS 资源的完整列表，请使用以下命令。 `custodian schema aws.rds`	DevOps 工程师

Task	描述	所需技能
创建Cloud Custodian策略。	使用 YAML 扩展名将 Cloud Custodian 策略文件下的代码保存在其他信息部分。	DevOps 工程师
定义 Cloud Custodian 操作以更改可公开访问的标志。	1. 找到保管人代码（例如 `/Users/abcd/custodian/lib/python3.9/site-packages/c7n/resources/rds.py`）。 2. 在 `rds.py` 中找到 `RDSSetPublicAvailability` 类，然后使用其他信息部分中的 c7n 资源 rds.py 文件下方的代码修改此类。	DevOps 工程师
执行试运行。	（可选）要在不对资源运行任何操作的情况下检查策略识别了哪些资源，请使用以下命令。 `custodian run -dryrun <policy_name>.yaml -s <output_directory>`	DevOps 工程师

部署策略

Task	描述	所需技能
使用 Lambda 部署策略。	要创建将运行策略的 Lambda 函数，请使用以下命令。	DevOps 工程师

Task	描述	所需技能
	`custodian run -s policy.yaml` 然后，该政策将由 AWS CloudTrail `CreateDBInstance` 活动启动。 因此，`false`对于符合条件的实例， AWS Lambda 会将可公开访问的标志设置为。	

相关资源

- AWS Lambda 网站
- 亚马逊 RDS 网站
- 云托管人文档

其他信息

Cloud Custodian 策略 YAML 文件

```
policies:
  - name: "block-public-access"
    resource: rds
    description: |
      This Enforcement blocks public access for RDS instances.
    mode:
      type: cloudtrail
      events:
        - event: CreateDBInstance # Create RDS instance cloudtrail event
          source: rds.amazonaws.com
          ids: requestParameters.dBInstanceIdentifier
      role: arn:aws:iam::1234567890:role/Custodian-compliance-role
    filters:
      - type: event
        key: 'detail.requestParameters.publiclyAccessible'
        value: true
```

```
    actions:
      - type: set-public-access
        state: false
```

c7n 资源 rds.py 文件

```
@actions.register('set-public-access')
 class RDSSetPublicAvailability(BaseAction):

     schema = type_schema(
         "set-public-access",
         state={'type': 'boolean'})
     permissions = ('rds:ModifyDBInstance',)

     def set_accessibility(self, r):
         client = local_session(self.manager.session_factory).client('rds')
         waiter = client.get_waiter('db_instance_available')
         waiter.wait(DBInstanceIdentifier=r['DBInstanceIdentifier'])
         client.modify_db_instance(
             DBInstanceIdentifier=r['DBInstanceIdentifier'],
             PubliclyAccessible=self.data.get('state', False))


     def process(self, rds):
         with self.executor_factory(max_workers=2) as w:
             futures = {w.submit(self.set_accessibility, r): r for r in rds}
             for f in as_completed(futures):
                 if f.exception():
                     self.log.error(
                         "Exception setting public access on %s  \n %s",
                         futures[f]['DBInstanceIdentifier'], f.exception())
         return rds
```

Security Hub 集成

可以与 Cloud Custodian 集成 AWS Security Hub，以发送安全调查结果并尝试补救措施。有关更多信息，请参阅宣布与云托管人集成 AWS Security Hub。

配置对 Amazon DynamoDB 的跨账户访问

由 Shashi Dalmia (AWS)、Esteban Serna Parra (AWS) 和 Imhoertha Ojior (AWS) 创作

摘要

此模式说明了使用基于资源的策略配置对 Amazon DynamoDB 的跨账户访问权限的步骤。对于使用 DynamoDB 的工作负载，使用工作负载隔离策略来最大限度地减少安全威胁并满足合规性要求变得越来越普遍。实施工作负载隔离策略通常需要使用 AWS Identity and Access Management 基于身份的 (IAM) 策略跨账户和跨区域访问 DynamoDB 资源。这包括设置 IAM 权限并在两者之间建立信任关系 AWS 账户。

Dynamo@@ DB 基于资源的策略极大地简化了跨账户工作负载的安全状况。此模式提供了步骤和示例代码，用于演示如何将 AWS Lambda 函数合而为一， AWS 账户 以便将数据写入其他账户的 DynamoDB 数据库表。

先决条件和限制

先决条件

- 两个活跃 AWS 账户。此模式将这些账户称为账户 A 和账户 B。
- AWS Command Line Interface (AWS CLI) 已安装并配置为访问账户 A，以创建 DynamoDB 表。此模式中的其他步骤提供了有关使用 IAM、DynamoDB 和 Lambda 控制台的说明。如果您打算 AWS CLI 改用，请将其配置为同时访问两个帐户。

限制

- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额页面，然后选择服务的链接。

架构

下图显示了单账户架构。 AWS Lambda、亚马逊 Elastic Compute Cloud（亚马逊 EC2）和 DynamoDB 都在同一个账户中。在这种情况下，Lambda 函数和亚马逊 EC2 实例可以访问 DynamoDB。要授予对 DynamoDB 表的访问权限，您可以在 IAM 中创建基于身份的策略，也可以在 DynamoDB 中创建基于资源的策略。

下图显示了多账户架构。如果一个中的资源 AWS 账户 需要访问其他账户中的 DynamoDB 表，则需要在 DynamoDB 中设置基于资源的策略来授予所需的访问权限。例如，在下图中，使用基于资源的策略向账户 B 中的 Lambda 函数授予对账户 A 中的 DynamoDB 表的访问权限。

此模式描述了 Lambda 和 DynamoDB 之间的跨账户访问。 AWS 服务 如果两个账户都配置了相应的权限，则可以对其他账户使用类似的步骤。例如，如果您想为账户 A 中的亚马逊简单存储服务 (Amazon S3) 存储段提供 Lambda 函数访问权限，则可以在 Amazon S3 中创建基于资源的策略，并将权限添加到账户 B 中的 Lambda 执行角色。

工具

AWS 服务

- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用您的 AWS 资源，从而帮助您安全地管理对这些资源的访问。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。

代码

此模式包括 “其他信息” 部分中的示例代码，用于说明如何配置账户 B 中的 Lambda 函数以写入账户 A 中的 DynamoDB 表。该代码仅用于说明和测试目的。如果您在生产环境中实现此模式，请使用该代码作为参考，并针对自己的环境对其进行自定义。

最佳实践

- 遵循 Dyn amoDB 文档中基于资源的策略的最佳实践。
- 遵循最低权限原则，授予执行任务所需的最低权限。有关更多信息，请参阅 IAM 文档中的授予最低权限和安全最佳实践。

操作说明

在账户 B 中为 Lambda 函数创建 IAM 策略和角色

Task	描述	所需技能
在账户 B 中创建策略。	此 IAM 策略允许对账户 A 中的 DynamoDB 表PutItem执行操作。 1. 在中登录账户 B AWS Management Console。 2. 打开 IAM 管理控制台。 3. 在导航窗格中，选择 策略，然后选择 创建策略。 4. 在指定权限页面上，对于策略编辑器，选择 JSON。 5. 输入以下策略。 `{ "Version": "2012-10-17", "Statement": [ { "Sid": "Statement1", "Effect": "Allow", "Action": "dynamodb:PutItem", "Resource": "arn:aws:dynamodb:<Region>:<Account-A-ID>:table/Table-Account-A" } ] }`	常规 AWS

Task	描述	所需技能
	6. 用您的值替换<Region>和<Account-A-ID> ，然后选择 “下一步”。 7. 在策略名称中，输入策略的唯一名称，例如DynamoDB-PutItem-Policy 。 8. （可选）添加策略描述。 9. 选择创建策略。	

Task	描述	所需技能
在账户 B 中创建角色。	账户 B 中的 Lambda 函数使用此 IAM 角色访问账户 A 中的 DynamoDB 表。 1. 打开 IAM 管理控制台。 2. 在导航窗格中，选择角色，然后选择创建角色。 3. 对于 Select trusted entity（选择受信任实体），选择 AWS 服务。 4. 在用例部分，选择 Lambda。 5. 选择下一步: 权限。 6. 在筛选器策略框中，输入 DynamoDB。 7. 在 DynamoDB 策略列表中，选择。`DynamoDB-PutItem-Policy` 8. 清除“筛选策略”框，然后输入 Lambda。 9. 在 Lambda 策略列表中，选择AWSLambda执行。 10选择下一步：名称、查看并创建。 11对于 Role name（角色名称），请为角色输入唯一名称，例如 `DynamoDB-PutItemAccess` 。 12.（可选）添加角色描述。 13.（可选）以键值对形式附加标签来向角色添加元数据。	常规 AWS

Task	描述	所需技能
	14选择 Create role（创建角色）。 有关创建角色的更多信息，请参阅 IAM 文档。	
记下 角色 ARN。	1. 打开 IAM 控制台。 2. 在导航窗格中，选择角色。 3. 在搜索框中输入DynamoDB-PutItemAccess ，然后选择角色。 4. 在该角色的摘要页面上，复制 Amazon 资源名称 (ARN)。您在设置 Lambda 函数时使用 ARN。	常规 AWS

在账户 A 中创建 DynamoDB 表

Task	描述	所需技能

创建 DynamoDB 表。	使用以下 AWS CLI 命令创建 DynamoDB 表。
	
	`aws dynamodb create-table \`
	`--table-name Table-Account-A \`
	`--attribute-definitions \`
	`Attribute Name=category,AttributeType=S \`
	`Attribute Name=item,AttributeType=S \`
	`--key-schema \`
		常规 AWS

Task	描述	所需技能

	`Attribute Name=category,KeyType=HASH \`
	`Attribute Name=item,KeyType=RANGE \`
	`--provisioned-throughput \`
	`ReadCapacityUnits=5,WriteCapacityUnits=5 \`
	`--resource-policy \`
	`'{`
	`"Version": "2012-10-17",`
	`"Statement": [`
	`{`
	`"Sid": "Statement1",`
	`"Effect": "Allow",`
	`"Principal": {`
	`"AWS": "arn:aws:iam::<Account-B-ID>:role/<Role-Name>"`
	`},`
	`"Action": "dynamodb:PutItem",`
	`"Resource": "arn:aws:dynamodb:<Region>:<Account-A-ID>:table/Table-Account-A"`
	`}`
	`]`
	`}'`	

Task	描述	所需技能
	替换此代码示例中的以下内容： • `<Account-B-ID>` 是账户 B 的 ID。 • `<Role-Name>` 是您创建的 IAM 角色的名称，例如`DynamoDB-PutItemAccess` 。 • `<Region>`是您创建 DynamoDB 表 AWS 区域 的地方。 • `<Account-A-ID>` 是账户 A 的 ID **Note** 您可以使用`--resource-policy`标志在`create-table` 语句中指定基于资源的策略配置。此策略引用账户 A 中 DynamoDB 表的 ARN 有关创建表的更多信息，请参阅 DynamoDB 文档。	

在账户 B 中创建 Lambda 函数

Task	描述	所需技能
创建一个 Lambda 函数，以向 DynamoDB 写入数据。	1. 在中登录账户 B AWS Management Console。 2. 打开 Lambda 控制台。 3. 在导航窗格中，选择函数，然后选择创建函数。 4. 对于名称，请输入 `lambda_write_function` 。 5. 对于运行时系统，请选择 Python 3.8 或更高版本。 6. 在 “更改默认执行角色” 下，选择 “使用现有角色”。 7. 对于现有角色，请选择您创建的 IAM 角色，例如`DynamoDB-PutItemAccess` 。 8. 选择 Create function (创建函数)。 9. 在 “代码” 选项卡中，粘贴此模式的 “其他信息” 部分中提供的示例代码。替换此代码示例中的以下内容： • `<Account-A-ID>` 是账户 A 的 ID • `<Region>`是您创建 DynamoDB 表 AWS 区域的地方。 10选择部署。 11选择测试。这将提示您配置测试事件。使用您的首	常规 AWS

Task	描述	所需技能
	选名称（例如）创建新事件`MyTestEventForWrite`，然后保存配置。 12.再次选择 Test (测试)。这将使用您提供的事件名称运行 Lambda 函数。 13.检查函数输出。它应表明该函数访问了账户 A 中的 DynamoDB 表并能够向该表写入数据。 有关创建 Lambda 函数的更多信息，请参阅 Lambda 文档。	

清理

Task	描述	所需技能
删除资源。	为避免产生与在此模式中创建的资源相关的成本，请执行以下操作来删除这些资源： 1. 在账户 B 中，删除您为连接 DynamoDB 而创建的 Lambda 函数。有关说明，请参阅 Lambda 文档。 2. 在账户 A 中，删除您创建的 DynamoDB 表。有关说明，请参阅 DynamoDB 文档。 3. 要了解安全最佳实践，请在不再需要 IAM 策略 (`DynamoDB-PutItem-Policy`) 时将其删除。有关	常规 AWS

Task	描述	所需技能
	更多信息，请参阅 IAM 文档。 4. 要了解安全最佳实践，请在不再需要 IAM 角色 (DynamoDB-PutItemAccess) 时将其删除。有关更多信息，请参阅 IAM 文档。	

故障排除

事务	解决方案
创建 Lambda 函数时，您会收到一个错误。ResourceNotFoundException	确认您已正确输入账户 A 的 AWS 区域 和 ID。这些是 DynamoDB 表的 ARN 的一部分。

相关资源

- DynamoDB 入门（DynamoDB 文档）
- Lambda 入门（Lambda 文档）
- 为 DynamoDB 使用基于资源的策略（Dynamo DB 文档）
- 创建 IAM policy（IAM 文档）
- 跨账户策略评估逻辑（IAM 文档）
- IAM JSON 策略元素参考（IAM 文档）

其他信息

示例代码

```
import boto3
from datetime import datetime

dynamodb_client = boto3.client('dynamodb')
```

```
def lambda_handler(event, context):
     now = datetime.now().isoformat()
     data = dynamodb_client.put_item(TableName='arn:aws:dynamodb:<Region>:<Account-
A-ID>:table/Table-Account-A', Item={"category": {"S": "Fruit"},"item": {"S":
 "Apple"},"time": {"S": now}})
     return data
```

> **Note**
>
> 实例化 DynamoDB 客户端时，将提供 DynamoDB 表的 ARN，而不是表名。这是必需的，这样 Lambda 函数才能在运行时连接到正确的 DynamoDB 表。

在 AWS 上的 SQL Server 的“始终打开”可用性组中配置只读路由

由 Subhani Shaik (AWS) 编写

摘要

此模式介绍如何在 SQL Server“始终打开”中使用备用辅助副本，方法是将只读工作负载从主副本卸载到辅助副本。

数据库镜像具有 one-to-one映射。您无法直接读取辅助数据库，因此必须创建快照。“始终打开”可用性组功能是在 Microsoft SQL Server 2012 中引入的。在后来的版本中，引入了主要功能，包括只读路由。在“始终打开”可用性组中，您可以通过将副本模式更改为只读来直接从辅助副本读取数据。

“始终打开”可用性组解决方案支持高可用性（HA）、灾难恢复（DR）和数据库镜像的替代方案。“始终打开”可用性组在数据库级别工作，可最大限度地提高一组用户数据库的可用性。

SQL Server 使用只读路由机制将传入的只读连接重定向到辅助只读副本。为此，应在连接字符串中添加以下参数和值：

- `ApplicationIntent=ReadOnly`
- `Initial Catalog=<database name>`

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account，拥有一个虚拟私有云（VPC）、两个可用区、私有子网和一个安全组
- 两台亚马逊弹性计算云 (Amazon EC2) 计算机在实例级别配置了 SQL Server 2019 企业版亚马逊计算机映像和 Windows 服务器故障转移群集 (WSFC)，主节点 () 和辅助节点 (WSFCNODE1) 之间在 SQL Server 级别配置了 Always On 可用性组，它们属于名为 Microsoft 的 AWS 目录服务 AWS Directory Active Directory 目录 `WSFCNODE2 tagechtalk.com`
- 配置的一个或多个节点，用于在辅助副本中接受 `read-only`
- 名为 `SQLAG1` 的侦听器，适用于“始终打开”可用性组
- SQL Server 数据库引擎，在两个节点上使用相同的服务账户运行
- 打开 SQL Server Management Studio (SSMS)
- 名为 `test` 的测试数据库

产品版本

- SQL Server 2014 及更高版本

架构

目标技术堆栈

- Amazon EC2
- AWS 托管的 Microsoft AD
- Amazon FSx

目标架构

下图显示了“始终打开”可用性组（AG）侦听器如何将连接中包含该 `ApplicationIntent` 参数的查询重定向到相应的辅助节点。

1. 将向“始终打开”可用性组侦听器发送请求。
2. 如果连接字符串没有 `ApplicationIntent` 参数，该请求会发送到主实例。
3. 如果连接字符串包含 `ApplicationIntent=ReadOnly`，则请求将发送到具有只读路由配置的辅助实例，即具有“始终打开”可用性组的 WSFC。

工具

Amazon Web Services

- 适用于 Microsoft Active Directory 的 AWS Directory Service 允许目录感知工作负载和 AWS 资源使用 Amazon Web Services Cloud 中的 Microsoft Active Directory。
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- Amazon FSx 提供的文件系统支持行业标准的连接协议，并可在 AWS 区域之间实现高可用性和复制。

其他服务

- SQL Server Management Studio（SSMS）是用于连接、管理和控制 SQL Server 实例的工具。
- sqlcmd 是一个命令行实用程序。

最佳实践

有关“始终打开”可用性组的更多信息，请参阅 SQL Server 文档。

操作说明

设置只读路由

Task	描述	所需技能
将副本更新为只读的。	要将主副本和辅助副本更新为只读，请从 SSMS 连接到主副本，然后运行其他信息部分中的步骤 1 代码。	数据库管理员
创建路由 URL。	要为两个副本创建路由 URL，请运行其他信息部分中的步骤 2 代码。此代码中，`tagechtalk.com` 是 AWS 托管的 Microsoft AD 目录的名称。	数据库管理员
创建路由列表。	要为两个副本创建路由列表，请运行其他信息部分中的步骤 3 代码。	数据库管理员
验证路由列表。	从 SQL Server Management Studio 连接到主实例，然后运行其他信息部分中的步骤 4 代码来验证路由列表。	数据库管理员

测试只读路由

Task	描述	所需技能
使用 ApplicationIntent 参数连接。	1. 从 SSMS 中，使用 `ApplicationIntent=ReadOnly;Initial Catalog=test` 连接到“始终打开”可用性组侦听器名称。 2. 与辅助副本的连接已建立。要进行测试，请运行以下命令，显示连接的服务器名称。 `SELECT SERVERPROPERTY('ComputerNamePhysicalNetBios')` 输出将显示当前的辅助副本名称（`WSFCNODE2`）。	数据库管理员
执行失效转移。	1. 从 SSMS 中，连接到“始终打开”可用性组侦听器名称。 2. 验证主数据库和辅助数据库是否同步，且没有数据丢失。 3. 执行失效转移，使当前的主副本成为辅助副本，辅助副本成为主副本。 4. 从 SSMS 中，使用 `ApplicationIntent=ReadOnly;Initial Catalog=test` 连接到“始终打开”可用性组侦听器名称。	数据库管理员

Task	描述	所需技能
	5. 与辅助副本的连接已建立。要对此进行测试，请通过运行以下命令，显示连接的服务器名称。 `SELECT SERVERPROPERTY('ComputerNamePhysicalNetBios')` 它将显示当前的辅助副本名称（WSFCNODE1 ）。	

使用 sqlcmd 命令行实用程序进行连接

Task	描述	所需技能
使用 sqlcmd 进行连接。	要从 sqlcmd 进行连接，请在命令提示符下运行其他信息部分中的步骤 5 代码。在连接后，请运行以下命令，显示连接的服务器名称。 `SELECT SERVERPROPERTY('ComputerNamePhysicalNetBios') .` 输出将显示当前的辅助副本名称 (WSFCNODE1)。	数据库管理员

故障排除

事务	解决方案
创建侦听器失败，并显示消息“WSFC 集群无法使网络名称资源联机”。	有关信息，请参阅 Microsoft 博客文章创建侦听器失败，并显示消息“WSFC 集群无法使网络名称资源联机”。
潜在问题，包括其他侦听器问题或网络访问问题。	请参阅 Microsoft 文档中的对“始终打开”可用性组配置（SQL Server）进行故障排除。

相关资源

- 在“始终打开”可用性组中配置只读路由
- 对“始终打开”可用性组配置（SQL Server）进行故障排除

其他信息

第 1 步：将副本更新为只读的

```
ALTER AVAILABILITY GROUP [SQLAG1] MODIFY REPLICA ON N'WSFCNODE1' WITH (SECONDARY_ROLE
 (ALLOW_CONNECTIONS = READ_ONLY))
GO
ALTER AVAILABILITY GROUP [SQLAG1] MODIFY REPLICA ON N'WSFCNODE2' WITH (SECONDARY_ROLE
 (ALLOW_CONNECTIONS = READ_ONLY))
GO
```

第 2 步：创建路由 URL

```
ALTER AVAILABILITY GROUP [SQLAG1] MODIFY REPLICA ON N'WSFCNODE1' WITH (SECONDARY_ROLE
 (READ_ONLY_ROUTING_URL = N'TCP://WSFCNode1.tagechtalk.com:1433'))
GO
ALTER AVAILABILITY GROUP [SQLAG1] MODIFY REPLICA ON N'WSFCNODE2' WITH (SECONDARY_ROLE
 (READ_ONLY_ROUTING_URL = N'TCP://WSFCNode2.tagechtalk.com:1433'))
GO
```

第 3 步：创建路由列表

```
ALTER AVAILABILITY GROUP [SQLAG1] MODIFY REPLICA ON N'WSFCNODE1' WITH
 (PRIMARY_ROLE(READ_ONLY_ROUTING_LIST=('WSFCNODE2','WSFCNODE1')));
GO
ALTER AVAILABILITY GROUP [SQLAG1] MODIFY REPLICA ON N'WSFCNODE2' WITH (PRIMARY_ROLE
 (READ_ONLY_ROUTING_LIST=('WSFCNODE1','WSFCNODE2')));
GO
```

第 4 步：验证路由列表

```
SELECT AGSrc.replica_server_name AS PrimaryReplica, AGRepl.replica_server_name AS
 ReadOnlyReplica, AGRepl.read_only_routing_url AS RoutingURL , AGRL.routing_priority
 AS RoutingPriority FROM sys.availability_read_only_routing_lists AGRL INNER JOIN
 sys.availability_replicas AGSrc ON AGRL.replica_id = AGSrc.replica_id INNER JOIN
 sys.availability_replicas AGRepl ON AGRL.read_only_replica_id = AGRepl.replica_id
 INNER JOIN sys.availability_groups AV ON AV.group_id = AGSrc.group_id ORDER BY
 PrimaryReplica
```

第 5 步。SQL 命令实用程序

```
sqlcmd -S SQLAG1,1433 -E -d test -K ReadOnly
```

在 pgAdmin 中使用 SSH 隧道进行连接

由 Jeevan Shetty (AWS) 和 Bhanu Ganesh Gudivada (AWS) 编写

摘要

出于安全考虑，最好将数据库放在私有子网中。通过亚马逊云网络服务 (AWS) 云上公有子网中的亚马逊弹性计算云 (Amazon EC2) 堡垒主机进行连接，即可对数据库进行查询。这需要在 Amazon 主机上安装开发人员 DBeaver或数据库管理员常用的软件，例如 pgadmin 或。 EC2

在 Linux 服务器上运行 pgAdmin 并通过网络浏览器访问它，需要安装其他依赖项、权限设置和配置。

作为替代解决方案，开发人员或数据库管理员可以使用 pgAdmin 从其本地系统启用 SSH 隧道，从而连接到 PostgreSQL 数据库。在这种方法中，pgadmin 在连接到数据库之前使用公有子网中的 EC2 Amazon 主机作为中间主机。架构部分的图表显示了设置情况。

> **Note**
>
> 确保连接到 PostgreSQL 数据库的安全组允许通过端口 5432 从亚马逊主机进行连接。 EC2

先决条件和限制

先决条件

- 现有 Amazon Web Services account
- 具有公有子网和私有子网的虚拟私有云（VPC）
- 附加了安全组的 EC2 实例
- 附加了安全组的 Amazon Aurora PostgreSQL 兼容版数据库
- 用于设置隧道的 Secure Shell (SSH) 密钥对

产品版本

- pgAdmin 版本 6.2+
- Amazon Aurora PostgreSQL 兼容版版本 12.7+

架构

目标技术堆栈

- Amazon EC2
- 兼容 Amazon Aurora PostgreSQL

目标架构

下图显示了使用带有 SSH 隧道的 pgadmin 通过互联网网关连接到连接到数据库 EC2 的实例。

工具

Amazon Web Services

- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩缩 PostgreSQL 部署。
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。

其他服务

- pgAdmin 是一种适用于 PostgreSQL 的开源管理工具。它提供了一个图形界面，可帮助您创建、维护和使用数据库对象。

操作说明

创建连接

Task	描述	所需技能
创建一个服务器。	在 pgAdmin 中，选择创建，然后选择服务器。有关设置 pgAdmin 以注册服务器、配置连接以及使用服务器对话框通过 SSH 隧道进行连接的其他帮助，请参阅相关资源部分中的链接。	数据库管理员
为服务器提供一个名称。	在常规选项卡上，输入名称。	数据库管理员

Task	描述	所需技能
输入数据库详细信息。	在连接选项卡上，输入以下各项的值： • 主机名/地址 • 端口 • 维护数据库 • 用户名 • 密码	数据库管理员
输入 Amazon EC2 服务器的详细信息。	在 SSH 隧道选项卡上，提供公有子网中的 Amazon EC2 实例的详细信息。 • 将使用 SSH 隧道设置为是，以指定 pgAdmin 在连接到指定服务器时应使用 SSH 隧道。 • 在隧道主机字段中，指定 SSH 主机的名称或 IP 地址（例如，10.x.x.x）。 • 在隧道端口字段中，指定 SSH 主机的端口（例如 22）。 • 在用户名字段中，指定具有 SSH 主机登录权限的用户名（例如，ec2-user）。 • 将身份验证类型指定为身份文件，以便 pgAdmin 在连接时使用私钥文件。 • 在身份文件字段中包括隐私增强邮件 (PEM) 文件的位置。.pem 文件是亚马逊 EC2 密钥对。	数据库管理员

Task	描述	所需技能
保存并连接。	选择保存以完成设置并使用 SSH 隧道连接到与 Aurora PostgreSQL 兼容的数据库。	数据库管理员

相关的资源

- 服务器对话框
- 连接到服务器

将 JSON Oracle 查询转换至 PostgreSQL 数据库 SQL

由 Pinesh Singal (AWS) 和 Lokesh Gurram (AWS) 编写

摘要

从本地迁移至 Amazon Web Services (AWS) Cloud 的迁移过程使用 AWS Schema Conversion Tool (AWS SCT) 将代码从 Oracle 数据库转换为 PostgreSQL 数据库。大部分代码价格 AWS SCT 自动转换。但是，与 JSON 相关的 Oracle 查询不自动转换。

从 Oracle 12.2 版开始，Oracle 数据库支持各种 JSON 函数，这些函数有助于将基于 JSON 的数据转换为基于行的数据。但是，AWS SCT 不会自动将基于 JSON 的数据转换至 PostgreSQL 支持的语言。

这种迁移模式主要侧重于手动使用 `JSON_OBJECT`、`JSON_ARRAYAGG` 和 `JSON_TABLE` 等函数将 JSON 相关的 Oracle 查询从 Oracle 数据库转换为 PostgreSQL 数据库。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 本地 Oracle 数据库实例（已启动并正在运行）
- Amazon Relational Database Service (Amazon RDS) for PostgreSQL 或Amazon Aurora PostgreSQL-Compatible Edition 数据库实例（已启动并运行）

限制

- 与 JSON 相关的查询需要固定的 `KEY` 和 `VALUE` 格式 不使用此格式会返回错误的结果。
- 如果 JSON 结构的任何更改在结果节中添加了新的 `KEY` 和 `VALUE` 对，则必须在 SQL 查询中更改相应的过程或函数。
- 早期版本的 Oracle 和 PostgreSQL 支持部分与 JSON 相关的函数，但功能较少。

产品版本

- Oracle 数据库版本 12.2 及更高版本
- Amazon RDS for PostgreSQL 或 Aurora PostgreSQL-Compatible 版本 9.5 和更高版本
- AWS SCT 最新版本（使用版本 1.0.664 进行了测试）

架构

源技术堆栈

- 19c 版本的 Oracle 数据库实例

目标技术堆栈

- Amazon RDS for PostgreSQL 或 Aurora PostgreSQL-Compatible 数据库实例，版本 13

目标架构

1. 使用带有 JSON 函数代码的 AWS SCT 将源代码从 Oracle 转换至 PostgreSQL。
2. 转换生成 PostgreSQL 支持的已迁移的 .sql 文件。
3. 手动将未转换的 Oracle JSON 函数代码转换至 PostgreSQL JSON 函数代码。
4. 在兼容 Aurora PostgreSQL 的目标数据库实例上运行 .sql 文件。

工具

Amazon Web Services

- Amazon Aurora 是与 MySQL 和 PostgreSQL 兼容的完全托管式的云端关系数据库引擎。
- Amazon Relational Database Service（Amazon RDS）for PostgreSQL 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展PostgreSQL 关系数据库。
- AWS Schema Conversion Tool（AWS SCT）通过自动将源数据库架构和大部分自定义代码转换为与目标数据库兼容的格式来支持异构数据库迁移。

其他服务

- Oracle SQL Developer 是一个集成的开发环境，可简化传统部署和基于云的部署中 Oracle 数据库的开发和管理。
- pgadmin 或 DBeaver。 pgadmin 是一款适用于 PostgreSQL 的开源管理工具。它提供了一个图形界面，可帮助您创建、维护和使用数据库对象。 DBeaver是一个通用的数据库工具。

最佳实践

使用 JSON_TABLE 函数时，Oracle 查询的默认类型为 CAST 类型。最佳实践是在 PostgreSQL 中也使用 CAST，使用双倍大于字符 (>>)。

有关更多信息，请参阅其他信息部分中的 Postgres_SQL_Read_JSON。

操作说明

在 Oracle 和 PostgreSQL 数据库生成 JSON 数据

Task	描述	所需技能
将 JSON 数据存储至 Oracle 数据库中。	在 Oracle 数据库中创建表，并在 CLOB 列中存储 JSON 数据。 使用其他信息部分的 Oracle_Table_Creation_Insert_Script。	迁移工程师
将 JSON 数据存储至 PostgreSQL 数据库中。	在 PostgreSQL 数据库中创建一个表，并在TEXT列中存储 JSON 数据。使用其他信息部分的 Postgres_Table_Creation_Insert_Script。	迁移工程师

将 JSON 转换为 ROW 格式

Task	描述	所需技能
转换 Oracle 数据库的 JSON 数据。	编写 Oracle SQL 查询，将 JSON 数据读取至 ROW 格式。有关更多详细信息和示例语法，请参阅其他信息部分中的 Oracle_SQL_Read_JSON。	迁移工程师
转换 PostgreSQL 数据库中的 JSON 数据。	编写 PostgreSQL 查询，将 JSON 数据读取至 ROW 格	迁移工程师

Task	描述	所需技能
	式。有关更多详细信息和示例语法，请参阅其他信息部分中的Postgres_SQL_Read_JSON。	

使用 SQL 查询手动转换 JSON 数据，并以 JSON 格式报告输出

Task	描述	所需技能
对 Oracle SQL 查询执行聚合与验证。	若要手动转换 JSON 数据，请对 Oracle SQL 查询执行联接、聚合和验证，并以 JSON 格式报告输出。使用其他信息部分的 Oracle_SQL_JSON_Aggregation_Join 中的代码。 1. JOIN — JSON 格式数据作为输入参数传递给查询。在此静态数据和 Oracle 数据库表 `aws_test_table` 中的 JSON 数据之间建立了一个内部联接。 2. 验证聚合 - JSON 数据具有 KEY 和 VALUE 带有值的参数，值例如 `accountNumber` 、`parentAccountNumber` 、`businessUnitId` 、`positionId` ，用于 SUM 和 COUNT 聚合。 3. JSON 格式 - 在联接和聚合之后，使用 JSON_OBJECT	迁移工程师

Task	描述	所需技能
	和 JSON_ARRAYAGG 以 JSON 格式报告数据。	
对 Postgres SQL 查询执行聚合与验证。	若要手动转换 JSON 数据，请对 Postgres SQL 查询执行联接、聚合和验证，并以 JSON 格式报告输出。使用其他信息部分的 Postgres_SQL_JSON_Aggregation_Join 中的代码。 1. JOIN — JSON 格式数据 (tab1) 作为输入参数传递给 WITH 子句查询。在此静态数据和表 tab 中的 JSON 数据之间建立了一个联接。还使用 WITH 子句进行联接，该子句在 aws_test_pg_table 表中有 JSON 数据。 2. 聚合 - JSON 数据具有 KEY 和 VALUE 带有值的参数，值例如 accountNumber 、parentAccountNumber 、businessUnitId 和 positionId ，用于 SUM 和 COUNT 聚合。 3. JSON 格式 - 在联接和聚合之后，使用 JSON_BUILD_OBJECT 和 JSON_AGG 以 JSON 格式报告数据。	迁移工程师

将 Oracle 程序转换为包含 JSON 查询的 PostgreSQL 函数

Task	描述	所需技能
将 Oracle 程序中的 JSON 查询转换为行。	对于 Oracle 程序示例，使用其他信息部分的 Oracle_procedure_with_JSON_Query 中的 Oracle 查询和代码。	迁移工程师
将具有 JSON 查询的 PostgreSQL 函数转换至基于行的数据。	对于 PostgreSQL 函数示例，使用之前的 PostgreSQL 查询以及其他信息部分的 Postgres_function_with_JSON_Query 中的代码。	迁移工程师

相关资源

- Oracle JSON 函数
- PostgreSQL JSON 函数
- 甲骨文 JSON 函数示例
- PostgreSQL JSON 函数示例
- AWS Schema Conversion Tool

其他信息

要将 JSON 代码从 Oracle 数据库转换至 PostgreSQL 数据库，请按顺序使用以下脚本。

1。Oracle_Table_Creation_Insert_Script

```
create table aws_test_table(id number,created_on date default sysdate,modified_on
 date,json_doc clob);

REM INSERTING into EXPORT_TABLE
SET DEFINE OFF;
Insert into aws_test_table (ID,CREATED_ON,MODIFIED_ON,json_doc)
values (1,to_date('02-AUG-2022 12:30:14','DD-MON-YYYY HH24:MI:SS'),to_date('02-AUG-2022
 12:30:14','DD-MON-YYYY HH24:MI:SS'),TO_CLOB(q'[{
  "metadata" : {
```

```
    "upperLastNameFirstName" : "ABC XYZ",
    "upperEmailAddress" : "abc@gmail.com",
    "profileType" : "P"
  },
  "data" : {
    "onlineContactId" : "032323323",
    "displayName" : "Abc, Xyz",
    "firstName" : "Xyz",
    "lastName" : "Abc",
    "emailAddress" : "abc@gmail.com",
    "productRegistrationStatus" : "Not registered",
    "positionId" : "0100",
    "arrayPattern" : " -'",
    "a]')
|| TO_CLOB(q'[ccount" : {
      "companyId" : "SMGE",
      "businessUnitId" : 7,
      "accountNumber" : 42000,
      "parentAccountNumber" : 32000,
      "firstName" : "john",
      "lastName" : "doe",
      "street1" : "retOdertcaShr ",
      "city" : "new york",
      "postalcode" : "XY ABC",
      "country" : "United States"
    },
    "products" : [
      {
        "appUserGuid" : "i0acc4450000001823fbad478e2eab8a0",
        "id" : "0000000046",
]')
|| TO_CLOB(q'[        "name" : "ProView",
        "domain" : "EREADER",
        "registrationStatus" : false,
        "status" : "11"
      }
    ]
  }
}]'));
Insert into aws_test_table (ID,CREATED_ON,MODIFIED_ON,json_doc) values (2,to_date('02-
AUG-2022 12:30:14','DD-MON-YYYY HH24:MI:SS'),to_date('02-AUG-2022 12:30:14','DD-MON-
YYYY HH24:MI:SS'),TO_CLOB(q'[{
  "metadata" : {
    "upperLastNameFirstName" : "PQR XYZ",
```

```
    "upperEmailAddress" : "pqr@gmail.com",
    "profileType" : "P"
  },
  "data" : {
    "onlineContactId" : "54534343",
    "displayName" : "Xyz, pqr",
    "firstName" : "pqr",
    "lastName" : "Xyz",
    "emailAddress" : "pqr@gmail.com",
    "productRegistrationStatus" : "Not registered",
    "positionId" : "0090",
    "arrayPattern" : " -'",
    "account" : {
      "companyId" : "CARS",
      "busin]')
|| TO_CLOB(q'[essUnitId" : 6,
      "accountNumber" : 42001,
      "parentAccountNumber" : 32001,
      "firstName" : "terry",
      "lastName" : "whitlock",
      "street1" : "UO  123",
      "city" : "TOTORON",
      "region" : "NO",
      "postalcode" : "LKM 111",
      "country" : "Canada"
    },
    "products" : [
      {
        "appUserGuid" : "ia744d7790000016899f8cf3f417d6df6",
        "id" : "0000000014",
        "name" : "ProView eLooseleaf",
      ]')
|| TO_CLOB(q'[  "domain" : "EREADER",
        "registrationStatus" : false,
        "status" : "11"
      }
    ]
  }
}]'));

commit;
```

2。Postgres_Table_Creation_Insert_Script

```
create table aws_test_pg_table(id int,created_on date ,modified_on date,json_doc text);
insert into aws_test_pg_table(id,created_on,modified_on,json_doc)
values(1,now(),now(),'{
  "metadata" : {
    "upperLastNameFirstName" : "ABC XYZ",
    "upperEmailAddress" : "abc@gmail.com",
    "profileType" : "P"
  },
  "data" : {
    "onlineContactId" : "032323323",
    "displayName" : "Abc, Xyz",
    "firstName" : "Xyz",
    "lastName" : "Abc",
    "emailAddress" : "abc@gmail.com",
    "productRegistrationStatus" : "Not registered",
    "positionId" : "0100",
    "arrayPattern" : " -",
    "account" : {
      "companyId" : "SMGE",
      "businessUnitId" : 7,
      "accountNumber" : 42000,
      "parentAccountNumber" : 32000,
      "firstName" : "john",
      "lastName" : "doe",
      "street1" : "retOdertcaShr ",
      "city" : "new york",
      "postalcode" : "XY ABC",
      "country" : "United States"
    },
    "products" : [
      {
        "appUserGuid" : "i0acc4450000001823fbad478e2eab8a0",
        "id" : "0000000046",
        "name" : "ProView",
        "domain" : "EREADER",
        "registrationStatus" : false,
        "status" : "11"
      }
    ]
  }
}');
```

```
insert into aws_test_pg_table(id,created_on,modified_on,json_doc)
values(2,now(),now(),'{
  "metadata" : {
    "upperLastNameFirstName" : "PQR XYZ",
    "upperEmailAddress" : "pqr@gmail.com",
    "profileType" : "P"
  },
  "data" : {
    "onlineContactId" : "54534343",
    "displayName" : "Xyz, pqr",
    "firstName" : "pqr",
    "lastName" : "Xyz",
    "emailAddress" : "a*b**@h**.k**",
    "productRegistrationStatus" : "Not registered",
    "positionId" : "0090",
    "arrayPattern" : " -",
    "account" : {
      "companyId" : "CARS",
      "businessUnitId" : 6,
      "accountNumber" : 42001,
      "parentAccountNumber" : 32001,
      "firstName" : "terry",
      "lastName" : "whitlock",
      "street1" : "UO  123",
      "city" : "TOTORON",
      "region" : "NO",
      "postalcode" : "LKM 111",
      "country" : "Canada"
    },
    "products" : [
      {
        "appUserGuid" : "ia744d7790000016899f8cf3f417d6df6",
        "id" : "0000000014",
        "name" : "ProView eLooseleaf",
        "domain" : "EREADER",
        "registrationStatus" : false,
        "status" : "11"
      }
    ]
  }
}');
```

3。Oracle_SQL_Read_JSON

以下代码块显示了如何将 Oracle JSON 数据转换为行格式。

查询和语法示例

```
SELECT   JSON_OBJECT(
 'accountCounts' VALUE JSON_ARRAYAGG(
           JSON_OBJECT(
               'businessUnitId' VALUE business_unit_id,
                       'parentAccountNumber' VALUE parent_account_number,
                       'accountNumber' VALUE account_number,
                       'totalOnlineContactsCount' VALUE online_contacts_count,
                       'countByPosition' VALUE
                   JSON_OBJECT(
                       'taxProfessionalCount' VALUE tax_count,
                       'attorneyCount' VALUE attorney_count,
                       'nonAttorneyCount' VALUE non_attorney_count,
                       'clerkCount' VALUE clerk_count
                              ) ) ) ) FROM
   (SELECT   tab_data.business_unit_id,
           tab_data.parent_account_number,
           tab_data.account_number,
           SUM(1) online_contacts_count,
           SUM(CASE WHEN tab_data.position_id = '0095' THEN  1 ELSE 0 END) tax_count,
           SUM(CASE    WHEN tab_data.position_id = '0100' THEN 1 ELSE 0 END)
 attorney_count,
           SUM(CASE    WHEN tab_data.position_id = '0090' THEN 1 ELSE 0 END)
 non_attorney_count,
           SUM(CASE    WHEN tab_data.position_id = '0050' THEN 1 ELSE 0 END)
 clerk_count
       FROM aws_test_table scco,JSON_TABLE ( json_doc, '$' ERROR ON ERROR
  COLUMNS (
         parent_account_number NUMBER PATH
          '$.data.account.parentAccountNumber',
          account_number NUMBER PATH '$.data.account.accountNumber',
          business_unit_id NUMBER PATH '$.data.account.businessUnitId',
          position_id VARCHAR2 ( 4 ) PATH '$.data.positionId'    )
          ) AS tab_data
          INNER JOIN JSON_TABLE ( '{
       "accounts": [{
         "accountNumber": 42000,
         "parentAccountNumber": 32000,
         "businessUnitId": 7
       }, {
         "accountNumber": 42001,
```

```
          "parentAccountNumber": 32001,
          "businessUnitId": 6
        }]
      }', '$.accounts[*]' ERROR ON ERROR
      COLUMNS (
      parent_account_number PATH '$.parentAccountNumber',
      account_number PATH '$.accountNumber',
      business_unit_id PATH '$.businessUnitId')
      ) static_data
      ON ( static_data.parent_account_number = tab_data.parent_account_number
           AND static_data.account_number = tab_data.account_number
           AND static_data.business_unit_id = tab_data.business_unit_id )
        GROUP BY
            tab_data.business_unit_id,
            tab_data.parent_account_number,
            tab_data.account_number );
```

JSON 文档将数据存储为集合。每个集合可以有 KEY 和 VALUE 对。每个 VALUE 都可以有嵌套 KEY 和 VALUE 对。下表提供了有关从 JSON 文档中读取特定 VALUE 的信息。

键	用于获取值的层次结构或者路径	值
profileType	metadata -> profileType	"P"
positionId	data -> positionId	"0100"
accountNumber	data -> 账户 -> accountNumber	42000

在上表中，KEY profileType 是 metadata KEY 中的 VALUE。KEY positionId 是 data KEY 中的 VALUE。KEY accountNumber 是 account KEY 中的 VALUE，account KEY 是 data KEY 中的 VALUE。

JSON 文档示例

```
{
  "metadata" : {
    "upperLastNameFirstName" : "ABC XYZ",
    "upperEmailAddress" : "abc@gmail.com",
```

```
"profileType" : "P"
  },
  "data" : {
    "onlineContactId" : "032323323",
    "displayName" : "Abc, Xyz",
    "firstName" : "Xyz",
    "lastName" : "Abc",
    "emailAddress" : "abc@gmail.com",
    "productRegistrationStatus" : "Not registered",
"positionId" : "0100",
    "arrayPattern" : " -",
    "account" : {
      "companyId" : "SMGE",
      "businessUnitId" : 7,
"accountNumber" : 42000,
      "parentAccountNumber" : 32000,
      "firstName" : "john",
      "lastName" : "doe",
      "street1" : "retOdertcaShr ",
      "city" : "new york",
      "postalcode" : "XY ABC",
      "country" : "United States"
    },
    "products" : [
      {
        "appUserGuid" : "i0acc4450000001823fbad478e2eab8a0",
        "id" : "0000000046",
        "name" : "ProView",
        "domain" : "EREADER",
        "registrationStatus" : false,
        "status" : "11"
      }
    ]
  }
}
```

用于从 JSON 文档中获取选定字段的 SQL 查询

```
select parent_account_number,account_number,business_unit_id,position_id from
 aws_test_table aws,JSON_TABLE ( json_doc, '$' ERROR ON ERROR
COLUMNS (
parent_account_number NUMBER PATH '$.data.account.parentAccountNumber',
account_number NUMBER PATH '$.data.account.accountNumber',
```

```
business_unit_id NUMBER PATH '$.data.account.businessUnitId',
position_id VARCHAR2 ( 4 ) PATH '$.data.positionId'
)) as sc
```

在前面的查询中，JSON_TABLE 是 Oracle 中的一个内置函数，用于将 JSON 数据转换为行格式。JSON_TABLE 函数要求参数使用 JSON 格式。

COLUMNS 中的每个项目都有预定义的 PATH，并且会以行格式返回与给定 KEY 对应的 VALUE。

上一次查询结果

PARENT_ACCOUNT_NUMBER	ACCOUNT_NUMBER	BUSINESS_UNIT_ID	POSITION_ID
32000	42000	7	0100
32001	42001	6	0090

4。Postgres_SQL_Read_JSON

查询和语法示例

```
select *
from (
select (json_doc::json->'data'->'account'->>'parentAccountNumber')::INTEGER as
 parentAccountNumber,
(json_doc::json->'data'->'account'->>'accountNumber')::INTEGER as accountNumber,
(json_doc::json->'data'->'account'->>'businessUnitId')::INTEGER as businessUnitId,
(json_doc::json->'data'->>'positionId')::VARCHAR as positionId
from aws_test_pg_table) d ;
```

在 Oracle 中，PATH 用于标识特定 KEY 和 VALUE。但是，PostgreSQL 使用 HIERARCHY 模型从 JSON 读取 KEY 和 VALUE。以下示例中使用了 Oracle_SQL_Read_JSON 下所述的相同的 JSON 数据。

不允许使用 CAST 类型 SQL 查询

(如果您强制输入 CAST，则查询会因语法错误而失败。)

```
select *
from (
```

```
select (json_doc::json->'data'->'account'->'parentAccountNumber') as
 parentAccountNumber,
(json_doc::json->'data'->'account'->'accountNumber')as accountNumber,
(json_doc::json->'data'->'account'->'businessUnitId') as businessUnitId,
(json_doc::json->'data'->'positionId')as positionId
from aws_test_pg_table) d ;
```

使用单个大于运算符 (>) 将返回为此 KEY 定义的 VALUE。例如，KEY: positionId 和 VALUE: "0100"。

当您使用单个大于号运算符 (>) 时，不允许使用类型 CAST。

允许使用 CAST 类型 SQL 查询

```
select *
from (
select (json_doc::json->'data'->'account'->>'parentAccountNumber')::INTEGER as
 parentAccountNumber,
(json_doc::json->'data'->'account'->>'accountNumber')::INTEGER as accountNumber,
(json_doc::json->'data'->'account'->>'businessUnitId')::INTEGER as businessUnitId,
(json_doc::json->'data'->>'positionId')::varchar as positionId
from aws_test_pg_table) d ;
```

要使用类型 CAST，必须使用双精度大于运算符。如果您使用单个大于运算符，则查询将返回为此定义的 VALUE（例如：和 KEY: positionId 和 VALUE: "0100"）。使用双精度大于运算符 (>>) 将返回为此定义的 KEY 实际值（例如 KEY:positionId 和 VALUE:0100，不带双引号）。

在前述例子中，parentAccountNumber 是类型 CAST 到 INT ，accountNumber 是类型 CAST 到 INT，businessUnitId 是类型 CAST 到INT，positionId 是类型 CAST 到 VARCHAR。

下表显示的查询结果解释了单个大于运算符 (>) 和双大于运算符 (>>) 的作用。

在第一个表中，查询使用单个大于运算符 (>) 。每列均采用 JSON 类型，无法转换至其他数据类型。

parentAccountNumber	accountNumber	businessUnitId	positionId
2003565430	2003564830	7	“0100”
2005284042	2005284042	6	“0090”
2000272719	2000272719	1	“0100”

在第二个表中，查询使用双精度大于运算符 (>>) 。每列都支持基于列值的 CAST 类型。例如，在此上下文中的 INTEGER。

parentAccountNumber	accountNumber	businessUnitId	positionId
2003565430	2003564830	7	0100
2005284042	2005284042	6	0090
2000272719	2000272719	1	0100

5。Oracle_SQL_JSON_Aggregation_Join

示例查询

```
SELECT
    JSON_OBJECT(
        'accountCounts' VALUE JSON_ARRAYAGG(
            JSON_OBJECT(
                'businessUnitId' VALUE business_unit_id,
                        'parentAccountNumber' VALUE parent_account_number,
                        'accountNumber' VALUE account_number,
                        'totalOnlineContactsCount' VALUE online_contacts_count,
                        'countByPosition' VALUE
                    JSON_OBJECT(
                        'taxProfessionalCount' VALUE tax_count,
                        'attorneyCount' VALUE attorney_count,
                        'nonAttorneyCount' VALUE non_attorney_count,
                        'clerkCount' VALUE clerk_count
                               ) ) ) )
FROM
    (SELECT
            tab_data.business_unit_id,
            tab_data.parent_account_number,
            tab_data.account_number,
            SUM(1) online_contacts_count,
            SUM(CASE WHEN tab_data.position_id = '0095' THEN  1 ELSE 0 END) tax_count,
            SUM(CASE    WHEN tab_data.position_id = '0100' THEN 1 ELSE 0 END)
 attorney_count,
            SUM(CASE    WHEN tab_data.position_id = '0090' THEN 1 ELSE 0 END)
 non_attorney_count,
```

```
            SUM(CASE    WHEN tab_data.position_id = '0050' THEN 1 ELSE 0 END)
clerk_count
       FROM aws_test_table scco,JSON_TABLE ( json_doc, '$' ERROR ON ERROR
 COLUMNS (
         parent_account_number NUMBER PATH
          '$.data.account.parentAccountNumber',
           account_number NUMBER PATH '$.data.account.accountNumber',
           business_unit_id NUMBER PATH '$.data.account.businessUnitId',
           position_id VARCHAR2 ( 4 ) PATH '$.data.positionId'    )
           ) AS tab_data
           INNER JOIN JSON_TABLE ( '{
       "accounts": [{
         "accountNumber": 42000,
         "parentAccountNumber": 32000,
         "businessUnitId": 7
       }, {
         "accountNumber": 42001,
         "parentAccountNumber": 32001,
         "businessUnitId": 6
       }]
     }', '$.accounts[*]' ERROR ON ERROR
     COLUMNS (
     parent_account_number PATH '$.parentAccountNumber',
     account_number PATH '$.accountNumber',
     business_unit_id PATH '$.businessUnitId')
     ) static_data
     ON ( static_data.parent_account_number = tab_data.parent_account_number
          AND static_data.account_number = tab_data.account_number
          AND static_data.business_unit_id = tab_data.business_unit_id )
       GROUP BY
           tab_data.business_unit_id,
           tab_data.parent_account_number,
           tab_data.account_number
   );
```

为了将行级数据转换至 JSON 格式，Oracle 内置了函数，例如JSON_OBJECT、JSON_ARRAY、JSON_OBJECTAGG 和 JSON_ARRAYAGG。

- JSON_OBJECT 接受两个参数：KEY 和 VALUE。该 KEY 参数本质上应该是硬编码的或静态的。该 VALUE 参数源自表输出。
- JSON_ARRAYAGG 接受 JSON_OBJECT 作为参数。这有助于将一组 JSON_OBJECT 元素分组为一个列表。例如，如果 JSON_OBJECT 元素有多条记录 (数据集中有多条 KEY 和 VALUE

对)，JSON_ARRAYAGG 会追加数据集并创建一个列表。根据数据结构语言，LIST 是一组元素。在此上下文中，LIST 是一组 JSON_OBJECT 元素。

以下示例显示了一个 JSON_OBJECT 元素。

```
{
   "taxProfessionalCount": 0,
   "attorneyCount": 0,
   "nonAttorneyCount": 1,
   "clerkCount": 0
}
```

下一个示例显示两个 JSON_OBJECT 元素，用方括号 ([ ]) 表示 LIST。

```
[
    {
        "taxProfessionalCount": 0,
        "attorneyCount": 0,
        "nonAttorneyCount": 1,
        "clerkCount": 0
      }
,
    {
        "taxProfessionalCount": 2,
        "attorneyCount": 1,
        "nonAttorneyCount": 3,
        "clerkCount":4
      }
]
```

SQL 查询示例

```
SELECT
    JSON_OBJECT(
        'accountCounts' VALUE JSON_ARRAYAGG(
            JSON_OBJECT(
                'businessUnitId' VALUE business_unit_id,
                        'parentAccountNumber' VALUE parent_account_number,
                        'accountNumber' VALUE account_number,
                        'totalOnlineContactsCount' VALUE online_contacts_count,
                        'countByPosition' VALUE
                    JSON_OBJECT(
```

```
                        'taxProfessionalCount' VALUE tax_count,
                        'attorneyCount' VALUE attorney_count,
                        'nonAttorneyCount' VALUE non_attorney_count,
                        'clerkCount' VALUE clerk_count
                               )
                        )
                                        )
             )
FROM
    (SELECT
            tab_data.business_unit_id,
            tab_data.parent_account_number,
            tab_data.account_number,
            SUM(1) online_contacts_count,
            SUM(CASE WHEN tab_data.position_id = '0095' THEN  1 ELSE   0 END
            )      tax_count,
            SUM(CASE    WHEN tab_data.position_id = '0100' THEN        1    ELSE
  0 END
            )      attorney_count,

            SUM(CASE    WHEN tab_data.position_id = '0090' THEN        1    ELSE
  0 END
            )      non_attorney_count,

            SUM(CASE    WHEN tab_data.position_id = '0050' THEN        1    ELSE
  0 END
            )      clerk_count

        FROM
            aws_test_table scco,  JSON_TABLE ( json_doc, '$' ERROR ON ERROR
            COLUMNS (
            parent_account_number NUMBER PATH '$.data.account.parentAccountNumber',
            account_number NUMBER PATH '$.data.account.accountNumber',
            business_unit_id NUMBER PATH '$.data.account.businessUnitId',
            position_id VARCHAR2 ( 4 ) PATH '$.data.positionId'    )
            ) AS tab_data
            INNER JOIN JSON_TABLE ( '{
        "accounts": [{
          "accountNumber": 42000,
          "parentAccountNumber": 32000,
          "businessUnitId": 7
        }, {
          "accountNumber": 42001,
          "parentAccountNumber": 32001,
```

```
          "businessUnitId": 6
        }]
      }', '$.accounts[*]' ERROR ON ERROR
      COLUMNS (
      parent_account_number PATH '$.parentAccountNumber',
      account_number PATH '$.accountNumber',
      business_unit_id PATH '$.businessUnitId')
      ) static_data ON ( static_data.parent_account_number =
 tab_data.parent_account_number
                         AND static_data.account_number = tab_data.account_number

                         AND static_data.business_unit_id =
 tab_data.business_unit_id )
        GROUP BY
            tab_data.business_unit_id,
            tab_data.parent_account_number,
            tab_data.account_number
    );
```

上一个 SQL 查询输出示例

```
{
  "accountCounts": [
    {
      "businessUnitId": 6,
      "parentAccountNumber": 32001,
      "accountNumber": 42001,
      "totalOnlineContactsCount": 1,
      "countByPosition": {
        "taxProfessionalCount": 0,
        "attorneyCount": 0,
        "nonAttorneyCount": 1,
        "clerkCount": 0
      }
    },
    {
      "businessUnitId": 7,
      "parentAccountNumber": 32000,
      "accountNumber": 42000,
      "totalOnlineContactsCount": 1,
      "countByPosition": {
        "taxProfessionalCount": 0,
        "attorneyCount": 1,
```

```
        "nonAttorneyCount": 0,
        "clerkCount": 0
      }
    }
  ]
}
```

6。 Postgres_SQL_JSON_Aggregation_Join

PostgreSQL 内置函数 JSON_BUILD_OBJECT 和 JSON_AGG 将行级数据转换为 JSON 格式。 PostgreSQL JSON_BUILD_OBJECT 和 JSON_AGG 等同于 Oracle JSON_OBJECT 和 JSON_ARRAYAGG

示例查询

```
select
JSON_BUILD_OBJECT ('accountCounts',
    JSON_AGG(
        JSON_BUILD_OBJECT ('businessUnitId',businessUnitId
        ,'parentAccountNumber',parentAccountNumber
        ,'accountNumber',accountNumber
        ,'totalOnlineContactsCount',online_contacts_count,
        'countByPosition',
            JSON_BUILD_OBJECT (
            'taxProfessionalCount',tax_professional_count
            ,'attorneyCount',attorney_count
            ,'nonAttorneyCount',non_attorney_count
            ,'clerkCount',clerk_count
            )
        )
    )
)
from (
with tab as (select * from (
select (json_doc::json->'data'->'account'->>'parentAccountNumber')::INTEGER as
 parentAccountNumber,
(json_doc::json->'data'->'account'->>'accountNumber')::INTEGER as accountNumber,
(json_doc::json->'data'->'account'->>'businessUnitId')::INTEGER as businessUnitId,
(json_doc::json->'data'->>'positionId')::varchar as positionId
from aws_test_pg_table) a ) ,
tab1 as ( select
(json_array_elements(b.jc -> 'accounts') ->> 'accountNumber')::integer accountNumber,
```

```
(json_array_elements(b.jc -> 'accounts') ->> 'businessUnitId')::integer
 businessUnitId,
(json_array_elements(b.jc -> 'accounts') ->> 'parentAccountNumber')::integer
 parentAccountNumber
from (
select '{
        "accounts": [{
          "accountNumber": 42001,
          "parentAccountNumber": 32001,
          "businessUnitId": 6
        }, {
          "accountNumber": 42000,
          "parentAccountNumber": 32000,
          "businessUnitId": 7
        }]
      }'::json as jc) b)
select
tab.businessUnitId::text,
tab.parentAccountNumber::text,
tab.accountNumber::text,
SUM(1) online_contacts_count,
SUM(CASE WHEN tab.positionId::text = '0095' THEN 1 ELSE 0  END)
  tax_professional_count,
SUM(CASE WHEN tab.positionId::text = '0100' THEN 1 ELSE 0  END)      attorney_count,
SUM(CASE  WHEN tab.positionId::text = '0090' THEN      1  ELSE      0 END)
  non_attorney_count,
SUM(CASE  WHEN tab.positionId::text = '0050' THEN      1  ELSE      0 END)
  clerk_count
from tab1,tab
where tab.parentAccountNumber::INTEGER=tab1.parentAccountNumber::INTEGER
and tab.accountNumber::INTEGER=tab1.accountNumber::INTEGER
and tab.businessUnitId::INTEGER=tab1.businessUnitId::INTEGER
GROUP BY      tab.businessUnitId::text,
            tab.parentAccountNumber::text,
            tab.accountNumber::text) a;
```

前述查询的输出示例

Oracle 和 PostgreSQL 输出完全相同。

```
{
  "accountCounts": [
    {
      "businessUnitId": 6,
```

```
      "parentAccountNumber": 32001,
      "accountNumber": 42001,
      "totalOnlineContactsCount": 1,
      "countByPosition": {
        "taxProfessionalCount": 0,
        "attorneyCount": 0,
        "nonAttorneyCount": 1,
        "clerkCount": 0
      }
    },
    {
      "businessUnitId": 7,
      "parentAccountNumber": 32000,
      "accountNumber": 42000,
      "totalOnlineContactsCount": 1,
      "countByPosition": {
        "taxProfessionalCount": 0,
        "attorneyCount": 1,
        "nonAttorneyCount": 0,
        "clerkCount": 0
      }
    }
  ]
}
```

7.Oracle_procedure_with_JSON_Query

此代码将 Oracle 程序转换为具有 JSON SQL 查询的 PostgreSQL 函数。它显示了查询如何将 JSON 转换至行，反之亦然。

```
CREATE OR REPLACE PROCEDURE p_json_test(p_in_accounts_json IN varchar2,
 p_out_accunts_json  OUT varchar2)
IS
BEGIN
/*
p_in_accounts_json paramter should have following format:
       {
        "accounts": [{
          "accountNumber": 42000,
          "parentAccountNumber": 32000,
          "businessUnitId": 7
        }, {
          "accountNumber": 42001,
```

```
          "parentAccountNumber": 32001,
          "businessUnitId": 6
        }]
      }
*/
SELECT
    JSON_OBJECT(
        'accountCounts' VALUE JSON_ARRAYAGG(
            JSON_OBJECT(
                'businessUnitId' VALUE business_unit_id,
                        'parentAccountNumber' VALUE parent_account_number,
                        'accountNumber' VALUE account_number,
                        'totalOnlineContactsCount' VALUE online_contacts_count,
                        'countByPosition' VALUE
                    JSON_OBJECT(
                        'taxProfessionalCount' VALUE tax_count,
                        'attorneyCount' VALUE attorney_count,
                        'nonAttorneyCount' VALUE non_attorney_count,
                        'clerkCount' VALUE clerk_count
                               ) ) ) )
into p_out_accunts_json
FROM
    (SELECT
            tab_data.business_unit_id,
            tab_data.parent_account_number,
            tab_data.account_number,
            SUM(1) online_contacts_count,
            SUM(CASE WHEN tab_data.position_id = '0095' THEN  1 ELSE 0 END) tax_count,
            SUM(CASE    WHEN tab_data.position_id = '0100' THEN 1 ELSE 0 END)
 attorney_count,
            SUM(CASE    WHEN tab_data.position_id = '0090' THEN 1 ELSE 0 END)
 non_attorney_count,
            SUM(CASE    WHEN tab_data.position_id = '0050' THEN 1 ELSE 0 END)
 clerk_count
        FROM aws_test_table scco,JSON_TABLE ( json_doc, '$' ERROR ON ERROR
            COLUMNS (
            parent_account_number NUMBER PATH '$.data.account.parentAccountNumber',
            account_number NUMBER PATH '$.data.account.accountNumber',
            business_unit_id NUMBER PATH '$.data.account.businessUnitId',
            position_id VARCHAR2 ( 4 ) PATH '$.data.positionId'    )
            ) AS tab_data
            INNER JOIN JSON_TABLE ( p_in_accounts_json, '$.accounts[*]' ERROR ON ERROR

      COLUMNS (
```

```
      parent_account_number PATH '$.parentAccountNumber',
      account_number PATH '$.accountNumber',
      business_unit_id PATH '$.businessUnitId')
      ) static_data
      ON ( static_data.parent_account_number = tab_data.parent_account_number
           AND static_data.account_number = tab_data.account_number
           AND static_data.business_unit_id = tab_data.business_unit_id )
        GROUP BY
            tab_data.business_unit_id,
            tab_data.parent_account_number,
            tab_data.account_number
    );
EXCEPTION
WHEN OTHERS THEN
   raise_application_error(-20001,'Error while running the JSON query');
END;
/
```

运行程序

以下代码块介绍了如何运行前面创建的 Oracle 程序，并将示例 JSON 输入用于该程序。它还会为您提供此程序的结果或输出。

```
set serveroutput on;
declare
v_out varchar2(30000);
v_in varchar2(30000):= '{
        "accounts": [{
          "accountNumber": 42000,
          "parentAccountNumber": 32000,
          "businessUnitId": 7
        }, {
          "accountNumber": 42001,
          "parentAccountNumber": 32001,
          "businessUnitId": 6
        }]
      }';
begin
  p_json_test(v_in,v_out);
  dbms_output.put_line(v_out);
end;
/
```

程序输出

```
{
  "accountCounts": [
    {
      "businessUnitId": 6,
      "parentAccountNumber": 32001,
      "accountNumber": 42001,
      "totalOnlineContactsCount": 1,
      "countByPosition": {
        "taxProfessionalCount": 0,
        "attorneyCount": 0,
        "nonAttorneyCount": 1,
        "clerkCount": 0
      }
    },
    {
      "businessUnitId": 7,
      "parentAccountNumber": 32000,
      "accountNumber": 42000,
      "totalOnlineContactsCount": 1,
      "countByPosition": {
        "taxProfessionalCount": 0,
        "attorneyCount": 1,
        "nonAttorneyCount": 0,
        "clerkCount": 0
      }
    }
  ]
}
```

8.Postgres_function_with_JSON_Query

示例函数

```
CREATE OR REPLACE  FUNCTION f_pg_json_test(p_in_accounts_json  text)
RETURNS text
LANGUAGE plpgsql
AS
$$
DECLARE
 v_out_accunts_json   text;
BEGIN
```

```
SELECT
JSON_BUILD_OBJECT ('accountCounts',
    JSON_AGG(
        JSON_BUILD_OBJECT ('businessUnitId',businessUnitId
        ,'parentAccountNumber',parentAccountNumber
        ,'accountNumber',accountNumber
        ,'totalOnlineContactsCount',online_contacts_count,
        'countByPosition',
            JSON_BUILD_OBJECT (
            'taxProfessionalCount',tax_professional_count
            ,'attorneyCount',attorney_count
            ,'nonAttorneyCount',non_attorney_count
            ,'clerkCount',clerk_count
            ))))
INTO v_out_accunts_json
FROM (
WITH tab AS (SELECT * FROM (
SELECT (json_doc::json->'data'->'account'->>'parentAccountNumber')::INTEGER AS
 parentAccountNumber,
(json_doc::json->'data'->'account'->>'accountNumber')::INTEGER AS accountNumber,
(json_doc::json->'data'->'account'->>'businessUnitId')::INTEGER AS businessUnitId,
(json_doc::json->'data'->>'positionId')::varchar AS positionId
FROM aws_test_pg_table) a ) ,
tab1 AS ( SELECT
(json_array_elements(b.jc -> 'accounts') ->> 'accountNumber')::integer accountNumber,
(json_array_elements(b.jc -> 'accounts') ->> 'businessUnitId')::integer businessUnitId,
(json_array_elements(b.jc -> 'accounts') ->> 'parentAccountNumber')::integer
 parentAccountNumber
FROM (
SELECT p_in_accounts_json::json AS jc) b)
SELECT
tab.businessUnitId::text,
tab.parentAccountNumber::text,
tab.accountNumber::text,
SUM(1) online_contacts_count,
SUM(CASE WHEN tab.positionId::text = '0095' THEN 1 ELSE 0  END)
  tax_professional_count,
SUM(CASE WHEN tab.positionId::text = '0100' THEN 1 ELSE 0  END)      attorney_count,
SUM(CASE  WHEN tab.positionId::text = '0090' THEN      1  ELSE      0 END)
  non_attorney_count,
SUM(CASE  WHEN tab.positionId::text = '0050' THEN      1  ELSE      0 END)
  clerk_count
FROM tab1,tab
WHERE tab.parentAccountNumber::INTEGER=tab1.parentAccountNumber::INTEGER
```

```
AND tab.accountNumber::INTEGER=tab1.accountNumber::INTEGER
AND tab.businessUnitId::INTEGER=tab1.businessUnitId::INTEGER
GROUP BY      tab.businessUnitId::text,
            tab.parentAccountNumber::text,
            tab.accountNumber::text) a;
RETURN v_out_accunts_json;
END;
$$;
```

运行函数

```
select    f_pg_json_test('{
        "accounts": [{
          "accountNumber": 42001,
          "parentAccountNumber": 32001,
          "businessUnitId": 6
        }, {
          "accountNumber": 42000,
          "parentAccountNumber": 32000,
          "businessUnitId": 7
        }]
      }')   ;
```

函数输出

以下输出与 Oracle 程序输出类似。不同之处在于此输出为文本格式。

```
{
  "accountCounts": [
    {
      "businessUnitId": "6",
      "parentAccountNumber": "32001",
      "accountNumber": "42001",
      "totalOnlineContactsCount": 1,
      "countByPosition": {
        "taxProfessionalCount": 0,
        "attorneyCount": 0,
        "nonAttorneyCount": 1,
        "clerkCount": 0
      }
    },
    {
      "businessUnitId": "7",
```

```
      "parentAccountNumber": "32000",
      "accountNumber": "42000",
      "totalOnlineContactsCount": 1,
      "countByPosition": {
        "taxProfessionalCount": 0,
        "attorneyCount": 1,
        "nonAttorneyCount": 0,
        "clerkCount": 0
      }
    }
  ]
}
```

使用跨账户复制 Amazon DynamoDB 表 AWS Backup

由 Ramkumar Ramanujam (AWS) 编写

摘要

在使用 Amazon DynamoDB AWS时，常见的用例是在开发、测试或暂存环境中将 DynamoDB 表与生产环境中的表数据复制或同步。作为标准做法，每个环境使用不同的环境 AWS 账户。

AWS Backup 支持 DynamoDB、亚马逊简单存储服务 (Amazon S3) Service 等的跨区域和跨账户备份和恢复数据。 AWS 服务此模式提供了使用 AWS Backup 跨账户备份和还原在两者之间复制 DynamoDB 表的步骤。 AWS 账户

先决条件和限制

先决条件

- 属于同 AWS 账户 一个组织的两个活跃分子 AWS Organizations
- 在两个账户中创建 DynamoDB 表的权限
- AWS Identity and Access Management (IAM) 创建和使用文件 AWS Backup 库的权限

限制

- 来源和目标 AWS 账户 应属于同一个组织 AWS Organizations。

架构

目标技术堆栈

- AWS Backup
- Amazon DynamoDB

目标架构

1. 在源账户的备份库中 AWS Backup 创建 DynamoDB 表备份。
2. 将备份复制至目标账户的备份库中。
3. 使用目标账户中备份库中的备份，还原目标账户中的 DynamoDB 表。

自动化和扩缩

您可以使用 AWS Backup 将备份安排为按特定时间间隔运行。

工具

- AWS Backup是一项完全托管的服务，用于在云端和内部 AWS 服务集中和自动化数据保护。使用此服务，您可以一站式配置备份策略并监控 AWS 资源活动。它允许您自动执行和整合以前由服务执行的备份任务，并且无需创建自定义脚本和手动流程。
- Amazon DynamoDB 是一种全托管 NoSQL 数据库服务，提供快速而可预测的性能，能够实现无缝扩展。

操作说明

开启源账户和目标账户中的 AWS Backup 功能

Task	描述	所需技能
启用 DynamoDB 和跨账户备份高级功能。	在源和目标系统中 AWS 账户，执行以下操作： 1. 在上 AWS Management Console，打开AWS Backup 控制台。 2. 选择设置。 3. 在Amazon DynamoDB 备份的高级功能下，确认已启用高级功能，或选择启用。 4. 在跨账户管理的跨账户备份中，选择启用。	AWS DevOps，迁移工程师

在源和目标账户中创建备份库

Task	描述	所需技能
创建备份库。	在源和目标系统中 AWS 账户，执行以下操作：	AWS DevOps，迁移工程师

Task	描述	所需技能
	1. 在AWS Backup 控制台上，选择 Backup Vaul ts。 2. 选择创建备份库。 3. 复制备份库的 Amazon 资源名称（ARN），并将其保存。 在 ARNs 源账户和目标账户之间复制 DynamoDB 表备份时，将需要源和目标备份存储库。	

使用备份库执行备份和还原

Task	描述	所需技能
在源账户中，创建 DynamoDB 表备份。	要在源账户中为 DynamoDB 表创建备份，请执行以下操作： 1. 在“ AWS Backup 控制面板”页面上，选择创建按需备份。 2. 在设置部分的资源类型中，选择DynamoDB，然后选择表名。 3. 在备份库下拉列表，选择您在源账户中创建的备份库。 4. 选择所需保留期。 5. 选择创建按需备份。 创建了新的备份作业。 要监视备份作业的状态，请在“ AWS Backup 作业”页面上选	AWS DevOps、数据库管理员、迁移工程师

Task	描述	所需技能
	择 “备份作业” 选项卡。此选项卡上列出了所有活动、正在进行和已完成的备份作业。	
将备份从源账户复制到目标账户。	备份任务完成后，将 DynamoDB 表备份从源账户的备份库复制到目标账户的备份保管库。 要复制备份库，请在源账户执行以下操作： 1. 在AWS Backup 控制台上，选择 Backup Vaul ts。 2. 在备份下，选择 DynamoDB 表备份。 3. 选择 Actions (操作) 和 Copy (复制)。 4. 输入 AWS 区域 目标账户的。 5. 对于外部库 ARN，请输入您在目标账户中创建的备份库的 ARN。 6. 若要将备份从源账户复制到目标账户，请在目标账户的备份库中启用其他账户的访问权限。	AWS DevOps，迁移工程师，数据库管理员

Task	描述	所需技能
还原目标账户中的备份。	在目标中 AWS 账户，执行以下操作： 1. 在AWS Backup 控制台上，选择 Backup Vaul ts。 2. 在备份下，选择您从源账户复制的备份。 3. 对于操作，选择还原。 4. 输入您想要还原的目标 DynamoDB 表名称。	AWS DevOps、数据库管理员、迁移工程师

相关资源

- AWS Backup 与 DynamoDB 搭配使用
- 跨创建备份副本 AWS 账户
- AWS Backup 定价

使用自定义实施，跨账户复制 Amazon DynamoDB 表

由 Ramkumar Ramanujam (AWS) 编写

摘要

在Amazon Web Services (AWS) 上使用 Amazon DynamoDB 时，常见的用例是将开发、测试或模拟环境中的 DynamoDB 表与生产环境中的表数据进行复制或同步。标准做法是，在每个环境使用不同的 Amazon Web Services account。

DynamoDB 现在支持使用 AWS Backup 执行跨账户备份。有关使用 AWS Backup 时的相关存储成本信息，请参阅 AWS Backup 定价。当您使用 AWS Backup 跨账户复制时，源账户和目标账户必须是 AWS Organizations 组织的组成部分。还有其他使用 AWS 服务（例如 AWS Glue）进行跨账户备份和恢复的解决方案。但是，使用这些解决方案会增加应用程序占用空间，因为有更多的 Amazon Web Services 需要部署和维护。

您也可使用 Amazon DynamoDB Streams 来捕获源账户中的表更改。然后，您可启动 AWS Lambda 函数，并在目标账户的目标表中进行相应的更改。但是从解决方案适用于源表和目标表必须始终保持同步的用例。它可能不适用于数据更新频率较高的开发、测试和模拟环境。

此模式提供了实施自定义解决方案，以将 Amazon DynamoDB 表从一个账户复制到另一个账户的步骤。这种模式可使用 C#、Java 和 Python 等常见编程语言来实现。我们建议使用 AWS SDK 支持的语言。

先决条件和限制

先决条件

- 两个有效的 Amazon Web Services account
- 两个账户中的 DynamoDB 表
- 了解 AWS Identity and Access Management (IAM) 角色和策略
- 了解如何使用任何常用编程语言（例如 C#、Java 或 Python）访问 Amazon DynamoDB 表

限制

这种模式适用于大约 2 GB 或以内的 DynamoDB 表。通过额外的逻辑处理连接或会话中断、节流以及失败和重试，它可以用于更大的表。

从源表读取项目的 DynamoDB 扫描操作，在一次调用中最多只能获取 1 MB 的数据。对于大于 2 GB 的大表，此限制会增加执行完整表复制的总时间。

架构

下图显示了源账户和目标 AWS 账户之间的自定义实现。IAM 策略和安全令牌用于自定义实现。从源账户中的亚马逊 DynamoDB 读取数据，然后将数据写入目标账户中的 DynamoDB。

自动化和扩缩

这种模式适用于较小（大约 2 GB）的 DynamoDB 表。

若要将此模式应用于较大的表，请解决以下问题：

- 在表复制操作期间，使用不同安全令牌维护两个活动会话。如表复制操作花费的时间超过令牌到期时间，则必须设置逻辑来刷新安全令牌。
- 如果未配置足够的读取容量单位 (RCUs) 和写入容量单位 (WCUs)，则源表或目标表的读取或写入可能会受到限制。请务必捕捉并处理异常情况。
- 处理任何其他失败或异常情况，并建立重试机制，以重试或从复制操作失败的地方继续操作。

工具

工具

- Amazon DynamoDB – Amazon DynamoDB 是一种全托管 NoSQL 数据库服务，提供快速而可预测的性能，能够实现无缝扩展。
- 所需其他工具将根据您为实现选择的编程语言而异。例如，如果你使用 C#，则需要微软 Visual Studio 和以下 NuGet 软件包：
  - `AWSSDK`
  - `AWSSDK.DynamoDBv2`

代码

以下 Python 代码段使用 Boto3 库，删除并重新创建 DynamoDB 表。

请勿使用 IAM 用户的 `AWS_ACCESS_KEY_ID` 和 `AWS_SECRET_ACCESS_KEY`，因为这些都是长期凭证，在编程访问 Amazon Web Services 时应避免使用这些凭证。有关临时凭证的更多信息，请参阅最佳实践部分。

以下代码段中使用的 AWS_ACCESS_KEY_ID 、AWS_SECRET_ACCESS_KEY 和 TEMPORARY_SESSION_TOKEN，是从 AWS Security Token Service (AWS STS) 获取的临时凭证。

```
import boto3
import sys
import json

#args = input-parameters = GLOBAL_SEC_INDEXES_JSON_COLLECTION,
 ATTRIBUTES_JSON_COLLECTION, TARGET_DYNAMODB_NAME, TARGET_REGION, ...

#Input param: GLOBAL_SEC_INDEXES_JSON_COLLECTION
#[{"IndexName":"Test-index","KeySchema":[{"AttributeName":"AppId","KeyType":"HASH"},
{"AttributeName":"AppType","KeyType":"RANGE"}],"Projection":
{"ProjectionType":"INCLUDE","NonKeyAttributes":["PK","SK","OwnerName","AppVersion"]}}]

#Input param: ATTRIBUTES_JSON_COLLECTION
#[{"AttributeName":"PK","AttributeType":"S"},
{"AttributeName":"SK","AttributeType":"S"},
{"AttributeName":"AppId","AttributeType":"S"},
{"AttributeName":"AppType","AttributeType":"N"}]

region = args['TARGET_REGION']
target_ddb_name = args['TARGET_DYNAMODB_NAME']

global_secondary_indexes = json.loads(args['GLOBAL_SEC_INDEXES_JSON_COLLECTION'])
attribute_definitions = json.loads(args['ATTRIBUTES_JSON_COLLECTION'])

# Drop and create target DynamoDB table
dynamodb_client = boto3.Session(
        aws_access_key_id=args['AWS_ACCESS_KEY_ID'],
        aws_secret_access_key=args['AWS_SECRET_ACCESS_KEY'],
        aws_session_token=args['TEMPORARY_SESSION_TOKEN'],
    ).client('dynamodb')

# Delete table
print('Deleting table: ' + target_ddb_name + ' ...')

try:
    dynamodb_client.delete_table(TableName=target_ddb_name)

    #Wait for table deletion to complete
    waiter = dynamodb_client.get_waiter('table_not_exists')
    waiter.wait(TableName=target_ddb_name)
```

```
    print('Table deleted.')
except dynamodb_client.exceptions.ResourceNotFoundException:
    print('Table already deleted / does not exist.')
    pass

print('Creating table: ' + target_ddb_name + ' ...')

table = dynamodb_client.create_table(
    TableName=target_ddb_name,
    KeySchema=[
        {
            'AttributeName': 'PK',
            'KeyType': 'HASH'  # Partition key
        },
        {
            'AttributeName': 'SK',
            'KeyType': 'RANGE'  # Sort key
        }
    ],
    AttributeDefinitions=attribute_definitions,
    GlobalSecondaryIndexes=global_secondary_indexes,
    BillingMode='PAY_PER_REQUEST'
)

waiter = dynamodb_client.get_waiter('table_exists')
waiter.wait(TableName=target_ddb_name)

print('Table created.')
```

最佳实践

临时凭证

作为安全最佳实践，在以编程方式访问 AWS 服务时，请避免使用 IAM 用户的AWS_ACCESS_KEY_ID和AWS_SECRET_ACCESS_KEY，因为这些都是长期证书。请务必尝试使用临时凭证，以编程方式访问 Amazon Web Services。

例如，开发人员在开发过程中对应用程序中的 IAM 用户的 AWS_ACCESS_KEY_ID 和 AWS_SECRET_ACCESS_KEY 进行了硬编码，但在将更改推送至代码存储库之前，无法删除硬编码值。这些暴露的凭证可能被意外用户或恶意用户使用，这可能会产生严重影响（尤其是在暴露的凭证具有管理员权限的情况下）。应使用 IAM 控制台或 AWS 命令行界面（AWS CLI）立即停用或删除这些公开的凭证。

要获得以编程方式访问 Amazon Web Services 的临时凭证，请使用 AWS STS。临时凭证仅在指定时间内有效（从 15 分钟到 36 小时不等）。临时凭证允许的最长持续时间因角色设置和角色链接等因素而异。有关 AWS STS 的更多信息，请参阅文档。

操作说明

设置 DynamoDB 表

Task	描述	所需技能
创建 DynamoDB 表。	在源和目标 Amazon Web Services account 中创建带索引的 DynamoDB 表。 将容量配置设置为按需模式，如此 DynamoDB 就可以根据工作负载动态扩展读/写容量。 或者，您可以将预配置容量与 4000 RCUs 和 4000 配合使用。 WCUs	应用程序开发人员，数据库管理员，迁移工程师
填充源表格。	使用测试数据填充源账户的 DynamoDB 表。拥有至少 50 MB 或更多的测试数据有助于您查看表复制期间的峰值和平均 RCUs 消耗量。然后，您可根据需要更改容量配置。	应用程序开发人员，数据库管理员，迁移工程师

设置用于访问 DynamoDB 表的凭证

Task	描述	所需技能
创建 IAM 角色，以访问源表和目标 DynamoDB 表。	在源账户中创建 IAM 角色，该角色有权访问（读取）源账户中的 DynamoDB 表。	AWS 应用程序开发人员 DevOps

Task	描述	所需技能
	将源账户添加至该角色的可信实体。 在目标账户中创建一个 IAM 角色，该角色有权访问（创建、读取、更新、删除）目标账户中的 DynamoDB 表。 将目标账户添加至该角色的可信实体。	

将表数据从一个账户复制至另一个账户

Task	描述	所需技能
获取 IAM 角色的临时凭证。	获取在源账户中创建的 IAM 角色的临时凭证。 获取在目标账户中创建的 IAM 角色的临时凭证。 获取 IAM 角色临时凭证的一种方法是在 AWS CLI 中使用 AWS STS。 `aws sts assume-role --role-arn arn:aws:iam::<account-id>:role/<role-name> --role-session-name <session-name> --profile <profile-name>` 使用相应的 AWS 配置文件（对应于源账户或目标账户）。	应用程序开发人员、迁移工程师

Task	描述	所需技能
	有关获取临时凭证的不同方式的详细信息，请参阅以下内容： • AWS Security Token Service API 参考 • 获取用于 CLI 访问的 IAM 角色凭证	
初始化 DynamoDB 客户端，以便访问源与目标 DynamoDB。	为源表和目标 DynamoDB 表初始化 AWS 开发工具包所提供的 DynamoDB 客户端。 • 对于源 DynamoDB 客户端，请使用从源账户获取的临时凭证。 • 对于目标 DynamoDB 客户端，请使用从目标账户获取的临时凭证。 有关使用 IAM 临时凭证发出请求的更多信息，请参阅 AWS 文档。	应用程序开发人员

Task	描述	所需技能
删除和重新创建目标表。	使用目标账户 DynamoDB 客户端，在目标账户中删除并重新创建目标 DynamoDB 表（以及索引）。 从 DynamoDB 表中删除所有记录是一项代价高昂的操作，因为它会消耗预配置。 WCUs删除并重新创建表，可以避免这些额外费用。 您可在创建表之后向其添加索引，但这会延长 2-5 分钟。通过将索引集传递至 `createTable` 调用，在创建表期间创建索引的效率更高。	应用程序开发人员

Task	描述	所需技能
执行表格复制。	重复以下步骤，直至复制完所有数据： • 使用源 DynamoDB 客户端对源账户的表执行扫描。每次 DynamoDB 扫描仅从表中检索 1 MB 数据，因此您必须重复此操作，直到读取所有项目或记录。 • 对于每组扫描的项目，使用适用于 DynamoDB 的 AWS 开发工具包中的 `BatchWriteItem` 调用，使用目标 DynamoDB 客户端将项目写入目标账户的表中。这将减少向 DynamoDB 发出的 `PutItem` 请求的数量。 • `BatchWriteItem` 限制为 25 次写入或放置，或最多 16 MB。在调用 `BatchWriteItem` 前，您必须添加逻辑以累积已扫描的项目，计数为 2。`BatchWriteItem` 返回无法成功复制的项目列表。使用此列表，添加重试逻辑，以便仅使用未成功的项目执行另一次 `BatchWriteItem` 调用。 有关更多信息，请参阅附件部分中的 C# 参考实现（用于删除、创建和填充表）。还附上了示例表配置 JavaScript 对象表示法 (JSON) 文件。	应用程序开发人员

相关资源

- Amazon DynamoDB 文档
- 在您的 Amazon Web Services account 中创建 IAM 用户
- AWS SDKs
- 将临时凭证用于 AWS 资源

其他信息

此模式是使用 C# 实现的，用于复制包含 200,000 个项目（平均项目大小为 5 KB，表大小为 250 MB）的 DynamoDB 表。目标 DynamoDB 表的预配置容量为 4000 和 4000。 RCUs WCUs

完整的表复制操作（从源账户到目标账户），包括删除和重新创建表，花了 5 分钟。消耗的总容量单位：30,000 个 RCUs 和大约 40 万个 WCUs。

有关 DynamoDB 容量模式的更多信息，请参阅 AWS 文档中的读/写容量模式。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

为 Amazon RDS 和 Amazon Aurora 创建详细的成本和使用情况报告

由 Lakshmanan Lakshmanan (AWS) 和 Sudarshan Narasimhan 创建

摘要

此模式显示如何通过配置用户定义的成本分配标签来跟踪 Amazon Relational Database Service（Amazon RDS）或 Amazon Aurora 集群的使用成本。您可以使用这些标签在 AWS Cost Explorer 成本管理服务中为跨多个维度的集群创建详细的成本和使用情况报告。例如，您可以在团队、项目或成本中心级别跟踪使用成本，然后在 Amazon Athena 中分析数据。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 一个或多个 Amazon RDS 或 Amazon Aurora 实例

限制

有关标记限制，请参阅 Amazon Web Services Billing 用户指南。

架构

目标技术堆栈

- Amazon RDS 或 Amazon Aurora
- AWS 成本和使用报告
- AWS Cost Explorer 成本管理服务
- Amazon Athena

工作流程和架构

标记和分析工作流程包含以下步骤：

1. 数据工程师、数据库管理员或 AWS 管理员为 Amazon RDS 或 Aurora 集群创建用户定义的成本分配标签。

2. AWS 管理员激活标签。
3. 标签将元数据报告给 AWS Cost Explorer 成本管理服务。
4. 数据工程师、数据库管理员或 AWS 管理员创建月度成本分配报告。
5. 数据工程师、数据库管理员或 AWS 管理员使用 Amazon Athena 分析月度成本分配报告。

下图显示了如何应用标签来跟踪 Amazon RDS 或 Aurora 实例的使用成本。

以下架构图显示了如何将成本分配报告与 Amazon Athena 集成以进行分析。

月度成本分配报告存储在您指定的 Amazon S3 存储桶中。当你使用 AW CloudFormation S 模板设置 AThena 时，如史诗部分所述，该模板会预配置几个额外的资源，包括 AWS Glue 爬虫、AWS Glue 数据库、亚马逊简单通知系统 (Amazon SNS) 事件、AWS Lambda 函数以及 Lambda 函数的 AWS 身份和访问管理 (IAM) 角色。当新的成本数据文件到达 S3 存储桶时，事件通知将用于将这些文件转发到 Lambda 函数进行处理。Lambda 函数启动 AWS Glue 爬网程序作业，以创建或更新 AWS Glue Data Catalog 中的表。然后，此表用于查询 Athena 中的数据。

工具

- Amazon Athena 是一种交互式查询服务，方便使用标准 SQL 分析 Amazon S3 的数据。
- Amazon Aurora 是与 MySQL 和 PostgreSQL 兼容的完全托管式的云端关系数据库引擎。
- Amazon Relational Database Service (Amazon RDS) 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展关系数据库。
- AWS CloudFormation 是一项基础设施即代码 (IaC) 服务，可让您轻松建模、配置和管理 AWS 和第三方资源。
- AWS Cost Explorer 成本管理服务可助您您查看和分析成本与使用情况。

操作说明

为您的 Amazon RDS 或 Aurora 集群创建和激活标签

Task	描述	所需技能
为您的 Amazon RDS 或 Aurora 集群创建用户定义的成本分配标签。	要向新的或现有的 Amazon RDS 或 Aurora 集群添加标签，请按照 Amazon Aurora 用户指南中的添加、列出和删除标签中的说明操作。 **Note** 有关如何设置亚马逊 Aurora 集群的信息，请参阅亚马逊 Aurora 用户指南中针对 MySQL 和 PostgreSQL 的说明。	AWS 管理员、数据工程师、数据库管理员
激活用户定义的成本分配标签。	按照 Amazon Web Services Billing 用户指南中的激活用户定义的成本分配标签中的说明操作。	AWS 管理员

创建成本和使用报告

Task	描述	所需技能
为集群创建和配置成本和使用情况报告。	1. 登录 AWS 管理控制台，然后打开 Amazon Web Services Billing Console。 2. 在导航窗格中，请选择成本和使用报告。 3. 选择创建报告。	应用程序所有者、AWS 管理员、数据库管理员、常规 AWS、数据工程师

Task	描述	所需技能
	4. 提供报告名称，保留其他选项的默认设置，然后选择下一步。 5. 选择配置并提供现有 S3 存储桶的详细信息。您还可以选择从此屏幕创建新的 S3 存储桶。选择下一步。 6. 验证将应用于存储桶的默认策略，选中确认复选框，然后选择保存。 7. 对于报告路径前缀，指定要添加到报告名称前面的前缀。 8. 对于时间粒度，选择每小时、每天或每月，具体取决于您希望为报告收集数据的频率。 9. 对于报告版本控制，选择是要单独创建报告的新版本还是用每个版本覆盖现有报告。 10对于启用报告数据集成，选择 Amazon Athena。验证压缩类型是否设置为 Parquet。 11选择下一步。 12查看报告设置，然后选择 查看并完成。 数据将在 24 小时内提供。	

分析成本和使用情况报告数据

Task	描述	所需技能
分析成本和使用报告数据。	1. 设置并使用 Athena 来分析报告数据。有关说明，请参阅 AWS 成本和使用情况报告用户指南中的使用 Amazon Athena 查询成本和使用情况报告。我们建议您使用 Athen a 提供的 AWS CloudFormation 模板。 2. 运行 Athena 查询。例如，您可以使用以下 SQL 查询来检查数据刷新的状态。 `select status from cost_and_usage_data_status` 有关更多信息，请参阅《AWS 成本和使用情况报告用户指南》中的运行 Amazon Athena 查询。 **Note** 运行 SQL 查询时，请确保从下拉列表中选择了正确的数据库。	应用程序所有者、AWS 管理员、数据库管理员、常规 AWS、数据工程师

相关资源

参考

- 使用 A CloudFormation WS 模板设置 Athena（推荐）

- 手动设置 Athena
- 运行 Amazon Athena 查询
- 将报告数据加载到其他资源

教程和视频

- 使用 Amazon Athena 分析成本和使用情况报告（YouTube 视频）

使用 Aurora PostgreSQL 中的自定义端点模拟 Oracle RAC 工作负载

由 HariKrishna Boorgadda (AWS) 创建

摘要

此模式描述了如何使用 Amazon Aurora PostgreSQL-Compatible Edition 以及在单个集群内的实例之间分配工作负载的自定义端点模拟 Oracle Real Application Clusters (Oracle RAC) 工作负载服务。该模式向您展示如何为 Amazon Aurora 数据库创建自定义端点。自定义端点使您能够在 Aurora 集群中的不同数据库实例集之间分配和负载均衡工作负载。

在 Oracle RAC 环境中，服务可以跨越一个或多个实例，并根据事务性能促进工作负载均衡。服务功能包括 end-to-end无人值守的恢复、按工作负载滚动更改以及完全的位置透明度。您可使用此模式模拟其中的一些功能。例如，您可模拟报表应用程序路由连接功能。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- PostgreSQL JDBC 驱动程序
- Aurora PostgreSQL-Compatible 数据库
- 已迁移到 Aurora PostgreSQL-Compatible 数据库的 Oracle RAC 数据库

限制

- 有关适用于自定义端点的限制，请参阅 Amazon RDS 文档中的指定自定义端点属性。

架构

源技术堆栈

- 三节点 Oracle RAC 数据库

目标技术堆栈

- Aurora PostgreSQL-Compatible 数据库，具有两个只读副本

源架构

下图显示了三节点 Oracle RAC 数据库架构。

目标架构

下图显示了具有两个只读副本的 Aurora PostgreSQL-Compatible 数据库的架构。三个不同的应用程序/服务使用自定义端点，为不同的应用程序用户提供服务，并在主副本和只读副本之间重定向流量和负载。

工具

- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩展 PostgreSQL 部署。
- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和在 AWS 上运行的应用程序的指标。
- 适用于 PostgreSQL 的 Amazon Relational Database Service（Amazon RDS）可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展 PostgreSQL 关系数据库。
- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。

操作说明

创建 Aurora PostgreSQL-Compatible 集群

Task	描述	所需技能
创建集群。	要创建集群，请参阅 Amazon RDS 文档中的创建数据库集群并连接到 Aurora PostgreSQL 数据库集群上的数据库。	AWS 管理员
为工作负载创建自定义参数组。	要创建参数组，请参阅 Amazon RDS 文档中的创建数据库集群参数组。	AWS 管理员

Task	描述	所需技能
创建事件通知和警报。	您可以使用事件通知和 Amazon CloudWatch 警报在集群状态发生变化时通知您，并在达到预定义阈值时捕获指标。 要创建 CloudWatch 警报，请参阅 CloudWatch 文档中的基于静态阈值创建 CloudWatch 警报。 要创建事件通知，请参阅 CloudWatch 文档中的创建在 CloudWatch 事件上触发的事件规则。	AWS 管理员

向 Aurora PostgreSQL-Compatible 数据库集群添加副本

Task	描述	所需技能
将只读副本添加到该集群。	1. 创建只读副本。 2. **Note** 将只读副本添加到数据库集群所在的同一可用区。：如果您的故障转移节点必须满足要求，则可以使用不同的可用区。	AWS 管理员
记下只读副本端点。	记录您的只读副本端点，以便以后在创建自定义端点时使用。	AWS 管理员

创建自定义端点

Task	描述	所需技能
输入自定义端点的名称。	对于您需要的每个端点，创建一个与您的工作负载或应用程序相关的唯一端点名称。	AWS 管理员
添加端点成员。	将您的只读副本端点添加到自定义群组。有关更多信息，请参阅 Amazon RDS 文档中的编辑自定义端点。	AWS 管理员
（可选）向集群添加未来实例。	如果您想向自定义组添加更多副本或端点，请参阅 Amazon RDS 文档中的将 Aurora 副本添加到数据库集群。	AWS 管理员
创建端点。	要创建端点，请参阅 Amazon RDS 文档中的创建自定义端点。	AWS 管理员

通过自定义端点测试应用程序连接

Task	描述	所需技能
与指向您的工作负载的应用程序共享自定义端点详细信息。	将您的自定义端点详细信息添加到您计划测试的报告应用程序中的数据库连接详细信息中。	AWS 管理员
使用自定义端点连接工作负载。	在报告应用程序中验证自定义端点的详细信息。	AWS 管理员
检查数据库中连接详细信息。	1. 测试应用程序用户名和连接数。	AWS 管理员

Task	描述	所需技能
	2. 检查工作负载之间的负载均衡，确保连接分布在不同的自定义端点（主副本和只读副本）上。	

相关资源

- Aurora 端点的类型
- 自定义端点的成员资格规则
- End-to-end 自定义终端节点的 AWS CLI 示例
- Amazon Aurora 作为 Oracle RAC 的替代品
- 从 Oracle 迁移到 PostgreSQL 时面临的挑战，以及如何克服这些挑战

在 Amazon RDS 中为 PostgreSQL 数据库实例启用加密连接

创建者：Rohit Kapoor (AWS)

摘要

Amazon Relational Database Service（Amazon RDS）支持适用于 PostgreSQL 数据库实例的 SSL 加密。使用 SSL，您可以加密应用程序与 Amazon RDS for PostgreSQL 数据库实例之间的 PostgreSQL 连接。默认情况下，亚马逊 RDS for Postgre SSL/TLS and expects all clients to connect by using SSL/TLS SQL 使用加密。Amazon RDS for PostgreSQL 支持 TLS 版本 1.1 和 1.2。

此模式介绍了如何为 Amazon RDS for PostgreSQL 数据库实例启用加密连接。您可使用相同进程为 Amazon Aurora PostgreSQL-Compatible Edition 启用加密连接。

先决条件和限制

- 一个有效的 Amazon Web Services account
- Amazon RDS for PostgreSQL 数据库实例
- SSL 捆绑包

架构

工具

- pgAdmin 是 PostgreSQL 的开源管理和开发平台。您可在 Linux、Unix、macOS 和 Windows 上使用 pgadmin 来管理 PostgreSQL 10 及更高版本中的数据库对象。
- PostgreSQL 编辑器提供了更加用户友好的界面，可帮助您创建、开发和运行查询，并按您的要求编辑代码。

最佳实践

- 监控不安全数据库连接。
- 审核数据库访问权限。
- 确保对备份和快照进行静态加密。
- 监控数据库访问。

- 避免使用不受限制的访问组。
- 使用 Amazon 增强您的通知效果 GuardDuty。
- 定期监控政策遵守情况。

操作说明

下载受信任的证书，并将其导入您的信任存储区

Task	描述	所需技能
将受信任的证书加载到您的计算机。	要将证书添加到计算机的受信任的根证书颁发机构存储中，请按照以下步骤操作。（这些指令以 Windows Server 为例。） 1. 在 Windows 服务器中，选择开始、运行，然后键入 mmc。 2. 在控制台中，选择文件、添加/删除管理单元。 3. 在可用管理单元下方，选择证书，然后选择添加。 4. 在此管理单元将始终管理证书下方，选择计算机账户、下一步。 5. 选择本地计算机、完成。 6. 如果您没有其他管理单元要添加到控制台，请选择确定。 7. 在控制台树中，双击证书。 8. 右键单击受信任的根证书颁发机构。 9. 依次选择所有任务和导入以导入已下载的证书。	DevOps 工程师、迁移工程师、DBA

Task	描述	所需技能
	10.按照“证书导入向导”中的步骤操作。	

强制执行 SSL 连接

Task	描述	所需技能
创建参数组和设置 rds.force_ssl 参数。	如果 PostgreSQL 数据库实例具有自定义参数组，请编辑该参数组并将 `rds.force_ssl` 更改为 1。 如果数据库实例使用未启用 `rds.force_ssl` 的默认参数组，则创建新的参数组。您可使用 Amazon RDS API 修改新的参数组，也可以按照以下说明手动修改新的参数组。 要创建新参数组，请执行以下操作： 1. 登录 AWS 管理控制台，打开托管数据库实例的 AWS 区域 的 Amazon RDS 控制台。 2. 在导航窗格中，选择参数组。 3. 选择创建参数组，然后设置以下值： • 对于参数组系列，选择 postgres14 或更高版本。	DevOps 工程师、迁移工程师、DBA

Task	描述	所需技能
	• 对于组名，键入 pgsql-<database_instance>-ssl。 • 对于描述，为要添加的参数组输入自由格式的描述。 • 选择创建。 4. 选择您创建的参数组。 5. 从 Parameter group actions (参数组操作) 中，选择 Edit (编辑)。 6. 找到 rds.force_ssl 并将其设置更改为 1。 **Note** 在更改此参数之前，请先进行客户端测试。 7. 选择 Save changes（保存更改）。 将参数组与您的 PostgreSQL 数据库实例相关联，请执行以下操作： 1. 在 Amazon RDS 控制台上的导航窗格中，选择数据库，然后选择 PostgreSQL 数据库实例。 2. 选择 Modify（修改）。 3. 在其他配置下方，选择新的参数组，然后选择继续。	

Task	描述	所需技能
	4. 在计划修改下方，选择立即应用。 5. 选择修改数据库实例。 有关更多信息，请参阅 Amazon RDS 文档。	
强制执行 SSL 连接。	连接到 Amazon RDS for PostgreSQL 数据库实例。不使用 SSL 的连接尝试会被拒绝，并显示错误消息。有关更多信息，请参阅 Amazon RDS 文档。	DevOps 工程师、迁移工程师、DBA

安装 SSL 扩展

Task	描述	所需技能

安装 SSL 扩展。	1. 以数据库管理员身份启动 psql 或 pgadmin 连接。
	2. 调用 ssl_is_used() 函数以判断是否在使用 SSL。
	
	`select ssl_is_used();`
	
	如果连接使用的是 SSL，则此函数将返回 t；否则返回 f。
	3. 安装 SSL 扩展。
	
	`create extension sslinfo;`
	`show ssl;`
	`select ssl_cipher();`
		DevOps 工程师、迁移工程师、DBA

Task	描述	所需技能
	有关更多信息，请参阅 Amazon RDS 文档。	

为您的 PostgreSQL 客户端配置 SSL

Task	描述	所需技能

为 SSL 配置客户端。	通过使用 SSL，您可启动支持使用 TLS 协议的加密连接的 PostgreSQL 服务器。服务器在同一 TCP 端口上侦听标准连接和 SSL 连接，并与任何连接客户端协商是否使用 SSL。默认情况下，这是一个客户端选项。
	
	如果您使用 psql 客户端：
	
	1. 确保已将 Amazon RDS 证书加载到您的本地计算机。
	2. 通过添加以下内容启动 SSL 客户端连接：
	
	`psql postgres -h SOMEHOST.amazonaws.com -p 8192 -U someuser sslmode=verify-full sslrootcert=rds-ssl-ca-cert.pem`
	`select ssl_cipher();`
	
	对于其他 PostgreSQL 客户端：
		DevOps 工程师、迁移工程师、DBA

Task	描述	所需技能
	• 修改相应的应用程序公有密钥参数。在 GUI 工具的连接页上，这可能以选项或连接字符串一部分或属性形式提供。 查看以下客户页面： • pgAdmin 文档 • JDBC 文档	

故障排除

事务	解决方案
无法下载 SSL 证书。	检查您与网站的连接，然后重试将证书下载到您的本地计算机。

相关资源

- Amazon RDS for PostgreSQL 文档
- 将 SSL 与 PostgreSQL 数据库实例结合使用（Amazon RDS 文档）
- 使用 SSL 保护 TCP/IP 连接（PostgreSQL 文档）
- 使用 SSL（JDBC 文档）

加密现有 Amazon RDS for PostgreSQL 数据库实例

由 Piyush Goyal (AWS)、Shobana Raghu (AWS) 和 Yaser Raja (AWS) 编写

摘要

此模式说明了如何在 AWS 云中加密适用于 PostgreSQL 数据库实例的现有亚马逊关系数据库服务 (Amazon RDS)，同时最大限度地减少停机时间。此进程也适用于 Amazon RDS for MySQL 数据库实例。

您可以在创建 Amazon RDS DB 数据库实例时为其启用加密，而不能在创建数据库实例之后启用加密。但是，您可以对未加密的数据库实例添加加密，方法是创建数据库实例快照，然后创建此快照的加密副本。然后，您可以从加密快照还原数据库实例，从而获得原始数据库实例的加密副本。如果您的项目允许在此活动期间停机（至少对于写入事务来说是如此），那么这就是您所需要做的。当数据库实例的新加密副本可用时，您可将应用程序指向新数据库。但是，如果您的项目不允许此活动出现大量停机时间，则您需要一种替代方法来帮助最大限度地减少停机时间。此模式使用 AWS Database Migration Service (AWS DMS) 迁移并持续复制数据，以便可以在最短的停机时间内完成到新的加密数据库的割接。

Amazon RDS 加密的数据库实例使用行业标准 AES-256 加密算法，对托管 Amazon RDS 数据库实例的服务器上的数据进行加密。在加密数据后，Amazon RDS 将以透明方式处理访问的身份验证和数据的解密，并且对性能产生的影响最小。您无需修改数据库客户端应用程序来使用加密。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 未加密的 Amazon RDS for PostgreSQL 数据库实例
- 有处理（创建、修改或停止）AWS DMS 任务的经验（请参阅 AWS DMS 文档中的处理 AWS DMS 任务）
- 熟悉用于加密数据库的 AWS Key Management Service (AWS KMS)（AWS KMS 文档）

限制

- 您只能在创建 Amazon RDS 数据库实例时而不是创建该数据库实例之后启用对该数据库实例的加密。
- 未记录表中的数据不会使用快照进行恢复。有关更多信息，请参阅 PostgreSQL 最佳实践。

- 您无法拥有未加密数据库实例的加密只读副本或加密数据库实例的未加密只读副本。
- 您不能将未加密的备份或快照还原到加密的数据库实例。
- AWS DMS 不会自动传输序列，因此需要采取额外的步骤来处理此问题。

有关更多信息，请参阅 Amazon RDS 文档中的 Amazon RDS 加密数据库实例限制。

架构

源架构

- 未加密的 RDS 数据库实例

目标架构

- 加密的 RDS 数据库实例
  - 目标 RDS 数据库实例是通过恢复源 RDS 数据库实例的数据库快照副本来创建的。
  - 还原快照时使用 AWS KMS 密钥加密。
  - AWS DMS 复制任务可用于迁移数据。

工具

用于启用加密的工具：

- 用于加密的 AWS KMS 密钥 - 创建加密数据库实例时，您可为 Amazon RDS 选择客户托管密钥或适用于 Amazon RDS 的 AWS 托管式密钥 来加密您的数据库实例。如果您没有为客户托管密钥指定密钥标识符，则 Amazon RDS 会将 AWS 托管式密钥用于您的新数据库实例。Amazon RDS 为您的 Amazon Web Services account 创建用于 Amazon RDS 的 AWS 托管式密钥。您的 Amazon Web Services account 对每个 AWS 区域都有不同的用于 Amazon RDS 的 AWS 托管式密钥。有关使用 KMS 密钥进行 Amazon RDS 加密的更多信息，请参阅加密 Amazon RDS 资源。

用于持续复制的工具：

- AWS DMS — 您可使用 AWS Database Migration Service (AWS DMS) 将更改从源数据库复制到目标数据库。保持源数据库和目标数据库同步，最大限度地减少停机时间，这一点很重要。有关设置 AWS DMS 和创建任务的信息，请参阅 AWS DMS 文档。

操作说明

创建源数据库实例快照并加密

Task	描述	所需技能
查看源 PostgreSQL 数据库实例详细信息。	在 Amazon RDS 控制台，选择源 PostgreSQL 数据库实例。在配置选项卡，确保未为实例启用加密。有关屏幕插图，请参阅其他信息部分。	数据库管理员
创建数据库快照。	为待加密的实例创建数据库快照。创建快照所用时间因数据库大小而异。有关说明，请参阅 Amazon RDS 文档中的创建数据库快照。	数据库管理员
加密快照。	在 Amazon RDS 控制台导航窗格，选择快照，然后选择您创建的数据库快照。对于 Actions (操作)，选择 Copy Snapshot (复制快照)。在相应的字段中提供目标 AWS 区域和数据库快照副本的名称。选中启用加密复选框。对于 Master Key，指定用于加密数据库快照副本的 KMS 密钥标识符。选择复制快照。有关更多信息，请参阅 Amazon RDS 文档中的复制快照。	数据库管理员

准备目标数据库实例

Task	描述	所需技能
还原数据库快照。	在 Amazon RDS 控制台中，选择快照。选择您创建的加密快照。对于操作，选择还原快照。对于数据库实例标识符，请为新数据库实例提供一个唯一名称。查看实例详细信息，然后选择还原数据库实例。将根据您的快照创建新的加密数据库实例。有关更多信息，请参阅 Amazon RDS 文档中从数据库快照还原。	数据库管理员
使用 AWS DMS 迁移数据。	在 AWS DMS 控制台，创建 AWS DMS 任务。对于迁移类型，请选择迁移现有数据并复制正在进行的更改。在任务设置，对于目标表格准备模式，选择截断。有关更多信息，请参阅 AWS DMS 文档中的创建任务。	数据库管理员
启用数据验证。	在任务设置中，选择启用验证。这使您能够将源数据与目标数据进行比较，以验证数据是否已准确迁移。	数据库管理员
禁用目标数据库实例限制。	在目标数据库实例的禁用所有触发器和外键约束，然后启动 AWS DMS 任务。有关禁用触发器和外键约束的更多信息，请参见 AWS DMS 文档。	数据库管理员
验证数据。	满载完成后，验证目标数据库实例数据，以查看其是否与	数据库管理员

Task	描述	所需技能
	源数据匹配。有关更多信息，请参阅 AWS DMS 文档中的 AWS DMS 数据验证。	

割接到目标数据库实例

Task	描述	所需技能
停止对源数据库实例的写入操作。	停止对源数据库实例的写入操作，以应用程序可以开始停机。验证 AWS DMS 是否已完成对管道数据的复制。启用目标数据库实例的触发器与外键。	数据库管理员
更新数据库序列	如果源数据库包含任何序列号，则验证并更新目标数据库中的序列。	数据库管理员
配置应用程序端点。	配置您的应用程序连接，以使用新 Amazon RDS 数据库实例端点。已加密数据库实例。	数据库管理员，应用程序所有者

相关资源

- 创建 AWS DMS 任务
- 使用 Amazon 监控复制任务 CloudWatch
- 监控 AWS DMS 任务
- 更新 Amazon RDS 加密密钥

其他信息

检查源 PostgreSQL 数据库实例的加密：

此示例的其他注意事项：

- 通过将 `rds.logical_replication` 参数设置为 1，在 PostgreSQL 上启用复制。

重要说明：复制槽会保留预写日志 (WAL) 文件，直到 `pg_recvlogical` 文件被外部使用，例如，通过提取、转换、加载（ETL）作业；或由 AWS DMS 使用。当您将 `rds.logical_replication` 参数值设置为 1 时，AWS DMS 会设置 `wal_level`、`max_wal_senders`、`max_replication_slots` 和 `max_connections` 参数。如果存在逻辑复制槽，但复制槽保留的 WAL 文件没有使用者，您可能会看到事务日志磁盘使用量增加，而可用存储空间不断减少。有关解决此问题的更多信息和步骤，请参阅文章如何确定是什么原因导致 Amazon RDS for PostgreSQL 上出现 “设备上没有剩余空间” 或 “” 错误？DiskFull 在 AWS Support 知识中心中。

- 创建数据库快照后对源数据库实例所做的任何架构更改，都不会出现在目标数据库实例上。
- 创建加密的数据库实例后，您无法更改该数据库实例使用的 KMS 密钥。请确保先确定您的 KMS 密钥要求，然后再创建加密的数据库实例。
- 运行 AWS DMS 任务之前，您必须在目标数据库实例上禁用触发器和外键。任务完成后，您可将它们重新启用。

在启动时强制对 Amazon RDS 数据库执行自动标记

由 Susanne Kangnoh (AWS) 和 Archit Mathur (AWS) 创建

摘要

Amazon Relational Database Service (Amazon RDS) 是一项 Web 服务，让用户能够在 Amazon Web Services (AWS) 云中更轻松地设置、操作和扩展关系数据库。它为行业标准的关系数据库提供了经济高效、可调整大小的容量，并管理常见的数据库管理任务。

您可使用标签按不同方式对 AWS 资源进行分类。当您的账户中有许多资源并且您希望根据标签快速识别特定资源时，关系数据库标记非常有用。您可使用 Amazon RDS 标签向您的 RDS 数据库实例添加自定义元数据。每个标签都由用户定义的键和值组成。我们建议您创建一组一致的标签以满足您的组织要求。

此模式提供了一个 AWS CloudFormation 模板来帮助您监控和标记 RDS 数据库实例。该模板创建了一个监视 AWS C CloudTrail re ate CloudWatch 事件的 Amazon Event DBInstance s 事件。（将 Amazon RDS 的 API 调用CloudTrail 捕获为事件。）当它检测到该事件时，它会调用 AWS Lambda 函数，该函数会自动应用您定义的标签键和值。该模板还使用 Amazon Simple Notification Service (Amazon SNS) 发出实例已被标记的通知。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 用于上传 Lambda 代码的 Amazon Simple Storage Service（Amazon S3）存储桶。
- 您希望接收标记通知的电子邮件地址。

限制

- 该解决方案支持 CloudTrail 创建DBInstance事件。它不会为任何其他事件创建通知。

架构

工作流程架构

自动化和扩缩

- 您可以针对不同的 AWS 区域和账户多次使用 AWS CloudFormation 模板。您只需在每个区域或账户中运行该模板一次。

工具

Amazon Web Services

- AWS CloudTrail — AWS CloudTrail 是一项 AWS 服务，可帮助您对 AWS 账户进行治理、合规以及运营和风险审计。用户、角色或 AWS 服务采取的操作在中记录为事件 CloudTrail。
- Amazon CloudWatch Events — Amazon CloudWatch Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。 CloudWatch 通过发送消息以响应环境、激活功能、进行更改和捕获状态信息，事件会在操作变化发生时意识到这些变化，并在必要时采取纠正措施。
- AWS Lambda — AWS Lambda 是一项计算服务，使您无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。您只需为消耗的计算时间付费 - 代码未运行时不产生费用。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项高度可扩展的对象存储服务，可用于各种存储解决方案，包括网站、移动应用程序、备份和数据湖。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 是一项 Web 服务，可让应用程序、终端用户和设备即时发送和接收来自云端的通知。

代码

此模式包括一个包含两个文件的附件：

- `index.zip` 是压缩文件，其中包含此模式的 Lambda 代码。
- `rds.yaml`是部署 Lambda 代码的 CloudFormation 模板。

有关如何使用这些文件的信息，请参阅操作说明部分。

操作说明

部署 Lambda 代码

Task	描述	所需技能
将代码上传到 S3 存储桶。	创建新的 S3 存储桶或使用现有 S3 存储桶上传附加的	云架构师

Task	描述	所需技能
	`index.zip` 文件（Lambda 代码）。此存储桶必须与要监控的资源 (RDS 数据库实例) 位于同一 AWS 区域中。	
部署 CloudFormation 模板。	在与 S3 存储桶相同的 AWS 区域中打开 Cloudformation 控制台，然后部署附件中提供的 `rds.yaml` 文件。在下一个操作说明中，提供模板参数的值。	云架构师

填写 CloudFormation 模板中的参数

Task	描述	所需技能
提供 S3 存储桶名称。	输入您在第一篇操作说明中创建或选择的 S3 存储桶的名称。此 S3 存储桶包含 Lambda 代码的.zip 文件，并且必须与模板和要监控的 RDS 数据库实例位于相同的 AWS 区域。CloudFormation	云架构师
提供 S3 密钥。	提供 Lambda 代码 .zip 文件在 S3 存储桶中的位置，不带前导斜杠（例如，`index.zip` 或 `controls/index.zip`）。	云架构师
提供电子邮箱地址。	提供要接收违规通知的活动电子邮件地址。	云架构师
指定日志记录级别。	指定日志级别和详细程度。`Info` 指定有关应用程序	云架构师

Task	描述	所需技能
	进度的详细信息消息，应仅用于调试。`Error` 指定仍允许应用程序继续运行的错误事件。`Warning` 表示潜在的有害情况。	
输入 RDS 数据库实例的标签键和值。	输入您想要自动应用于 RDS 实例的所需标签键和值。有关更多信息，请参阅 AWS 文档中的为 Amazon RDS 资源添加标签。	云架构师

确认订阅

Task	描述	所需技能
确认电子邮件订阅。	成功部署 CloudFormation 模板后，它会向您提供的电子邮件地址发送订阅电子邮件。要在标记实例时接收通知，您必须确认此电子邮件订阅。	云架构师

相关资源

- 创建存储桶（Amazon S3 文档）
- 为 Amazon RDS 资源添加标签（Amazon Aurora 文档）
- 上传对象（Amazon S3 文档）
- 使用 AWS 创建在 AWS API 调用时触发 CloudWatch 的事件规则 CloudTrail（亚马逊 CloudWatch 文档）

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

估算按需容量的 DynamoDB 表成本

由 Moinul Al-Mamun (AWS) 创建

摘要

Amazon DynamoDB 是 NoSQL 事务数据库，即使在 PB 级规模下，也能提供个位数毫秒的延迟。此款 Amazon Web Services (AWS) 无服务器产品因其稳定的性能和可扩展性而越来越受欢迎。 您无需配置基础设施。您的单个表最多可以扩展至 PB 级。

在按需容量模式下，您按请求为应用程序在表上执行的数据读取和写入付费。AWS 的费用基于一个月内累积的读取请求单位 (RRUsWRUs) 和写入请求单位 ()。DynamoDB 会在整个月中持续监控您的表大小，确定您的存储费用。它支持使用 point-in-time-recovery (PITR) 进行连续备份。DynamoDB 会在整个月中持续监控您的启用 PITR 的表大小，确定您的备份费用。

要估算项目的 DynamoDB 成本，计算在产品生命周期的不同阶段将消耗多少 RRU、WRU 和存储非常重要。要进行粗略的成本估算，您可以使用 AWS Pricing Calculator RRUs，但您必须提供表的大致数量 WRUs、和存储要求。这些在项目开始时可能很难估计。AWS Pricing Calculator 不考虑数据增长率或项目大小，也不会分别考虑基表和全局二级索引 (GSIs) 的读取和写入次数。要使用 AWS 定价计算器，您必须估算所有这些方面，以假设 WRU、RRU 和存储大小的大致数字，以获得成本估算。

此模式提供了一种机制和可重复使用的 Microsoft Excel 模板来估算按需容量模式的基本 DynamoDB 成本因素，例如写入、读取、存储、备份和恢复成本。它比 AWS Pricing Calculator 更精细，它会独立考虑基表和 GSIs 需求。它还考虑了每月项目数据的增长率，并预测了三年的成本。

先决条件和限制

先决条件

- DynamoDB 和 DynamoDB 数据模型设计基础知识
- DynamoDB 定价、WRU、RRU、存储以及备份和恢复的基础知识（有关更多信息，请参阅按需容量定价）
- 了解 DynamoDB 中的数据、数据模型以及项目大小
- 对 DynamoDB 的了解 GSIs

限制

- 该模板为您提供近似计算，但并非适用于所有配置。要获得更准确的估算值，您必须测量基表中每件商品的单个商品尺寸，然后 GSIs。

- 为了进行更准确的估计，您必须考虑平均每月每个项目的预期写入（插入、更新和删除）和读取次数。
- 此模式仅支持根据固定的数据增长假设估算未来几年的写入、读取、存储以及备份和恢复成本。

工具

Amazon Web Services

- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。

其他工具

- AWS 定价计算器是一款基于 Web 的规划工具，您可用来针对您的 AWS 用例创建估算值。

最佳实践

为了帮助降低成本，请考虑以下 DynamoDB 设计最佳实践。

- 分区键设计 - 使用高基数分区键均匀分配负载。
- 邻接列表设计模式 — 使用此设计模式进行 one-to-many管理和 many-to-many关系。
- 稀疏索引 — 为你使用稀疏索引。 GSIs创建 GSI 时，您可以指定分区键和排序键（可选）。只有基表中包含相应 GSI 分区键的项目才会显示在稀疏索引中。这有助于保持 GSIs 体积更小。
- 索引过载 - 使用相同的 GSI 为各种类型的项目编制索引。
- GSI 写入分片 - 明智地分片，将数据分布到各个分区，以实现更高效、更快速的查询。
- 大项目 - 仅在表中存储元数据，将 blob 保存在 Amazon S3 中，将引用保留在 DynamoDB 中。将大项目分成多个项目，并使用排序键有效地编制索引。

有关更多设计最佳实践，请参阅 Amazon DynamoDB Developer Guide。

操作说明

从 DynamoDB 数据模型提取项目信息

Task	描述	所需技能
获取项目大小。	1. 检查您要在桌面里存放多少不同类型的物品。 2. 要计算每个项目的大小（以千字节为单位），请添加每个属性的键和值大小。 3. 计算基表和每个 GSI 项目的大小。	数据工程师

估算写入成本。	要估算按需容量模式下的写入成本，首先必须测量一个月内 WRUs 将消耗多少容量。为此，您要考虑以下因素：
	• 一个月内针对每个项目执行的创建、更新和删除操作次数。
	• 可用数量 GSIs。单独考虑每项索引。
	  • 索引项的平均大小
	  • 索引的同步次数
	• 每月将在表格中添加多少新内容（例如组件或产品）？每个月添加的内容数量可能有所不同，但您可以根据您的业务案例假设平均增长率。
	
	有关更多信息，请参阅其他信息部分。
		数据工程师

Task	描述	所需技能
估算读取成本。	要估算按需模式下的读取成本，首先必须衡量一个月内 RRUs 将消耗多少读取成本。为此，您要考虑以下因素： • 可用数量 GSIs。单独考虑每项索引。 • 索引项的平均大小 • 每个产品每月的平均读取次数。 • DynamoDB 表中可用物品（组件或产品）的总数。	数据工程师、应用程序开发人员

Task	描述	所需技能
估算存储大小和成本。	首先，根据表中的项目大小估算平均每月存储要求。然后，通过将存储大小乘以您的 AWS 区域的每 GB 存储价格来计算存储成本。 如果您已输入数据来估计写入成本，则无需再次输入数据来计算存储大小。否则，要估计存储大小，您需要考虑以下因素： • 基于您的表格设计的模块（产品）中的数据项数。 • 以千字节为单位的平均项目大小。 • 可用数量 GSIs。单独考虑每项索引。 • 索引项的平均大小 • 每个月将在表格中添加多少新产品？ 每个月的新产品数量可能会有所不同，但您可以根据您的业务案例假设平均增长率。此示例每月平均使用 1000 万个新产品。	数据工程师

在 Excel 模板中输入项目与对象信息

Task	描述	所需技能
从“附件”部分下载 Excel 模板，并针对您的用例表进行调整。	1. 下载 Excel 模板。 2. 调整业务模块 GSIs，并根据您的表格设计进行调整。	数据工程师

Task	描述	所需技能
在 Excel 模板输入信息。	1. 更新工作表中的项目信息。仅更新橙色单元格内的数据。 2. 调整对象数目：每个月可以在表中添加多少对象？ 3. 更新您的 AWS 区域的 WRU 和 RRU 每百万分的价格。 4. 更新您的 AWS 区域每月每 GB 的存储和备份价格。 5. 更新您的 AWS 区域每 GB 的恢复价格。 在模板中，存在三个项目或实体：信息、元数据和关系。有两个 GSIs。对于您的用例，如果您需要更多项目，则请创建新行。如果您需要更多 GSIs，请复制现有的 GSI 块，然后粘贴以创建所需数量的 GSI 块。然后调整 SUM 和 TOTAL 列的计算。	数据工程师

相关资源

参考

- 按需容量的 Amazon DynamoDB 定价
- 适用于 DynamoDB 的 AWS 定价计算器
- 使用 DynamoDB 进行设计和架构的最佳实践
- DynamoDB 入门

指南和模式

- 使用 Amazon DynamoDB 对数据建模
- 估算 Amazon DynamoDB 表的存储成本

其他信息

写成本计算示例

DynamoDB 数据模型设计显示了一个产品的三个项目，平均项目大小为 4 KB。当您将新产品添加到 DynamoDB 基表时，它会消耗项目数量 * (项目大小/1 KB 写入单位) = 3 * (4/1) = 12 WRU。在此示例中，每写入 1 KB 时，产品消耗 1 WRU。

读取成本计算示例

要获得 RRU 估算值，请考虑每个项目在一个月内被读取的平均次数。例如，信息项平均每月被读取 10 次，元数据项平均被读取 2 次，关系项平均被读取 5 次。在示例模板中，所有组件的总 RRU = 每月创建的新组件数量 * 每个组件每月的 RRU = 1000 万 * 17 RRU = 每月 1.7 亿 RRU = 每月 1.7 亿 RRU。

每个月都会添加新的东西（组件或产品），并且产品总数将随着时间的推移而增长。因此，RRU 要求也会随时间的推移而增长。

- 第一个月的 RRU，消费量将为 1.7亿。
- 第二个月的 RRU 消耗量将为 2*1.7 亿=3.4 亿。
- 第三个月的RRU消耗量将为 3*1.7 亿=5.1 亿。

下图显示了每月 RRU 消耗量与成本预测。

请注意：图表中的价格仅供说明之用。要为您的使用案例创建准确的预测，请检查 AWS 定价页面，并在 Excel 工作表中使用这些价格。

存储、备份以及恢复成本计算示例

DynamoDB 存储、备份和恢复都是相互连接的。备份与存储直接相关，恢复与备份大小直接相关。随着表大小的增加，相应的存储、备份和恢复成本也会成比例增加。

存储大小和成本

根据您的数据增长率，存储成本将随着时间的推移而增加。例如，假设基表中组件或产品的平均大小 GSIs 为 11 KB，并且每月将向数据库表中添加 1000 万个新产品。在这种情况下，DynamoDB 表大小将增长 (11 KB * 1000 万) /1024/1024 = 每月 105 GB。在第一个月，您的表存储大小将为 105 GB，第二个月将为 105 + 105 = 210 GBs，依此类推。

- 对于您的 AWS 区域，第一个月存储成本将为每 GB 105 GB * 的存储价格。
- 第二个月的存储成本将为您所在地区每 GB 210 GB * 存储价格。
- 第三个月，您所在地区存储成本将为每 GB 315 GB * 的存储价格。

有关未来三年的存储大小和成本，请参阅存储大小和预测部分。

备份成本

根据您的数据增长率，备份成本将随着时间的推移而增加。使用 point-in-time-recovery (PITR) 开启连续备份时，持续备份费用基于每月平均存储 GB。在一个日历月中，平均备份大小将与表存储大小相同，尽管实际大小可能略有不同。因每个月都会添加新产品，因此总存储大小和备份大小将随着时间的推移而增长。例如，在第一个月，105 GB 的平均备份大小可能会在第二个月增长至 210 GB。

- 对于您的 AWS 区域，第一个月的备份费用将为每 GB 105 GB * 连续备份价格。
- 第二个月的备份费用将为您所在地区的每月 210 GB*每 GB 的连续备份价格。
- 第三个月的备份成本将为每月 315 GB * 您所在区域的每 GB 连续备份价格。
- 等等

备份成本包含在存储大小和成本预测部分的图表中。

恢复成本

当您在启用 PITR 的情况下进行连续备份时，恢复操作费用将根据恢复的大小而定。每次还原时，您都要根据恢复的数据量 (GB) 付费。如果您的表很大，并且一个月内执行多次恢复，那么成本会很高。

为了估算恢复成本，此示例假设您每月在月底执行一次 PITR 恢复。该示例使用每月平均备份大小作为该月的恢复数据大小。第一个月的平均备份大小为 105 GB，月底恢复时，恢复数据大小为 105 GB。第二个月将是 210 GBs，依此类推。

根据您的数据增长率，恢复成本将随着时间的推移而增加。

- 对于您的 AWS 区域，第一个月的恢复成本为每 GB 105 GB * 的恢复价格。
- 第二个月，恢复成本将为 210 GB * 您所在区域的每 GB 恢复价格。

- 第三个月，恢复成本将为 315 GB * 您所在区域的每 GB 恢复价格。

有关详细信息，请参阅 Excel 模板中的存储、备份和恢复选项卡以及下一节中的图表。

存储大小和成本预测

在模板中，实际计费存储大小是通过减去标准表类每月 25 GB 的免费套餐来计算的。在该表中，您将获得一个分为每月值的预测图表。

以下示例图表预测了未来 36 个日历月的每月存储大小（以 GB 为单位）、计费存储成本、按需备份成本和恢复成本。所有费用以美元表示。从图中可以清楚地看出，存储、备份和恢复成本随着存储大小的增加而成比例增加。

请注意，图表中所用的价格仅用于说明目的。要为您的用例创建准确的价格，请查看 AWS 定价页面，然后在 Excel 模板使用这些价格。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

估算 Amazon DynamoDB 表的存储成本

由 Moinul Al-Mamun (AWS) 创建

摘要

Amazon DynamoDB 是 NoSQL 事务数据库，即使在 PB 级规模下，也能提供个位数毫秒的延迟。这款广受欢迎的无服务器产品可 AWS 提供稳定的性能和可扩展性。您无需预置存储空间，您的单个表最多可以增长到 PB。

DynamoDB 会在整个月中持续监控您的表的大小，以确定您的存储费用。 AWS 然后按平均存储容量（以 GB 为单位）向您收费。您的桌面随着时间的推移而增长的越多，您的存储成本就会增长的越多。要计算存储成本，你可以使用AWS Pricing Calculator，但你需要提供表的大致大小，包括全局二级索引 (GSIs)，这在项目开始时很难估计。此外， AWS 定价计算器不考虑数据增长率。

这种模式提供了一种机制和可重复使用的 Microsoft Excel 模板，以计算 DynamoDB 存储大小和成本。它考虑了基表和 GSIs 独立表的存储需求。它通过考虑各个项目的大小和数据增长率随时间而计算存储尺寸。

要获取估计值，请将两个信息插入模板：

- 基表的单个项目大小（以千字节为单位）和 GSIs
- 平均一个月内可以向表格中添加的新对象或产品数量（例如，1000 万个）

该模板生成未来三年的存储和成本预测图，如以下示例所示。

先决条件和限制

先决条件

- DynamoDB 基础知识，包括 DynamoDB 存储和定价
- 了解 DynamoDB 中的数据、数据模型以及项目大小
- DynamoDB 全局二级索引知识 () GSIs

限制

- 该模板为您提供近似计算，但并非适用于所有配置。要获得更准确的估算值，您必须测量基表中每件商品的单个商品尺寸，然后 GSIs。

- 该模式支持仅根据固定数据增长假设来估算未来几年的存储尺寸和成本。

工具

Amazon Web Services

- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。

其他工具

- AWS P@@ ric ing Calculator 是一款基于 Web 的规划工具，可用于为 AWS 用例创建估算值。

操作说明

从 DynamoDB 数据模型提取项目信息

Task	描述	所需技能
获取项目大小。	1. 确定要在表格中存储多少种不同的物品类型。 2. 要计算每个项目的大小（以千字节为单位），请将每个属性的`Key`和`Value`大小相加。 3. 计算基表和每个 GSI 项目的大小。	数据工程师
获取一个月内添加的对象数量。	估计一个月内平均将有多少组件或对象添加至 DynamoDB 表中。	数据工程师

在 Excel 模板中输入项目与对象信息

Task	描述	所需技能
下载并调整 Excel 电子表格。	1. 从所附文档中下载 Excel 模板。 2. 根据您的表格设计调整业务模块和 GSIs。	数据工程师
在 Excel 模板输入信息。	1. 更新工作表中的项目信息。 2. 调整对象数目：每个月可以在表中添加多少对象？ 3. 更新您的每月每 GB 存储价格。 AWS 区域	数据工程师

相关资源

- 亚马逊 DynamoD@@ B 按需容量的定价
- AWS DynamoDB 定价计算器

其他信息

请注意，随附的模板预测仅存储大小和标准存储表类的成本。根据对存储成本的预测，并考虑单个项目的规模和产品或对象增长率，您可以估计以下内容：

- 数据导出成本
- 备份和恢复成本
- 数据存储要求。

Amazon DynamoDB 数据存储成本

DynamoDB 会持续监控您的表的大小，以确定您的存储费用。DynamoDB 通过将数据的原始字节大小加上数据的原始字节大小加上按项目的存储开销（这取决于您启用的功能）来衡量计费数据的大小。有关更多信息，请参阅 DynamoDB 开发人员指南。

数据存储的价格取决于您的表类。如果您使用 DynamoDB 标准表类，则每月存储的前 25 GB 是免费的。有关不同标准和不频繁访问表类别的存储成本的更多信息 AWS 区域，请参阅按需容量定价。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWR 报告估计 Oracle 数据库的 Amazon RDS 引擎大小

由 Abhishek Verma (AWS) 和 Eduardo Valentim (AWS) 编写

摘要

当您将 Oracle 数据库迁移至 Amazon Relational Database Service (Amazon RDS) 或 Amazon Aurora 时，计算目标数据库的 CPU、内存和磁盘 I/O 是一项关键要求。您可以通过分析 Oracle Oracle Automatic Workload Repository (AWR) 报告来估计目标数据库所需容量。此模式说明了如何使用 AWR 报告估计这些值。

源 Oracle 数据库可以位于本地或托管在亚马逊弹性计算云 (Amazon EC2) 实例上，也可以是 Amazon RDS for Oracle 数据库实例。目标数据库可是任何 Amazon RDS 或 Aurora 数据库。

> **Note**
>
> 如果您的目标数据库引擎是 Oracle，则容量估计会更加精确。对于其他 Amazon RDS 数据库，由于数据库架构差异，引擎大小可能会有所不同。

我们建议您在迁移 Oracle 数据库前进行性能测试。

先决条件和限制

先决条件

- Oracle Database Enterprise Edition 许可证和 Oracle Diagnostics Pack 许可证，用于下载 AWR 报告。

产品版本

- 版本 11g（版本 11.2.0.3.v1 及更高版本）和 12.2 以及 18c、19c 的所有 Oracle 数据库版本。
- 这种模式不包括 Oracle Engineered Systems 或 Oracle Cloud Infrastructure (OCI)。

架构

源技术堆栈

下列情况之一：

- 本地 Oracle 数据库
- EC2 实例上的 Oracle 数据库
- Amazon RDS for Oracle 数据库实例

目标技术堆栈

- 任何 Amazon RDS 或 Amazon Aurora 数据库

目标架构

有关完整迁移过程的信息，请参阅使用 AWS DMS 和 AWS SCT 将 Oracle 数据库迁移至 Aurora PostgreSQL 的模式。

自动化和扩缩

如果您有多个 Oracle 数据库要迁移并且想要使用其他性能指标，您可以按照博客文章根据 Oracle 性能指标大规模调整 Amazon RDS 实例中描述的步骤来自动化该过程。

工具

- Oracle Automatic Workload Repository (AWR)是一个内置在 Oracle 数据库中的存储库。它定期收集和存储系统活动和工作负载数据，然后由 Automatic Database Diagnostic Monitor (ADDM) 对其进行分析。AWR 定期拍摄系统性能数据的快照（默认情况下，每 60 分钟一次）并存储信息（默认情况下，最多 8 天）。 您可使用 AWR 视图和报告来分析这些数据。

最佳实践

- 若要计算目标数据库的资源要求，您可以使用单个 AWR 报告、多个 AWR 报告或动态 AWR 视图。我们建议您在高峰负载期间使用多个 AWR 报告估算处理这些峰值负载所需资源。此外，动态视图提供更多数据点，帮助您更准确地计算资源需求。
- 您应该仅估算计划迁移的数据库的 IOPS，而不是使用该磁盘的其他数据库和进程的 IOPS。
- 要计算数据库使用了多少 I/O，请不要使用 AWR 报告的负载配置文件部分中的信息。请改用“I/O 配置文件”部分（如果可用），或者跳至“实例活动统计信息”部分并查看物理读写操作的总值。
- 当您估计 CPU 利用率时，我们建议您使用数据库指标方法而不是操作系统 (OS) 统计信息，因为它仅基于数据库使用的 CPU。（操作系统统计信息还包含其他进程的 CPU 使用率。）您还应查看 ADDM 报告中与 CPU 相关的建议，以提高迁移后的性能。

- 在确定正确的实例类型时，请考虑特定实例大小的 I/O 吞吐量限制（Amazon Elastic Block Store (Amazon EBS) 吞吐量和网络吞吐量）。
- 迁移前运行性能测试，以验证引擎大小。

操作说明

创建 AWR 报告

Task	描述	所需技能
启用 AWR 报告。	若要启用该报告，请按照 Oracle 文档中的说明进行操作。	数据库管理员
查看保留期。	若要查看 AWR 报告的保留期，请使用以下查询。 `SQL> SELECT snap_interval,retention FROM dba_hist_wr_control;`	数据库管理员
生成快照。	如果 AWR 快照间隔的精细度不足以捕获峰值工作负载的峰值，您可以手动生成 AWR 报告。要生成手动 AWR 快照，请使用以下查询。 `SQL> EXEC dbms_workload_repository.create_snapshot;`	数据库管理员
查看最近的快照。	要查看最近的 AWR 快照，请使用以下查询。 `SQL> SELECT snap_id, to_char(begin_interval_time,'dd/MON/`	数据库管理员

Task	描述	所需技能
	yy hh24:mi') Begin_Interval, to_char(end_interval_time,'dd/MON/yy hh24:mi') End_Interval FROM dba_hist_snapshot ORDER BY 1;	

估算磁盘 I/O 需求

Task	描述	所需技能
选择方法。	IOPS 是存储设备上每秒输入和输出操作的标准衡量标准，包括读取和写入操作。 如果您要将本地数据库迁移至 AWS，则需要确定数据库使用的磁盘 I/O 峰值。您可以使用以下方法来估计目标数据库的磁盘 I/O： • AWR 报告的加载配置文件部分 • AWR 报告的实例活动统计信息部分（使用此部分获取 Oracle Database 12c 或更高版本） • AWR 报告的 I/O 配置文件部分（使用此部分获得 12c 之前的 Oracle 数据库版本） • AWR 视图 以下步骤介绍了这四种方法。	数据库管理员

Task	描述	所需技能
选项 1：使用负载配置文件。	下表显示了 AWR 报告的负载配置文件部分的示例。 ⚠ Important 要获得更准确的信息，我们建议您使用选项 2（I/O 配置文件）或选项 3（实例活动统计信息）来代替负载配置文件。 （每秒） 每笔交易 每位高管 每次通话 数据库时间（秒 26.6 0.2 0.00 0.02 数据库 CPU 18.0 0.1 0.00 0.01 后台 CPU 0.2 0.0 0.00 0.00	数据库管理员

Task	描述		所需技能
	重做大小（字节）	2,45 17,0 .9	
	逻辑读取（块	3,37 23,4 .5	
	块更改：	21,6 150. 4	
	物理读取（块	13,5 94.4 7	
	物理写入（块	3,46 24.1 7	
	读取IO请求：	3,58 24.9 6	

Task	描述	所需技能
	写入 IO 请求： 574.1 4.0 读取 IO (MB) 106.1 0.7 写入 IO (MB) 27.1 0.2 IM 扫描行： 0.0 0.0 会话逻辑读取 IM： 用户调用： 1,245 8.7	

Task	描述	所需技能
	解析 (SQL 4,62(32.2 硬解析 (SQL 8.9 0.1 SQL 工作区 (MB) 824.9 5.7 登录： 1.7 0.0 执行次数 (SQL 136,(950.4 回滚： 22.9 0.2 事务： 143.8 根据这些信息，您可以按如下方式计算 IOPs 吞吐量： IOPS = 读取 I/O 请求：+ 写入 I/O 请求 = 3,586.8 + 574.7 = 4134.5	

Task	描述	所需技能
	吞吐量 = 物理读取（块）+ 物理写入（块）= 13,575.1 + 3,467.3 = 17,042.4 由于 Oracle 中的块大小为 8 KB，因此您可按如下方式计算总吞吐量： 以 MB 为单位的总吞吐量是 17042.4 * 8 * 1024/1024/1024 = 133.2 MB **⚠ Warning** 请勿使用负载配置文件来估算实例大小。它不如实例活动统计数据或 I/O 配置文件精确。	

Task	描述	所需技能
选项 2：使用实例活动统计信息。	如果您使用的是 12c 之前的 Oracle 数据库，则可使用 AWR 报告的实例活动统计信息部分来估计 IOPS 和吞吐量。下表显示了本部分的示例。 Statist 总计 （每秒） 每次传输 物理读取总数 IO 请求数 2,547,,217 3,610. 25.11 物理读取总字节数 80,776,124, 114,48 26.26 796,149 8 物理写入总量 IO 请求数 534,19 08 757.1 5.27 物理写入总字节数 25,517 8,849, 36,165 1.84 251,508 8 根据这些信息，您可按如下方式计算总 IOPS 和吞吐量：	数据库管理员

Task	描述	所需技能
	IOPS 总数 = 3,610.28 + 757.11 = 4367 总计 Mbps = 114,482,426.26 + 36,165,631.84 = 150648058.1/1024/1024 = 143 Mbps	

Task	描述	所需技能
选项 3：使用 I/O 配置文件。	在 Oracle Database 12c，AWR 报告包括一个 I/O 配置文件部分，该部分在单个表中显示所有信息，并提供有关数据库性能的更准确的数据。下表显示了本部分的示例。 每秒读取+写入次数 / 每秒读取次数 / 每秒写入次数 请求总数：4,367. / 3,610. / 757.1 数据库请求：4,161. / 3,586. / 574.7 优化的请求：0.0 / 0.0 / 0.0 重做请求：179.3 / 2.8 / 176.6 总计 (MB)：143.7 / 109.2 / 34.5 数据库 (MB)：133.1 / 106.1 / 27.1	数据库管理员

Task	描述	所需技能
	优化总数 (MB)： 0.0 0.0 0.0 重做 (MB)： 7.6 2.7 4.9 数据库（块） 17,042 13,575 3,467.3 通过缓冲区缓存（块）： 5,898. 5,360. 537.6 直连（块） 11,143 8,214. 2,929.7 该表提供了以下吞吐量和总 IOPS 值： 吞吐量 = 143 MBPS（从标有总计的第五行、第二列开始） IOPS = 4,367.4(从标有总计的第一行、第二列开始)	

Task	描述	所需技能
选项 4：使用 AWR 视图。	您可使用 AWR 视图查看相同的 IOPS 和吞吐量信息。若要获取此信息，请使用以下查询： `break on report compute sum of Value on report select METRIC_NAME,avg(AVERAGE) as "Value" from dba_hist_sysmetric_summary where METRIC_NAME in ('Physical Read Total IO Requests Per Sec','Physical Write Total IO Requests Per Sec') group by metric_name;`	数据库管理员

估计 CPU 需求

Task	描述	所需技能
选择方法。	您可通过三种方式估算目标数据库所需 CPU： • 通过处理器的实际可用内核 • 通过基于操作系统统计信息的已用内核 • 通过基于数据库统计信息的已用内核 如果您要查看已使用的内核数，我们建议您使用数据库指	数据库管理员

Task	描述	所需技能
	标方法而不是操作系统统计信息，因为它仅基于您计划迁移的数据库使用的 CPU。（操作系统统计信息还包含其他进程的 CPU 使用率。）您还应查看 ADDM 报告中与 CPU 相关的建议，以提高迁移后的性能。 您还可根据 CPU 生成估算需求。如果您使用不同的 CPU 代，则可以按照白皮书《揭开 v CPUs 数的神秘面纱，实现最佳工作负载性能》中的说明来估算目标数据库所需的 CPU。	

Task	描述	所需技能

选项 1：根据可用内核估算需求。	在 AWR 报告：
	• CPUs 指的是逻辑和虚拟 CPUs。
	• 内核是物理 CPU 芯片组的处理器数量。
	• 插槽是一种将芯片连接至主板的物理设备。多核处理器的插槽带多个 CPU 内核。
	
	您可通过两种方式估算可用内核：
	• 通过使用 OS 命令
	• 通过使用 AWR 报告
	
	使用操作系统命令估算可用内核
	
	使用以下命令计算处理器内核。
	`$ cat /proc/cpuinfo | grep "cpu cores"|uniq`
	`cpu cores : 4`
	`cat /proc/cpuinfo | egrep "core id|physical id" | tr -d "\n" | sed s/physical/\\nphysical/g | grep -v ^$ | sort | uniq | wc -l`
	
	使用以下命令计算处理器插槽数量。
		数据库管理员

Task	描述	所需技能

	`grep "physical id" /proc/cpuinfo | sort -u`
	`physical id : 0`
	`physical id : 1`
	
	Note
	我们不建议使用 nmon 和 sa r 等操作系统命令提取 CPU 利用率。原因是这些计算包括其他进程的 CPU 使用率，可能无法反映数据库使用的实际 CPU。
	
	通过 AWR 报告估算可用内核
	
	您也可从 AWR 报告的第一部分得出 CPU 使用率。报告摘录如下。
	
	数据库名称 | 数据库 ID | 实例 | 实例数 | 开始时间 | 发布版本 | RA
	X | <DE | XX> | 1 | 05-Sep 23:09 | 12.1 0 | 否2	

Task	描述	所需技能
	主机名称 平台 CPU 内核 套接字 内存 (GB) <host> Linux x86 64 位 80 80 2 441.7 在此示例中，CPUs 计数为 80，这表示它们是逻辑的（虚拟）CPUs。您还可以看到，此配置有两个插槽，每个插槽上有一个物理处理器（总共两个物理处理器），每个物理处理器或插槽有 40 个内核。	

<table>
<tr><th>Task</th><th>描述</th><th>所需技能</th></tr>
<tr><td>选项 2：使用操作系统统计信息估计 CPU 使用率。</td><td>您可以直接在操作系统中查看操作系统 CPU 使用率统计信息（使用 sar 或其他主机操作系统实用程序），也可以通过查看 AWR 报告的操作系统统计信息部分中的 IDLE/(IDLE+BUSY) 值来查看操作系统 CPU 使用率统计信息。您可以直接从 v$osstat 中查看消耗的 CPU 秒数。AWR 和 Statspack 报告还在操作系统统计信息部分显示了这些数据。

如果同一个框上有多个数据库，则它们对于 BUSY_TIME 都有相同的 v$osstat 值。

<table><tr><th>Statistic</th><th>值</th><th>终止值</th></tr><tr><td>FREE_MRY_BYT</td><td>6,810,67,248</td><td>12,280,799,232</td></tr><tr><td>INACTIVMEMORTES</td><td>175,627,33,632</td><td>160,380,653,568</td></tr><tr><td>SWAP_F_BYTES</td><td>17,145,64,336</td><td>17,145,872,384</td></tr><tr><td>BUSY_T</td><td>1,305,56,937</td><td></td></tr><tr><td>IDLE_TI</td><td>4,312,71,839</td><td></td></tr></table></td><td>数据库管理员</td></tr>
</table>

Task	描述		所需技能
	IOWAIT_ME	53,417,14	
	NICE_TI	29,815	
	SYS_TIM	148,567,70	
	USER_T	1,146,91,783	
	LOAD	25	29
	VM_IN_E ES	593,920	
	VM_OUT TES	327,680	
	PHYSIC/ MEMOR TES	474,362,17,152	
	NUM_CF	80	
	NUM_CF ORES	80	
	NUM_CF OCKETS	2	
	GLOBAL CEIVE_S E_MAX	4,194,30	
	GLOBAL ND_SIZE AX	2,097,15	

Task	描述	所需技能
	TCP_RE 87,380 VE_SIZE EFAULT TCP_RE 6,291,45 VE_SIZE AX TCP_RE 4,096 VE_SIZE IN TCP_SE 16,384 SIZE_DE ULT TCP_SE 4,194,30 SIZE_MA TCP_SE 4,096 SIZE_MI 如果系统中没有其他主要的 CPU 使用者，请使用以下公式计算 CPU 利用率的百分比： 利用率 = 忙碌时间/总时间 繁忙时间 = 需求 = v$osstat.BUSY_TIME C = 总时间（忙碌 + 空闲） C = 容量 = v$ostat.BUSY_TIME + v$ostat.IDLE_TIME	

Task	描述	所需技能
	利用率 = BUSY_TIME/(BUSY_TIME + IDLE_TIME) = -1,305,569,937/(1,305,569,937 + 4,312,718,839) = 23% 利用率	

Task	描述	所需技能
选项 3：使用数据库指标估算 CPU 利用率。	如果系统中运行多个数据库，您可以使用报告开头显示的数据库指标。 		快照 ID	快照时间	会话	光标/会话
---	---	---	---	---
开始快照：	1846	2020 年 9 月 28 日 09:0(	1226	35.8
结束快照：	1854	2020 年 10 月 06 日 13：00：20	1876	41.1
用时：		11,7（分钟）		
数据库		312,(0（分钟）				数据库管理员

Task	描述	所需技能
	时 间： 要获取 CPU 利用率的指标，请使用以下公式： 数据库 CPU 使用率（可用的 CPU 能耗百分比）= CPU 时间/NUM_CPUS /已用时间 其中，CPU 使用率由 CPU 时间描述，表示在 CPU 上花费的时间，而非等待 CPU 的时间。此计算结果为： = 312,625.40/11,759.64/80 = 33% 使用中 CPU 内核数量 (33%) * 80 = 26.4 个内核 内核总数 = 26.4 * (120%) = 31.68 个内核 您可使用这两个值中的较大值来计算 Amazon RDS 或 Aurora 数据库实例的 CPU 使用率。 **Note** 在 IBM AIX 上，计算出的利用率与操作系统或数据库中的值不匹	

Task	描述	所需技能
	配。这些值在其他操作系统确实匹配。	

估算内存需求

Task	描述	所需技能
通过内存统计数据估算内存需求。	您可以使用 AWR 报告计算源数据库的内存并在目标数据库中进行匹配。您还应该检查现有数据库的性能并减少内存需求以节省成本，或增加内存需求以提高性能。这需要对 AWR 响应时间和应用程序的服务水平协议（SLA）进行详细分析。使用 Oracle 系统全局区域 (SGA) 和程序全局区域 (PGA) 使用率之和作为 Oracle 的估计内存利用率。为操作系统额外添加 20%，以确定目标内存大小要求。对于 Oracle RAC，使用所有 RAC 节点上的估计内存利用率的总和并减少总内存，因为它存储在公共块上。 1. 检查“实例效率百分比”表中的指标。表格使用以下术语： • 缓冲区命中百分比是在缓冲区高速缓存中发现特定块而不是执行物理 I/O 的次数百分比。为了提高性能，请以 100% 为目标。	数据库管理员

Task	描述	所需技能
	• 缓冲区 Nowait% 应接近 100%。 • 锁定命中百分比应接近 100%。 • % 非解析 CPU 是在非解析活动中花费的 CPU 时间的百分比。此值应该接近 100%。 实例效率百分比（目标 100%） 缓冲区 Nowa%：99.99 重做 NoWa%：100.00 缓冲区命中率%：99.84 内存中排序%：100.00 库命中%：748.7 软解析%：99.81 执行到解 96.61 锁定命 100.00	

Task	描述	所需技能
	析 %： 中 %： 解析 CPU 到解析已用时间 %： 72.73 % 非解析 CPU 99.21 闪存缓存命中 %： 0.00 在这个例子中，所有指标看起来都很好，因此您可以使用现有数据库的 SGA 和 PGA 作为容量规划要求。 2. 检查内存统计信息部分，并计算 SGA/PGA。 开始 结束	

Task	描述	所需技能
	主机内存 (MB)： 452,387 452,387.3 SGA 使用量 (MB)： 220,544 220,544.0 PGA 使用量 (MB)： 36,874.9 45,270.0 使用中实例总内存 = SGA + PGA = 220 GB + 45 GB = 265 GB 添加 20% 缓冲区： 实例总内存 = 1.2 * 265 GB = 318 GB 由于 SGA 和 PGA 占主机内存 70%，因此总内存需求为： 主机总内存 = 318/0.7 = 464 GB **Note** 当您迁移到 Amazon RDS for Oracle 时，PGA 和 SGA 是根据预定义的公式进行预	

Task	描述	所需技能
	先计算的。确保预先计算的值接近您的估值。	

确定目标数据库数据库实例类型

Task	描述	所需技能
根据磁盘 I/O、CPU 和内存估计确定数据库实例类型。	根据前面步骤中的估算，目标 Amazon RDS 或 Aurora 数据库的容量应为： • 68 个 CPU 内核 • 143 MBPS 吞吐量 • 磁盘 I/O 的 4367 IOPS • 464 GB 的内存 在目标 Amazon RDS 或 Aurora 数据库中，您可将这些值映射到 db.r5.16xlarge 实例类型，该实例类型的容量为 32 个内核、512 GB 的 RAM 和 13,600 Mbps 的吞吐量。有关更多信息，请参阅 AWS Blog 文章基于 Oracle 性能指标大规模调整 Amazon RDS 实例大小。	数据库管理员

相关的资源

- Aurora 数据库实例类（Amazon Aurora 文档）
- Amazon RDS 数据库实例存储（Amazon RDS 文档）
- AWS 矿工工具（GitHub 存储库）

使用 AWS DMS 将 Amazon RDS for SQL Server 表导出至 S3 存储桶

由 Subhani Shaik (AWS) 编写

摘要

Amazon Relational Database Service (Amazon RDS) for SQL Server 不支持将数据加载到 Amazon Web Services (AWS) 云上的其他数据库引擎链接服务器上。相反，您可以使用 AWS Database Migration Service (AWS DMS) 将 Amazon RDS for SQL Server 表导出到 Amazon Simple Storage Service (Amazon S3) 存储桶，其中的数据可供其他数据库引擎使用。

AWS DMS 可帮助您快速安全地将数据库迁移到 AWS。源数据库可在迁移过程中保持全面运行，从而最大程度地为依赖该数据库的应用程序减少停机时间。AWS DMS 可以在最广泛使用的商用和开源数据库之间迁移数据。

此模式在配置 AWS DMS 端点时使用 AWS Secrets Manager。 服务可帮助您保护访问您的应用程序、服务和 IT 资源所需密钥。您可以使用 Secrets Manager 在数据库凭证、API 密钥和其他密钥的整个生命周期内对其进行轮换、管理和检索。用户和应用程序通过调用 Secrets Manager 来检索机密，从而减少对敏感信息进行硬编码的需要。Secrets Manager 使用 Amazon RDS、Amazon Redshift 和 Amazon DocumentDB 的内置集成提供密钥轮换。此外，该服务还可扩展到其他类型的机密，包括 API 密钥和 OAuth 令牌。Secrets Manager 使您能够使用精细权限控制对机密的访问，并集中审计 AWS Cloud、第三方服务和本地资源的密钥轮换。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 一个 S3 存储桶
- 虚拟私有云（VPC）
- 数据库子网
- Amazon RDS for SQL Server
- 一个 AWS Identity and Acess Management (IAM) 角色，该角色具有代表 Amazon RDS 实例对 S3 存储桶的访问（列出、获取和放置对象）。
- 用于存储 RDS 实例凭证的 Secrets Manager。

架构

技术堆栈

- Amazon RDS for SQL Server
- AWS DMS
- Amazon S3
- AWS Secrets Manager

目标架构

下图显示了借助 AWS DMS 将数据从 Amazon RDS 实例导入到 S3 存储桶的架构。

1. 通过源端点连接到源 Amazon RDS 实例的 AWS DMS 迁移任务
2. 从源 Amazon RDS 实例复制数据
3. 通过目标端点连接到目标 S3 存储桶的 AWS DMS 迁移任务
4. 以逗号分隔值 (CSV) 格式将复制的数据导出至 S3 存储桶

工具

Amazon Web Services

- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到 Amazon Web Services Cloud，或者在云和本地设置的组合之间迁移。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- Amazon Relational Database Service (Amazon RDS) 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展关系数据库。
- Amazon Simple Storage Service(Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Secrets Manager 帮助您将代码中的硬编码凭证（包括密码）替换为对 Secrets Manager 的 API 调用，以便以编程方式检索密钥。

其他服务

- Microsoft SQL Server Management Studio (SSMS) 是一款用于管理 SQL Server 的工具，包括访问、配置和管理 SQL Server 组件。

操作说明

配置 Amazon RDS for SQL Server 实例

Task	描述	所需技能
创建 Amazon RDS for SQL Server 实例。	1. 打开 AWS 管理控制台，选择 RDS，然后使用标准创建选项创建具有所需版本的 Amazon RDS 实例，例如 SQL Server Express 版、SQL Server 标准版或 SQL Server 企业版。对于版本，请选择 2016 年或更高版本。 2. 在模板下，选择开发/测试。	数据库管理员、工程师 DevOps
设置实例的凭证。	1. 为实例输入名称。 2. 提供 Amazon RDS 实例的用户名和密码。	数据库管理员、工程师 DevOps
配置实例类别、存储、自动扩缩以及可用性。	1. 从列表中选择数据库实例类：标准类、内存优化和突发性能。选择可分配为此数据库实例规划的工作负载所需计算、网络和内存容量的数据库实例类型。有关更多信息，请参阅 AWS 文档。 2. 从列表中选择存储类型：通用型 SSD、预调配 IOPS SSD或磁性介质。根据需要分配默认存储大小。	数据库管理员、工程师 DevOps

Task	描述	所需技能
	3. 选择启用存储自动扩缩，根据容量规划增加 Amazon RDS 存储空间。 4. AWS DMS 支持带有复制实例的多可用区部署。如果可用区、内部硬件或网络发生中断，AWS DMS 将创建一个备用实例，并通过自动失效转移到备用副本来提供高可用性 (HA)。根据您导入的大小，选择相应选项。	
指定虚拟私有云（VPC）、子网组、公共访问和安全组。	根据需要选择 VPC、数据库子网组和 VPC 安全组创建 Amazon RDS 实例。遵循最佳实践，例如： • 请勿启用 RDS 数据库实例的公共访问权限。 • 请勿在安全组中使用 CIDR 0.0.0/0。 • 仅使用所需 IP 地址和端口详细信息来访问 RDS 实例。	数据库管理员、工程师 DevOps
配置监控、备份和维护。	1. 指定所需备份选项。默认情况下，启用备份，保留期为一天。 2. 选择相应的 auto 次要版本升级和维护窗口设置，以将 Amazon RDS 的待处理修改或维护应用于数据库。 3. 选择 Create database(创建数据库)。	数据库管理员、工程师 DevOps

设置数据库和示例数据

Task	描述	所需技能
创建表和加载示例数据。	在新数据库中创建一个表。使用其他信息部分中的示例代码将数据加载到表中。	数据库管理员、工程师 DevOps

设置凭证

Task	描述	所需技能
创建密钥。	1. 打开 Secrets Manager 控制台并选择存储新密钥。 2. 输入 Amazon RDS for SQL Server 数据库用户名和密码。 此密钥将用于 AWS DMS 源端点。	数据库管理员、工程师 DevOps

在数据库和 S3 存储桶间设置访问权限

Task	描述	所需技能
创建 IAM 角色以便访问 Amazon RDS	1. 在控制台上，选择 IAM，然后创建一个 IAM 角色，授予 S3 存储桶对 Amazon RDS 的读/写访问权限。 2. 在功能，选择S3 集成。	数据库管理员、工程师 DevOps

创建 S3 存储桶

Task	描述	所需技能
创建 S3 存储桶。	要保存来自 Amazon RDS for SQL Server 的数据，请在控制台上选择 S3，然后选择创建存储桶。确保 S3 存储桶不可公开访问。	数据库管理员、工程师 DevOps

在 AWS DMS 和 S3 存储桶间设置访问权限

Task	描述	所需技能
创建一个用于访问 Athena 和 Amazon S3 的 IAM 角色	创建一个 IAM 角色，允许 AWS DMS 列出、获取与放置 S3 存储桶中的对象。	数据库管理员、工程师 DevOps

配置 AWS DMS

Task	描述	所需技能
为源创建 AWS DMS 端点。	1. 在控制台上，选择 Database Migration Service，然后选择端点。创建源端点，选中选择 RDS 数据库实例复选框。 2. 对于源引擎，请选择 Microsoft SQL Server。 3. 在访问端点数据库，选择 AWS Secrets Manager，然后输入您之前创建的密钥和 IAM 角色以及数据库名称。 4. 测试源端点。	数据库管理员、工程师 DevOps

Task	描述	所需技能
为目标创建 AWS DMS 端点。	创建目标端点，选择 Amazon S3 作为目标引擎。 提供您之前创建的 IAM 角色的 S3 存储桶名称和文件夹名称。	数据库管理员、工程师 DevOps
创建 AWS DMS 复制实例。	在同一个 VPC、子网和安全组中，创建 AWS DMS 复制实例。有关数据库实例类选项的详细信息，请参阅 AWS 文档。	数据库管理员、工程师 DevOps
启动 AWS DMS 迁移任务。	要将数据从 Amazon RDS for SQL Server 导出至 S3 存储桶，请创建数据库迁移任务。对于迁移类型，请选择迁移现有数据。选择您创建的 AWS DMS 端点和复制实例。	数据库管理员、工程师 DevOps

将数据导出至 S3 存储桶

Task	描述	所需技能
运行数据库迁移任务。	若要导出 SQL Server 表数据，请启动数据库迁移任务。该任务将以 CSV 格式将数据从 Amazon RDS for SQL Server 导出至 S3 存储桶。	数据库管理员、工程师 DevOps

清理资源

Task	描述	所需技能
删除资源。	为了避免产生额外费用，请使用控制台按以下顺序删除资源： 1. 迁移任务 2. 复制实例 3. 了解如何查看、监控和管理 SageMaker 端点。 4. S3 存储桶 5. 数据库实例	数据库管理员、工程师 DevOps

相关资源

- AWS DMS
- Amazon S3
- Amazon RDS for SQL Server
- Amazon S3 集成

其他信息

若要创建数据库和表并加载示例数据，请使用以下代码。

```
--Step1: Database creation in RDS SQL Server
CREATE DATABASE [Test_DB]
 ON  PRIMARY
( NAME = N'Test_DB', FILENAME = N'D:\rdsdbdata\DATA\Test_DB.mdf' , SIZE = 5120KB ,
 FILEGROWTH = 10%)
 LOG ON
( NAME = N'Test_DB_log', FILENAME = N'D:\rdsdbdata\DATA\Test_DB_log.ldf' , SIZE =
 1024KB , FILEGROWTH = 10%)
GO

--Step2: Create Table
```

```
USE Test_DB
GO
Create Table Test_Table(ID int, Company Varchar(30), Location Varchar(20))

--Step3: Load sample data.
USE Test_DB
GO
Insert into Test_Table values(1,'AnyCompany','India')
Insert into Test_Table values(2,'AnyCompany','USA')
Insert into Test_Table values(3,'AnyCompany','UK')
Insert into Test_Table values(4,'AnyCompany','Hyderabad')
Insert into Test_Table values(5,'AnyCompany','Banglore')
```

在 Aurora PostgreSQL 中处理动态 SQL 语句中的匿名块

创建者：anuradha chintha (AWS)

摘要

此模式向您展示了如何避免在处理动态 SQL 语句中的匿名块时出现的错误。当您使用 AWS Schema Conversion Tool 将 Oracle 数据库转换为 Aurora PostgreSQL-Compatible Edition 数据库时，您会收到一条错误消息。为避免错误，必须知道 OUT 绑定变量的值，但是要等到运行 SQL 语句之后才能知道 OUT 绑定变量的值。该错误是由于 AWS Schema Conversion Tool（AWS SCT）不理解动态 SQL 语句中的逻辑造成的。AWS SCT 无法转换 PL/SQL 代码（即函数、过程和软件包）中的动态 SQL 语句。

先决条件和限制

先决条件

- 有效的 Amazon Web Services account
- Aurora PostgreSQL 数据库（DB）实例
- Amazon Relational Database Service（Amazon RDS）for Oracle 数据库实例
- PostgreSQL 交互式终端 (psql)
- SQL *Plus
- 目标数据库中的 `AWS_ORACLE_EXT` 架构（AWS SCT 扩展包的一部分）
- 最新版本的 AWS Schema Conversion Tool（AWS SCT）及其所需驱动程序

架构

源技术堆栈

- 本地 Oracle 数据库 10g 及更高版本

目标技术堆栈

- Amazon Aurora PostgreSQL
- Amazon RDS for PostgreSQL
- AWS Schema Conversion Tool (AWS SCT)

迁移架构

下图显示了如何使用 AWS SCT 和 Oracle 0UT 绑定变量来扫描应用程序代码中是否存在嵌入式 SQL 语句，并将代码转换为 Aurora 数据库可以使用的兼容格式。

图表显示了以下工作流：

1. 使用 Aurora PostgreSQL 作为目标数据库，为源数据库生成 AWS SCT 报告。
2. 识别动态 SQL 代码块中的匿名块（AWS SCT 对此提出了错误）。
3. 手动转换代码块并将代码部署到目标数据库上。

工具

Amazon Web Services

- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩展 PostgreSQL 部署。
- Amazon Relational Database Service (Amazon RDS) for Oracle 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展 Oracle 关系数据库。
- AWS Schema Conversion Tool（AWS SCT）通过自动将源数据库架构和大部分数据库代码对象转换为与目标数据库兼容的格式，帮助您预测异构数据库迁移。

其他工具

- pgAdmin 允许您连接数据库服务器并与之交互。
- Oracle SQL Developer 是一个集成的开发环境，您可以使用它来开发和管理 Oracle 数据库中的数据库。您可以使用 SQL *Plus 或 Oracle SQL Developer 来实现这种模式。

操作说明

配置 Oracle 源数据库

Task	描述	所需技能
在亚马逊 RDS 或亚马逊上创建 Oracle 实例 EC2。	要在 Amazon RDS 上创建 Oracle 数据库实例，请参阅	数据库管理员

Task	描述	所需技能
	Amazon RDS 文档中的创建 Oracle 数据库实例并连接到 Oracle 数据库实例上的数据库。 要在亚马逊弹性计算云 (亚马逊 EC2) 上创建 Oracle 数据库实例，请参阅 AWS Prescriptive G EC2 uidence 文档中的 Amazon for Oracle。	
创建用于迁移的数据库架构和对象。	您可以使用 Amazon Cloud Directory 创建数据库架构。有关更多信息，请参阅 Cloud Directory 文档中的创建架构。	数据库管理员
配置入站和出站安全组。	要创建和配置安全组，请参阅 Amazon RDS 文档中的使用安全组控制访问权限。	数据库管理员
确认数据库正在运行。	要检查数据库的状态，请参阅 Amazon RDS 文档中的查看 Amazon RDS 事件。	数据库管理员

配置 Aurora PostgreSQL 目标数据库

Task	描述	所需技能
在 Amazon RDS 中创建 Aurora PostgreSQL 实例。	要创建 Aurora PostgreSQL 数据库实例，请参阅 Amazon RDS 文档中的创建数据库集群并连接到 Aurora PostgreSQL 数据库集群上的数据库。	数据库管理员
配置入站和出站安全组。	要创建和配置安全组，请参阅 Aurora 文档中的通过创建安全	数据库管理员

Task	描述	所需技能
	组提供对 VPC 中数据库集群的访问。	
确认 Aurora PostgreSQL 数据库正在运行。	要检查数据库的状态，请参阅 Aurora 文档中的查看 Amazon RDS 事件。	数据库管理员

设置 AWS SCT

Task	描述	所需技能
将 AWS SCT 连接到源数据库。	要将 AWS SCT 连接到源数据库，请参阅 AWS SCT 文档中的作为源连接到 PostgreSQL。	数据库管理员
将 AWS SCT 连接到目标数据库。	要将 AWS SCT 连接到目标数据库，请参阅 AWS Schema Conversion Tool 用户指南中的什么是 AWS Schema Conversion Tool？。	数据库管理员
在 AWS SCT 中转换数据库架构，并将自动转换后的代码保存为 SQL 文件。	要保存 AWS SCT 转换后的文件，请参阅 AWS Schema Conversion Tool 用户指南中的在 AWS SCT 中保存和应用转换后的架构。	数据库管理员

迁移代码

Task	描述	所需技能
获取用于手动转换的 SQL 文件。	在 AWS SCT 转换后的文件中，提取需要手动转换的 SQL 文件。	数据库管理员
更新脚本。	手动更新 SQL 文件。	数据库管理员

相关资源

- Amazon RDS
- Amazon Aurora 功能

其他信息

下面的示例代码显示了如何配置 Oracle 源数据库：

```
CREATE or replace PROCEDURE calc_stats_new1 (
  a NUMBER,
  b NUMBER,
  result out NUMBER)
IS
BEGIN
result:=a+b;
END;
/
```

```
set serveroutput on ;

DECLARE
  a NUMBER := 4;
  b NUMBER := 7;
  plsql_block VARCHAR2(100);
  output number;
BEGIN
  plsql_block := 'BEGIN calc_stats_new1(:a, :b,:output); END;';
  EXECUTE IMMEDIATE plsql_block USING a, b,out output;
```

```
  DBMS_OUTPUT.PUT_LINE('output:'||output);

END;
```

下面的示例代码显示了如何配置 Aurora PostgreSQL 目标数据库：

```
 w integer,
 x integer)
RETURNS integer
AS
$BODY$
DECLARE
begin
return w + x ;
end;
$BODY$
LANGUAGE  plpgsql;


CREATE OR REPLACE FUNCTION test_pg.init()
RETURNS void
AS
$BODY$
BEGIN
if aws_oracle_ext.is_package_initialized
      ('test_pg' ) then
      return;
    end if;
    perform aws_oracle_ext.set_package_initialized
      ('test_pg' );

PERFORM aws_oracle_ext.set_package_variable('test_pg', 'v_output', NULL::INTEGER);
PERFORM aws_oracle_ext.set_package_variable('test_pg', 'v_status', NULL::text);
END;
$BODY$
LANGUAGE  plpgsql;


DO $$
declare
v_sql text;
v_output_loc int;
a integer :=1;
```

```
b integer :=2;
BEGIN
perform  test_pg.init();
--raise notice 'v_sql %',v_sql;
execute 'do $a$ declare v_output_l int; begin select * from test_pg.calc_stats_new1('||
a||','||b||') into v_output_l;
PERFORM aws_oracle_ext.set_package_variable(''test_pg'', ''v_output'', v_output_l) ;
 end; $a$'  ;
v_output_loc := aws_oracle_ext.get_package_variable('test_pg', 'v_output');
raise notice 'v_output_loc %',v_output_loc;
END ;
$$
```

在 Aurora PostgreSQL 兼容中处理重载的 Oracle 函数

由 Sumana Yanamandra (AWS) 编写

摘要

您从本地 Oracle 数据库迁移到 Amazon Aurora PostgreSQL 兼容版的代码可能包含重载函数。这些函数具有相同的定义，即相同的函数名称以及相同的输入 (IN) 参数数量和数据类型，但输出 (OUT) 参数的数据类型或数量可能会有所不同。

这些参数不匹配可能会导致 PostgreSQL 出现问题，因为很难确定要运行哪个函数。此模式说明了在将数据库代码迁移到 Aurora PostgreSQL 兼容版时如何处理重载函数。

先决条件和限制

先决条件

- 作为源数据库的 Oracle 数据库实例
- 一个与 Aurora PostgreSQL 兼容的数据库实例作为您的目标数据库（请参阅 Aurora 文档中的说明）

产品版本

- Oracle 数据库 9i 或更高版本
- Oracle SQL 开发人员版本 18.4.0.376
- pgAdmin 4 客户端
- Aurora PostgreSQL 兼容版本 11 或更高版本（请参阅 Aurora 文档中的识别 Amazon Aurora PostgreSQL 的版本）

工具

Amazon Web Services

- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩展 PostgreSQL 部署。

其他工具

- Oracle SQL Developer 是免费的集成开发环境，用于在传统部署和云部署中在 Oracle 数据库中使用 SQL。
- pgAdmin 是一种适用于 PostgreSQL 的开源管理工具。它提供了一个图形界面，可帮助您创建、维护和使用数据库对象。

操作说明

创建一个简单函数

Task	描述	所需技能
在 PostgreSQL 中创建一个具有一个输入参数和一个输出参数的函数。	以下示例说明了 Aurora PostgreSQL 兼容中名为 `test_overloading` 的函数。此函数有两个参数：一个输入文本参数和一个输出文本参数。 `CREATE OR REPLACE FUNCTION public.test_overloading( str1 text, OUT str2 text) LANGUAGE 'plpgsql' COST 100 VOLATILE AS $BODY$ DECLARE BEGIN str2 := 'Success'; RETURN ; EXCEPTION WHEN others THEN RETURN ; END; $BODY$;`	数据工程师，Aurora PostgreSQL 兼容

Task	描述	所需技能

在 PostgreSQL 中运行该函数。	运行您在上一步中创建的函数。
	`select public.test_overloading('Test');`
	应显示以下输出。
	`Success`
		数据工程师，Aurora PostgreSQL 兼容

重载函数

Task	描述	所需技能
使用相同的函数名在 PostgreSQL 中创建重载函数。	在 Aurora PostgreSQL 兼容版中创建一个重载函数，该函数使用的函数名称与之前的函数名称相同。以下示例也被命名 `test_overloading` ，但它具有三个参数：一个输入文本参数、一个输出文本参数和一个输出整数参数。 `CREATE OR REPLACE FUNCTION public.test_overloading( str1 text, OUT str2 text, OUT num1 integer) LANGUAGE 'plpgsql' COST 100 VOLATILE AS $BODY$`	数据工程师，Aurora PostgreSQL 兼容

Task	描述	所需技能

	DECLARE
	str3 text;
	
	BEGIN
	
	str2 :=
	'Success';
	num1 := 100;
	
	RETURN ;
	EXCEPTION
	WHEN others THEN
	RETURN ;
	END;
	$BODY$;	

在 PostgreSQL 中运行该函数。	当您运行此函数时，它会失败并显示以下错误消息。 ERROR: cannot change return type of existing function HINT: Use DROP FUNCTION test_over loading(text) first. 之所以发生这种情况，是因为 Aurora PostgreSQL 兼容版不直接支持函数重载。它无法确定要运行哪个函数，因为尽管输入参数相同，但函数的第二个版本中的输出参数数量不同。	数据工程师，Aurora PostgreSQL 兼容

应用解决方法

Task	描述	所需技能
将 INOUT 添加到第一个输出参数中。	解决方法是，通过将第一个输出参数表示为 INOUT 来修改函数代码。 `CREATE OR REPLACE FUNCTION public.test_overloading( str1 text, INOUT str2 text, OUT num1 integer) LANGUAGE 'plpgsql' COST 100 VOLATILE AS $BODY$ DECLARE str3 text; BEGIN str2 := 'Success'; num1 := 100; RETURN ; EXCEPTION WHEN others THEN RETURN ; END; $BODY$;`	数据工程师，Aurora PostgreSQL 兼容
运行修改后的函数。	使用以下查询运行已更新的函数。您传递一个空值作为此函数的第二个参数，因为您声明此参数为 INOUT 以避免错误。	数据工程师，Aurora PostgreSQL 兼容

Task	描述	所需技能

	`select public.test_overloading('Test', null);`
	
	该函数现已成功创建。
	
	`Success, 100`	

验证结果。	验证带有重载函数的代码是否已成功转换。	数据工程师，Aurora PostgreSQL 兼容

相关资源

- 使用 Amazon Aurora PostgreSQL（Aurora 文档）
- Oracle 中的函数重载（Oracle 文档）
- PostgreSQL 中的函数重载（PostgreSQL 文档）

帮助强制执行 DynamoDB 标签

由 Mansi Suratwala (AWS) 编写

摘要

此模式在预定义的 Amazon DynamoDB 标签丢失或从 Amazon Web Services（AWS）云上的 DynamoDB 资源中删除时设置自动通知。

DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测的性能和可扩展性。借助 DynamoDB，您可以减轻操作和扩展分布式数据库的管理负担。使用 DynamoDB 时，您不必担心硬件预置、设置和配置、复制、软件修补或集群扩展。

该模式使用 AWS CloudFormation 模板，该模板用于创建亚马逊活动 CloudWatch 事件和一个 AWS Lambda 函数。该事件使用 AWS 监视任何新的或现有的 DynamoDB 标签信息。 CloudTrail如果缺少或删除了预定义的标签，则会 CloudWatch 触发 Lambda 函数，该函数会向您发送亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 通知，告知您违规行为。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- Lambda .zip 文件的 Amazon Simple Storage Service（Amazon S3）存储桶，其中包含用于运行 Lambda 函数的 Python 脚本

限制

- 该解决方案仅在`TagResource`或`UntagResource` CloudTrail 事件发生时才起作用。它不会为任何其他事件创建通知。

架构

目标技术堆栈

- Amazon DynamoDB
- AWS CloudTrail
- Amazon CloudWatch

- AWS Lambda
- Amazon S3
- Amazon SNS

目标架构

自动化和扩缩

您可以针对不同的 AWS 区域和账户多次使用 AWS CloudFormation 模板。您只需在每个区域或账户中运行该模板一次。

工具

工具

- Amazon DynamoDB – DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测的性能和可扩展性。
- AWS CloudTrail — CloudTrail 是一项 AWS 服务，可帮助您对 AWS 账户进行治理、合规以及运营和风险审计。用户、角色或 AWS 服务采取的操作在中记录为事件 CloudTrail。
- Amazon CloudWatch Events — Amazon CloudWatch Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。
- AWS Lambda — AWS Lambda 是一项计算服务，使您无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项高度可扩展的对象存储服务，可用于各种存储解决方案，包括网站、移动应用程序、备份和数据湖。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 是一项 Web 服务，可让应用程序、终端用户和设备即时发送和接收来自云端的通知。

代码

- 该项目的 .zip 文件作为附件提供。

操作说明

删除 S3 存储桶

Task	描述	所需技能
定义 S3 存储桶。	在 Amazon S3 控制台上，选择或创建一个具有不包含前导斜杠的唯一名称的 S3 存储桶。此 S3 存储桶将托管 Lambda 代码 .zip 文件。您的 S3 存储桶必须与正在监控的 DynamoDB 资源位于同一 AWS 区域。	云架构师

将 Lambda 代码上传至 S3 存储桶

Task	描述	所需技能
将 Lambda 代码上传至 S3 存储桶。	将附件部分中提供的 Lambda 代码 .zip文件上传至 S3 存储桶。S3 存储桶必须与正在监控的 DynamoDB 资源位于同一区域。	云架构师

部署 AWS CloudFormation 模板

Task	描述	所需技能
部署 AWS CloudFormation 模板。	在 AWS CloudFormation 控制台上，部署 “附件” 部分中提供的 AWS CloudFormation 模板。在下一个操作说明中，为参数提供值。	云架构师

填写 AWS CloudFormation 模板中的参数

Task	描述	所需技能
命名 S3 存储桶。	输入您在第一个操作说明中创建或选择的 S3 存储桶的名称。	云架构师
提供 Amazon S3 密钥。	提供 Lambda 代码 .zip 文件在 S3 存储桶中的位置，不带前导斜杠（例如，`<folder>/<file-name>.zip`）。	云架构师
提供电子邮件地址	提供有效的电子邮件地址以接收 Amazon SNS 通知。	云架构师
定义日志记录级别。	定义 Lambda 函数的日志记录级别和频率。`Info` 指明有关应用程序进度的详细信息消息。`Error` 指明仍允许应用程序继续运行的错误事件。`Warning` 指明潜在的有害情况。	云架构师
输入所需 DynamoDB 标签键。	确保标记之间用逗号分隔，它们之间没有空格（例如，`ApplicationId,CreatedBy,Environment,Organization`）。Events CloudWatch 事件会搜索这些标签，如果找不到这些标签，则会发送通知。	云架构师

确认订阅。

Task	描述	所需技能
确认订阅。	成功部署模板后，它会向你提供的电子邮件地址发送订阅电子邮件。要接收违规通知，您必须确认此电子邮件订阅。	云架构师

相关资源

- 创建 S3 存储桶
- 将文件上传到 S3 存储桶
- 在 DynamoDB 中为资源添加标签
- 使用 AWS 创建在 AWS API 调用时触发 CloudWatch 的事件规则 CloudTrail

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

通过 AWS DMS 和 Amazon Aurora 实施跨区域灾难恢复

由 Mark Hudson (AWS) 编写

摘要

自然或人为灾害可能随时发生，并可能影响在给定 AWS 区域中运行的服务和工作负载的可用性。为了降低风险，您必须制定一项集成 Amazon Web Services 内置跨区域功能的灾难恢复 (DR) 计划。对于本质上不能提供跨区域功能的 Amazon Web Services，灾难恢复计划还必须提供解决方案来处理其在 AWS 区域间的故障转移。

此模式引导您完成灾难恢复设置，该设置涉及单个区域中的两个 Amazon Aurora MySQL 兼容版数据库集群。为满足灾难恢复要求，数据库集群配置为使用 Amazon Aurora Global Database 功能，单个数据库跨越多个 AWS 区域。AWS Database Migration Service (AWS DMS) 任务在本地区域的集群间复制数据。但是，AWS DMS 目前不支持区域间的任务故障转移。此模式包含解决该限制，并在两个区域独立配置 AWS DMS 所需步骤。

先决条件和限制

先决条件

- 支持 Amazon Aurora Global Database 的精选主和辅助 AWS 区域。
- 两个独立的 Amazon Aurora MySQL 兼容版数据库集群位于主区域单个账户中。
- 数据库实例类 db.r5 或以上（推荐）。
- 主区域中的 AWS DMS 任务，在现有数据库集群间执行持续复制。
- 灾难恢复区域资源已就绪，可以满足创建数据库实例的要求。有关更多信息，请参见在 VPC 中使用数据库实例。

限制

- 有关 Amazon Aurora Global Database 限制的完整列表，请参阅 Amazon Aurora Global Database 限制。

产品版本

- Amazon Aurora MySQL-Compatible Edition 5.7 或 8.0。有关更多信息，请参阅 Amazon Aurora 版本。

架构

目标技术堆栈

- Amazon Aurora MySQL-Compatible Edition 全局数据库集群
- AWS DMS

目标架构

下图显示了两个 AWS 区域的全球数据库，一个包含主数据库和报告数据库以及 AWS DMS 复制，另一个包含辅助主数据库和报告数据库。

自动化和扩缩

您可以使用 AWS CloudFormation 在辅助区域创建必备基础设施，例如虚拟私有云 (VPC)、子网和参数组。您还可以使用 AWS CloudFormation 在灾难恢复区域创建辅助集群并将其添加到全局数据库中。如果您使用 CloudFormation 模板在主区域创建数据库集群，则可以用其他模板对其进行更新或扩充，以创建全局数据库资源。有关更多信息，请参阅创建包含两个数据库实例的 Amazon Aurora 数据库集群和为 Aurora MySQL 创建全局数据库集群。

最后，您可以在发生故障转移和故障恢复事件 CloudFormation 后使用在主区域和次要区域创建 AWS DMS 任务。有关更多信息，请参阅 AWS::DMS::ReplicationTask。

工具

- Amazon Aurora 是一个完全托管的关系数据库引擎，与 MySQL 和 PostgreSQL 兼容。此模式使用 Amazon Aurora MySQL 兼容版。
- Amazon Aurora 全球数据库专为分布在全球的应用程序而设计。一个 Amazon Aurora Global Database 可以跨越多个 AWS 区域。它可以在不影响数据库性能的情况下复制数据。它还支持在每个区域以低延迟实现快速本地读取，并可以从区域范围的中断中提供灾难恢复。
- AWS DMS 提供一次性迁移或持续复制。正在进行的复制任务可使源数据库与目标数据库保持同步。设置完成后，正在进行的复制任务会以最小延迟持续将源更改应用于目标。所有的 AWS DMS 功能，例如数据验证和转换，可用于任何复制任务。

操作说明

准备主区域的现有数据库集群

Task	描述	所需技能
修改数据库集群参数组。	在现有的数据库集群参数组中，通过将 `binlog_format` 参数设置为行值来激活行级二进制日志记录。 在执行持续的复制或更改数据捕获 (CDC) 时，AWS DMS 要求对兼容 MySQL 的数据库执行行级二进制日志记录。有关更多信息，请参阅 使用与 AWS 托管的 MySQL 兼容数据库作为 AWS DMS 的来源。	AWS 管理员

更新数据库二进制日志保留期。	使用安装在最终用户设备上的 MySQL 客户端或亚马逊弹性计算云 (Amazon EC2) 实例，在主数据库集群的写入器节点上运行亚马逊关系数据库服务 (Amazon RDS) 提供的以下存储过程，其中XX是保留日志的小时数。
	
	`call mysql.rds_set_configuration('binlog retention hours', XX)`
	
	可以通过运行以下命令确认设置。
	
	`call mysql.rds_show_configuration;`
		数据库管理员

Task	描述	所需技能
	由 AWS 托管的 MySQL 兼容数据库会尽快清除二进制日志。因此，保留期必须足够长，以确保在 AWS DMS 任务运行前不会清除日志。24 小时值通常就足够了，但该值应基于在灾难恢复区域设置 AWS DMS 任务所需时间。	

更新主区域中现有 AWS DMS 任务

Task	描述	所需技能

记录 AWS DMS 任务 ARN。	使用 Amazon 资源名称（ARN）获取 AWS DMS 任务名称以便稍后使用。要检索 AWS DMS 任务 ARN，请在控制台查看该任务或运行以下命令。
	`aws dms describe-replication-tasks`
	ARN 如下所示。
	`arn:aws:dms:us-east-1:<accountid>:task:AN6HFFMPM246XOZVEUHCNSOVF7MQCLTOZUIRAMY`
	最后一个冒号之后的字符对应后面步骤中使用的任务名称。
		AWS 管理员

Task	描述	所需技能
修改现有 AWS DMS 任务，以记录检查点。	AWS DMS 创建包含信息的检查点，以便复制引擎知道更改流的恢复点。若要记录检查点信息，请在控制台中执行以下步骤： 1. 停止 AWS DMS 任务。 2. 使用任务中的 JSON 编辑器将 TaskRecoveryTableEnabled 参数设置为 true。 3. 启动 AWS DMS 任务。	AWS 管理员
验证检查点信息。	使用连接到集群写入器端点的 MySQL 客户端，在报告者数据库集群中查询新的元数据表，以验证该表是否存在并包含复制状态信息。运行以下命令。 `select * from awsdms_control.awsdms_txn_state;` ARN 中的任务名称应在 `Task_Name` 此表的列中找到。	数据库管理员

将两个 Amazon Aurora 集群扩展至灾难恢复区域

Task	描述	所需技能
在灾难恢复区域创建基础设施。	创建创建和访问 Amazon Aurora 集群所需基本组件： • 虚拟私有云（VPC） • 子网	AWS 管理员

Task	描述	所需技能
	• 安全组 • 网络访问控制列表 • 子网组 • 数据库参数组 • 数据库集群参数组 确保两个参数组的配置与主区域配置相匹配。	
将灾难恢复区域添加至两个 Amazon Aurora 集群。	向主集群和报告集群 Amazon Aurora 添加辅助区域（灾难恢复区域）。有关更多信息，请参阅将 AWS 区域 添加到 Amazon Aurora Global Database。	AWS 管理员

执行故障转移

Task	描述	所需技能
停止 AWS DMS 任务。	故障转移发生后，主区域中的 AWS DMS 任务无法正常运行，应停止该任务以避免出错。	AWS 管理员
执行托管故障转移。	对主数据库集群执行灾难恢复区域托管故障转移。有关说明，请参阅执行 Amazon Aurora Global Database 的托管计划内故障转移。主数据库集群的故障转移完成后，在报告器数据库集群执行相同的活动。	AWS 管理员，数据库管理员

Task	描述	所需技能
将数据加载至主数据库。	将测试数据插入灾难恢复数据库集群中的主数据库写入器节点。此数据将用于验证复制是否正常运行。	数据库管理员
创建 AWS DMS 复制实例。	要在灾难恢复区域创建 AWS DMS 复制实例，请参阅创建复制实例。	AWS 管理员，数据库管理员
创建 AWS DMS 源和目标端点。	要在灾难恢复区域创建 AWS DMS 源和目标端点，请参阅创建源和目标端点。源应指向主数据库集群写入器实例。目标应指向报告器数据库集群写入器实例。	AWS 管理员，数据库管理员
获取复制检查点。	若要获取复制检查点，请使用 MySQL 客户端通过对灾难恢复区域报告器数据库集群中的写入节点，运行以下命令来查询元数据表。 `select * from awsdms_control.awsdms_txn_state;` 在表中，找到与 AWS DMS 任务的 ARN 相对应的 task_name 值，该值存在于您在第二个操作说明中获得的主要区域中。	数据库管理员

Task	描述	所需技能
创建 AWS DMS 任务。	使用控制台在灾难恢复区创建 AWS DMS 任务。在任务中，指定仅复制数据更改的迁移方法。有关更多信息，请参阅创建任务。 1. 在任务设置中，利用向导指定以下内容： • 源事务的 CDC 启动模式 – 启用自定义 CDC 启动模式 • 源事务的自定义 CDC 起点 - 指定恢复检查点 2. 在恢复检查点框，输入先前通过对 `awsdms_txn_state` 表的数据库查询获得的复制检查点值。 3. 在任务设置部分，选择 JSON 编辑器，并将TaskRecoveryTableEnabled参数设置为 true。 将 AWS DMS 任务启动迁移任务设置为创建时自动。	AWS 管理员，数据库管理员
记录 AWS DMS 任务 ARN。	使用 ARN 获取 AWS DMS 任务名称，以供日后使用。若要检索 AWS DMS 任务 ARN，请运行以下命令。 `aws dms describe-replication-tasks`	AWS 管理员，数据库管理员

Task	描述	所需技能
验证复制数据。	在灾难恢复区域中查询报告器数据库集群，以确认加载到主数据库集群中的测试数据已被复制。	数据库管理员

执行失效自动恢复

Task	描述	所需技能
停止 AWS DMS 任务。	失效自动恢复后，灾难恢复区域中的 AWS DMS 任务无法正常运行，应停止任务以避免出错。	AWS 管理员
执行托管失效自动恢复。	将主数据库集群故障恢复至主区域。有关说明，请参阅执行 Amazon Aurora Global Database 的托管计划内故障转移。主数据库集群的失效自动恢复完成后，在报告器数据库集群执行相同的活动。	AWS 管理员，数据库管理员
获取复制检查点。	若要获取复制检查点，请使用 MySQL 客户端通过对灾难恢复区域报告器数据库集群中的写入节点，运行以下命令来查询元数据表。 `select * from awsdms_control.awsdms_txn_state;` 在表中，找到与您在第四篇操作说明中获得的灾难恢复区域中存在的 AWS DMS 任务的	数据库管理员

Task	描述	所需技能
	ARN 相对应的 `task_name` 值。	
更新 AWS DMS 源和目标端点。	数据库集群故障恢复后，请检查主区域中集群以确定哪些节点是写入器实例。然后验证主区域中的现有 AWS DMS 源和目标端点是否指向写入器实例。如果不是，请使用写入器实例的域名系统（DNS）名称更新端点。	AWS 管理员

Task	描述	所需技能
创建 AWS DMS 任务。	通过控制台在主区域创建 AWS DMS 任务。在任务中，指定仅复制数据更改的迁移方法。有关更多信息，请参阅创建任务。 1. 在任务设置中，使用向导和指定以下内容： • 源事务的 CDC 启动模式 – 启用自定义 CDC 启动模式 • 源事务的自定义 CDC 起点 - 指定恢复检查点 2. 在恢复检查点框，输入先前通过对 `awsdms_txn_state` 表的数据库查询获得的复制检查点值。 3. 同样在任务设置部分中，选择 JSON 编辑器并将TaskRecoveryTableEnabled参数设置为 true。 4. 最后，将 AWS DMS 任务启动迁移任务设置为创建时自动。	AWS 管理员，数据库管理员

Task	描述	所需技能
记录 AWS DMS 任务 Amazon 资源名称（ARN）。	使用 ARN 获取 AWS DMS 任务名称，以供日后使用。若要检索 AWS DMS 任务 ARN，请运行以下命令： `aws dms describe-replication-tasks` 在执行其他托管故障转移或灾难恢复场景事，将需要任务名称。	AWS 管理员，数据库管理员
删除 AWS DMS 任务。	删除主区域中的原始（当前已停止）AWS DMS 任务和辅助区域中的现有 AWS DMS 任务（当前已停止）。	AWS 管理员

相关资源

- 配置 Amazon Aurora 数据库集群
- 使用 Amazon Aurora Global Database
- 使用 Amazon Aurora MySQL
- 使用 AWS DMS 复制实例
- 使用 AWS DMS 端点
- 使用 AWS DMS 任务
- 什么是 AWS CloudFormation？

其他信息

本示例中使用 Amazon Aurora Global Database 进行灾难恢复，因为它们提供了 1 秒的有效恢复时间目标（RTO）和小于 1 分钟的恢复点目标（RPO），两者都低于传统的复制解决方案，非常适合灾难恢复方案。

Amazon Aurora Global Database 还具有许多其他优势，包含：

- 具有本地延迟的全球读取 - 全球消费者可以在本地延迟的情况下访问本地区域的信息。
- 可扩展的辅助 Amazon Aurora 数据库集群 — 辅助集群可独立扩展，最多可添加 16 个只读副本。
- 从主到辅助 Amazon Aurora DB 集群的快速复制 – 复制对主集群的性能影响很小。它发生在存储层，典型的跨区域复制延迟应小于 1 秒。

此模式还使用 AWS DMS 复制。Amazon Aurora 数据库提供了创建只读副本的功能，这可简化复制过程和灾难恢复设置。但是，当需要进行数据转换或目标数据库需要源数据库没有的其他索引时，通常会使用 AWS DMS 进行复制。

将含有 100 多个参数的 Oracle 函数和过程迁移到 PostgreSQL

创建者：Srinivas Potlachervoo (AWS)

摘要

此模式显示如何将含有 100 多个参数的 Oracle 数据库函数和过程迁移到 PostgreSQL。例如，您可以使用此模式将 Oracle 函数和过程迁移到以下与 PostgreSQL 兼容的 AWS 数据库服务之一：

- Amazon Relational Database Service (Amazon RDS) for PostgreSQL
- Amazon Aurora PostgreSQL 兼容版

PostgreSQL 不支持含有 100 多个参数的函数和过程。解决方法是，您可以定义一种新的数据类型，其类型字段与源函数的参数相匹配。然后，您可以创建并运行使用自定义数据类型作为参数的 PL/pgSQL 函数。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- Amazon RDS Oracle 数据库（DB）实例
- 一个 Amazon RDS for PostgreSQL 数据库实例或 Aurora PostgreSQL-Compatible 数据库实例

产品版本

- Amazon RDS Oracle 数据库实例版本 10.2 及更高版本
- Amazon RDS PostgreSQL 数据库实例版本 9.4 及更高版本，或者 Aurora PostgreSQL-Compatible 数据库实例版本 9.4 及更高版本
- Oracle SQL 开发人员版本 18 及更高版本
- pgAdmin 版本 4 及更高版本

架构

源技术堆栈

- Amazon RDS Oracle 数据库实例版本 10.2 及更高版本

目标技术堆栈

- Amazon RDS PostgreSQL 数据库实例版本 9.4 及更高版本，或者 Aurora PostgreSQL-Compatible 数据库实例版本 9.4 及更高版本

工具

Amazon Web Services

- Amazon Relational Database Service（Amazon RDS）for PostgreSQL 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展 PostgreSQL 关系数据库。
- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩展 PostgreSQL 部署。

其他服务

- Oracle SQL Developer 是一个集成的开发环境，可简化传统部署和基于云的部署中 Oracle 数据库的开发和管理。
- pgAdmin 是一种适用于 PostgreSQL 的开源管理工具。它提供了一个图形界面，可帮助您创建、维护和使用数据库对象。

最佳实践

确保您创建的数据类型与源 Oracle 函数或过程中包含的类型字段相匹配。

操作说明

运行含有 100 多个参数的 Oracle 函数或过程

Task	描述	所需技能
创建或识别一个含有 100 多个参数的现有 Oracle/PLSQL 函数或过程。	创建一个含有 100 多个参数的 Oracle/PLSQL 函数或过程。 –或者–	Oracle/PLSQL 知识

Task	描述	所需技能
	识别一个含有 100 多个参数的现有 Oracle/PLSQL 函数或过程。 有关更多信息，请参阅 Oracle 数据库文档中的第 14.7 节 创建函数语句和第 14.11 节创建过程语句。	
编译 Oracle/PLSQL 函数或过程。	编译 Oracle/PLSQL 函数或过程。 有关更多信息，请参阅 Oracle 数据库文档中的编译函数。	Oracle/PLSQL 知识
运行 Oracle/PLSQL 函数。	运行 Oracle/PLSQL 函数或过程。然后，保存输出。	Oracle/PLSQL 知识

定义与源函数或过程的参数相匹配的新数据类型

Task	描述	所需技能
在 PostgreSQL 中定义一种新的数据类型。	在 PostgreSQL 中定义一种新的数据类型，该数据类型包括源 Oracle 函数或过程的参数中出现的所有相同字段。 有关更多信息，请参阅 PostgreSQL 文档中的创建类型。	PostgreSQL PL/pgSQL 知识

创建一个包含新的 TYPE 参数的 PostgreSQL 函数

Task	描述	所需技能
创建一个包含新的数据类型的 PostgreSQL 函数。	创建一个包含新的 TYPE 参数的 PostgreSQL 函数。 要查看示例函数，请参阅此模式的其他信息部分。	PostgreSQL PL/pgSQL 知识
编译 PostgreSQL 函数。	编译 PostgreSQL 中的 函数。如果新的数据类型字段与源函数或过程的参数相匹配，则该函数成功编译。	PostgreSQL PL/pgSQL 知识
运行 PostgreSQL 函数。	运行 PostgreSQL 函数。	PostgreSQL PL/pgSQL 知识

故障排除

事务	解决方案
函数返回以下错误： 错误：“<statement>”附近有语法错误	确保函数的所有语句都以分号（；）结尾。
函数返回以下错误： 错误：“<variable>”不是已知变量	确保函数正文中使用的变量列在函数的 DECLARE部分中。

相关资源

- 使用Amazon Aurora PostgreSQL（适用于 Aurora 的 Amazon Aurora 用户指南）
- 创建类型（PostgreSQL 文档）

其他信息

包含 TYPE 参数的 PostgreSQL 函数示例

```
CREATE OR REPLACE FUNCTION test_proc_new
(
    IN p_rec type_test_proc_args
)
RETURNS void
AS
$BODY$
BEGIN

    /*
    **************
    The body would contain code to process the input values.
    For our testing, we will display couple of values.
    ***************
    */
    RAISE NOTICE USING MESSAGE = CONCAT_WS('', 'p_acct_id: ', p_rec.p_acct_id);
    RAISE NOTICE USING MESSAGE = CONCAT_WS('', 'p_ord_id: ', p_rec.p_ord_id);
    RAISE NOTICE USING MESSAGE = CONCAT_WS('', 'p_ord_date: ', p_rec.p_ord_date);

END;
$BODY$
LANGUAGE plpgsql
COST 100;
```

将 Amazon RDS for Oracle 数据库实例迁移到使用 AMS 的其他账户

由 Pinesh Singal (AWS) 编写

摘要

此模式说明如何将 Amazon Relational Database Service （ Amazon RDS ） for Oracle 数据库实例从一个 Amazon Web Services account 迁移到另一个 Amazon Web Services account。该模式适用于源 Amazon Web Services account 不使用 AWS Managed Services（ AMS ）但目标账户使用 AMS 的情况。您可以通过在 AMS 中使用更改请求（ RFC ）来完成迁移，而不是使用 AWS 管理控制台执行数据库操作。此方法为具有大量事务的多 TB Oracle 源数据库提供了最短的停机时间。例如，400–900 GB 数据库的停机时间可能持续大约两到三个小时。数据库迁移时间与 Amazon RDS for Oracle 数据库实例的大小成正比。

> **Important**
>
> 此模式要求您在源账户中拍摄 Amazon RDS for Oracle 数据库实例的数据库快照，将快照复制到使用 AMS 的目标账户，然后通过引发从该快照创建新的数据库实例 RFCs。

先决条件和限制

先决条件

- 源账户的有效 Amazon Web Services account
- 将 AMS 用于目标账户的有效 Amazon Web Services account
- Amazon RDS for Oracle 数据库实例，已启动并正在运行

限制

- 源账户中数据库实例的相同属性或配置将复制到 AMS 上的新目标数据库实例。
- 此迁移方法中使用的 RFC 方法具有支持 Amazon RDS for Oracle 的功能有限。通过使用 AWS CloudFormation 模板执行数据库迁移，您可以访问 Amazon RDS for Oracle 的全部功能。
- 您可能会遇到应用程序中断几个小时的情况，因为迁移必须在计划的停机时间内完成。在停机期间，您可以停止源账户中的数据库实例，然后上线到目标账户中的新数据库实例。
- 此迁移方法不适用于将数据库实例从一个 AWS 区域 迁移到同一 Amazon Web Services account 中的另一个区域。

产品版本

- Oracle 数据库标准版 2 (SE2) 12.1.0.2.v2 实例及更高版本在 Amazon RDS for Oracle 上市
- 不再支持 Amazon RDS for Oracle 11g（有关更多信息，请参阅 Amazon RDS 文档中的 Amazon RDS for Oracle。

架构

源技术堆栈

- 适用于 Oracle 的 Amazon RDS 上的 Oracle 数据库 SE2 12.1.0.2.v2 实例
- Amazon RDS 子网组
- Amazon RDS 选项组（如果需要）
- Amazon RDS 参数组（如果需要）
- Amazon Virtual Private Cloud (Amazon VPC) 安全组
- 具有 AWS 托管式密钥或客户托管密钥的 AWS Key Management Service（AWS KMS）
- AWS Identity and Access Management (IAM) 角色（如果需要）

目标技术堆栈

- 适用于 Oracle 的 Amazon RDS 上的 Oracle 数据库 SE2 12.1.0.2.v2 实例
- Amazon RDS 子网组
- Amazon RDS 选项组（如果需要）
- Amazon RDS 参数组（如果需要）
- Amazon VPC 安全组
- AWS Managed Services (AMS)
- 具有 AWS 托管式密钥和客户托管密钥的 AWS KMS
- IAM 角色（如果需要）

源迁移和目标迁移架构

下图显示了一个 Amazon Web Services account 中的 Amazon RDS for Oracle 数据库实例迁移到另一个使用 AMS 的 Amazon Web Services account 中的 Amazon RDS for Oracle 数据库实例。

图表显示了以下工作流：

1. 在源账户中拍摄 Amazon RDS for Oracle 数据库实例的数据库快照。
2. 将快照复制到目标账户中的 AMS。
3. 从目标账户中的快照创建新的 Amazon RDS for Oracle 数据库实例。

自动化和扩缩

您可以通过使用 CloudFormation 模板和在 AMS RFCs 中创建来自动执行和扩展迁移。CloudFormation 使您能够使用 Amazon RDS for Oracle 的所有功能，包括在使用快照创建 Amazon RDS for Oracle 数据库实例时配置和还原数据库实例。

工具

- Amazon Relational Database Service (Amazon RDS) for Oracle 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展 Oracle 关系数据库。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥，以帮助保护您的数据。
- AWS Managed Services (AMS) 可帮助您更高效、更安全地运营 AWS 基础设施。

操作说明

准备在目标账户上进行割接

Task	描述	所需技能
创建自定义 AWS KMS 密钥。	1. 引发名为创建 KMS 密钥的自动 RFC，以从您的目标账户创建自定义 KMS 密钥。 2. **Note** 与源账户共享您的自定义 KMS 密钥。：您无法共享使用亚马逊 RDS 默认 AWS 托管密钥的 Amazon RDS for Oracle 数	AWS、SAM

Task	描述	所需技能
	据库实例 (aws/rds)。相反，您可以通过从 KMS 密钥重新加密数据库实例来共享数据库实例。	
创建安全组。	引发名为 Create security group 的自动 RFC，以从您的目标账户为您的 VPC 创建安全组。 请务必指定以下内容： • 新安全组名称 • TCP 和 UDP 入口和出口规则 • 标准标签	AWS、SAM

Task	描述	所需技能
（可选）查看您的 Amazon RDS 资源。	创建 Amazon RDS for Oracle 数据库实例时，将创建以下资源： • Amazon RDS 子网组（基于子网 ID） • Amazon RDS 选项组（基于源数据库实例的快照） • Amazon RDS 参数组（基于数据库实例的快照） 如果要查看在创建数据库实例时创建的 Amazon RDS 资源，则可以连接到 Oracle 数据库实例并在 Amazon RDS 控制台中查找子网组、选项组和参数组。	AWS

割接源账户

Task	描述	所需技能
停止应用程序。	停止应用程序及其相关服务。您必须停止到源账户中数据库的所有流量。	应用程序所有者
手动创建快照。	在源账户中手动创建 Amazon RDS for Oracle 数据库实例的数据库快照。	AWS
停止数据库实例。	停止 Amazon RDS for Oracle 数据库实例。	AWS

Task	描述	所需技能
复制快照。	将数据库快照复制到同一源账户，然后使用从目标账户共享的自定义 KMS 密钥重新加密复制的数据库快照文件。	AWS
共享快照。	与目标账户共享新快照（使用自定义 KMS 密钥复制）。	AWS

在目标账户上割接

Task	描述	所需技能
复制快照。	引发名为复制 RDS 快照的自动 RFC，将数据库快照复制到同一目标账户，并使用为重新加密创建的默认 AWS 托管 KMS 密钥。 这是使目标账户成为新快照的所有者，并使从快照创建的 Amazon RDS for Oracle 数据库实例与选项组关联（如果需要）所必需的。	AWS、SAM
从快照创建数据库实例。	引发名为从快照创建数据库的自动 RFC，以从快照创建 Amazon RDS for Oracle 数据库实例。 请务必指定以下内容： • 在上一步中创建的新快照 ID • VPC ID • 子网 ID • RDS 实例 ID	AWS、SAM

Task	描述	所需技能
	• 标准标签	
将实例附加到安全组并进行配置更新。	1. 引发名为更新其他的手动 RFC，以将您之前创建的 Amazon RDS for Oracle 数据库实例与您之前创建的 VPC 安全组连接起来。 2. 对 Amazon RDS for Oracle 数据库实例配置进行任何其他更改。	AWS、SAM
测试数据库实例。	通过登录到同一安全组上托管的任何实例或应用程序服务器并使用 telnet 连接到 1521 端口，测试新的 Amazon RDS for Oracle 数据库实例端点连接。有关更多信息，请参阅 Amazon RDS 文档中的连接到 Amazon RDS 数据库实例。 **Note** 如果主用户登录凭证可用，则可以通过从任何 SQL 客户端（例如 Oracle SQL 开发人员）登录来测试 Amazon RDS for Oracle 数据库实例。	AWS、数据库管理员

相关资源

- AWS Managed Services（AWS 文档）
- RFCs 工作原理（AWS Managed Services 文档）

- 共享加密快照（Amazon RDS 用户指南）
- 如何与其他账户共享加密的 Amazon RDS 数据库快照？（AWS Knowledge Center）
- 什么是 Amazon Relational Database Service (Amazon RDS)？（Amazon RDS 用户指南）
- Amazon RDS for Oracle（Amazon RDS 用户指南）
- 使用 AMS 控制台（AWS Managed Services 文档）

其他信息

回滚迁移

如果要回滚迁移，请完成以下步骤：

1. 从目标账户提起手动 RFC（更新其他），以删除在目标账户中创建的数据库堆栈。
2. 更新应用程序配置以指向源账户中的 Amazon RDS for Oracle 数据库实例。
3. 在源账户中启动 Amazon RDS for Oracle 数据库实例。

将 Oracle OUT 绑定变量迁移到 PostgreSQL 数据库

由 Bikash Chandra Rout (AWS) 和 Vinay Paladi (AWS) 编写

摘要

此模式显示如何将 Oracle 数据库 OUT 绑定变量迁移到以下任一与 PostgreSQL 兼容的 AWS 数据库服务：

- Amazon Relational Database Service (Amazon RDS) for PostgreSQL
- Amazon Aurora PostgreSQL 兼容版

PostgreSQL 不支持 OUT 绑定变量。要在 Python 语句中获得相同的功能，您可以创建一个使用 GET 和 SET 包变量的自定义 PL/pgSQL 函数。为了应用这些变量，此模式中提供的示例包装函数脚本使用了 AWS Schema Conversion Tool (AWS SCT) 扩展包。

> **Note**
>
> 如果 Oracle SELECT 语EXECUTE IMMEDIATE句是最多可以返回一行的语句，则最佳做法是执行以下操作：

- 将 OUT 绑定变量（定义）放在 INTO 子句中
- 将 IN 绑定变量放在 USING 子句中

有关更多信息，请参阅 Oracle 文档中的 EXECUTE IMMEDIATE 语句。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 一个本地数据中心中的 Oracle 数据库 10g（或更高版本）源数据库
- 一个 Amazon RDS for PostgreSQL 数据库实例或 Aurora PostgreSQL-Compatible 数据库实例

架构

源技术堆栈

- 本地 Oracle 数据库 10g（或更高版本）数据库

目标技术堆栈

- 一个 Amazon RDS for PostgreSQL 数据库实例或 Aurora PostgreSQL-Compatible 数据库实例

目标架构

下图显示了将 Oracle 数据库0UT绑定变量迁移到兼容 PostgreSQL 的 AWS 数据库的示例工作流程。

图表显示了以下工作流：

1. AWS SCT 将源数据库架构和大部分自定义代码转换为与目标 PostgreSQL 兼容的 AWS 数据库兼容的格式。
2. 任何无法自动转换的数据库对象都会被 PL/pgSQL 函数标记。然后手动转换已标记的对象以完成迁移。

工具

- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩展 PostgreSQL 部署。
- Amazon Relational Database Service（Amazon RDS）for PostgreSQL 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展PostgreSQL 关系数据库。
- AWS Schema Conversion Tool（AWS SCT）通过自动将源数据库架构和大部分自定义代码转换为与目标数据库兼容的格式来支持异构数据库迁移。
- pgAdmin 是一种适用于 PostgreSQL 的开源管理工具。它提供了一个图形界面，可帮助您创建、维护和使用数据库对象。

操作说明

使用自定义 PL/pgSQL 函数和 AWS SCT 迁移 Oracle OUT 绑定变量

Task	描述	所需技能
连接到与 PostgreSQL 兼容的 AWS 数据库。	创建数据库实例后，您可以使用任何标准 SQL 客户端应用程序连接数据库集群中的数据库。例如，您可使用 pgAdmin 连接至您的数据库实例。 有关更多信息，请参阅以下任一项： • Amazon RDS 用户指南中的连接到 Amazon RDS 数据库实例 • 《Amazon Aurora 用户指南》中的连接到 Amazon Aurora 数据库集群	迁移工程师
将此模式中的示例包装函数脚本添加到目标数据库的主架构中。	从此模式的其他信息部分复制示例 PL/pgSQL 包装函数脚本。然后，将该函数添加到目标数据库的主架构中。 有关更多信息，请参阅 PostgreSQL 文档中的 CREATE FUNCTION。	迁移工程师
（可选）更新目标数据库主架构中的搜索路径，使其包含 Test_pg 架构。	为了提高性能，您可以更新 PostgreSQL search_path 变量，使其包含 Test_pg 架构名称。如果在搜索路径中包含架构名称，调用 PL/pgSQL 函数时就无需指定名称。	迁移工程师

Task	描述	所需技能
	有关更多信息，请参阅 PostgreSQL 文档中的第 5.9.3 部分“架构搜索路径”。	

相关资源

- AWS Schema Conversion Tool
- OUT 绑定变量（Oracle 文档）
- 使用绑定变量提高 SQL 查询性能（Oracle 博客）

其他信息

pl/pgSQL 函数示例

```
/* Oracle */

CREATE or replace PROCEDURE test_pg.calc_stats_new1 (
                                                      a NUMBER,
                                                      b NUMBER,
                                                      result out NUMBER
                                                    )

IS
BEGIN
result:=a+b;
END;
/
/* Testing */
set serveroutput on
DECLARE
  a NUMBER := 4;
  b NUMBER := 7;
  plsql_block VARCHAR2(100);
  output number;
BEGIN
  plsql_block := 'BEGIN test_pg.calc_stats_new1(:a, :b,:output); END;';
  EXECUTE IMMEDIATE plsql_block USING a, b,out output;  -- calc_stats(a, a, b, a)
  DBMS_OUTPUT.PUT_LINE('output:'||output);
```

```
END;

output:11


PL/SQL procedure successfully completed.

--Postgres--

/* Example : 1 */
CREATE OR REPLACE FUNCTION test_pg.calc_stats_new1(
                                                    w integer,
                                                    x integer
                                                  )
RETURNS integer
AS
$BODY$
begin
      return w + x ;
end;
$BODY$
LANGUAGE  plpgsql;


CREATE OR REPLACE FUNCTION aws_oracle_ext.set_package_variable(
                                                                 package_name name,
                                                                 variable_name name,
                                                                 variable_value
 anyelement
                                                               )
    RETURNS void
    LANGUAGE 'plpgsql'

    COST 100
    VOLATILE
AS $BODY$
begin
    perform set_config
      ( format( '%s.%s',package_name, variable_name )
      , variable_value::text
      , false );
  end;
$BODY$;
```

```
CREATE OR REPLACE FUNCTION aws_oracle_ext.get_package_variable_record(
                                                                        package_name
 name,
                                                                        record_name name
                                                                      )
RETURNS text
LANGUAGE 'plpgsql'
    COST 100
    VOLATILE
AS $BODY$
begin
    execute 'select ' || package_name || '$Init()';

    return aws_oracle_ext.get_package_variable
        (
         package_name := package_name
        , variable_name := record_name || '$REC' );
end;
$BODY$;

--init()--
CREATE OR REPLACE FUNCTION test_pg.init()
RETURNS void
AS
$BODY$
BEGIN
if aws_oracle_ext.is_package_initialized('test_pg' ) then
      return;
    end if;
    perform aws_oracle_ext.set_package_initialized
      ('test_pg' );
PERFORM aws_oracle_ext.set_package_variable('test_pg', 'v_output', NULL::INTEGER);
PERFORM aws_oracle_ext.set_package_variable('test_pg', 'v_status', NULL::text);
END;
$BODY$
LANGUAGE  plpgsql;

/* callable for 1st Example */

DO $$
declare
v_sql text;
v_output_loc int;
a integer :=1;
```

```
b integer :=2;
BEGIN
perform  test_pg.init();
--raise notice 'v_sql %',v_sql;
execute 'do $a$ declare v_output_l int; begin select * from test_pg.calc_stats_new1('||
a||','||b||') into v_output_l;
PERFORM aws_oracle_ext.set_package_variable(''test_pg'', ''v_output'', v_output_l) ;
 end; $a$'  ;
v_output_loc := aws_oracle_ext.get_package_variable('test_pg', 'v_output');
raise notice 'v_output_loc %',v_output_loc;
END ;
$$

/*In above Postgres example we have set the value of v_output using v_output_l in the
 dynamic anonymous block to mimic the
  behaviour of oracle out-bind variable .*/

--Postgres Example : 2 --
CREATE OR REPLACE FUNCTION test_pg.calc_stats_new2(
 w integer,
 x integer,
inout status text,
 out result integer)
AS
$BODY$
DECLARE
begin
result :=  w + x ;
status := 'ok';
end;
$BODY$
LANGUAGE  plpgsql;

/* callable for 2nd Example */
DO $$
declare
v_sql text;
v_output_loc int;
v_staus text:= 'no';
a integer :=1;
b integer :=2;
BEGIN
perform  test_pg.init();
```

```
execute 'do $a$ declare v_output_l int; v_status_l text; begin select * from
 test_pg.calc_stats_new2(''||a||'',''||b||'','''||v_staus||''') into v_status_l,v_output_l;
PERFORM aws_oracle_ext.set_package_variable(''test_pg'', ''v_output'', v_output_l) ;
PERFORM aws_oracle_ext.set_package_variable(''test_pg'', ''v_status'', v_status_l) ;
end; $a$'  ;
v_output_loc := aws_oracle_ext.get_package_variable('test_pg', 'v_output');
v_staus := aws_oracle_ext.get_package_variable('test_pg', 'v_status');
raise notice 'v_output_loc %',v_output_loc;
raise notice 'v_staus %',v_staus;
END ;
$$
```

使用具有相同主机名的 SAP HSR 将 SAP HANA 迁移至 AWS

由 Pradeep Puliyampatta (AWS) 编写

摘要

可以使用多个选项执行 SAP HANA 向亚马逊网络服务 (AWS) 的迁移，包括备份和还原、导出和导入以及 SAP HANA 系统复制 (HSR)。特定选项的选择取决于源数据库和目标 SAP HANA 数据库间的网络连接、源数据库的大小、停机注意事项和其他因素。

当源系统和目标系统之间存在稳定的网络并且整个数据库（SAP HANA 数据库复制快照）可以在 1 天内完全复制时，用于将 SAP HANA 工作负载迁移至 AWS 的 SAP HSR 选项效果良好，这是 SAP HSR 网络吞吐量要求所规定的。这种方法的停机时间要求仅限于在目标 AWS 环境上执行接管、SAP HANA 数据库备份和迁移后任务。

SAP HSR 支持使用不同的主机名（映射到不同 IP 地址的主机名）来传输主系统（源系统）与辅助系统或目标系统之间的复制流量。为此，您可通过在 `global.ini` 中的 `[system_replication_hostname_resolution]` 部分定义这些特定的主机名集来实现。在本节中，必须在每台主机定义主站点和辅助站点的所有主机。有关详细配置步骤，请参阅 SAP 文档。

此设置的关键要点是，主系统中的主机名必须与辅助系统中的主机名不同。否则，可能会出现如下错误。

- `"each site must have a unique set of logical hostnames"`
- `"remoteHost does not match with any host of the source site. All hosts of source and target site must be able to resolve all hostnames of both sites correctly"`

但是，通过在目标 AWS 环境中使用相同的 SAP HANA 数据库主机名，可以减少迁移后的步骤数量。

此模式为使用 SAP HSR 选项时在源环境和目标环境中使用相同主机名提供了一种解决方法。使用此模式，您可使用 SAP HANA 主机名重命名选项。您可为目标 SAP HANA 数据库分配一个临时主机名，以便 SAP HSR 的主机名具有唯一性。迁移完成目标 SAP HANA 环境的接管里程碑后，您可将目标系统的主机名恢复为源系统的主机名。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- 带有虚拟专用网络（VPN）端点或路由器的虚拟私有云（VPC）。
- AWS Client VPN 或 AWS Direct Connect 配置为将文件从源传输到目标。
- 源环境和目标环境的 SAP HANA 数据库。在相同 SAP HANA 平台版本中，目标 SAP HANA 数据库补丁级别应等于或高于源 SAP HANA 数据库补丁级别。例如：无法在 HANA 1.0 和 HANA 2.0 系统之间设置复制。若要获取更多信息，请参见 SAP Note 第 15 题：1999880 – 常见问题解答：SAP HANA 系统复制。
- 目标环境的 SAP 应用程序服务器。
- 在目标环境红的 Amazon Elastic Block Store（Amazon EBS）卷

限制

以下 SAP 文档列表涵盖了与此变通方法相关的已知问题，包含与 SAP HANA 动态分层和横向扩展迁移相关的限制：

- 2956397 — 重命名 SAP HANA 数据库系统失败
- 2222694 — 尝试重命名 HANA 系统时，会出现以下错误：“源文件不归初始 sidadm 用户所有 (uid = xxxx)”
- 2607227 — hdblcm：register_rename_system：重命名 SAP HANA 实例失败
- 2630562 — HANA 主机名重命名失败且 HANA 无法启动
- 2935639 — sr_register 没有使用 global.ini 部分中 system_replication_hostname_resolution 指定的主机名
- 2710211 — 错误：源系统和目标系统的逻辑主机名重叠
- 2693441 — 由于出现错误，无法重命名 SAP HANA 系统
- 2519672 — HANA 主级和辅助版的系统 PKI SSFS 数据和密钥不同，或无法检查
- 2457129 — 当动态分层是环境一部分时，不允许重命名 SAP HANA 系统主机
- 2473002 — 使用 HANA 系统复制迁移横向扩展系统（SAP 在横向扩展 SAP HANA 系统使用这种主机名重命名方法时，SAP 没有提供任何限制。但是，必须在每台主机上重复该过程。其他横向扩展迁移限制也适用于这种方法。 ）

产品版本

- 此解决方案适用于 SAP HANA 数据库平台 1.0 和 2.0 版。

架构

源设置

源环境安装 SAP HANA 数据库。所有 SAP 应用程序服务器连接和数据库接口都采用相同的主机名进行客户机连接。下图显示了示例源主机名 hdbhost 及其对应的 IP 地址。

目标设置

AWS Cloud 目标环境使用相同的主机名运行 SAP HANA 数据库。AWS 上的目标环境包含以下内容：

- SAP HANA 数据库
- SAP 应用程序服务器
- EBS 卷

中间配置

在下图中，AWS 目标环境上的主机名被临时重命名temp-host为源环境和目标环境的主机名是唯一的。迁移完成目标环境的接管里程碑后，将使用原始名称重命名目标系统的虚拟主机名为 hdbhost。

中间配置包含以下选项之一：

- AWS Client VPN 使用 Client VPN 端点
- AWS Direct Connect 连接到路由器

AWS 目标环境上的 SAP 应用程序服务器可以在复制设置之前或接管之后安装。但是，在设置复制之前安装应用程序服务器可帮助减少安装、配置高可用性和备份过程中的停机时间。

工具

AWS 服务

- AWS Client VPN是一项基于客户端的托管 VPN 服务，可让您安全地访问本地网络中的 AWS 资源和资源。

- AWS Direct Connect通过标准以太网光纤电缆将您的内部网络链接到某个 AWS Direct Connect 位置。通过此连接，您可以直接创建面向公众的虚拟接口 AWS 服务，绕过网络路径中的互联网服务提供商。
- 亚马逊弹性区块存储 (Amazon EBS) Elastic Block Store 提供块级存储卷，用于亚马逊弹性计算云 (Ama EC2 zon) 实例。EBS 卷的行为类似于原始、未格式化的块储存设备。您可以将这些卷作为设备挂载在实例上。

其他工具

- SAP 应用程序服务器 — SAP 应用程序服务器为程序员提供了表达业务逻辑的方法。SAP 应用程序服务器根据业务逻辑执行数据处理。实际数据存储在数据库内，该数据库是一个独立的组件。
- SAP HANA cockpit 和 SAP HANA Studio – Both SAP HANA cockpit 和 SAP HANA Studio 都为 SAP HANA 数据库提供了管理界面。在 SAP HANA Studio 中，SAP HANA 管理控制台为系统视图，它为 SAP HANA 数据库管理提供相关内容。
- SAP HANA System Replication – SAP HANA System Replication (SAP HSR) 是 SAP 提供的用于复制 SAP HANA 数据库的标准程序。SAP HSR 所需可执行文件是 SAP HANA 服务器内核本身的一部分。

操作说明

准备源和目标环境

Task	描述	所需技能
安装与配置 SAP HANA 数据库。	在源环境和目标环境中，确保按照 SAP HANA 的最佳实践安装和配置 SAP HANA 数据库。有关更多信息，请参阅上的 SAP HANA AWS。	SAP Basis 管理
映射 IP 地址。	在目标环境中，确保将临时主机名分配至内部 IP 地址。 1. 导航到 “ EC2 实例”、“操作”、“联网”、“管理 IP IPv4 地址” EC2、“分配新 IP 地	AWS 管理

Task	描述	所需技能
	址”，在 AWS 管理控制台上为实例分配辅助地址。 2. 要从操作系统为 EC2 网络适配器 (NIC) 分配相同的地址，请以 root 用户身份运行命令`ip addr add <IP>/32 dev eth0`，替换`<IP>`为步骤 1 中的 IP 地址。	
解析目标主机名。	在辅助 SAP HANA 数据库上，通过更新 `/etc/hosts` 文件中的相关主机名，确认已为 SAP HANA 复制网络解析两个主机名（`hdbhost` 和 `temp-host`）。	Linux 管理
备份源与目标 SAP HANA 数据库。	使用 SAP HANA Studio 或 SAP HANA cockpit 在 SAP HANA 数据库上执行备份。	SAP Basis 管理
交换系统 PKI 凭证。	(仅适用于 SAP HANA 2.0 及以上版本)在主数据库和辅助数据库之间的文件系统 (SSFS) 存储区中的系统公钥基础设施 (PKI) 安全存储区中交换凭证。有关更多信息，请参阅 SAP Note 2369981 — 使用 SAP HANA 系统复制进行身份验证所需配置步骤。	SAP Basis 管理

重命名目标 SAP HANA 数据库

Task	描述	所需技能
停止目标客户端连接。	在目标环境中，关闭 SAP 应用程序服务器与其他客户端连接。	SAP Basis 管理

将目标 SAP HANA 数据库重命名为临时主机名。	1. 以根用户身份，使用常驻 hdblcm 用户将目标 SAP HANA 数据库主机名重命名为临时主机名。
	`root $> cd /hana/shared/<SID/hdblcm`
	`root $> ./hdblcm`
	2. 选择选项 9 | rename_system | Rename the SAP HANA Database System。
	3. 提供新名称： temp-host 。
	4. 您可根据需要验证其他选项。但是，请确保不要将主机重命名与 SID 更改混为一谈 (SAP Note 2598814 — hdblcm：SID 重命名失败)。
	
	SAP HANA 数据库的停止和启动将由 hdblcm 控制。
		SAP Basis 管理

分配复制网络。	在源系统的 global.ini 文件的 [system_replication_hostname_resolution] 标题下，提供源和目标复制网络的详细信息。然后	SAP Basis 管理

Task	描述	所需技能
	将这些条目复制到目标系统的 `global.ini` 文件中。	
在主服务器上启用复制功能。	若要在源 SAP HANA 数据库上启用复制，请运行以下命令。 `hdbnsutil -sr_enable --name=siteA`	SAP Basis 管理

将目标 SAP HANA 数据库注册至辅助系统。	要将目标 SAP HANA 数据库注册为 SAP HSR 的辅助系统，请选择异步复制。
	`(sid)adm $> HDB stop`
	`(sid)adm $> hdbnsutil -sr_register –name=siteB –remotehost=hdbhost / --remoteInstance=00 –replicationMode=async –operationMode=logreplay`
	`(sid)adm $> HDB start`
	或者，您可选择注册 `–online` 选项。在这种情况下，您无需停止和启动 SAP HANA 数据库。
		SAP Basis 管理

Task	描述	所需技能

验证同步。	在源 SAP HANA 数据库上，确认所有日志均已应用于目标系统（因为它是异步复制）。
	
	若要验证复制，请在源上运行以下命令。
	
	`(sid)adm $> cdpy`
	`(sidadm $> python systemReplicationStatus.py`
		SAP Basis 管理

关闭源 SAP 应用程序与 SAP HANA 数据库。	在迁移割接期间，关闭源系统 (SAP 应用程序和 SAP HANA 数据库)。	SAP Basis 管理
对目标执行接管。	若要在 AWS 上对目标执行接管，请运行命令 `hdbnsutil -sr_takeover` 。	SAP Basis 管理
在目标 SAP HANA 数据库，关闭复制。	若要清除复制元数据，请运行命令停止目标系统上的复制 `hdbnsutil -sr_disable` 。 **Note** 这符合 SAP Note 2693441 — 由于错误而无法重命名 SAP HANA 系统。	SAP Basis 管理
备份目标 SAP HANA 数据库。	接管成功后，我们建议执行完整 SAP HANA 数据库备份。	SAP Basis 管理

恢复至目标系统中的原始主机名

Task	描述	所需技能

将目标 SAP HANA 数据库主机名恢复至原始主机名。	1. 要将目标 SAP HANA 数据库主机名恢复至原始虚拟主机名，请使用 resident hdblcm。
	`root $> cd /hana/shared/<SID>/hdblcm`
	`root $> ./hdblcm`
	2. 选择选项 9 | rename_system | Rename the SAP HANA Database System。
	3. 提供新名称：hdbhost。
	
	您可根据需要验证其他选项。但是，请确保不要将主机重命名与 SID 更改混为一谈 (SAP Note 2598814 — hdblcm：SID 重命名失败)。
		SAP Basis 管理

调节 hdbuserstore。	调整指向源 hdbuserstore 细节的 schema/user 细节。有关详细步骤，请参阅 SAP 文档。 若要验证此步骤，请运行命令 R3trans -d。结果应反映成功连接至 SAP HANA 数据库。	SAP Basis 管理
启动客户端连接。	在目标环境中，启动 SAP 应用程序服务器和其他客户端连接。	SAP Basis 管理

相关资源

SAP 参考

SAP 经常更新 SAP 文档参考资料。若要了解最新信息，请参阅 SAP Note 2407186 — SAP HANA 高可用性操作指南和白皮书。

附加 SAP 说明

- 2550327 — 如何重命名 SAP HANA 系统
- 1999880 — 常见问题解答：SAP HANA 系统复制
- 2078425 — SAP HANA 平台生命周期管理工具 hdblcm 故障排除
- 2592227 — HANA 系统的 FQDN 后缀更改
- 2048681 — 在没有 SSH 或根凭证的情况下，在多主机系统上执行 SAP HANA 平台生命周期管理管理任务

SAP 文档

- 系统复制网络连接
- 系统复制的主机名解析

AWS 参考文献

- 将 SAP HANA 从其他平台迁移到 AWS

其他信息

作为主机名重命名活动的一部分，`hdblcm` 所执行的更改合并至以下详细日志中。

使用分布式可用性组将 SQL Server 迁移至 AWS

由 Praveen Marthala (AWS) 编写

摘要

Microsoft SQL Server Always On 可用性组为 SQL Server 提供了高可用性 (HA) 和灾难恢复 (DR) 解决方案。可用性组由一个接受读/写流量的主副本，和最多八个接受读取流量的辅助副本组成。可用性组是在具有两个或更多节点的 Windows 服务器失效转移群集 (WSFC) 上配置。

Microsoft SQL Server Always On 分布式可用性组提供了一种在两个独立可用性组之间配置两个独立可用性组的解决方案 WFSCs。属于分布式可用性组的可用性组不必位于同一数据中心。一个可用性组可以位于本地，另一个可用性组可以位于另一个域中的亚马逊云服务 (AWS) 云上的亚马逊弹性计算云 (Amazon EC2) 实例上。

此模式概述了使用分布式可用性组将属于现有可用性组的本地 SQL Server 数据库迁移到在 Amazon 上设置可用组的 SQL Server 的步骤 EC2。通过遵循该模式，您可以将数据库迁移至 Amazon Web Services Cloud 中，同时最大限度地减少割接期间的停机时间。割接后，这些数据库立即在 AWS 上具有高可用性。您还可使用此模式将底层操作系统从本地更改为 AWS，同时保持相同版本的 SQL Server。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- AWS Direct Connect 或 AWS Site-to-Site VPN
- 安装在本地和 AWS 上的两个节点上相同版本 SQL Server

产品版本

- SQL Server 版本 2016 和更高版本
- SQL Server 企业版

架构

源技术堆栈

- 本地具有 Always On 可用性组的 Microsoft SQL Server 数据库

目标技术堆栈

- AWS 云端亚马逊 EC2 上带永久开启可用组的微软 SQL Server 数据库

迁移架构

术语

- WSFC 1 — WSFC 本地
- WSFC 2 — Amazon Web Services Cloud 上的 WSFC
- AG 1 — 第一可用性组，位于 WSFC 1 中
- AG 2 — 第二可用性组，位于 WSFC 2 中
- SQL Server 主副本 — AG 1 中被视为写入操作的全局主节点
- SQL Server 转发程序 — AG 2 中从 SQL Server 主副本异步接收数据的节点
- SQL Server 辅助副本 — AG 1 或 AG 2 中同步接收来自主副本或转发器数据的节点

工具

- AWS Direct Connect – AWS Direct Connect 通过标准的以太网光纤电缆将您的内部网络链接到 AWS Direct Connect 位置。通过此连接，您可以创建直接连接到公有 Amazon Web Services 的虚拟接口，从而绕过网络路径中的互联网服务提供商。
- 亚马逊 EC2 — 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要使用 Amazon EC2 启动任意数量或数量的虚拟服务器，也可以进行横向扩展或扩展。
- AWS Site-to-Site VP Site-to-Site N — AWS VPN 支持创建 site-to-site虚拟专用网络 (VPN)。您可以将 VPN 配置为在 AWS 上启动的实例和您自己的远程网络之间传递流量。
- Microsoft SQL Server Management Studio – Microsoft SQL Server Management Studio (SSMS) 是一个用于管理 SQL Server 基础设施的集成环境。它提供了用户界面和一组工具，其中包含与 SQL Server 交互的丰富脚本编辑器。

操作说明

在 AWS 上设置第二可用性组

Task	描述	所需技能
在 AWS 上创建 WSFC。	在带有两个节点的 Amazon EC2 实例上创建 WSFC 2，用于 HA。您将使用此失效转移群集在 AWS 上创建第二可用性组 (AG 2)。	系统管理员、 SysOps 管理员
在 WSFC 2 上创建第二可用性组。	使用 SSMS 在 WSFC 2 中的两个节点创建 AG 2。WSFC 2 中的第一节点将充当转发器。WSFC 2 中的第二节点将作为 AG 2 的辅助副本。 在此阶段，AG 2 中没有可用数据库。其为设置分布式可用性组的起点。	数据库管理员、开发人员
在 AG 2 上创建无恢复选项的数据库。	备份本地可用性组 (AG 1) 数据库。 在没有恢复选项的情况下，将数据库还原至 AG 2 的转发器和辅助副本。还原数据库时，请指定足够磁盘空间，以存放数据库数据文件和日志文件。 在该阶段，数据库处于还原状态。它们不是 AG 2 或分布式可用性组的一部分，也非同步。	数据库管理员、开发人员

配置分布式可用性组

Task	描述	所需技能
在 AG 1 上创建分布式可用性组。	要在 AG 1 上创建分布式可用性组，请使用带 DISTRIBUTED 选项的 CREATE AVAILABILITY GROUP。 1. 使用 AG 1 和 AG 2 的 LISTENER_URL 端点地址。 2. 对于 AVAILABILITY-MODE，用于 ASYNCHRONOUS_COMMIT 避免网络延迟（如有）。这并非影响数据库的性能。 3. 对于 FAILOVER_MODE ，请使用 MANUAL。这是唯一适用于分布式可用性组的可用性模式。 4. 要在 AG 2 上手动还原数据库并对较大的数据库进行更多控制，请使用 SEEDING_MODE 的 MANUAL。	数据库管理员、开发人员
在 AG 2 上创建分布式可用性组。	要在 AG 2 上创建分布式可用性组，请通过 DISTRIBUTED 选项使用 ALTER AVAILABILITY GROUP 。 1. 使用 AG 1 和 AG 2 的 LISTENER_URL 端点地址。	数据库管理员、开发人员

Task	描述	所需技能
	2. 对于 AVAILABILITY-MODE，用于 ASYNCHRONOUS_COMMIT 避免网络延迟（如有）。这并非影响数据库的性能。 3. 对于 FAILOVER_MODE ，请使用 MANUAL。这是唯一适用于分布式可用性组的可用性模式。 4. 要在 AG 2 上手动还原数据库并对较大的数据库进行更多控制，请使用 SEEDING_MODE 的 MANUAL。 分布式可用性组在 AG 1 和 AG 2 间创建。 AG 2 中的数据库尚未配置为：参与从 AG 1 到 AG 2 的数据流。	

Task	描述	所需技能
将数据库添加至 AG 2 上的转发器和辅助副本。	在 AG 2 的转发器和辅助副本中使用 `SET HADR AVAILABILITY GROUP` 选项的 `ALTER DATABASE` 将数据库添加到分布式可用性组。 这将启动 AG 1 和 AG 2 上的数据库间的异步数据流。 全局主服务器接受写入，将数据同步发送至 AG 1 上的辅助副本，并异步向 AG 2 上的转发器发送数据。AG 2 上的转发器将数据同步发送至 AG 2 上的辅助副本。	数据库管理员、开发人员

监控 AG 1 和 AG 2 间的异步数据流

Task	描述	所需技能
使用 DMVs 和 SQL Server 日志。	使用动态管理视图 (DMVs) 和 SQL Server 日志，监控两个可用性组之间数据流的状态。 DMVs 值得监视的内容包括 `sys.dm_hadr_availability_replica_states` 和`sys.dm_hadr_automatic_seeding` 。 要了解转发器同步的状态，在转发器上的 SQL Server 日志中监控同步状态。	数据库管理员、开发人员

为最终迁移执行割接活动

Task	描述	所需技能
停止所有流向主副本流量。	在 AG 1 中阻止主副本传入流量，这样数据库上就不会发生写入活动，数据库就可以迁移了。	应用程序所有者、开发人员
更改 AG 1 上的分布式可用性组的可用性模式。	在主副本上，将分布式可用性组可用性模式设置为同步。 将可用性模式更改为同步模式后，数据将从 AG 1 主副本同步发送至 AG 2 中的转发器。	数据库管理员、开发人员
LSNs 在两个可用性组中选中。	检查 AG 1 和 AG 2 中的最后一个日志序列号 (LSNs)。由于 AG 1 的主副本中没有发生写入操作，因此数据已同步，两个可用性组的最后一次 LSNs 数据应匹配。	数据库管理员、开发人员
将 AG 1 更新至辅助角色。	当您将 AG 1 更新至辅助角色时，AG 1 将失去主副本角色并且不接受写入，两个可用性组之间的数据流将停止。	数据库管理员、开发人员

故障转移至第二个可用性组

Task	描述	所需技能
手动故障转移至 AG 2。	在 AG 2 的转发器，更改分布式可用性组以允许数据丢失。由于您已经检查并确认 AG 1 和 AG 2 LSNs 上的最后一个匹配，因此数据丢失不是问题。	数据库管理员、开发人员

Task	描述	所需技能
	当您允许 AG 2 中转发器的数据丢失时，AG 1 和 AG 2 的角色会发生变化： • AG 2 成为包含主副本和辅助副本的可用性组。 • AG 1 成为带有转发器和辅助副本的可用性组。	
更改 AG 2 上的分布式可用性组的可用性模式。	在 AG 2 主副本上，将可用性模式更改为异步。 这会将数据从 AG 2 移动至 AG 1、从同步移动更改为异步移动。此步骤是为了避免 AG 2 和 AG 1 间的网络延迟（如有），并且不会影响数据库的性能。	数据库管理员、开发人员
开始向新主副本发送流量。	更新连接字符串，以使用 AG 2 上的侦听器 URL 端点向数据库发送流量。 AG 2 现在接受写入并向 AG 1 中转发器发送数据，同时在 AG 2 中将数据发送到自己的辅助副本。数据从 AG 2 异步移动至 AG 1。	应用程序所有者、开发人员

执行割接后的活动

Task	描述	所需技能
在 AG 2 上删除分布式可用性组。	在计划时间内监控迁移情况。在 AG 2 上删除分布式可用性组，以删除 AG 2 和 AG 1 之	数据库管理员、开发人员

Task	描述	所需技能
	间的分布式可用性组设置。这将删除分布式可用性组配置，从 AG 2 至 AG 1 的数据流将停止。 目前，AG 2 在 AWS 上具有高可用性，主副本需写入，辅助副本位于同一个可用性组中。	
停用本地服务器。	停用属于 AG 1 的 WSFC 1 中的本地服务器。	系统管理员、 SysOps 管理员

相关资源

- 分布式可用性组
- SQL 文档： 分布式可用性组
- SQL 文档：Always On 可用性组：高可用性和灾难恢复解决方案

使用和 AWS DMS 从 Oracle 8i 或 9i 迁移到适用于 Oracle 的 Amazon RD SharePlex S

由 Ramu Jagini (AWS) 编写

摘要

此模式描述了如何使用 Oracle 数据泵将 Oracle 数据库从本地数据中心迁移到适用于 Oracle 数据库实例的 Amazon Relational Database Service（Amazon RDS）。您可以使用这种模式通过使用Quest进行同步复制，在减少停机时间 SharePlex 的情况下完成迁移。

您必须使用中间 Oracle 数据库实例迁移，因为 AWS Database Migration Service (AWS DMS) 不支持 Oracle 8i 或 9i 作为源环境。您可以使用 SharePlex 7.6.3 将以前的 Oracle 数据库版本复制到更高版本的 Oracle 数据库版本。中间 Oracle 数据库实例可作为 SharePlex 7.6.3 的目标兼容，并支持作为 AWS DMS 或更高版本的来源。 SharePlex此支持允许将数据向前复制到 Amazon RDS for Oracle 目标环境。

请考虑一下，一些已弃用的数据类型和功能可能会影响从 Oracle 8i 或 9i 迁移至最新版本的 Oracle 数据库。为了减轻这种影响，这种模式使用 Oracle 11.2.0.4 作为中间数据库版本，在迁移至 Amazon RDS for Oracle 目标环境之前帮助优化架构代码。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 本地环境的源 Oracle 8i 或 9i 数据库
- 用于在亚马逊弹性计算云 (亚马逊CR2) 上暂存的 Oracle Database 12c 版本 2 (12) EC2
- 任务 SharePlex 7.6.3（商业级）

限制

- RDS for Oracle 的限制

产品版本

- 作为源数据库的 Oracle 8i 或 9i
- CR2 用于暂存数据库的 Oracle 12（必须与 Amazon RDS for Oracle 版本匹配）

- 目标数据库的 Oracle 12 CR2 或更高版本（适用于 Oracle 的 Amazon RDS）

架构

源技术堆栈

- Oracle 8i 或 9i 数据库
- SharePlex

目标技术堆栈

- Amazon RDS for Oracle

迁移架构

下图显示了如何将 Oracle 8i 或 9i 数据库从本地环境迁移至 Amazon Web Services Cloud 中的 Amazon RDS for Oracle 数据库实例。

图表显示了以下工作流：

1. 为 Oracle 源数据库启用归档日志模式、强制日志记录和补充日志记录。
2. 使用恢复管理器 (RMAN) point-in-time 恢复和 FLASHBACK_SCN 从 Oracle 源数据库恢复 Oracle 暂存数据库。
3. 使用FLASHBACK_SCN（在 RMAN 中使用）配置 SharePlex 为从 Oracle 源数据库读取重做日志。
4. 开始 SharePlex 复制，将数据从 Oracle 源数据库同步到 Oracle 临时数据库。
5. 使用FLASHBACK_SCN EXPDP 和 IMPDP 恢复 Amazon RDS for Oracle 目标数据库。
6. 使用 FLASHBACK_SCN（在 EXPDP 中使用）将 AWS DMS 及其源任务配置为 Oracle 暂存数据库，将 Amazon RDS for Oracle 配置为目标数据库。
7. 启动 AWS DMS 任务以将数据从 Oracle 暂存数据库同步到 Oracle 目标数据库。

工具

- Amazon Relational Database Service (Amazon RDS) 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展关系数据库。

- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到 Amazon Web Services Cloud，或者在云和本地设置的组合之间迁移。
- Quest SharePlex 是一款 Oracle-to-Oracle数据复制工具，用于在最短的停机时间内移动数据，并且不会丢失任何数据。
- 恢复管理器 (RMAN) 是 Oracle 数据库客户端，可对数据库执行备份和恢复任务。它极大地简化了数据库文件的备份、还原和恢复流程。
- Data Pump Export 可帮助您将数据和元数据上传到一组称为转储文件集的操作系统文件中。转储文件集只能由 Data Pump Import 实用程序或 DBMS_DATAPUMP 包导入。

操作说明

在亚马逊上设置 SharePlex 和 Oracle 暂存数据库 EC2

Task	描述	所需技能
创建实 EC2 例。	1. 创建实 EC2 例。 2. 在 EC2 实例CR2 上安装 Oracle 12 以用作 Oracle 暂存数据库。	Oracle 管理
准备暂存数据库。	从 Oracle 8i 或 9i 数据库源环境中获取 RMAN 备份，为 Oracle 暂存数据库做好准备，以便在 Oracle 12 CR2 上作为升级进行恢复。 有关更多信息，请参阅 Oracle 文档中的 Oracle 9i Recovery Manager 用户指南和数据库备份和恢复用户指南。	Oracle 管理
配置 SharePlex。	将 SharePlex 源配置为本地 Oracle 8i 或 9i 数据库，并将目标配置为托管在亚马逊上的 Oracle 12 CR2 暂存数据库。EC2	SharePlex，甲骨文管理

为 Amazon RDS for Oracle 设置您的环境。

Task	描述	所需技能
创建 Oracle 数据库实例	创建 Amazon RDS for Oracle 数据库，然后将 Oracle 12 CR2 连接到该数据库。 有关更多信息，请参阅 Amazon RDS 文档中的创建 PostgreSQL 数据库实例并连接到 PostgreSQL 数据库实例上的数据库。	数据库管理员
从暂存数据库恢复 Amazon RDS for Oracle。	1. 使用 FLASHBACK_SCN 从 Oracle 暂存数据库服务器中获取 EXPDP 备份。 2. 从暂存数据库恢复 Amazon RDS for Oracle。 有关更多信息，请参阅 Oracle Database 文档中的 DBMS_MLE。	数据库管理员

设置 AWS DMS

Task	描述	所需技能
为数据库创建端点。	为 Oracle 暂存数据库创建源端点，为 Amazon RDS for Oracle 数据库创建目标端点。 有关更多信息，请参阅 AWS Knowledge Center 中的如何使用 AWS DMS 创建源或目标端点？。	数据库管理员

Task	描述	所需技能
创建复制实例。	使用 AWS DMS 启动 Oracle 暂存数据库到 Amazon RDS for Oracle 数据库的复制实例。 有关更多信息，请参阅 AWS Knowledge Center 中的如何创建 AWS DMS 复制实例？	数据库管理员
创建并启动复制任务。	使用来自 EXPDP 的 FLASHBACK_SCN 变更数据捕获 (CDC) 创建 AWS DMS 复制任务（因为已通过 EXPDP 完成了满载）。 有关更多信息，请参阅 AWS DMS 文档中的创建任务。	数据库管理员

割接 Amazon RDS for Oracle

Task	描述	所需技能
停止应用程序工作负载。	在计划的割接窗口期间停止应用程序服务器及其应用程序。	应用程序开发人员、数据库管理员
验证本地 Oracle 暂存数据库与实例的 EC2 同步。	EC2 通过在本地源数据库上执行几次日志切换，确认从 SharePlex 复制实例到 Amazon 上的 Oracle 暂存数据库的复制任务的所有消息都已发布。 有关更多信息，请参见 Oracle 文档中的 6.4.2 切换日志文件。	数据库管理员

Task	描述	所需技能
验证 Oracle 暂存数据库与 Amazon RDS for Oracle 数据库的同步。	确认您的所有 AWS DMS 任务是否低延迟且没有错误，然后检查任务的验证状态。	数据库管理员
停止复制 SharePlex 和 Amazon RDS。	如果 SharePlex 和 AWS DMS 复制均未显示任何错误，则停止这两个复制。	数据库管理员
将应用程序重新映射至 Amazon RDS。	与应用程序服务器及其应用程序共享 Amazon RDS for Oracle 端点的详细信息，然后启动应用程序以恢复业务运营。	应用程序开发人员、数据库管理员

测试 AWS 目标环境

Task	描述	所需技能
在 AWS 上测试 Oracle 暂存数据库环境。	1. 测试 SharePlex 复制并确认 Oracle 暂存数据库上没有同步间隙或复制错误。 2. 通过本地环境中定义的基准验证应用程序的行为是否符合预期。	SharePlex，甲骨文管理
测试 Amazon RDS 环境。	1. 验证复制后传播到 Amazon RDS 的所有数据是否没有错误。 2. 将另一应用程序指向 Amazon RDS 数据库实例，然后运行性能测试以验证预期行为。	Oracle 管理

Task	描述	所需技能
	有关更多信息，请参阅 Amazon RDS 文档中的Amazon RDS for Oracle。	

相关资源

- 放心迁移
- Amazon EC2
- Amazon RDS for Oracle
- AWS Database Migration Service
- 调试您的 AWS DMS Migrations：出现问题时该怎么做（第 1 部分）
- 调试您的 AWS DMS Migrations：出现问题时该怎么做 (第 2 部分)
- 调试您的 AWS DMS Migrations：出现问题时该怎么做（第 3 部分）
- SharePlex 用于数据库复制
- SharePlex：适用于任何环境的数据库复制

监控 Amazon Aurora 以查找未加密的实例

由 Mansi Suratwala (AWS) 编写

摘要

此模式提供了一个亚马逊网络服务 (AWS) CloudFormation 模板，您可以部署该模板，以便在未开启加密的情况下创建 Amazon Aurora 实例时设置自动通知。

Aurora 是一个与 MySQL 和 PostgreSQL 兼容的完全托管式的关系数据库引擎。在某些工作负载条件下，Aurora 最多可以将 MySQL 吞吐量增加 5 倍，将 PostgreSQL 的吞吐量增加 3 倍，而无需对大多数现有应用程序进行更改。

该 CloudFormation 模板创建了一个亚马逊 CloudWatch 事件和一个 AWS Lambda 函数。该事件使用 AWS CloudTrail 来监控任何 Aurora 实例的创建或现有实例的时间点恢复。Cloudwatch Events 事件启动 Lambda 函数，该函数检查是否启用了加密。如果未启用加密，Lambda 函数会发送 Amazon Simple Notification Service (Amazon SNS) 通知，通知您存在违规情况。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account

限制

- 此服务控制仅适用于 Amazon Aurora 实例。它不支持其他 Amazon Relational Database Service (Amazon RDS)实例。
- CloudFormation 模板必须仅针对`CreateDBInstance`和 `RestoreDBClusterToPointInTime` 部署。

产品版本

- Amazon Aurora 中支持的 PostgreSQL 版本
- Amazon Aurora 中支持的 MySQL 版本

架构

目标技术堆栈

- Amazon Aurora
- AWS CloudTrail
- Amazon CloudWatch
- AWS Lambda
- Amazon Simple Storage Service (Amazon S3)
- Amazon SNS

目标架构

自动化和扩缩

您可以为不同的地区和账户多次使用该 CloudFormation 模板。您只需在每个区域或账户中运行一次。

工具

工具

- Amazon Aurora - Amazon Aurora 是与 MySQL 和 PostgreSQL 兼容的完全托管式的云端关系数据库引擎。
- AWS CloudTrail — AWS CloudTrail 可帮助您管理 AWS 账户的治理、合规以及运营和风险审计。用户、角色或 AWS 服务采取的操作在中记录为事件 CloudTrail。
- Amazon CloudWatch Events — Amazon CloudWatch Events 提供一系列描述了 AWS 资源变化的系统事件。 near-real-time
- AWS Lambda – AWS Lambda 是一项计算服务，支持无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。
- Amazon S3 - Amazon Simple Storage Service (Amazon S3)是一种高度可扩展的对象存储服务，可用于各种存储解决方案，包括网站、移动应用程序、备份和数据湖。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS)是一项托管服务，可通过 Lambda、HTTP、电子邮件、手机推送通知和手机短信 (SMS) 的形式提供消息。

代码

该项目的 .zip 文件作为附件提供。

操作说明

为 Lambda 脚本创建 S3 存储桶

Task	描述	所需技能
定义 S3 存储桶。	打开 Amazon S3 控制台并选择或创建 S3 存储桶。此 S3 存储桶将托管 Lambda 代码 .zip 文件。您的 S3 存储桶需要与 Aurora 位于同一区域。S3 存储桶名称不能包含前导斜杠。	云架构师

将 Lambda 代码上传至 S3 存储桶

Task	描述	所需技能
上传 Lambda 代码。	将附件部分中提供的 Lambda 代码 .zip 文件上传到您定义的 S3 存储桶。	云架构师

部署 CloudFormation 模板

Task	描述	所需技能
部署 CloudFormation 模板。	在 CloudFormation 控制台上，部署作为该模式附件提供的`RDS_Aurora_Encryption_At_Rest.yml` CloudFormation 模板。在下一个操作说明中，提供模板参数的值。	云架构师

填写 CloudFormation 模板中的参数

Task	描述	所需技能
提供 S3 存储桶名称。	输入您在第一个操作说明中创建或选择的 S3 存储桶的名称。	云架构师
提供 S3 密钥。	提供 Lambda 代码 .zip 文件在 S3 存储桶中的位置，不带前导斜杠（例如，`<directory>/<file-name>.zip`）。	云架构师
提供电子邮箱地址。	提供有效的电子邮件地址以接收 Amazon SNS 通知。	云架构师
定义日志记录级别。	定义 Lambda 函数的日志记录级别和频率。`Info` 指明有关应用程序进度的详细信息消息。`Error` 指明仍允许应用程序继续运行的错误事件。`Warning` 指明潜在的有害情况。	云架构师

确认订阅

Task	描述	所需技能
确认订阅。	成功部署模板后，它将向提供的电子邮件地址发送订阅电子邮件。要接收通知，您必须确认此电子邮件订阅。	云架构师

相关资源

- 创建 S3 存储桶

- 将文件上传到 S3 存储桶
- 创建 Amazon Aurora 数据库集群
- 使用 AWS 创建在 AWS API 调用时触发 CloudWatch 的事件规则 CloudTrail

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用亚马逊监控 Oracle GoldenGate 日志 CloudWatch

由 Chithra Krishnamurthy (AWS) 创建

摘要

Oracle GoldenGate 为甲骨文数据库提供亚马逊关系数据库服务（Amazon RDS）之间或亚马逊弹性计算云（亚马逊 EC2）上托管的 Oracle 数据库之间的实时复制。它支持单向和双向复制。

在使用 GoldenGate 复制时，监控对于验证 GoldenGate 过程是否已启动并运行，以及确保源数据库和目标数据库同步至关重要。

此模式说明了对 GoldenGate 错误日志实施 Amazon CloudWatch 监控的步骤，以及如何设置警报以发送特定事件的通知，例如`STOP`您可以采取适当措施快速恢复复制。`ABEND`

先决条件和限制

先决条件

- GoldenGate 已在 EC2 实例上安装和配置，因此您可以对这些 EC2 实例设置 CloudWatch 监控。如果要 GoldenGate 跨AWS区域监控双向复制，则必须在运行该 GoldenGate 进程的每个 EC2 实例中安装 CloudWatch 代理。

限制

- 此模式说明了如何使用监视 GoldenGate 进程 CloudWatch。 CloudWatch 不监控复制期间的复制延迟或数据同步问题。您必须运行单独的 SQL 查询来监控复制延迟或与数据相关的错误，如GoldenGate 文档中所述。

产品版本

- 本文档基于 Linux x86-64 上适用于 Oracle 的 Oracle GoldenGate 19.1.0.0.4 的实现。但是，此解决方案适用于的所有主要版本 GoldenGate。

架构

目标技术堆栈

- GoldenGate 安装在实例上的 Oracle 二进制文件 EC2

- 亚马逊 CloudWatch
- Amazon Simple Notification Service (Amazon SNS)

目标架构

工具

Amazon Web Services

- Amazon CloudWatch 是一种监控服务，在此模式中用于监控 GoldenGate 错误日志。
- Amazon SNS 是一种消息通知服务，在此模式中用于发送电子邮件通知。

其他工具

- Oracle GoldenGate 是一款数据复制工具，可用于适用于 Amazon RDS for Oracle 数据库或托管在亚马逊上的 Oracle 数据库 EC2。

高级实施步骤

1. 为代理创建 AWS Identity and Access Managem CloudWatch ent (IAM) 角色。
2. 将 IAM 角色附加到生成 GoldenGate 错误日志的 EC2 实例。
3. 在 EC2 实例上安装 CloudWatch 代理。
4. 配置 CloudWatch 代理配置文件：`awscli.conf`和`awslogs.conf`。
5. 启动代 CloudWatch 理。
6. 在日志组中创建指标筛选器。
7. 设置 Amazon SNS。
8. 为指标筛选条件创建警报。当这些筛选条件捕获事件时，Amazon SNS 会发送电子邮件提醒。

有关详细说明，请参阅下一部分。

操作说明

第 1 步：为 CloudWatch 代理创建 IAM 角色

Task	描述	所需技能
创建 IAM 角色。	访问 AWS 资源需要权限，因此您可以创建 IAM 角色以包含每台服务器运行 CloudWatch 代理所需的权限。 要创建 IAM 角色： 1. 登录 AWS 管理控制台并在上打开 IAM 控制台https://console.aws.amazon.com/iam/。 2. 在导航窗格中，选择角色，然后选择创建角色。 3. 在可信实体类型中选择 Amazon Web Services。 4. 对于常见用例，请选择 EC2，然后选择下一步。 5. 在策略列表中，选中旁边的复选框CloudWatchAgentServerPolicy。如有必要，请使用搜索框查找该策略。 6. 选择下一步。 7. 对于角色名称，请输入新角色的名称，如 `goldengate-cw-monitoring-role` 或所需其他名称。 8. （可选）对于角色描述，请输入描述。	AWS 常规

Task	描述	所需技能
	9. 确认它CloudWatc hAgentServerPolicy显示在“策略名称” 下。 10. （可选）添加一个或多个标签（键值）对，以组织、追踪或控制对此角色的访问，然后选择创建角色。	

第 2 步：将 IAM 角色附加到 GoldenGate EC2 实例

Task	描述	所需技能
将 IAM 角色附加到生成 GoldenGate 错误日志的 EC2 实例。	GoldenGate 必须将生成的错误日志填充到 CloudWatch 并进行监控，因此您需要将您在步骤 1 中创建的 IAM 角色附加到 GoldenGate 正在运行的 EC2 实例。 将 IAM 角色附加到实例： 1. 打开亚马逊 EC2 控制台，网址为https://console.aws.amazon.com/ec2/。 2. 在导航窗格中，选择实例，然后找到 GoldenGate 正在运行的实例。 3. 选择实例，然后依次选择操作、安全性、修改 IAM 角色。 4. 选择第一步中创建的 IAM 角色，将其附加到实例，然后选择保存。	AWS 常规

步骤 3-5。在 Gold CloudWatch EC2 engate 实例上安装和配置代理

Task	描述	所需技能
在 GoldenGate EC2 实例上安装 CloudWatch 代理。	要安装代理，请运行命令： `sudo yum install -y awslogs`	AWS 常规

编辑代理配置文件。	1. 运行以下命令。
	`sudo su -`
	2. 编辑此文件以根据需要更新 AWS 区域。
	`cat /etc/awslogs/conf [plugins] cwlogs = cwlogs [default] region = us-east-1`
	3. 编辑`/etc/awslogs/awslogs.conf` 文件以更新文件名、日志组名称以及与中的日期格式相匹配`ggserror.log` ；否则，日志流将不会流入 CloudWatch。date/time format. You must specify the date/time例如：
	`datetime_format = %Y-%m-%dT%H:%M:%S%z file = /u03/oracle/oragg/ggserr.log log_group_name = goldengate_monitor`
		AWS 常规

Task	描述	所需技能
启动代 CloudWatch 理。	要启动代理，请使用以下命令。 `$ sudo service awslogsd start` 启动代理后，可以在 CloudWatch 控制台中查看日志组。日志流将包含文件的内容。	AWS 常规

步骤 6. 创建日志组的指标筛选条件

Task	描述	所需技能
为关键字 ABEND 和 STOPPED 创建指标筛选器。	当您为日志组创建指标筛选条件时，只要在错误日志中识别出筛选条件，它就会启动警报并根据 Amazon SNS 配置发送电子邮件通知。 创建指标筛选条件： 1. 打开 CloudWatch 控制台，网址为https://console.aws.amazon.com/cloudwatch/。 2. 选择日志组的名称。 3. 选择 Actions（操作），然后选择 Create metric filter（创建指标筛选条件）。 4. 对于筛选器模式，指定一个模式，例如 ABEND。	CloudWatch

Task	描述	所需技能
	5. 选择 Next（下一步），然后为指标筛选条件输入名称。 6. 在指标详细信息下，在指标命名空间中，输入要发布指标的 CloudWatch 命名空间的名称。如果该命名空间尚不存在，请确保选中 Create new（新建）。 7. 对于指标值，输入 1，因为您的指标筛选条件正在计算筛选条件中关键字的出现次数。 8. 将单位设置为无。 9. 选择 Create metric filter（创建指标筛选条件）。可以从导航窗格找到您创建的指标筛选条件。 10.为 STOPPED 模式创建另一个指标筛选器。在一个日志组中，您可以创建多个指标筛选器并单独设置告警。	

第 7 步。设置 Amazon SNS

Task	描述	所需技能
创建 SNS 主题。	在此步骤中，您将配置 Amazon SNS 以为指标筛选条件创建警报。 创建 SNS 主题：	Amazon SNS

Task	描述	所需技能
	1. 在家中登录亚马逊 SNS 控制台。https://console.aws.amazon.com/sns/ 2. 在创建主题框中，输入主题的名称，例如 `goldengate-alert` ，然后选择下一步。 3. 对于类型，选择标准。 4. 滚动到表单末尾，选择 Create topic（创建主题）。控制台会打开新主题的 Details（详细信息）页面。	

Task	描述	所需技能
创建订阅。	要创建主题订阅： 1. 在左侧导航窗格中，选择订阅。 2. 在订阅页面上，选择创建订阅。 3. 在创建订阅页面上，选择主题 ARN字段以查看您的 Amazon Web Services account 中的主题列表。 4. 选择在先前步骤中创建的主题。 5. 对于协议，选择电子邮件。 6. 对于 Endpoint（端点），输入可以接收通知的电子邮件地址。 7. 选择创建订阅。控制台会打开新订阅的详细信息页面。 8. 查看收件箱中是否有来自 AWS 通知的电子邮件，然后在电子邮件中选择确认订阅。 Amazon SNS 会打开您的 Web 浏览器，并显示带有您的订阅 ID 的订阅确认信息。	Amazon SNS

步骤 8：创建警报以发送指标筛选器的通知

Task	描述	所需技能
为 SNS 主题创建警报。	要根据日志组指标筛选条件创建警报：	CloudWatch

Task	描述	所需技能
	1. 打开 CloudWatch 控制台，网址为https://console.aws.amazon.com/cloudwatch/。 2. 从左侧导航窗格中，选择 Logs（日志），然后选择 Log groups（日志组）。 3. 选择包含您的指标筛选器的日志组。 4. 选择 Metric filters（指标筛选条件）。 5. 在指标筛选条件选项卡中，选中要用于创建警报的指标筛选条件对应的复选框。 6. 选择创建警报。 7. 对于条件，请在每个部分中指定以下内容： • 对于阈值类型，选择静态。 • 对于每当 metric_name 为. . .，选择大于。 • 对于比. . .，指定 0。 8. 选择下一步。 9. 在通知下： • 对于 Alarm state trigger（告警状态触发器），选择 In alarm（在告警中）。 • 对于发送通知到以下 SNS 主题，选择选择现有的主题。	

Task	描述	所需技能
	• 在电子邮件框中，选择您在上一步中创建的 Amazon SNS 主题。 10选择下一步。 11对于名称和描述，输入告警的名称和描述。 Note 对于描述，您可以指定实例名称，以便通知电子邮件具有描述性。 12对于预览和创建，确保您的配置正确，然后选择创建警报。 完成这些步骤后，每当在您监控的 GoldenGate 错误日志文件 (`ggserr.log`) 中检测到这些模式时，您都会收到一封电子邮件通知。	

故障排除

事务	解决方案
GoldenGate 错误日志中的日志流不会流入 CloudWatch。	检查`/etc/awslogs/awslogs.conf` 文件以验证文件名、日志组名称以及与中的日期格式相匹配`ggserror.log` 。date/time format. You must specify the date/time否则，日志流将不会流入 CloudWatch。

相关资源

- 亚马逊 CloudWatch 文档
- 使用 CloudWatch 代理收集指标和日志
- Amazon SNS 文档

从 Oracle Database Enterprise Edition 更换平台到 Amazon RDS for Oracle 上的 Standard Edition 2。

由 Lanre（Lan-Ray）showunmi（AWS）和 Tarun Chawla（AWS）创作

摘要

Oracle Database Enterprise Edition (EE) 是许多企业运行应用程序的流行选择。然而，在某些情况下，应用程序很少或根本不使用 Oracle Database EE 功能，因此没有理由承担巨额许可成本。迁移到 Amazon RDS 时，您可以将此类数据库降级为 Oracle 数据库标准版 2 (SE2)，从而节省成本。

此模式描述了从本地迁移到 Amazon RDS for Oracle SE2 时，如何从 Oracle Database EE 降级到 Oracle 数据库。如果您的 EE Oracle 数据库已经在 Amazon RDS 或亚马逊弹性计算云 (Amazon EC2) 实例上运行，则此模式中介绍的步骤也适用。

有关更多信息，请参阅 AWS Prescriptive Guidance 指南，了解评估将 Oracle 数据库降级到 Standard Edition 2 on AWS。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- Oracle Database Enterprise Edition
- 客户端工具，例如 Oracle SQL Developer 或 SQL*Plus，用于连接和在 Oracle 数据库上运行 SQL 命令
- 用于执行评测的数据库用户。例如以下用户之一：
  - 拥有足够权限运行 AWS Schema Conversion Tool (AWS SCT) 评测的用户
  - 具有足够权限的用户，可以对 Oracle 数据库字典表运行 SQL 查询
- 用于执行数据库迁移的数据库用户，例如以下用户之一：
  - 拥有足够权限运行 AWS Database Migration Service(AWS DMS) 的用户
  - 具有执行 Oracle Data Pump 导出和导入的足够权限的用户
  - 拥有足够权限运行 Oracle 的用户 GoldenGate

限制

- Amazon RDS for Oracle 具有最大数据库大小。有关更多信息，请参阅 Amazon RDS 数据库实例存储。

产品版本

此文档中描述的一般逻辑适用于 9i 及更高版本的 Oracle。有关自托管数据库和 Amazon RDS for Oracle 数据库的支持版本，请参阅 AWS DMS 文档。

若要在不支持 AWS SCT 的情况下确定功能使用情况，请在源数据库上运行 SQL 查询。要从不支持 AWS DMS 和 Oracle Data Pump 的早期版本 Oracle 迁移，请使用 Oracle 导出和导入实用程序。

有关当前支持的版本的列表，请参阅 AWS 文档中的 Oracle on Amazon RDS。有关定价和支持的实例类的详细信息，请参阅 Amazon RDS for Oracle pricing。

架构

源技术堆栈

- 在本地或亚马逊上运行的 Oracle 数据库企业版 EC2

使用原生 Oracle 工具瞄准技术堆栈

- 运行 Oracle 数据库的 Amazon RDS for Oracle SE2

1. 使用 Oracle Data Pump 导出数据。
2. 通过数据库链接将转储文件复制至 Amazon RDS。
3. 使用 Oracle Data Pump 将转储文件导入 Amazon RDS。

使用 AWS DMS 瞄准技术堆栈

- 运行 Oracle 数据库的 Amazon RDS for Oracle SE2
- AWS DMS

1. 使用 Oracle Data Pump 和 FLASHBACK_SCN 导出数据。

2. 通过数据库链接将转储文件复制至 Amazon RDS。
3. 使用 Oracle Data Pump 将转储文件导入 Amazon RDS。
4. 使用 AWS DMS 更改数据捕获 (CDC)。

工具

Amazon Web Services

- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到 Amazon Web Services Cloud，或者在云和本地设置的组合之间迁移。
- Amazon Relational Database Service (Amazon RDS) 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展关系数据库。此模式使用了 Amazon RDS for Oracle。
- AWS SCT 提供了一个基于项目的用户界面，用于自动评测、转换源 Oracle 数据库的数据库架构，并将其复制为与 Amazon RDS for Oracle 兼容的格式。AWS SCT 让您可以分析将 Oracle 许可证类型从 Enterprise 版更改为 Standard Edition 可以实现的潜在成本节省。AWS SCT 报告的许可评估和云支持部分提供了有关正在使用的 Oracle 功能的详细信息，因此您可在迁移至 Amazon RDS for Oracle 时做出明智的决定。

其他工具

- 原生 Oracle 导入和导出实用工具可以将 Oracle 数据移入和移出 Oracle 数据库。Oracle 提供两种类型的数据库导入和导出实用工具：源 Export and Import（适用于早期版本）和 Oracle Data Pump Export and Import（适用于 Oracle Database 10g 版本 1 及更高版本）。
- Oracle GoldenGate 提供了实时复制功能，因此您可以在初始加载后同步目标数据库。此选项有助于帮助减少上线期间的应用程序停机时间。

操作说明

执行迁移前评测

Task	描述	所需技能
验证应用程序数据库要求。	确保您的应用程序经过认证，可以在 Oracle 数据库上运行 SE2。直接咨询软件供应商、开发人员或应用程序文档。	应用程序开发者、数据库管理员、应用程序所有者

Task	描述	所需技能
直接在数据库中研究 EE 功能的使用。	若要确定 EE 功能的使用，请执行以下操作之一： • 为您的 EE 数据库生成 AWS SCT 评测报告。此报告将告诉您如果要更改许可证类型，应该移除当前 EE 数据库中的哪些功能。 • 如果您有 Oracle Support 账户，请获取并运行支持文档 1317265.1 中的脚本 `options_packs_usage_statistics.sql`，以生成有关您的 Oracle 数据库上正在使用的选项和功能的报告。 • 查询数据库管理员_FEATURE_USAGE_STATISTICS 显示正在使用的所有功能的详细信息。	应用程序所有者，数据库管理员，应用程序开发者

Task	描述	所需技能
确定 EE 功能在运营活动中的使用。	数据库或应用程序管理员有时依赖仅限 EE 的功能进行操作活动。常见的示例包括在线维护活动（索引重建、表移动）和批处理作业对并行性的使用。 只要有可能，就可以通过修改操作来缓解这些依赖项。确定这些功能的用途，并按成本和收益做出决策。 使用 “比较 Oracle 数据库 EE 和 SE2 功能” 表作为指南，确定 Oracle 数据库中可用的功能 SE2。	应用程序开发者、数据库管理员、应用程序所有者
查看 EE Oracle 数据库的工作负载模式。	Oracle 数据库随时 SE2 自动将使用量限制为最多 16 个 CPU 线程。 如果您的 Oracle EE 数据库已获得使用 Oracle 诊断包的许可，请使用自动工作量存储库 (AWR) 工具或 DBA_HIST_* 视图来分析数据库工作负载模式，以确定降级到时，16 个 CPU 线程的最大限制是否会对服务级别产生负面影响。 SE2 确保您的评测涵盖活动高峰期，如日末、月末或年末处理。	应用程序所有者，数据库管理员，应用程序开发者

在 AWS 上准备目标基础设施

Task	描述	所需技能
部署和配置联网基础设施。	创建虚拟私有云（VPC）和子网、安全组和网络访问控制列表。	AWS 管理员、云架构师、网络管理员、 DevOps 工程师
预配置 Amazon RDS for Oracle SE2 数据库。	预置目标 Amazon RDS for Oracle SE2 数据库，以满足应用程序的性能、可用性和安全要求。建议对生产工作负载使用多可用区配置。但是，为了提高迁移性能，您可将启用多可用区推迟到数据迁移之后。	云管理员、云架构师、数据库管理员、 DevOps 工程师、AWS 管理员
自定义 Amazon RDS 环境。	配置自定义参数和选项，并启用其他监控。有关更多信息，请参阅迁移至 Amazon RDS for Oracle 的最佳实践。	AWS 管理员、AWS 系统管理员、云管理员、数据库管理员、云架构师

执行迁移试运行和应用程序测试

Task	描述	所需技能
迁移数据（试运行）。	使用最适合您的特定环境的方法，将数据从源 Oracle EE SE2 数据库迁移到 Amazon RDS for Oracle 数据库实例。按规模、复杂性和可用停机时间等因素选择迁移策略。使用以下方法之一或组合使用二者： • Oracle 原生工具，例如 Oracle Data Pump（推荐）、Oracle 导入导出实用	数据库管理员

Task	描述	所需技能
	程序和 Oracle。 GoldenGate • AWS DMS，使用满负荷通过 CDC 执行持续复制。	
验证目标数据库。	对数据库存储与代码对象执行迁移后验证。查看迁移日志，并修复所有已发现问题。有关更多信息，请参阅指南将 Oracle 数据库迁移至 AWS Cloud。	数据库管理员
测试应用程序。	应用程序和数据库管理员应酌情执行功能、性能和操作测试。有关更多信息，请参阅迁移至 Amazon RDS for Oracle 的最佳实践。 最后获得利益相关者对测试结果的签字。	应用程序开发者，应用程序所有者，数据库管理员，迁移工程师，迁移主管

割接

Task	描述	所需技能
刷新 Oracle 数据库 EE 数据。	根据应用程序可用性要求，选择数据刷新方法。有关更多信息，请参阅将 Oracle 数据库迁移至 AWS 的策略中的迁移方法。 例如，通过使用诸如 Oracle GoldenGate 或 AWS DMS 之类的工具进行持续复制，您可以实现近乎零的停机时间。如	应用程序所有者，割接主管，数据库管理员，迁移工程师，迁移主管

Task	描述	所需技能
	果停机窗口允许，则可以使用离线方法（例如 Oracle Data Pump 或 Original Export-Import 实用程序）执行最终的数据割接。	
将应用程序指向目标数据库实例。	更新应用程序和其他客户端中的连接参数，使其指向 Amazon RDS for Oracle SE2 数据库。	应用程序开发者，应用程序所有者，迁移工程师，迁移主管，割接主管
执行迁移后活动。	执行数据迁移后的任务，例如启用多可用区、数据验证以及其他检查。	数据库管理员，迁移工程师
执行割接后监控。	使用诸如亚马逊 CloudWatch和亚马逊 RDS Performance Insights 之类的工具来监控 Amazon RDS for Oracle SE2 数据库。	应用程序开发者，应用程序所有者，AWS 管理员，数据库管理员，迁移工程师

相关资源

AWS Prescriptive Guidance

- 将 Oracle 数据库迁移至 Amazon Web Services Cloud（指南）
- 评估将 Oracle 数据库降级为 Standard Edition 2 on AWS（指南）
- 将本地 Oracle 数据库迁移至至 Amazon RDS for Oracle（模式）
- 使用 Oracle Data Pump 将本地 Oracle 数据库迁移到 Amazon RDS for Oracle（模式）

博客文章

- 使用 AWS DMS 以近乎零停机时间迁移 Oracle 数据库
- 使用 Amazon RDS for Oracle 分析 Oracle SE 中的性能管理
- 使用 Amazon RDS for Oracle 管理 Oracle SE 中的 SQL 计划

- 在 Oracle Standard Edition 中实现表分区：第 1 部分

使用 Precission Connect 将大型机数据库复制到 AWS

由 Lucio Pereira (AWS)、Balaji Mohan (AWS) 和 Sayantan Giri (AWS) 编写

摘要

此模式概述了使用 Precission Connect 近乎实时地将数据从大型机数据库复制到 Amazon 数据存储的步骤。它通过 Amazon Managed Streaming for Apache Kafka (Amazon MSK) 和云中的自定义数据库连接器实现了基于事件的架构，以提高可扩展性、弹性和性能。

Precisely Connect 是一种复制工具，可从遗留大型机系统捕获数据并将其集成到云环境中。通过使用具有低延迟和高吞吐量的异构数据管道的近乎实时的消息流，通过变更数据捕获 (CDC) 将数据从大型机复制到 AWS。

该模式还涵盖了具有多区域数据复制和失效转移路由功能的弹性数据管道的灾难恢复策略。

先决条件和限制

先决条件

- 要复制到 AWS 云的现有大型机数据库，例如 IBM DB2、IBM 信息管理系统 (IMS) 或虚拟存储访问方法 (VSAM)
- 一个有效的 Amazon Web Services account
- 从您的企业环境到 AWS 的 AWS Direct Connect 或 AWS Virtual Private Network (AWS VPN)
- 具有可由您的旧平台访问的子网的虚拟私有云

架构

源技术堆栈

至少包含以下数据库之一的大型机环境：

- IBM IMS 数据库
- IBM DB2 数据库
- VSAM 文件

目标技术堆栈

- Amazon MSK
- Amazon Elastic Kubernetes Service (Amazon EKS) 和 Amazon EKS Anywhere
- Docker
- AWS 关系或 NoSQL 数据库，如下所示：
  - Amazon DynamoDB
  - Amazon Relational Database Service (Amazon RDS) for Oracle、Amazon RDS for PostgreSQL 或 Amazon Aurora
  - ElastiCache 适用于 Redis 的 Amazon
  - Amazon Keyspaces（Apache Cassandra 兼容）

目标架构

将大型机数据复制至 AWS 数据库

下图说明了将大型机数据复制到 AWS 数据库，例如 DynamoDB、Amazon RDS、Amazon 或 Amazon Keyspac ElastiCache es 的情况。通过在本地大型机环境中使用 Precisely Capture 和 Publisher、在本地分布式环境中使用 Amazon EKS Anywhere 上的 Precisely Dispatcher 以及在 Amazon Web Services Cloud 中使用 Precisely Apply Engine 和数据库连接器，可以近乎实时地进行复制。

图表显示了以下工作流：

1. Precisely Capture 从 CDC 日志中获取大型机数据，并将数据维护在内部临时存储中。
2. Precisely Publisher 侦听内部数据存储中的更改，并通过 TCP/IP 连接将 CDC 记录发送到 Precisely Dispatcher。
3. Precisely Dispatcher 从 Publisher 接收 CDC 记录并将其发送到 Amazon MSK。调度程序根据用户配置和多个工作任务创建 Kafka 键以并行推送数据。当记录存储在 Amazon MSK 中后，调度程序会向 Publisher 发送确认信息。
4. Amazon MSK 在云环境中保存 CDC 记录。主题的分区大小取决于您的事务处理系统 (TPS) 对吞吐量的要求。Kafka 密钥对于进一步的转换和事务排序是必需的。
5. Precisely Apply Engine 监听来自 Amazon MSK 的 CDC 记录，并根据目标数据库要求转换数据（例如，通过筛选或映射）。您可将自定义逻辑添加至 Precission SQD 脚本。（SQD 是 Precist 的专有语言。）Precisely Apply Engine 将每条 CDC 记录转换为 Apache Avro 或 JSON 格式，并根据您的要求将其分发到不同的主题。

6. 目标 Kafka 主题根据目标数据库保存多个主题中的 CDC 记录，并且 Kafka 根据定义的 Kafka 键促进事务排序。分区键与相应的分区对齐以支持顺序过程。
7. 数据库连接器（自定义 Java 应用程序)监听来自 Amazon MSK 的 CDC 记录并将其存储在目标数据库中。
8. 可根据您的要求选择目标数据库。此模式同时支持 NoSQL 与关系数据库。

灾难恢复

业务连续性是组织成功的关键。Amazon Web Services Cloud 提供高可用性 (HA) 和灾难恢复 (DR) 功能，并支持贵组织的失效转移和备用计划。此模式遵循主动/被动灾难恢复策略，并为实施满足 RTO 和 RPO 要求的灾难恢复策略提供高级指导。

下图说明了 DR 的工作流。

此图显示以下内容：

1. 如果 AWS 区域 1 发生任何故障，则需要进行半自动失效转移。如果区域 1 出现故障，系统必须启动路由更改，才能将 Precisly Dispatcher 连接至区域 2。
2. Amazon MSK 在不同区域间通过镜像复制数据，因此，在失效转移期间，区域 2 中的 Amazon MSK 集群必须提升为主要领导者。
3. Precisely Apply Engine 和数据库连接器是无状态应用程序，可以在任何区域中工作。
4. 数据库同步取决于目标数据库。例如，DynamoDB 可以使用全局表，也可以使用全局数据 ElastiCache 存储。

通过数据库连接器执行低延迟和高吞吐量处理

数据库连接器是此模式中的关键组件。连接器采用基于侦听器的方法从 Amazon MSK 收集数据，并通过任务关键型应用程序（第 0 层和第 1 层)的高吞吐量和低延迟处理将事务发送到数据库。下图阐明了此过程。

该模式支持通过多线程处理引擎开发具有单线程消耗的定制应用程序。

1. 连接器主线程使用来自 Amazon MSK 的 CDC 记录并将其发送至线程池进行处理。
2. 线程池中的线程处理 CDC 记录并将其发送至目标数据库。

3. 如果所有线程都处于繁忙状态，则线程队列保留 CDC 记录。
4. 主线程等待从线程队列中清除所有记录，并将偏移量提交至 Amazon MSK 中。
5. 子线程处理故障。如果在处理过程中发生故障，则失败的消息将发送到 DLQ（死信队列）主题。
6. 子线程根据大型机时间戳启动条件更新（参见 DynamoDB 文档中的条件表达式），以避免数据库中的任何重复 out-of-order或更新。

有关如何实现具有多线程功能的 Kafka 消费者应用程序的信息，请参阅 Confluent 网站上的博客文章 Apache Kafka 消费者使用多线程消息消费。

工具

Amazon Web Services

- Amazon Managed Streaming for Apache Kafka (Amazon MSK) 是一项完全托管式服务，可帮助您构建并运行使用 Apache Kafka 来处理流数据的应用程序。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可帮助您在 AWS 上运行 Kubernetes，而无需安装或维护您自己的 Kubernetes 控制面板或节点。
- Amazon EKS Anywhere 帮助您部署、使用和管理在您自己的数据中心运行的 Kubernetes 集群。
- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- Amazon Relational Database Service (Amazon RDS) 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展关系数据库。
- Amazon ElastiCache 可帮助您在 AWS 云中设置、管理和扩展分布式内存缓存环境。
- Amazon Keyspaces（Apache Cassandra 兼容）是一项托管数据库服务，可帮助您在 Amazon Web Services Cloud 中迁移、运行和扩展 Cassandra 工作负载。

其他工具

- Precission Connect 将来自传统大型机系统（例如 VSAM 数据集或 IBM 大型机数据库)的数据集成到下一代云和数据平台中。

最佳实践

- 找到 Kafka 分区和多线程连接器的最佳组合，从而平衡最佳性能和成本。由于 MIPS（每秒百万条指令）消耗量更高，多个 Precist Capture 和 Dispatcher 实例可能会增加成本。

- 避免向数据库连接器添加数据操作和转换逻辑。为此，请使用 Precisely Apply Engine，它提供以微秒为单位的处理时间。
- 在数据库连接器中创建对数据库的定期请求或运行状况检查调用（检测信号），以频繁预热连接并减少延迟。
- 实现线程池验证逻辑，以了解线程队列中的待处理任务，并在下一次 Kafka 轮询之前等待所有线程完成。这有助于避免节点、容器或进程崩溃时数据丢失。
- 通过运行状况端点公开延迟指标，通过控制面板和跟踪机制增强可观测性。

操作说明

准备源环境（本地）

Task	描述	所需技能
设置大型机过程（批处理或在线实用程序），以从大型机数据库启动 CDC 过程。	1. 确定大型机环境。 2. 确定将参与 CDC 流程的大型机数据库。 3. 在大型机环境中，开发一个启动 CDC 工具来捕获源数据库中的更改的过程。有关说明，请参阅大型机文档。 4. 记录 CDC 过程，包括配置。 5. 在测试和生产环境部署该过程。	大型机工程师
激活大型机数据库日志流。	1. 在大型机环境中配置日志流以捕获 CDC 日志。有关说明，请参阅大型机文档。 2. 测试日志流以确保捕获了必要的数据。 3. 在测试和生产环境部署日志流。	大型机数据库专家

Task	描述	所需技能
使用捕获组件捕获 CDC 记录。	1. 在大型机环境中安装与配置 Precision Capture 组件。有关说明，请参阅 Precisely 文档。 2. 测试配置以确保 Capture 组件正常运行。 3. 设置复制过程以通过捕获组件复制捕获的 CDC 记录。 4. 记录每个源数据库的捕获配置。 5. 开发监控系统以确保 Capture 组件随着时间的推移正确收集日志。 6. 在测试和生产环境中部署安装与配置。	大型机工程师、Precisely Connect SME
配置 Publisher 组件以侦听 Capture 组件。	1. 在大型机环境中安装并配置 Precisely Publisher 组件。有关说明，请参阅 Precisely 文档。 2. 测试配置以确保 Publisher 组件正常工作。 3. 设置复制流程，将 CDC 记录从 Publisher 发布至 Precist Dispatcher 组件。 4. 记录 Publisher 配置。 5. 开发一个监控系统以确保 Publisher 组件随着时间的推移正常工作。 6. 在测试和生产环境中部署安装与配置。	大型机工程师、Precisely Connect SME

Task	描述	所需技能
在本地分布式环境中预配 Amazon EKS Anywhere。	1. 在本地基础设施上安装 Amazon EKS Anywhere，并确保其配置正确。有关说明，请参阅 Amazon EKS Anywhere 文档。 2. 为 Kubernetes 集群设置安全的网络环境，包括防火墙。 3. 实施并测试将示例应用程序部署到 Amazon EKS Anywhere 集群。 4. 实现集群的自动伸缩功能。 5. 制定并实施备份和灾难恢复程序。	DevOps 工程师
在分布式环境中部署和配置 Dispatcher 组件，以便在 Amazon Web Services Cloud 中发布主题。	1. 配置和容器化 Precisely Dispatcher 组件。有关说明，请参阅 Precisely 文档。 2. 将 Dispatcher Docker 映像部署至本地 Amazon EKS Anywhere 环境中。 3. 在 Amazon Web Services Cloud 和 Dispatcher 之间建立安全连接。 4. 开发监控系统以确保 Dispatcher 组件随着时间的推移正常工作。 5. 在测试和生产环境中部署安装与配置。	DevOps 工程师，Precission Conn

准备目标环境 (AWS)

Task	描述	所需技能
在指定的 AWS 区域配置 Amazon EKS 集群。	1. 登录您的 Amazon Web Services account 并对其进行配置，以确保拥有创建和管理 Amazon EKS 集群所需权限。 2. 在所选 AWS 区域中创建虚拟私有云（VPC）和子网。有关说明，请参阅 Amazon EKS 文档。 3. 创建和配置必要的网络安全组，以允许 Amazon EKS 集群与 VPC 中的其他资源间进行通信。有关更多信息，请参阅 Amazon EKS 文档。 4. 创建 Amazon EKS 集群并使用正确的节点组大小和实例类型对其进行配置。 5. 通过部署示例应用程序，以验证 Amazon EKS 集群。	DevOps 工程师、网络管理员
配置 MSK 集群并配置适用 Kafka 主题。	1. 配置您的 Amazon Web Services account，确保拥有创建和管理 MSK 集群所需权限。 2. 创建和配置必要的网络安全组，以允许 MSK 集群与 VPC 中的其他资源进行通信。有关更多信息，请参阅 Amazon VPC 文档。 3. 创建 MSK 集群，并将其配置为包含应用程序将使用的 Kafka 主题。有关更多信息	DevOps 工程师、网络管理员

Task	描述	所需技能
	，请参阅 Amazon MSK 文档。	
配置 Apply Engine 组件，以侦听复制的 Kafka 主题。	1. 配置和容器化 Precisely Apply Engine 组件。 2. 将 Apply Engine Docker 映像部署到您的 Amazon Web Services account 中的 Amazon EKS 集群。 3. 设置 Apply Engine 以侦听 MSK 主题。 4. 在 Apply Engine 中开发和配置 SQD 脚本来处理筛选和转换。有关更多信息，请参阅 Precisely 文档。 5. 在测试与生产环境中部署应用引擎。	Precisely Connect SME

Task	描述	所需技能
在 Amazon Web Services Cloud 中配置 DB 实例。	1. 配置 Amazon Web Services account，确保拥有创建和管理数据库集群和表所需权限。有关说明，请参阅您希望使用的 AWS 数据库服务的 AWS 文档。（有关链接，请参阅资源部分。） 2. 在选定的 AWS 区域创建 VPC 和子网。 3. 创建和配置必要的网络安全组，以允许数据库实例与 VPC 中的其他资源进行通信。 4. 创建数据库，并将其配置为包含应用程序将使用的表。 5. 设计和验证数据库架构。	数据工程师、 DevOps 工程师
配置和部署数据库连接器以侦听 Apply Engine 发布的主题。	1. 设计数据库连接器，将 Kafka 主题与您在前面步骤中创建的 AWS 数据库连接。 2. 根据目标数据库开发连接器。 3. 配置连接器以侦听 Apply Engine 发布的 Kafka 主题。 4. 将连接器部署到 Amazon EKS 集群中。	应用程序开发人员、云架构师、数据工程师

设置业务连续性和灾难恢复

Task	描述	所需技能
为您的业务应用程序定义灾难恢复目标。	1. 根据您的业务需求和影响分析，定义 CDC 渠道 RPO 和 RTO 目标。 2. 定义沟通和通知程序，以确保所有利益相关者都了解灾难恢复计划。 3. 确定实施灾难恢复计划所需预算以及资源。 4. 记录灾难恢复目标，包含 RPO 和 RTO 目标。	云架构师、数据工程师、应用程序所有者
根据定义 RTO/RPO 设计灾难恢复策略。	1. 根据您的关键程度和恢复要求，确定最适合 CDC 管道的灾难恢复策略。 2. 定义灾难恢复架构和拓扑。 3. 定义 CDC 管道的失效转移和失效自动恢复程序，以确保它们能够快速、无缝地切换到备份区域。 4. 记录灾难恢复策略和程序，并确保所有利益相关者对设计有清晰的了解。	云架构师、数据工程师
配置灾难恢复集群和配置。	1. 为灾难恢复配置辅助 AWS 区域。 2. 在辅助的 AWS 区域 中，创建与主 AWS 区域 相同的环境。 3. 在主区域和辅助区域 MirrorMaker 之间配置 Apache Kafka。有关更多信	DevOps 工程师、网络管理员、云架构师

Task	描述	所需技能
	息，请参阅 Amazon MSK 文档。 4. 在辅助区域中配置备用应用程序。 5. 配置主区域和辅助区域之间的数据库复制。	
测试 CDC 管道灾难恢复。	1. 定义 CDC 管道灾难恢复测试的范围和目标，包括测试场景和要实现的 RTO。 2. 确定用于进行灾难恢复测试的测试环境和基础设施。 3. 准备测试数据集和脚本来模拟故障场景。 4. 验证数据的完整性和一致性，确保数据不丢失。	应用程序所有者、数据工程师、云架构师

相关资源

AWS 资源

- Amazon DynamoDB
- Amazon DynamoDB 的条件表达式
- Amazon EKS
- Amazon EKS Anywhere
- Amazon ElasticCache
- Amazon Keyspaces
- Amazon MSK
- Amazon RDS 和 Amazon Aurora
- Amazon VPC

Precisely Connect 资源

- Precisely Connect 概述
- 使用 Precise Connect 更改数据捕获

Confluent 资源

- Apache Kafka 消费者使用多线程消息消费

使用 Lambda 和 Secrets Manager 计划适用于 Amazon RDS for PostgreSQL 和 Aurora PostgreSQL 的任务

由 Yaser Raja (AWS) 创建

摘要

对于本地数据库和托管在亚马逊弹性计算云 (Amazon EC2) 实例上的数据库，数据库管理员通常使用 cron 实用程序来安排作业。

例如，使用 cron 可轻松地安排数据提取任务或数据清除任务。对于这些任务，数据库凭证通常为硬编码或存储于属性文件。但是，当您迁移至 Amazon Relational Database Service (Amazon RDS) 或 Amazon Aurora PostgreSQL-Compatible Edition，您将无法登录主机实例安排 cron 任务。

此模式描述了迁移后如何使用 AWS Lambda 和 AWS Secrets Manager 安排适用于 PostgreSQL 的亚马逊 RDS 和兼容 Aurora PostgreSQL 的数据库的作业。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- Amazon RDS for PostgreSQL 或 Aurora PostgreSQL-Compatible 数据库

限制

- 任务必须在 15 分钟内完成，这是 Lambda 函数的超时限值。有关其他限制，请参阅AWS Lambda 文档。
- 任务代码必须按 Lambda 支持语言编写。

架构

源技术堆栈

此堆栈包含通过 Bash、Python 和 Java 等语言编写的任务。数据库凭证存储于属性文件，任务使用 Linux cron 调度。

目标技术堆栈

此堆栈包含 Lambda 函数，该函数使用存储在 Secrets Manager 中的凭证连接至数据库并执行活动。Lambda 函数通过使用亚马逊 CloudWatch 事件按计划的时间间隔启动。

目标架构

工具

- Amazon CloudWatch Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。使用可以快速设置的简单规则，您可以匹配事件并将它们路由到一个或多个目标函数或流。CloudWatch 事件在发生时就会意识到操作变化。其可响应这些操作更改并在必要时采取纠正措施，方式是发送消息以响应环境、激活函数、进行更改并捕获状态信息。您还可以使用 Ev CloudWatch ents 来安排自动操作，这些操作在特定时间使用 cron 或速率表达式自行启动。
- AWS Lambda 是一项计算服务，可使您无需调配或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。您只需按消耗的计算时间付费；代码未运行时不产生费用。借助 Lambda，您几乎可以为任何类型的应用程序或后端服务运行代码，无需任何管理。Lambda 在高可用性计算基础设施上运行您的代码，并管理所有计算资源，包括服务器和操作系统维护、容量配置和自动扩展、代码监控和日志记录。您只需使用 Lambda 支持的语言之一提供您的代码即可。
- AWS Secrets Manager帮助您保护用于访问应用程序、服务和 IT 资源的秘密。您可以在数据库凭证、API 密钥和其他密钥的整个生命周期内轻松地对其进行轮换、管理和检索。用户和应用程序通过调用 Secrets Manager 来检索机密 APIs，无需以纯文本格式对敏感信息进行硬编码。Secrets Manager 使用 Amazon RDS、Amazon Redshift 和 Amazon DocumentDB 的内置集成提供密钥轮换。该服务可扩展到其他类型的机密，包括 API 密钥和 OAuth 令牌。Secrets Manager 使您能够使用细粒度的权限控制对机密的访问 AWS Cloud，并集中审核第三方服务和内部资源的密钥轮换。

操作说明

将数据库凭证存储至 Secrets Manager

Task	描述	所需技能
为 Lambda 函数 创建数据库用户。	最好将不同的数据库用户用于不同的应用程序部分。如果您的 cron 任务已有单独的数据库用户，请使用该用户。否	数据库管理员

Task	描述	所需技能
	则，创建一个新数据库用户。有关更多信息，请参阅管理 PostgreSQL 用户和角色AWS（博客文章）。	
将数据库凭证作为密钥存储至 Secrets Manager。	按照创建数据库密钥（Secrets Manager 文档）中的说明。	数据库管理员， DevOps

编写 Lambda 函数的代码。

Task	描述	所需技能
选择 Lambda 支持的编程语言。	有关支持的语言列表，请参阅 Lambda 运行时系统（Lambd 文档）。	开发人员
编写从 Secrets Manager 获取数据库凭证的逻辑。	有关示例代码，请参阅如何使用安全地向 Lambda 函数提供数据库凭证 AWS Secrets Manager（AWS 博客文章）。	开发人员
编写执行计划数据库活动的逻辑。	将您在本地使用的调度任务的现有代码迁移到 Lambda 函数。有关更多信息，请参阅部署 Lambda 函数（Lambda 文档）。	开发人员

部署代码并创建 Lambda 函数

Task	描述	所需技能
创建 Lambda 函数部署包。	此数据包包含代码及其依赖项。有关更多信息，请参阅部署包（Lambda 文档）。	开发人员

Task	描述	所需技能
创建 Lambda 函数。	在 Lambda 控制台中，选择创建函数，输入函数名称，选择运行时环境，然后选择创建函数。	DevOps
构建部署程序包。	选择您创建的 Lambda 函数，以打开其配置。您可以直接在代码部分编写代码或上传部署包。若要上传数据包，请前往函数代码部分，选择要上传 .zip 文件的代码条目类型，然后选择该数据包。	DevOps
根据您的要求配置 Lambda 函数。	例如，您可以将超时参数设置为您的 Lambda 函数预计花费的时间。有关更多信息，请参阅配置函数选项（Lambda 文档）。	DevOps
为 Lambda 函数角色设置 Secrets Manager 访问权限。	有关说明，请参阅在 AWS Lambda 函数中使用密钥（Secrets Manager 文档）。	DevOps
测试 Lambda 函数	手动启动 Lambda 函数以确保其按预期运行。	DevOps

使用事件安排 Lambda 函数 CloudWatch

Task	描述	所需技能
创建按计划运行 Lambda 函数的规则。	使用 CloudWatch 事件安排 Lambda 函数。有关说明，请参阅使用 CloudWatch 事件安	DevOps

Task	描述	所需技能
	排 Lambda 函数（CloudWatch 事件教程）。	

相关资源

- AWS Secrets Manager
- Lambda 入门
- 创建在 CloudWatch 事件上触发的事件规则
- AWS Lambda 限制
- 从无服务器应用程序查询 AWS 数据库（博客文章）

使用本地 SMTP 服务器和数据库邮件发送 Amazon RDS for SQL Server 数据库实例通知

创建者：Nishad Mankar (AWS)

摘要

数据库邮件（Microsoft 文档）使用简单邮件传输协议 (SMTP) 服务器从 Microsoft SQL Server 数据库发送电子邮件，例如通知或警报。Amazon Relational Database Service (Amazon RDS) for Microsoft SQL Server 文档提供了使用 Amazon Simple Email Service (Amazon SES) 作为数据库邮件的 SMTP 服务器的说明。有关详细信息，请参阅在 Amazon RDS for SQL Server 上使用 Database Mail。作为替代配置，此模式说明了如何使用本地 SMTP 服务器作为邮件服务器，将数据库邮件配置为从 Amazon RDS for SQL Server 数据库（DB）实例发送电子邮件。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 运行标准版或企业版 SQL Server 的 Amazon RDS 数据库实例
- 本地 SMTP 服务器的 IP 地址或主机名
- 入站安全组规则，允许从 SMTP 服务器的 IP 地址连接到 Amazon RDS for SQL Server 数据库实例
- 您的本地网络与包含 Amazon RDS 数据库实例的虚拟私有云（VPC）之间的连接，例如 AWS Direct Connect 连接

限制

- 不支持 SQL Server Express 版。
- 有关限制的更多信息，请参阅 Amazon RDS 文档中的在 Amazon RDS for SQL Server 上使用数据库邮件中的限制。

产品版本

- RDS 中支持的 SQL Server 版本的标准版和企业版

架构

目标技术堆栈

- Amazon RDS for SQL Server 数据库实例
- Amazon Route 53 转发规则
- 数据库邮件
- 本地 SMTP 服务器
- Microsoft SQL Server Management Studio (SSMS)

目标架构

下图显示了此模式的目标架构。当发生启动有关数据库实例的通知或警报的事件或操作时，Amazon RDS for SQL Server 使用数据库邮件发送电子邮件通知。数据库邮件通过本地 SMTP 服务器发送电子邮件。

工具

Amazon Web Services

- Amazon Relational Database Service (Amazon RDS) for Microsoft SQL Server 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展 SQL Server 关系数据库。
- Amazon Route 53 是一种可用性高、可扩展性强的 DNS Web 服务。

其他工具

- 数据库邮件是一种从 SQL Server 数据库引擎向用户发送电子邮件（例如通知和警报）的工具。
- Microsoft SQL Server Management Studio (SSMS) 是一款用于管理 SQL Server 的工具，包括访问、配置和管理 SQL Server 组件。在这种模式中，您可使用 SSMS 运行 SQL 命令在 Amazon RDS for SQL Server 数据库实例上设置数据库邮件。

操作说明

启用与本地 SMTP 服务器的网络连接

Task	描述	所需技能
从 RDS 数据库实例中删除多可用区。	如果您使用的是多可用区 RDS 数据库实例，请将多可用区实例转换为单可用区实例。完成配置数据库邮件后，您会将数据库实例转换回多可用区部署。然后，主节点和辅助节点都具有数据库邮件配置。有关说明，请参阅从 Microsoft SQL Server 数据库实例删除多可用区。	数据库管理员
为本地 SMTP 服务器上的 Amazon RDS 端点或 IP 地址创建允许列表。	SMTP 服务器位于 AWS 网络之外。在本地 SMTP 服务器上，创建一个允许列表，允许服务器与 Amazon RDS 实例或托管在 Amazon RDS 上的亚马逊弹性计算云 (Amazon EC2) 实例的出站终端节点或 IP 地址进行通信。此进程因组织而异。有关数据库实例端点的更多信息，请参阅查找数据库实例端点和端口号。	数据库管理员
移除端口 25 限制。	默认情况下，AWS 限制 EC2 实例上的端口 25。要取消端口 25 限制，请执行以下操作： 1. 使用您的 Amazon Web Services account 登录，然后打开申请取消电子邮件发送限制表单。	常规 AWS

Task	描述	所需技能
	2. 输入您的电子邮件地址，以便 Amazon Web Services Support 联系您，告知您的请求的最新情况。 3. 在用例描述字段提供所需信息。 4. 选择提交。 Note • 如果您在多个 AWS 区域有实例，请为每个区域提交单独的请求。 • 处理您的请求最长可能需要 48 小时。	
添加 Route 53 规则，以解析 SMTP 服务器 DNS 查询。	使用 Route 53 解析您的 AWS 资源和本地 SMTP 服务器之间的 DNS 查询。您必须创建将 DNS 查询转发到 SMTP 服务器域的规则，例如 `example.com` 。有关说明，请参阅 Route 53 文档中的创建转发规则。	网络管理员

在 Amazon RDS for SQL Server 数据库实例上设置数据库邮件

Task	描述	所需技能
启用数据库邮件。	为数据库邮件创建参数组，将 `database mail xps` 参数设置为 `1`，然后将数据库邮	数据库管理员

Task	描述	所需技能
	件参数组与目标 RDS 数据库实例关联。有关说明，请参阅 Amazon RDS 文档中的启用数据库邮件。不要继续进入到这些说明中的配置数据库邮件部分。本地 SMTP 服务器配置与 Amazon SES 不同。	
连接到数据库实例。	从堡垒主机，使用 Microsoft SQL Server Managemen t Studio (SSMS) 连接至 Amazon RDS for SQL Server 数据库实例。有关说明，请参阅连接到运行 Microsoft SQL Server 数据库引擎的数据库实例。如果您遇到任何错误，请参阅相关资源部分中的连接疑难解答参考资料。	数据库管理员

Task	描述	所需技能
创建配置文件。	在 SSMS 中，输入以下 SQL 语句创建数据库邮件配置文件。替换以下值： • 对于 `profile_name` ，请为新配置文件输入一个名称。 • 对于 `description` ，请输入新配置文件的简要描述。 有关此存储过程及其参数的更多信息，请参阅 Microsoft 文档中的 sysmail_add_profile_sp。 `EXECUTE msdb.dbo.sysmail_add_profile_sp @profile_name = 'SQL Alerts profile', @description = 'Profile used for sending outgoing notifications using OM SMTP Server.';`	数据库管理员

Task	描述	所需技能
在个人资料中添加主体。	输入以下 SQL 语句，将公共或私有主体添加到数据库邮件配置文件中。委托人指可以请求获取 SQL Server 资源的实体。替换以下值： • 对于 `profile_name` ，请输入您之前创建的配置文件名称。 • 对于 `principal_name` ，请输入数据库用户或角色的名称。数据库主体必须映射到 SQL Server 身份验证用户、Windows 身份验证用户或 Windows 身份验证组。 有关此存储过程及其参数的更多信息，请参阅 Microsoft 文档中的 sysmail_add_principalprofile_sp。 `EXECUTE msdb.dbo.sysmail_add_principalprofile_sp @profile_name = 'SQL Alerts profile', @principal_name = 'public', @is_default = 1 ;`	数据库管理员

Task	描述	所需技能
创建账户。	输入以下 SQL 语句创建数据库邮件账户。替换以下值： • 对于 `account_name` ，请为新账户输入名称。 • 对于 `description` ，请输入新账户的简要描述。 • 对于 `email_address` ，请输入用于发送数据库邮件消息的电子邮件地址。 • 对于 `display_address` ，请输入用于该账户外发消息的显示名称，例如 `SQL Server Automated Notification` 。您也可以使用为 `email_address` 输入的值。 • 对于 `mailserver_name` ，请输入 SMTP 邮件服务器名称或 IP 地址。 • 对于 `port`，请保留值 25。 • 对于 `enable_ssl` ，如果您不想让数据库邮件使用 SSL 加密通信，请将该值保留为 1 或输入 0。 • 对于 `username`，请输入用于登录 SMTP 邮件服务器的用户名。如果服务器不需要身份验证，请输入 NULL。 • 对于 `password`，请输入用于登录 SMTP 邮件服务器的	数据库管理员

Task	描述	所需技能
	密码。如果服务器不需要身份验证，请输入 NULL。 有关此存储过程及其参数的更多信息，请参阅 Microsoft 文档中的 sysmail_add_account_sp。 `EXECUTE msdb.dbo.sysmail_add_account_sp @account_name = 'SQL Alerts account', @description = 'Database Mail account for sending outgoing notifications.', @email_address = 'xyz@example.com', @display_name = 'xyz@example.com', @mailserver_name = 'test_smtp.example.com', @port = 25, @enable_ssl = 1, @username = 'SMTP-username', @password = 'SMTP-password';`	

Task	描述	所需技能
将账户添加到配置文件中。	输入以下 SQL 语句以将数据库邮件账户添加到数据库邮件配置文件中。替换以下值： • 对于 `profile_name` ，请输入您之前创建的配置文件名称。 • 对于 `account_name` ，请输入您之前创建的账户的名称。 有关此存储过程及其参数的更多信息，请参阅 Microsoft 文档中的 sysmail_add_profileaccount_sp。 `EXECUTE msdb.dbo.sysmail_add_profileaccount_sp @profile_name = 'SQL Alerts profile', @account_name = 'SQL Alerts account', @sequence_number = 1;`	数据库管理员
（可选）向 RDS 数据库实例添加多可用区。	如果要添加带数据库镜像 (DBM) 或始终开启可用性组 (AGs) 的多可用区，请参阅向 Microsoft SQL Server 数据库实例添加多可用区中的说明。	数据库管理员

相关的资源

- 在 Amazon RDS for SQL Server 上使用数据库邮件（Amazon RDS 文档）
- 处理文件附件（Amazon RDS 文档）

- 排除与 SQL Server 数据库实例的连接故障（Amazon RDS 文档）
- 无法连接到 Amazon RDS 数据库实例（Amazon RDS 文档）

在 IBM Db2 on AWS 上为 SAP 设置灾难恢复

由 Ambarish Satarkar (AWS) 和 Debasis Sahoo (AWS) 编写

摘要

此模式概述了使用 IBM Db2 作为数据库平台并在 Amazon Web Services (AWS) 云上运行的 SAP 工作负载设置灾难恢复 (DR) 系统的步骤。目标是提供一种低成本解决方案，以在发生中断时提供业务连续性。

该图案使用指示灯方法。通过在 AWS 上实施指示灯灾难恢复，您可以减少停机时间并保持业务连续性。指示灯方法侧重于在 AWS 中设置最小的灾难恢复环境，包括 SAP 系统和与生产环境同步的备用 Db2 数据库。

该解决方案具有可扩展性。您可以根据需要将其扩展至全面的灾难恢复环境。

先决条件和限制

先决条件

- 在亚马逊弹性计算云 (Amazon EC2) 实例上运行的 SAP 实例
- IBM Db2 数据库
- SAP Product Availability Matrix (PAM) 支持的操作系统
- 生产数据库主机和备用数据库主机的物理数据库主机名不同
- AWS 区域 中的 Amazon Simple Storage Service (Amazon S3) 存储桶，且已启用跨区域复制 (CRR)

产品版本

- IBM Db2 数据库 11.5.7 或更高版本

架构

目标技术堆栈

- Amazon EC2
- Amazon Simple Storage Service（Amazon S3）
- Amazon 虚拟私有云（VPC 对等连接）

- Amazon Route 53
- IBM Db2 高可用性灾难恢复 (HADR)

目标架构

该架构以 Db2 作为数据库平台，为 SAP 工作负载实现了灾难恢复解决方案。生产数据库部署在 AWS 区域 1 中，备用数据库部署在第二个区域。备用数据库称为灾难恢复系统。Db2 数据库支持多个备用数据库（最多三个）。它使用 Db2 HADR 设置灾难恢复数据库并在生产数据库和备用数据库之间自动进行日志传送。

如果发生灾难导致区域 1 不可用，DR 区域中的备用数据库将接管生产数据库角色。SAP 应用程序服务器可以提前构建，也可以使用 AWS Elastic Disaster Recovery 或亚马逊机器映像（AMI）满足恢复时间目标（RTO）要求。此示例使用 AMI。

Db2 HADR 实现生产备用设置，其中生产充当主服务器，所有用户都连接到它。所有事务都写入日志文件，并使用 TCP/IP 将日志文件传输到备用服务器。备用服务器通过前滚传输的日志记录来更新其本地数据库，这有助于确保其与生产服务器保持同步。

使用 VPC 对等互连，以便生产区域和灾难恢复区域中的实例可以相互通信。Amazon Route 53 将最终用户路由至互联网应用程序。

1. 在区域 1 为应用程序创建 AMI，复制 AMI 至区域 2。发生灾难时，使用 AMI 启动区域 2 服务器。
2. 在生产数据库 (在区域 1)和备用数据库 (在区域 2) 之间设置 Db2 HADR 复制。
3. 发生灾难时，请更改 EC2 实例类型以匹配生产实例。
4. 在区域 1 中，将 LOGARCHMETH1 设置为 db2remote: S3 path。
5. 在区域 2 中，将 LOGARCHMETH1 设置为 db2remote: S3 path。
6. 跨区域复制在S3存储桶之间执行。

工具

Amazon Web Services

- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。

- Amazon Route 53 是一种可用性高、可扩展性强的 DNS Web 服务。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。此模式使用 VPC 对等互连。

最佳实践

- 网络在决定 HADR 复制模式方面起着关键作用。对于跨 AWS 区域 的灾难恢复，我们建议您使用 Db2 HADR ASYNC 或 SUPERASYNC 模式。
- 有关 Db2 HADR 复制模式的更多信息，请参阅 IBM 文档。
- 您可以使用 AWS 管理控制台 或 AWS 命令行界面（AWS CLI）针对现有 SAP System 创建新的 AMI。然后，您可以使用 AMI 恢复现有 SAP 系统或者创建克隆。
- AWS Systems Manager Automation 可以帮助完成 EC2 实例和其他 AWS 资源的常见维护和部署任务。
- AWS 提供多种本机服务，监控和管理 AWS 上的基础设施和应用程序。诸如 Amazon CloudWatch 和 AWS 之类的服务 CloudTrail 可以分别用于监控您的底层基础设施和 API 操作。有关更多详细信息，请参阅 SAP on AWS – IBM Db2 HADR， 带 Pacemaker。

操作说明

准备环境

Task	描述	所需技能
检查系统和日志。	1. 确认 Db2 系统上的生产 SAP 已设置。 2. 确认日志备份已打开，并配置为将日志保存在 S3 存储桶中。这可通过 Db2 参数 `LOGARCHMETH1` 进行检查。 3. 创建附加应用服务器 AMI。	AWS 管理员、SAP Basis 管理员

设置服务器和复制

Task	描述	所需技能
创建 SAP 和数据库服务器。	1. 要为灾难恢复区域部署基础设施，请使用 AWS CloudFormation 脚本或生产实例的 AMI。作为指示灯方法的一部分，您可以使用与生产 EC2 实例同一个系列中的较小实例。例如，如果您的生产实例类型为 `r6i.12xlarge` ，则可以将 `r6i.xlarge` 实例类型用于灾难恢复构建。但是，请确保灾难恢复实例上分配相同的存储容量，以恢复生产数据库备份。 2. 为 `/sapmnt/<SID>/` 创建 Amazon Elastic File System (Amazon EFS) 挂载点，并确保将其 设置为从主系统复制。 3. 从生产系统进行完整数据库备份（在线或离线）。您将使用此备份构建灾难恢复数据库。 4. 在灾难恢复系统中，使用 SAP 软件配置管理器 (SWPM) 系统复制方法和使用系统副本来backup/restore for HA/DR构建 DR SAP 系统。 5. 当 SWPM 要求时，使用从生产中获取的备份恢复灾难	SAP Basis 管理员

Task	描述	所需技能
	恢复中的数据库。DR 数据库将处于前滚挂起状态。 恢复完整备份后，默认设置前滚挂起状态。前滚挂起状态指示数据库正在恢复，并且可能需要应用一些更改。有关更多信息，请参阅 IBM 文档。	

Task	描述	所需技能

检查配置。	1. 要为 HADR 设置日志归档，生产数据库和灾难恢复数据库都必须能够从所有日志归档位置自动检索日志。验证灾难恢复数据库中的 LOGARCHMETH1 参数是否设置为与生产数据库中的参数相同的位置。如果因区域限制而无法访问同一位置，请确保灾难恢复系统可以自动从主系统获取日志。
	2. 要启用 TCP/IP 端口以启用数据库复制，请通过添加以下两个 /etc/services 条目在生产和灾难恢复主机中进行修改。在代码中，<SID> 指的是 Db2 数据库的系统 ID (SID)(例如 PR1)。
	`<SID>_HADR_1 55001/tcp # DB2 HADR Port1`
	`<SID>_HADR_2 55002/tcp # DB2 HADR Port2`
	确认两个端口都允许主端口和备用端口间的入站和出站流量。
	3. 检查 /etc/hosts 生产主机和灾难恢复主机，确认生产主机和备用主机的主机名都指向正确的 IP 地址。
		AWS 管理员、SAP Basis 管理员

Task	描述	所需技能

设置从生产数据库到灾难恢复数据库的复制（使用异步模式）。	1. 在生产数据库中，执行以下命令更新参数。
	
	`db2 UPDATE DB CFG FOR <SID> USING HADR_LOCAL_HOST HOST1`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_LOCAL_SVC <SID>_HADR_1`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_REMOTE_HOST HOST2`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_REMOTE_SVC <SID>_HADR_2`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_REMOTE_INST db2<sid>`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_TIMEOUT 120`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_SYNCMODE ASYNC`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_SPOOL_LIMIT 1000`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_PEER_WINDOW 240`
	`db2 UPDATE DB CFG FOR <SID> USING indexrec RESTART logindexbuild ON`
	
	2. 在灾难恢复数据库，中执行以下命令更新参数。
		SAP Basis 管理员

Task	描述	所需技能

	`db2 UPDATE DB CFG FOR <SID> USING HADR_LOCAL_HOST HOST2`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_LOCAL_SVC <SID>_HADR_2`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_REMOTE_HOST HOST1`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_REMOTE_SVC <SID>_HADR_1`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_REMOTE_INST db2<sid>`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_TIMEOUT 120`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_SYNCMODE ASYNC`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_SPOOL_LIMIT 1000`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_PEER_WINDOW 240`
	`db2 UPDATE DB CFG FOR <SID> USING indexrec RESTART logindexbuild ON`
	
	需要这些参数向两个数据库提供 HADR 相关信息。在 Db2 数据库，根据先前设置的每个参数的值激活 HADR。有关这些参数的更多信息，请参阅 IBM 文档。	

Task	描述	所需技能

	3. 首先使用以下命令，在新创建的备用数据库上启动 HADR。
	
	`db2 deactivate db <SID>`
	`db2 start hadr on db <SID> as standby`
	
	4. 使用以下命令，对生产数据库启动 HADR。
	
	`db2 deactivate db <SID>`
	`db2 start hadr on db <SID> as primary`
	
	5. 检查生产数据库和备用 Db2 数据库是否同步，以及日志传送是否正在进行中。
	
	要监控 HADR 复制状态，请使用以下 db2pd 命令。
	
	`db2pd -d <SID> -hadr`
	
	有关监控 HADR 的更多信息，请参阅 IBM 文档。	

测试灾难恢复失效转移任务

Task	描述	所需技能
规划灾难恢复测试生产业务停机时间。	确保在生产环境中计划所需业务停机时间，以测试灾难恢复失效转移场景。	SAP Basis 管理员

Task	描述	所需技能
创建测试用户。	创建可在灾难恢复主机中进行验证的测试用户（或任何测试更改），以确认灾难恢复失效转移后的日志复制。	SAP Basis 管理员
在控制台上，停止生产 EC2 实例。	在此步骤中启动非正常关闭，以模拟灾难场景。	AWS 系统管理员

扩展 DR EC2 实例以满足要求。	在 EC2 控制台上，更改灾难恢复区域中的实例类型。
	1. 停止实例：如果实例正在运行，您必须先停止，然后才能更改其实例类型。在 EC2 控制台上，选择实例，然后选择停止。
	2. 修改实例类型：在 EC2 控制台上，选择实例，然后选择操作、实例设置、更改实例类型。选择与主实例匹配的实例类型，然后选择应用。
	3. 启动实例：实例类型更改完成后，从 EC2 控制台启动实例，方法是选择实例并选择启动。
	4. 要启动 Db2 数据库，请使用以下命令：
	`db2start`
	`db2 start HADR on db <SID> as standby`
		SAP Basis 管理员

Task	描述	所需技能

发起接管。	在灾难恢复系统 (host2) 中，启动接管进程并启动灾难恢复数据库作为主数据库。
	`db2 takeover hadr on database <SID> by force`
	或者，您可设置以下参数，根据实例类型自动调整数据库内存分配。该 INSTANCE_MEMORY 值可以根据分配给 Db2 数据库的专用内存部分来决定。
	`db2 update db cfg for <SID> using INSTANCE_MEMORY <FIXED VALUE> IMMEDIATE;`
	`db2 get db cfg for <SID> | grep -i DATABASE_MEMORY AUTOMATIC IMMEDIATE;`
	`db2 update db cfg for <SID> using self_tuning_mem ON IMMEDIATE;`
	请使用以下命令验证更改。
	`db2 get db cfg for <SID> | grep -i MEMORY`
	`db2 get db cfg for <SID> | grep -i self_tuning_mem`
		SAP Basis 管理员

在灾难恢复区域中启动适用于 SAP 的应用程序服务器。	使用您在生产系统中创建的 AMI，在灾难恢复区域中启动新的附加应用程序服务器。	SAP Basis 管理员

Task	描述	所需技能
在启动 SAP 应用程序前执行验证。	1. 验证 /etc/hosts 和 /etc/fstab 条目。 2. 在灾难恢复系统上安装 /sapmnt/<SID>/ 。 3. 验证灾难恢复文件系统 /sapmnt/<SID>/ 是否与生产 /sapmnt/<SID>/ 文件系统同步。 4. 登录 <sid>adm 用户，运行 R3trans -d 并验证 trans.log 文件中的输出。trans.log 文件是在运行 R3trans -d 命令的同一位置生成的。	AWS 管理员、SAP Basis 管理员

在灾难恢复系统启动 SAP 应用程序。	使用 <sid>adm 用户在灾难恢复系统上启动 SAP 应用程序。使用以下代码，代码 XX 代表您的 SAP ABAP SAP 中央服务 (ASCS) 服务器的实例号，YY 代表您的 SAP 应用程序服务器的实例号。
	`sapconrol -nr XX -function StartService <SID>`
	`sapconrol -nr XX -function StartSystem`
	`sapconrol -nr YY -function StartService <SID>`
	`sapconrol -nr YY -function StartSystem`
		SAP Basis 管理员

Task	描述	所需技能
执行 SAP 验证。	这是作为灾难恢复测试执行的，以提供证据或检查灾难恢复区域的数据复制是否成功。	测试工程师

执行灾难恢复失效自动恢复任务

Task	描述	所需技能
启动生产 SAP 和数据库服务器。	在控制台上，启动在生产系统中托管 SAP 和数据库的 EC2 实例。	SAP Basis 管理员

启动生产数据库并设置 HADR。	登录至生产系统 (host1)，使用以下命令验证数据库是否处于恢复模式。
	`db2start`
	`db2 start HADR on db P3V as standby`
	`db2 connect to <SID>`
	验证 HADR 状态是否为 `connected` 。复制状态应为 `peer`。
	`db2pd -d <SID> -hadr`
	如果数据库并非不一致且未处于 `connected` 和 `peer` 状态，则可能需要进行备份和恢复，以使数据库 (开启 host1) 与当前活动的数据库 (在灾难恢复区域 host2 中) 同步。在这种情况下，请将数据库备份从 host2 灾难恢复区域的数据库
		SAP Basis 管理员

Task	描述	所需技能
	还原到 host1 生产区域的数据库。	

将数据库故障恢复至生产区域。	在正常 business-as-usual情况下，此步骤是在计划停机时间内执行的。在灾难恢复系统上运行的应用程序将停止，数据库将故障恢复到生产区域 (区域 1)，以便从生产区域恢复操作。
	1. 登录灾难恢复区域中的 SAP 应用程序服务器，然后停止 SAP 应用程序。
	2. /sapmnt/<SID> 从灾难恢复系统中卸载，确保将更改反向复制到 /sapmnt/<SID> 生产系统。
	3. 登录生产区域中的数据库服务器 (host1)，然后执行接管。
	`db2 takeover hadr on database <SID>`
	4. 检查 HADR 状态：HADR_ROLE 应为 host1 的 PRIMARY 和 host2 的 StandBy。
	`db2pd -d <SID> -hadr`
		SAP Basis 管理员

Task	描述	所需技能
在启动 SAP 应用程序前执行验证。	1. 验证 /etc/hosts 和 /etc/fstab 条目。 2. 在生产系统上安装 /sapmnt/<SID>/ 。 3. 确保它与灾难恢复系统 /sapmnt/<SID>/ 同步。 4. 登录 <sid>adm 用户，运行 R3trans -d 并验证 trans.log 文件中的输出。trans.log 文件是在运行 R3trans -d 命令的同一位置生成的。	AWS 管理员、SAP Basis 管理员

启动 SAP 应用程序。	1. 使用 <sid>adm 用户在生产系统上启动 SAP 应用程序。使用以下代码，代码 XX 代表您的 SAP ASCS 服务器的实例号，YY 代表您的 SAP 应用程序服务器的实例号。
	`sapconrol -nr XX -function StartService <SID>`
	`sapconrol -nr XX -function StartSystem`
	`sapconrol -nr YY -function StartService <SID>`
	`sapconrol -nr YY -function StartSystem`
	2. 要确认应用程序服务器是否可用，请登录 SAP 并使用 SICK 和 SM51 事务进行检查。
		SAP Basis 管理员

故障排除

事务	解决方案
用于解决 HADR 相关问题的密钥日志文件和命令	• `db2 get db cfg | grep -i hadr` • `db2pd -d sid -hadr` • `Db2diag.log` （此文件通常位于 db2dump 目录内，db2dump 路径由参数 DIAGPATH 定义。）
SAP 注意事项，用于对 Db2 UDB 上的 HADR 问题进行疑难解答	请参阅 SAP Note 1154013- DB6：HADR 环境中的数据库问题。（您需要 SAP 门户凭证才能访问此笔记。）

相关资源

- AWS 上 Db2 数据库灾难恢复方法（博客文章）
- SAP on AWS — 带有 Pacemaker 的 IBM Db2 HADR
- 在数据库之间 DB2 设置 HADR 复制的分步过程
- Db2 HADR Wiki

其他信息

使用这种模式，您可为在 Db2 数据库上运行的 SAP 系统设置灾难恢复系统。在灾难情况下，业务应能够继续执行您定义的恢复时间目标（RTO）和恢复点目标（RPO）要求：

- RTO 是指服务中断和服务恢复之间可接受的最大延迟。这决定了当服务不可用时，什么时间段被视为可接受的时间窗口。
- RPO 是指自上一个数据恢复点以来可接受的最长时间。这决定了从上一个恢复点到服务中断之间可接受的数据丢失情况。

有关 HADR 的 FAQs 相关信息，请参阅 SAP 注释 #1612105- DB6：Db2 高可用性灾难恢复 (HADR) 常见问题解答。(您需要 SAP 门户凭证才能访问此笔记。)

使用 Terraform 为数据库迁移设置 CI/CD 管道

由 Rahul Sharad Gaikwad 博士 (AWS)、Aarti Rajput (AWS)、Ashish Bhatt (AWS)、Aniket Dekate (AWS)、Naveen Suthar (AWS)、Nadeem Rahaman (AWS)、Ruchika Modi (AWS) 和 Tamilselvan P (AWS) 创作

摘要

这种模式旨在建立持续集成和持续部署 (CI/CD) 管道，以便以可靠和自动化的方式管理数据库迁移。它涵盖了使用 Terraform（一种基础设施即代码 (IaC) 工具）配置必要的基础架构、迁移数据和自定义架构更改的过程。

具体而言，该模式设置了一个 CI/CD 管道，用于将本地 Microsoft SQL Server 数据库迁移到亚马逊关系数据库服务 (Amazon RDS)。 AWS您也可以使用这种模式将位于虚拟机 (VM) 或其他云环境中的 SQL Server 数据库迁移到 Amazon RDS。

这种模式解决了与数据库管理和部署相关的以下难题：

- 手动部署数据库既耗时又容易出错，而且各环境之间缺乏一致性。
- 协调基础架构配置、数据迁移和架构更改可能既复杂又难以管理。
- 确保数据完整性并最大限度地减少数据库更新期间的停机时间对于生产系统至关重要。

这种模式具有以下优点：

- 通过实施用于数据库迁移的 CI/CD 管道，简化更新和部署数据库更改的过程。这降低了出错的风险，确保了跨环境的一致性，并最大限度地减少了停机时间。
- 有助于提高可靠性、效率和协作。缩短上市时间，减少数据库更新期间的停机时间。
- 帮助您采用现代数据库管理 DevOps 实践，从而提高软件交付流程的敏捷性、可靠性和效率。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 本地计算机上安装了 Terraform 0.12 或更高版本（有关说明，请参阅 Terraform 文档）
- Terraform P AWS rovider 版本 3.0.0 或更高版本 HashiCorp （请参阅此提供程序的GitHub 存储库）
- 最低权限 AWS Identity and Access Management (IAM) 策略（请参阅博客文章《编写最低权限 IAM 策略的技巧》）

架构

此模式实现了以下架构，该架构为数据库迁移过程提供了完整的基础架构。

在此架构中：

- 源数据库是位于本地、虚拟机 (VM) 上或由其他云提供商托管的 SQL Server 数据库。该图假设源数据库位于本地数据中心。
- 本地数据中心和 AWS 通过 VPN 或 AWS Direct Connect 连接进行连接。这提供了源数据库和 AWS 基础架构之间的安全通信。
- 目标数据库是 Amazon RDS 数据库，它借助数据库配置管道托管在虚拟私 AWS 有云 (VPC) 内。
- AWS Database Migration Service (AWS DMS) 将您的本地数据库复制到。 AWS它用于配置源数据库到目标数据库的复制。

下图显示了使用不同级别的数据库迁移过程设置的基础架构，其中包括配置、 AWS DMS 设置和验证。

在此过程中：

- 验证管道会验证所有检查。当所有必要的验证完成后，集成管道将进入下一步。
- 数据库配置管道由各个 AWS CodeBuild 阶段组成，这些阶段对为数据库提供的 Terraform 代码执行 Terraform 操作。完成这些步骤后，它将在目标 AWS 账户中部署资源。
- 该 AWS DMS 管道由多个 CodeBuild 阶段组成，这些阶段执行测试，然后使用 IaC 配置用于执行迁移 AWS DMS 的基础架构。

工具

AWS 服务 和工具

- AWS CodeBuild是一项完全托管的持续集成服务，用于编译源代码、运行测试和生成 ready-to-deploy软件包。
- AWS CodePipeline是一项完全托管的持续交付服务，可帮助您实现发布管道的自动化，从而实现快速可靠的应用程序和基础架构更新。

- Amazon Relational Database Service（Amazon RDS）可帮助您在中设置、操作和扩展关系数据库 AWS Cloud。
- Amazon Simple Storage Service (Amazon S3) 是一种对象存储服务，可提供出色的可扩展性、数据可用性、安全性和性能。
- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到云和本地设置的组合中，AWS Cloud 或者迁移到云端和本地设置的组合之间。

其他服务

- Terraform 是一款 IaC 工具 HashiCorp ，可帮助您创建和管理云和本地资源。

代码存储库

此模式的代码可在使用 Terraform 示例 GitHub 存储库的数据库迁移 DevOps 框架中找到。

最佳实践

- 对数据库迁移实施自动测试，以验证架构更改的正确性和数据完整性。这包括单元测试、集成测试和 end-to-end测试。
- 为您的数据库实施强大的备份和还原策略，尤其是在迁移之前。这样可以确保数据完整性，并在出现故障时提供备用选项。
- 实施强大的回滚策略，以便在迁移过程中出现故障或问题时还原数据库更改。这可能涉及回滚到以前的数据库状态或还原单个迁移脚本。
- 设置监控和记录机制以跟踪数据库迁移的进度和状态。这可以帮助您快速识别和解决问题。

操作说明

设置本地工作站

Task	描述	所需技能
在本地工作站上设置和配置 Git。	按照 Git 文档中的说明在本地工作站上安装和配置 Git。	DevOps 工程师
创建项目文件夹并添加 GitHub 存储库中的文件。	1. 打开此模式的GitHub 存储库。	DevOps 工程师

Task	描述	所需技能

	2. 选择 “代码” 以查看克隆选项，然后复制 HTTPS 选项卡中提供的 URL。
	3. 在您的工作站上为您的项目创建一个文件夹。
	4. 打开终端并导航到此文件夹。
	5. 克隆 GitHub 存储库：
	`git clone <github-repository-url>`
	您在步骤 2 中复制的 URL 在哪里`<github-repository-url>` 。
	6. 克隆完成后，转到项目文件夹中的克隆存储库：
	`cd <folder-name>/aws-terraform-db-migration-framework-samples`
	7. 在您选择的集成开发环境 (IDE) 中打开此项目。	

配置目标架构

Task	描述	所需技能
更新必需的参数。	该`ssm-parameters.sh` 文件存储所有必需的 AWS Systems Manager 参数。您可以使用项目的自定义值来配置这些参数。	DevOps 工程师

Task	描述	所需技能
	在本地工作站的`setup/db-ssm-params` 文件夹中，打开`ssm-parameters.sh` 文件并设置这些参数，然后再运行 CI/CD 管道。	
初始化 Terraform 配置。	在该`db-cicd-integration` 文件夹中，输入以下命令来初始化包含 Terraform 配置文件的工作目录： `terraform init`	DevOps 工程师
预览 Terraform 计划。	要创建 Terraform 计划，请输入以下命令： `terraform plan -var-file="terraform.sample"` Terraform 会评估配置文件，以确定已声明资源的目标状态。然后，其将比较目标状态与当前状态，并创建计划。	DevOps 工程师
验证计划。	查看计划并确认它已在您的目标 AWS 账户中配置了所需的架构。	DevOps 工程师

Task	描述	所需技能
部署解决方案。	1. 输入以下命令以应用计划： `terraform apply -var-file="terraform.sample"` 2. 输入 `yes`以确认。Terraform 创建、更新或销毁基础设施，以实现配置文件中声明的目标状态。有关序列的更多信息，请参阅此模式的“架构”部分。	DevOps 工程师

验证部署

Task	描述	所需技能
验证部署。	验证`db-cicd-integration` 管道的状态以确认数据库迁移已完成。 1. 登录 AWS Management Console，然后打开AWS CodePipeline 控制台。 2. 在导航窗格中，选择管道。 3. 选择`db-cicd-integration` 管道。 4. 验证管道执行是否已成功完成。	DevOps 工程师

使用后清理基础架构

Task	描述	所需技能
清理基础架构。	1. 项目完成后，使用以下命令清理您创建的基础架构： `terraform destroy --var-file=terrafo rm.sample` 2. 输入 `yes`以确认。	DevOps 工程师

相关资源

AWS 文档

- 开始使用 Terraform 产品

Terraform 文档

- Terraform installation（Terraform 安装）
- Terraform 后端配置
- Terraform AWS 提供者文档

使用有效备用数据库为 Amazon RDS Custom 上的 Oracle 电子商务套件设置 HA/DR 架构

由西蒙·坎宁安 (AWS) 和 Nitin Saxena 编写

摘要

此模式描述了如何在 Amazon Relational Database Service (Amazon RDS) 定制上架构 Oracle 电子商务解决方案，以实现高可用性 (HA) 和灾难恢复 (DR)，方法是在另一个 Amazon Web Services 可用区中设置 Amazon RDS Custom 只读副本数据库,并将其转换为活动备用数据库。Amazon RDS Custom 只读副本的创建，是通过 AWS 管理控制台完全自动化的。

此模式不讨论添加额外应用程序层和共享文件系统的步骤，这些也可以是 HA/DR 架构的一部分。有关这些主题的更多信息，请参阅以下 Oracle Support 注意事项：1375769.1、1375670.1 和 1383621.1（第 5 节，高级克隆选项）。（访问需要有 Oracle Support 账户。）

要将电子商务套件系统迁移到 Amazon Web Services (AWS) 上的单层单可用区架构，请参阅 将 Oracle 电子商务套件迁移到 Amazon RDS Custom 的说明。

Oracle 电子商务套件是一种企业资源规划（ERP）解决方案，用于自动化企业范围内的流程，例如财务、人力资源、供应链和制造。它具有三层架构：客户端、应用程序和数据库。以前，您必须在自行管理的亚马逊弹性计算云 (Amazon EC2) 实例上运行电子商务套件数据库，但现在您可以从 Amaz on RDS 定制版中受益。

先决条件和限制

先决条件

- Amazon RDS Custom 上安装的现有电子商务套件；参见模式将 Oracle 电子商务套件迁移至 Amazon RDS Custom
- 如果您想将只读副本更改为只读副本并使用它来卸载向备用副本的报告，请获得 Oracle Active Data Guard 数据库许可证（请参阅 Oracle Technology 商业价目表）

限制

- Amazon RDS Custom 上的 Oracle 数据库的限制和不支持的配置自定义
- 适用于 Oracle 的 Amazon RDS Custom 只读副本的相关限制

产品版本

有关 Amazon RDS Custom 支持的 Oracle Database 版本和实例类型，请参阅 Amazon RDS Custom for Oracle 的要求和限制

架构

下图展示了 AWS 上 E-Business Suite 的代表性架构，其中包括主动/被动设置中的多个可用区和应用程序层。该数据库使用 Amazon RDS Custom 数据库实例和 Amazon RDS Custom 只读副本。只读副本使用 Active Data Guard 复制到另一可用区。您还可以使用只读副本卸载主数据库上的读取流量并用于报告目的。

有关更多信息，请参阅 Amazon RDS 文档中的使用适用于 Oracle 的 Amazon RDS Custom 只读副本。

默认情况下，Amazon RDS Custom 只读副本是在安装时创建的。但是，如果您想将一些只读工作负载卸到备用数据库以减轻主数据库的负载，则可以按照操作说明部分中的步骤手动将已装载副本的模式更改为只读。典型的用例是从备用数据库运行报告。更改为只读，需要活动备用数据库许可证。

当您在 AWS 创建只读副本时，系统会秘密使用 Oracle Data Guard 代理。 此配置是在最大性能模式下自动生成和设置的，如下所示：

```
DGMGRL> show configuration
Configuration - rds_dg
  Protection Mode: MaxPerformance
  Members:
  vis_a - Primary database
    vis_b - Physical standby database
Fast-Start Failover: DISABLED
Configuration Status:
SUCCESS   (status updated 58 seconds ago)
```

工具

Amazon Web Services

- 适用于 Oracle 的 Amazon RDS Custom 是一项托管式数据库服务，适用于需要访问底层操作系统和数据库环境的旧版、自定义和打包应用程序。它可以自动执行数据库管理任务和操作，并使您能够作为数据库管理员访问和自定义数据库环境和操作系统。

其他工具

- Oracle Data Guard 是一款可帮助您创建和管理 Oracle 备用数据库的工具。此模式采用 Oracle Data Guard 在 Amazon RDS Custom 上设置活动备用数据库。

操作说明

创建只读副本

Task	描述	所需技能
创建 Amazon RDS Custom 数据库实例的只读副本。	要创建只读副本，请按 Amazon RDS 文档中的说明进行操作，并使用您创建的 Amazon RDS Custom 数据库实例（请参阅先决条件部分）作为源数据库。 默认情况下，Amazon RDS Custom 只读副本创建为物理备用副本，并处于已装载状态。这样做是为了确保遵守 Oracle Active Data Guard 许可。请按以下步骤将只读副本转换为只读模式。	数据库管理员

将只读副本更改为只读活动备用数据库

Task	描述	所需技能
连接到 Amazon RDS Custom 只读副本。	使用以下命令将物理备用数据库转换为活动备用数据库。 ⚠ Important 这些命令需要 Oracle 活动备用许可证。要	数据库管理员

Task	描述	所需技能
	获得许可证，请联系 Oracle 代表。	

	<pre>$ sudo su - rdsdb
	-bash-4.2$ sql
	SQL> select process,s
	tatus,sequence# from v
	$managed_standby;
	
	PROCESS STATUS
	SEQUENCE#
	--------- ------------
	---------- 	
	ARCH CLOSING
	3956
	ARCH CONNECTED
	0
	ARCH CLOSING
	3955
	ARCH CLOSING
	3957
	RFS IDLE
	0
	RFS IDLE
	3958
	MRP0 APPLYING_LOG
	3958
	SQL> select name,
	database_role,
	open_mode from v
	$database;
	
	NAME DATABASE_
	ROLE OPEN_MODE
	--------- ---------
	------- ---------
	----------- 	
	VIS PHYSICAL
	STANDBY MOUNTED</pre>	

Task	描述	所需技能

	`SQL> alter database recover managed standby database cancel;`
	`Database altered.`
	`Open the standby database`
	`SQL> alter database open;`
	`Database altered.`
	`SQL> select name, database_role, open_mode from v$database;`
	
	`NAME DATABASE_ROLE OPEN_MODE`
	`--------- ---------- ----------`
	`VIS PHYSICAL STANDBY READ ONLY`	

使用实时日志应用启动介质恢复。	要启用实时日志应用功能，请使用以下命令。它们将备用数据库（只读副本）转换为活动备用数据库并进行验证，因此您可连接和运行只读查询。
	
	`SQL> alter database recover managed standby database using current logfile disconnect from session;`
	`Database altered`
		数据库管理员

Task	描述	所需技能

检查数据库状态。	要查看数据库的状态，请使用以下命令。
	
	`SQL> select name, database_role, open_mode from v$database;`
	`NAME DATABASE_ROLE OPEN_MODE`
	`--------- ---------- ----------`
	`VIS PHYSICAL STANDBY READ ONLY WITH APPLY`
		数据库管理员

Task	描述	所需技能

检查重做应用模式。	若要查看重做应用模式，请使用以下命令。
	
	`SQL> select process,status,sequence# from v$managed_standby;`
	`PROCESS STATUS SEQUENCE#`
	`--------- ------------ ----------`
	`ARCH CLOSING 3956`
	`ARCH CONNECTED 0`
	`ARCH CLOSING 3955`
	`ARCH CLOSING 3957`
	`RFS IDLE 0`
	`RFS IDLE 3958`
	`MRP0 APPLYING_LOG 3958`
	
	`SQL> select open_mode from v$database;`
	`OPEN_MODE`
	`--------------------`
	`READ ONLY WITH APPLY`
		数据库管理员

相关的资源

- 将 Oracle 电子商务套件迁移到 Amazon RDS Custom (AWS Prescriptive Guidance)
- 使用 Amazon RDS Custom（Amazon RDS 文档）
- 使用 Amazon RDS Custom for Oracle 只读副本（Amazon RDS 文档）
- Amazon RDS Custom for Oracle：数据库环境中的新控制功能（AWS 新闻博客）

- 在 AWS 上迁移 Oracle 电子商务套件（AWS 白皮书）
- AWS 上的 Oracle 电子商务套件架构（AWS 白皮书）

EC2 使用 GTID 在亚马逊 RDS for MySQL 和亚马逊上的 MySQL 之间设置数据复制

由 Rajesh Madiwale (AWS) 编写

摘要

此模式描述了如何使用 MySQL 原生全局交易标识符 (GTID) 复制，在亚马逊网络服务 (AWS) 云上设置适用于 MySQL 数据库实例的亚马逊关系数据库服务 (Amazon RDS) 和亚马逊弹性计算云 (Amazon EC2) 实例上的 MySQL 数据库之间的数据复制。

使用 GTIDs，当事务提交到源服务器并由副本应用时，可以识别和跟踪事务。在失效转移期间启动新副本时，不需要参考日志文件。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 已部署 Amazon Linux 实例

限制

- 此设置需要内部团队运行只读查询。
- 源 MySQL 版本必须相同。
- 复制设置在同一 AWS 区域 和虚拟私有云（VPC）中。

产品版本

- Amazon RDS 5.7.23 和更高的 Amazon RDS 版本，这些版本支持 GTID

架构

源技术堆栈

- Amazon RDS for MySQL

目标技术堆栈

- 亚马逊上的 MySQL EC2

目标架构

工具

Amazon Web Services

- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- Amazon Relational Database Service (Amazon RDS) for MySQL 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展 MySQL 关系数据库。

其他服务

- 全局事务标识符 (GTIDs) 是为已提交的 MySQL 事务生成的唯一标识符。
- mysqldump 是客户端实用程序，用于通过生成 SQL 语句来执行逻辑备份，运行这些语句可以重现源数据库对象定义和表数据。
- mysql 是 MySQL 的命令行客户端。

操作说明

创建与准备 Amazon RDS for MySQL 数据库实例

Task	描述	所需技能
创建 RDS for MySQL 实例。	若要创建 RDS for MySQL 实例，请使用下一个任务中介绍的参数值，按照 Amazon RDS 文档中的步骤进行操作。	数据库管理员、工程师 DevOps
在数据库参数组中启用 GTID 相关的设置。	在 Amazon RDS for MySQL 数据库参数组启用以下参数。	数据库管理员

Task	描述	所需技能
	将 `enforce_gtid_consistency` 设置为 on，将 `gtid-mode` 设置为 on。	
重启 Amazon RDS for MySQL 实例。	若要使参数更改生效，必须先重启系统。	数据库管理员

创建用户并授予其复制权限	要安装 MySQL，请使用以下命令。
	`CREATE USER 'repl'@'%' IDENTIFIED BY 'xxxx';`
	`GRANT REPLICATION SLAVE ON *.* TO 'repl'@'%' ;`
	`FLUSH PRIVILEGES;`
		数据库管理员

在亚马逊 EC2 实例上安装和准备 MySQL

Task	描述	所需技能

在 Amazon Linux 上安装 MySQL。	要安装 MySQL，请使用以下命令。
	`sudo yum update`
	`sudo wget https://dev.mysql.com/get/mysql57-community-release-el7-11.noarch.rpm`
	`sudo yum localinstall mysql57-community-release-el7-11.noarch.rpm`
		数据库管理员

Task	描述	所需技能
	`sudo yum install mysql-community-server` `sudo systemctl start mysqld`	
在 EC2 实例上登录 MySQL 并创建数据库。	数据库名称应与 Amazon RDS for MySQL 数据库名称相同。在以下示例中，数据库名为 `replication` 。 `create database replication;`	数据库管理员

编辑 MySQL 配置文件，然后重新启动此数据库。	通过添加以下参数来编辑位于 `my.conf` 中的文件 `/etc/`。
	`server-id=3`
	`gtid_mode=ON`
	`enforce_gtid_consistency=ON`
	`replicate-ignore-db=mysql`
	`binlog-format=ROW`
	`log_bin=mysql-bin`
	然后重新启动 `mysqld` 服务。
	`systemctl mysqld restart`
		数据库管理员

设置复制

Task	描述	所需技能
从 Amazon RDS for MySQL 数据库导出数据转储。	若要从 Amazon RDS for MySQL 导出转储，请使用以下命令。	数据库管理员

Task	描述	所需技能
	`mysqldump --single-transaction -h mydb.xxxxxxx.amazonaws.com -uadmin -p --databases replication > replication-db.sql`	
在亚马逊 EC2的 MySQL 数据库中恢复.sql 转储文件。	要将转储文件导入到 Amazon 上的 MySQL 数据库 EC2，请使用以下命令。 `mysql -D replication -uroot -p < replication-db.sql`	数据库管理员

将 Amazon 上的 MySQL 数据库配置 EC2 为副本。	要开始复制并检查复制状态，请登录 Amazon 上的 MySQL 数据库 EC2，然后使用以下命令。
	`CHANGE MASTER TO MASTER_HOST="mydb.xxxxxxx.amazonaws.com", MASTER_USER="repl", MASTER_PASSWORD="rep123", MASTER_PORT=3306, MASTER_AUTO_POSITION = 1;`
	`START SLAVE;`
	`SHOW SLAVE STATUS\G`
		数据库管理员

相关的资源

- 适用于 Linux 实例的 Amazon EC2 用户指南
- 使用 MySQL Yum Repository 在 Linux 上安装 MySQL

- 使用全局交易标识符复制
- 在 Amazon RDS for MySQL 中使用基于 GTID 的复制

在 Amazon RDS 上为 Oracle PeopleSoft 应用程序过渡角色适用于 Oracle 定制

创建者：sampath kathirvel (AWS)

摘要

要在亚马逊网络服务 (AWS) 上运行 Oracle PeopleSoft 企业资源规划 (ERP) 解决方案，您可以使用亚马逊关系数据库服务 (Amazon RDS) 或 Amazon RDS Custom for Oracle，后者支持需要访问底层操作系统 (OS) 和数据库环境的传统、定制和打包应用程序。有关规划迁移时需要考虑的关键因素，请参阅 AWS Prescriptive Guidance 中的 Oracle 数据库迁移策略。

此模式侧重于为在 Amazon RDS Custom 上运行的 PeopleSoft 应用程序数据库执行 Oracle Data Guard 切换或角色过渡的步骤，该数据库是带有只读副本数据库的主数据库。该模式包括配置快速启动失效转移 (FSFO) 的步骤。在此过程中，Oracle Data Guard 配置中的数据库将继续以其新角色运行。Oracle Data Guard 切换的典型用例包括灾难恢复 (DR) 演练、数据库的定期维护活动以及备用优先补丁应用滚动补丁。有关更多信息，请参阅博客文章缩短 Amazon RDS Custom 中的数据库补丁停机时间。

先决条件和限制

先决条件

- 使用只读副本模式完成 PeopleSoft 在 Amazon RDS 自定义上向 Oracle 添加 HA。

限制

- 适用于 Oracle 的 RDS Custom 的限制和不支持的配置
- 适用于 Oracle 的 Amazon RDS Custom 只读副本的相关限制

产品版本

- 有关 Amazon RDS Custom 支持的 Oracle 数据库版本，请参阅适用于 Oracle 的 RDS Custom。
- 有关 Amazon RDS Custom 支持的 Oracle 数据库实例类，请参阅适用于 Oracle 的 RDS Custom 支持的数据库实例类。

架构

技术堆栈

- 适用于 Oracle 的 Amazon RDS Custom

目标架构

下图显示了 Amazon RDS Custom 数据库实例和 Amazon RDS Custom 只读副本。Oracle Data Guard 在灾难恢复失效转移期间提供角色转换。

有关 PeopleSoft 在 AWS 上使用 Oracle 的代表性架构，请参阅在 AWS 上设置高可用 PeopleSoft 架构。

工具

Amazon Web Services

- 适用于 Oracle 的 Amazon RDS Custom 是一种托管式数据库服务，适用于需要访问底层操作系统和数据库环境的旧版、自定义和打包应用程序。
- AWS Secrets Manager 帮助您将代码中的硬编码凭证（包括密码）替换为对 Secrets Manager 的 API 调用，以便以编程方式检索密钥。在这种模式中，您可以从 Secrets Manager 中使用密钥名称 `do-not-delete-rds-custom-+<<RDS Resource ID>>+-dg` 检索 `RDS_DATAGUARD` 的数据库用户密码。

其他服务

- Oracle Data Guard 可帮助您创建、维护、管理和监控备用数据库。这种模式使用 Oracle Data Guard 最高性能来转换角色（Oracle Data Guard 切换）。

最佳实践

对于生产部署，我们建议在与主节点和只读副本节点分开的第三个可用区中启动观察器实例。

操作说明

启动角色转换

Task	描述	所需技能
暂停主数据库和副本数据库的自动化。	尽管 RDS Custom 自动化框架不会干扰角色转换过程，但在 Oracle Data Guard 切换期间暂停自动化是一种不错的做法。 要暂停和恢复 RDS Custom 数据库自动化，请按照暂停和恢复 RDS Custom 自动化中的说明进行操作。	云管理员、数据库管理员

检查 Oracle Data Guard 状态。	要检查 Oracle Data Guard 状态，请登录到主数据库。此模式包括用于使用多租户容器数据库（CDB）或非 CDB 实例的代码。
	
	非 CDB
	
	`-bash-4.2$ dgmgrl RDS_DATAGUARD@RDS_CUSTOM_ORCL_A`
	`DGMGRL for Linux: Release 19.0.0.0.0 - Production on Mon Nov 28 20:55:50 2022`
	`Version 19.10.0.0.0`
	`Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved.`
	`Welcome to DGMGRL, type "help" for informati on.`
	`Password:`
	`Connected to "ORCL_A"`
		数据库管理员

Task	描述	所需技能
	Connected as SYSDG. DGMGRL> show configura tion Configuration - rds_dg Protection Mode: MaxAvailability Members: orcl_a - Primary database orcl_d - Physical standby database Fast-Start Failover: Disabled Configuration Status: SUCCESS (status updated 59 seconds ago) DGMGRL> CDB CDB-bash-4.2$ dgmgrl C##RDS_DATAGUARD@R DS_CUSTOM_RDSCDB_A DGMGRL for Linux: Release 19.0.0.0.0 - Production on Wed Jan 18 06:13:07 2023 Version 19.16.0.0.0 Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved. Welcome to DGMGRL, type "help" for informati on. Password: Connected to "RDSCDB_A " Connected as SYSDG. DGMGRL> show configura tion	

Task	描述	所需技能

	`Configuration - rds_dg`
	`  Protection Mode:`
	` MaxAvailability`
	`  Members:`
	`  rdscdb_a - Primary`
	` database`
	`  rdscdb_b - Physical`
	` standby database`
	`Fast-Start Failover:`
	` Disabled`
	`Configuration Status:`
	`SUCCESS  (status`
	` updated 52 seconds ago)`
	`DGMGRL>`	

验证实例角色。	打开 AWS 管理控制台，然后导航到 Amazon RDS 控制台。在数据库的复制部分的连接和安全选项卡上，验证主数据库和副本数据库的实例角色。 主角色应与 Oracle Data Guard 主数据库匹配，副本角色应与 Oracle Data Guard 物理备用数据库相匹配。	云管理员、数据库管理员

Task	描述	所需技能

执行切换。	要执行切换，请从主节点连接至 DGMGRL。
	
	非 CDB
	
	`DGMGRL> switchover to orcl_d;`
	`Performing switchover NOW, please wait...`
	`Operation requires a connection to database "orcl_d"`
	`Connecting ...`
	`Connected to "ORCL_D"`
	`Connected as SYSDG.`
	`New primary database "orcl_d" is opening...`
	`Operation requires start up of instance "ORCL" on database "orcl_a"`
	`Starting instance "ORCL"...`
	`Connected to an idle instance.`
	`ORACLE instance started.`
	`Connected to "ORCL_A"`
	`Database mounted.`
	`Database opened.`
	`Connected to "ORCL_A"`
	`Switchover succeeded, new primary is "orcl_d"`
	`DGMGRL>`
	
	CDB
	
	`DGMGRL> switchover to rdscdb_b`
	`Performing switchover NOW, please wait...`
		数据库管理员

Task	描述	所需技能
	New primary database "rdscdb_b" is opening... Operation requires start up of instance "RDSCDB" on database "rdscdb_a" Starting instance "RDSCDB"... Connected to an idle instance. ORACLE instance started. Connected to "RDSCDB_A" Database mounted. Database opened. Connected to "RDSCDB_A" Switchover succeeded, new primary is "rdscdb_b"	

Task	描述	所需技能

验证 Oracle Data Guard 连接。	切换后，验证 Oracle Data Guard 从主节点连接到 DGMGRL。
	
	非 CDB
	
	`DGMGRL> show configura tion; Configuration - rds_dg Protection Mode: MaxAvailability Members: orcl_d - Primary database orcl_a - Physical standby database Fast-Start Failover: Disabled Configuration Status: SUCCESS (status updated 60 seconds ago) DGMGRL>`
	
	`DGMGRL> show configura tion lag; Configuration - rds_dg Protection Mode: MaxAvailability Members: orcl_d - Primary database orcl_a - Physical standby database Transport Lag: 0 seconds (computed 0 seconds ago) Apply Lag: 0 seconds (computed 0 seconds ago) Fast-Start Failover: Disabled`
		数据库管理员

Task	描述	所需技能
	Configuration Status: SUCCESS (status updated 44 seconds ago) DGMGRL>	
	CDB DGMGRL> show configura tion DGMGRL> show configura tion Configuration - rds_dg Protection Mode: MaxAvailability Members: rdscdb_b - Primary database rdscdb_a - Physical standby database Fast-Start Failover: Disabled Configuration Status: SUCCESS (status updated 52 seconds ago) DGMGRL> DGMGRL> show configura tion lag Configuration - rds_dg Protection Mode: MaxAvailability Members: rdscdb_b - Primary database rdscdb_a - Physical standby database Transport Lag: 0 seconds (computed 0 seconds ago)	

Task	描述	所需技能

	`Apply Lag: 0 seconds (computed 0 seconds ago)`
	`Fast-Start Failover: Disabled`
	`Configuration Status: SUCCESS (status updated 53 seconds ago)`
	`DGMGRL>`	

在 Amazon RDS 控制台上验证实例角色。	执行角色切换后，Amazon RDS 控制台在数据库下的连接和安全选项卡的复制部分下显示新角色。复制状态可能需要几分钟才能从空升级至正在复制。	数据库管理员

配置 FSFO

Task	描述	所需技能
重置切换。	将切换设置回主节点。	数据库管理员
安装并启动观察器。	观察器过程是 DGMGRL 客户端组件，通常运行在与主数据库和备用数据库不同的计算机上。观察器的 ORACLE HOME 安装可以采用 Oracle Client Administrator 安装，也可以安装 Oracle 数据库企业版或个人版。有关数据库版本的观察器安装的更多信息，请参阅安装和启动观察器。要为观察器过程配置高可用性，您可能想要执行以下操作：	数据库管理员

Task	描述	所需技能
	• 为运行观察者的EC2 实例启用 EC2 实例自动恢复。作为操作系统启动的一部分，您需要自动执行观察器启动过程。 • 在 EC2 实例中部署观察者并配置大小为一 (1) 的 Amazon A EC2 uto Scaling 组。如果 EC2 实例出现故障，自动伸缩组会自动启动另一个 EC2 实例。 对于 Oracle 12c 版本 2 及更高版本，您最多可以部署三个观察器。其中一个观察器是主观察器，其余的则是备用观察器。当主观察器失效时，其中一个备用观察器将扮演主角色。	

Task	描述	所需技能

从观察器主机连接到 DGMGRL。	观察器主机配有用于主数据库和备用数据库连接的 `tnsnames.ora` 条目。只要数据丢失在FastStartFailoverLagLimit配置范围内（以秒为单位），您就可以启用具有最高性能保护模式的 FSFO。但是，您必须使用最大可用性保护模式才能实现零数据丢失（RPO=0）。
	
	非 CDB
	
	`DGMGRL> show configuration;`
	`Configuration - rds_dg`
	`Protection Mode: MaxAvailability`
	`Members:`
	`orcl_a - Primary database`
	`orcl_d - Physical standby database`
	`Fast-Start Failover: Disabled`
	`Configuration Status:`
	`SUCCESS (status updated 58 seconds ago)`
	`DGMGRL> show configuration lag`
	`Configuration - rds_dg`
	`Protection Mode: MaxAvailability`
	`Members:`
	`orcl_a - Primary database`
	`orcl_d - Physical standby database`
		数据库管理员

Task	描述	所需技能
	Transport Lag: 0 seconds (computed 1 second ago) Apply Lag: 0 seconds (computed 1 second ago) Fast-Start Failover: Disabled Configuration Status: SUCCESS (status updated 5 seconds ago) DGMGRL>	
	CDB -bash-4.2$ dgmgrl C##RDS_DATAGUARD@RDS_CUSTOM_RDSCDB_A DGMGRL for Linux: Release 19.0.0.0.0 - Production on Wed Jan 18 06:55:09 2023 Version 19.16.0.0.0 Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved. Welcome to DGMGRL, type "help" for information. Password: Connected to "RDSCDB_A" Connected as SYSDG. DGMGRL> show configuration Configuration - rds_dg Protection Mode: MaxAvailability Members: rdscdb_a - Primary database	

Task	描述	所需技能
	rdscdb_b - Physical standby database Fast-Start Failover: Disabled Configuration Status: SUCCESS (status updated 18 seconds ago) DGMGRL>	

Task	描述	所需技能

将备用数据库修改为失效转移目标。	从主节点或观察器节点连接到一个备用数据库。（尽管您可能配有多个备用数据库，但此时您只需要连接到一个备用数据库。）
	
	非 CDB
	
	`DGMGRL> edit database orcl_a set property FastStartFailoverTarget='orcl_d';`
	`Property "faststartfailovertarget" updated`
	`DGMGRL> edit database orcl_d set property FastStartFailoverTarget='orcl_a';`
	`Property "faststartfailovertarget" updated`
	`DGMGRL> show database orcl_a FastStartFailoverTarget;`
	`FastStartFailoverTarget = 'orcl_d'`
	`DGMGRL> show database orcl_d FastStartFailoverTarget;`
	`FastStartFailoverTarget = 'orcl_a'`
	`DGMGRL>`
	
	CDB
	
	`DGMGRL> edit database orcl_a set property`
		数据库管理员

Task	描述	所需技能
	FastStartFailoverT arget='rdscdb_b'; Object "orcl_a" was not found DGMGRL> edit database rdscdb_a set property FastStartFailoverT arget='rdscdb_b'; Property "faststar tfailovertarget" updated DGMGRL> edit database rdscdb_b set property FastStartFailoverT arget='rdscdb_a'; Property "faststar tfailovertarget" updated DGMGRL> show database rdscdb_a FastStart FailoverTarget; FastStartFailoverT arget = 'rdscdb_b' DGMGRL> show database rdscdb_b FastStart FailoverTarget; FastStartFailoverT arget = 'rdscdb_a' DGMGRL>	

Task	描述	所需技能

配置 FastStartFailoverThreshold 与 DGMGRL 的连接。	在 Oracle 19c 中，默认值为 30 秒，最小值为 6 秒。较低的值可能会缩短失效转移期间的恢复时间目标（RTO）。较高的值有助于降低主数据库上出现不必要的失效转移瞬时错误的可能性。
	
	适用于 Oracle 的 RDS Custom 自动化框架监控数据库运行状况并每隔几秒钟执行一次纠正操作。因此，我们建议将值设置 FastStartFailoverThreshold 为大于 10 秒的值。以下示例将阈值配置为 35 秒。
	
	非 CBD 或 CDB
	
	`DGMGRL> edit configuration set property FastStartFailoverThreshold=35;`
	`Property "faststartfailoverthreshold" updated`
	`DGMGRL> show configuration FastStartFailoverThreshold;`
	`FastStartFailoverThreshold = '35'`
	`DGMGRL>`
		数据库管理员

Task	描述	所需技能

通过从主节点或观察器节点连接到 DGMGRL 来启用 FSFO。	如果数据库未启用闪回数据库，则会显示警告消息 ORA-16827 。如果FastStartFailoverAutoReinstate配置属性设置为TRUE（这是默认值），则可选的闪回数据库有助于自动将出现故障的主数据库恢复到故障转移之前的时间点。
	
	非 CDB
	
	`DGMGRL> enable fast_start failover;`
	`Warning: ORA-16827: Flashback Database is disabled`
	`Enabled in Zero Data Loss Mode.`
	`DGMGRL>`
	`DGMGRL> show configuration`
	`Configuration - rds_dg`
	`Protection Mode: MaxAvailability`
	`Members:`
	`orcl_a - Primary database`
	`Warning: ORA-16819: fast-start failover observer not started`
	`orcl_d - (*) Physical standby database`
	`Warning: ORA-16819: fast-start failover observer not started`
	`Fast-Start Failover: Enabled in Zero Data Loss Mode`
		数据库管理员

Task	描述	所需技能
	Configuration Status: WARNING (status updated 29 seconds ago) DGMGRL> CDB DGMGRL> enable fast_star t failover; Warning: ORA-16827: Flashback Database is disabled Enabled in Zero Data Loss Mode. DGMGRL> show configura tion; Configuration - rds_dg Protection Mode: MaxAvailability Members: rdscdb_a - Primary database Warning: ORA-16819 : fast-start failover observer not started rdscdb_b - (*) Physical standby database Fast-Start Failover: Enabled in Zero Data Loss Mode Configuration Status: WARNING (status updated 11 seconds ago) DGMGRL>	

Task	描述	所需技能

启动观察器进行 FSFO 监控，然后验证状态。	您可以在启用 FSFO 之前或之后启动观察器。如果 FSFO 已启用，则观察器会立即开始监控状态以及与主备用数据库和目标备用数据库的连接。如果未启用 FSFO，则观察器要等到 FSFO 启用后才会开始监控。
	
	启动观察器时，将显示主数据库配置而不显示任何错误消息，前述 `show configuration` 命令就是明证。
	
	非 CDB
	
	`DGMGRL> start observer;`
	`[W000 2022-12-01T06:16:51.271+00:00] FSFO target standby is orcl_d`
	`Observer 'ip-10-0-1-89' started`
	`[W000 2022-12-01T06:16:51.352+00:00] Observer trace level is set to USER`
	
	`DGMGRL> show configuration`
	`Configuration - rds_dg`
	`Protection Mode: MaxAvailability`
	`Members:`
	`orcl_a - Primary database`
	`orcl_d - (*) Physical standby database`
		数据库管理员

Task	描述	所需技能
	Fast-Start Failover: Enabled in Zero Data Loss Mode Configuration Status: SUCCESS (status updated 56 seconds ago) DGMGRL> DGMGRL> show observer Configuration - rds_dg Primary: orcl_a Active Target: orcl_d Observer "ip-10-0- 1-89" - Master Host Name: ip-10-0-1 -89 Last Ping to Primary: 1 second ago Last Ping to Target: 1 second ago DGMGRL> CDB DGMGRL> start observer; Succeeded in opening the observer file "/home/oracle/fsfo _ip-10-0-1-56.dat". [W000 2023-01-1 8T07:31:32.589+00:00] FSFO target standby is rdscdb_b Observer 'ip-10-0- 1-56' started The observer log file is '/home/oracle/obse rver_ip-10-0-1-56. log'.	

Task	描述	所需技能

	DGMGRL> show configura
	tion
	Configuration - rds_dg
	  Protection Mode:
	 MaxAvailability
	  Members:
	  rdscdb_a - Primary
	 database
	  rdscdb_b - (*)
	 Physical standby
	 database
	Fast-Start Failover:
	 Enabled in Zero Data
	 Loss Mode
	Configuration Status:
	SUCCESS (status
	 updated 12 seconds ago)
	DGMGRL>
	
	DGMGRL> show observer;
	Configuration - rds_dg
	  Primary:
	 rdscdb_a
	  Active Target:
	 rdscdb_b
	Observer "ip-10-0-
	1-56" - Master
	  Host Name:
	  ip-10-0-1-56
	  Last Ping to Primary:
	  1 second ago
	  Last Ping to Target:
	  2 seconds ago
	DGMGRL>	

Task	描述	所需技能
验证失效转移。	在这种情况下，可以通过手动停止主 EC2 实例来执行故障转移测试。在停止 EC2 实例之前，请使用`tail`命令根据您的配置监控观察者日志文件。以用户 RDS_DATAGUARD 的身份使用 DGMGRL 登录备用数据库 `orcl_d`，并检查 Oracle Data Guard 状态。此时应显示 `orcl_d` 为新的主数据库。 **Note** 在此故障转移测试场景中，`orcl_d`是非 CDB 数据库。 在失效转移之前，已在 `orcl_a` 上启用闪回数据库。在先前的主数据库恢复在线状态并以 MOUNT 状态启动后，观察器将其恢复到新的备用数据库中。恢复后的数据库充当新主数据库的 FSFO 目标。您可以在观察器日志中验证详细信息。 `DGMGRL> show configura tion Configuration - rds_dg Protection Mode: MaxAvailability Members: orcl_d - Primary database`	数据库管理员

Task	描述	所需技能

	<pre>Warning: ORA-16824
	: multiple warnings,
	 including fast-star
	t failover-related
	 warnings, detected for
	 the database
	orcl_a - (*) Physical
	 standby database
	 (disabled)
	ORA-16661: the standby
	 database needs to be
	 reinstated
	Fast-Start Failover:
	 Enabled in Zero Data
	 Loss Mode
	Configuration Status:
	WARNING (status updated
	 25 seconds ago)
	DGMGRL></pre>
	以下显示了 observer.log 中的示例输出。
	<pre>$ tail -f /tmp/obse
	rver.log
	Unable to connect
	 to database using
	 rds_custom_orcl_a
	[W000 2023-01-1
	8T07:50:32.589+00:00]
	 Primary database cannot
	 be reached.
	[W000 2023-01-1
	8T07:50:32.589+00:00]
	 Fast-Start Failover
	 threshold has expired.
	[W000 2023-01-1
	8T07:50:32.590+00:00]
	 Try to connect to the
	 standby.</pre>	

Task	描述	所需技能
	[W000 2023-01-18T07:50:32.590+00:00] Making a last connection attempt to primary database before proceeding with Fast-Start Failover. [W000 2023-01-18T07:50:32.591+00:00] Check if the standby is ready for failover. [S002 2023-01-18T07:50:32.591+00:00] Fast-Start Failover started... 2023-01-18T07:50:32.591+00:00 Initiating Fast-Start Failover to database "orcl_d"... [S002 2023-01-18T07:50:32.592+00:00] Initiating Fast-start Failover. Performing failover NOW, please wait... Failover succeeded, new primary is "orcl_d" 2023-01-18T07:55:32.101+00:00 [S002 2023-01-18T07:55:32.591+00:00] Fast-Start Failover finished... [W000 2023-01-18T07:55:32.591+00:00] Failover succeeded. Restart pinging. [W000 2023-01-18T07:55:32.603+00:00] Primary database has changed to orcl_d.	

Task	描述	所需技能
	[W000 2023-01-18T07:55:33.618+00:00] Try to connect to the primary. [W000 2023-01-18T07:55:33.622+00:00] Try to connect to the primary rds_custom_orcl_d. [W000 2023-01-18T07:55:33.634+00:00] The standby orcl_a needs to be reinstated [W000 2023-01-18T07:55:33.654+00:00] Try to connect to the new standby orcl_a. [W000 2023-01-18T07:55:33.654+00:00] Connection to the primary restored! [W000 2023-01-18T07:55:35.654+00:00] Disconnecting from database rds_custom_orcl_d. [W000 2023-01-18T07:55:57.701+00:00] Try to connect to the new standby orcl_a. ORA-12170: TNS:Connect timeout occurred	

配置 Oracle Peoplesoft 应用程序和数据库之间的连接

Task	描述	所需技能
在主数据库中创建并启动该服务。	在配置中有一条 TNS 条目同时包含主数据库端点和备用数	数据库管理员

Task	描述	所需技能
	据库端点，使用此条目可以避免在角色转换期间更改应用程序配置。您可以定义两个基于角色的数据库服务来支持读/写和只读工作负载。在以下示例中，orcl_rw 是主数据库上处于有效状态的读/写服务。orcl_ro 是只读服务，在以只读模式打开的备用数据库上处于有效状态。 `SQL> select name,open _mode from v$database; NAME OPEN_MODE --------- --------- ----------- ORCL READ WRITE SQL> exec dbms_serv ice.create_service ('orcl_rw','orcl_r w'); PL/SQL procedure successfully completed . SQL> exec dbms_serv ice.create_service ('orcl_ro','orcl_r o'); PL/SQL procedure successfully completed .`  `SQL> exec dbms_serv ice.start_service('orcl_rw'); PL/SQL procedure successfully completed . SQL>`	

Task	描述	所需技能
在备用数据库中启动服务。	要在只读备用数据库中启动该服务，请使用以下代码。 `SQL> select name,open _mode from v$database; NAME OPEN_MODE --------- --------- ----------- ORCL READ ONLY WITH  APPLY SQL> exec dbms_serv ice.start_service( 'orcl_ro'); PL/SQL procedure  successfully completed . SQL>`	数据库管理员

Task	描述	所需技能

在主数据库重新启动时自动启动服务。	要在主数据库重新启动时自动启动该服务，请使用以下代码。
	
	SQL> CREATE OR REPLACE
	TRIGGER TrgDgServ
	ices after startup on
	database
	DECLARE
	db_role VARCHAR(30);
	db_open_mode VARCHAR(3
	0);
	BEGIN
	SELECT DATABASE_ROLE,
	OPEN_MODE INTO db_role,
	db_open_mode FROM V
	$DATABASE;
	IF db_role = 'PRIMARY'
	THEN
	DBMS_SERV 2 ICE.START
	_SERVICE('orcl_rw');
	END IF;
	IF db_role = 'PHYSICAL
	STANDBY' AND db_open_m
	ode LIKE 'READ ONLY%'
	THEN
	DBMS_SERVICE.START_SER
	VICE('orcl_ro');
	END IF;
	END;
	/
	Trigger created.
	SQL>
		数据库管理员

Task	描述	所需技能
配置读/写数据库和只读数据库之间的连接。	您可以将以下应用程序配置示例用于读/写和只读连接。 `ORCL_RW = (DESCRIPTION = (CONNECT_TIMEOUT= 120)(RETRY_COUNT=20)(RETRY_DELAY=3)(TRANSPORT_CONNECT_ TIMEOUT=3) (ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST=devpsftdb.******.us-west-2 .rds.amazonaws.com) (PORT=1521)) (ADDRESS = (PROTOCOL = TCP)(HOST=psftread .******.us-west-2. rds.amazonaws.com) (PORT=1521))) (CONNECT_DATA=(SERVIC E_NAME = orcl_rw))) ORCL_RO = (DESCRIPTION = (CONNECT_TIMEOUT= 120)(RETRY_COUNT=20)(RETRY_DELAY=3)(TRANSPORT_CONNECT_ TIMEOUT=3) (ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST=devpsftdb.******.us-west-2 .rds.amazonaws.com) (PORT=1521)) (ADDRESS = (PROTOCOL = TCP)(HOST=psftread .******.us-west-2.`	数据库管理员

Task	描述	所需技能
	rds.amazonaws.com) (PORT=1521))) (CONNECT_DATA=(SERVIC E_NAME = orcl_ro)))	

相关的资源

- 在适用于 Oracle 的 Amazon RDS Custom 上使用 Data Guard 启用高可用性（AWS 技术指南）
- 将 Amazon RDS 配置为 Oracle PeopleSoft 数据库（AWS 白皮书）
- Oracle Data Guard 代理指南（Oracle 参考文档）
- Data Guard 概念和管理（Oracle 参考文档）
- Oracle Data Guard 特定的 FAN 和 FCF 配置要求（Oracle 参考文档）

跨账户将数据从 Amazon Redshift 集群卸载到亚马逊 S3

由安德鲁·卡梅尔 (AWS) 创作

摘要

在测试应用程序时，将生产数据放在测试环境中会很有帮助。使用生产数据可以更准确地评估正在开发的应用程序。

这种模式将数据从生产环境中的 Amazon Redshift 集群提取到亚马逊网络服务开发环境中的亚马逊简单存储服务 (Amazon S3) 存储桶 ()。AWS

该模式会逐步设置 DEV 和 PROD 帐户，包括以下内容：

- 所需的 资源
- AWS Identity and Access Management (IAM) 角色
- 调整子网、安全组和虚拟私有云 (VPC) 以支持 Amazon Redshift 连接
- 一个带有 Python 运行时的示例 AWS Lambda 函数，用于测试架构

要授予对 Amazon Redshift 集群的访问权限，该模式 AWS Secrets Manager 用于存储相关证书。好处是拥有直接连接到 Amazon Redshift 集群所需的所有信息，而无需知道 Amazon Redshift 集群所在的位置。此外，您还可以监控密钥的使用情况。

保存在 Secrets Manager 中的密钥包括 Amazon Redshift 集群的主机、数据库名称、端口和相关凭证。

有关使用此模式时的安全注意事项的信息，请参阅 “最佳实践” 部分。

先决条件和限制

先决条件

- 在 PROD 账户中运行的 Amazon Redshift 集群
- 在 DEV 账户中创建的 S3 存储桶
- D@@ EV 账户和 PROD 账户之间的 VPC 对等，并相应调整路由表
- 为两个对等设备@@ 启用了 DNS 主机名和 DNS 解析 VPCs

限制

- 根据您要查询的数据量，Lambda 函数可能会超时。

  如果您的运行时间超过了 Lambda 的最大超时时间（15 分钟），请对您的 Lambda 代码使用异步方法。此模式的代码示例使用适用于 Python 的 psycopg2 库，该库目前不支持异步处理。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额页面，然后选择服务的链接。

架构

下图显示了目标架构，包括 DEV 和 PROD 帐户。

图表显示了以下工作流：

1. 开发者账户中的 Lambda 函数扮演访问生产账户中 Secrets Manager 中的 Amazon Redshift 证书所需的 IAM 角色。

   然后，Lambda 函数检索亚马逊 Redshift 集群密钥。
2. 开发者账户中的 Lambda 函数使用这些信息通过对等连接到 PROD 账户中的 Amazon Redshift 集群。 VPCs

   然后，Lambda 函数发送卸载命令来查询 PROD 账户中的 Amazon Redshift 集群。
3. PROD 账户中的 Amazon Redshift 集群扮演相关的 IAM 角色来访问开发者账户中的 S3 存储桶。

   Amazon Redshift 集群将查询的数据卸载到开发者账户中的 S3 存储桶。

查询来自亚马逊 Redshift 的数据

下图显示了用于检索亚马逊 Redshift 凭证和连接到亚马逊 Redshift 集群的角色。该工作流程由 Lambda 函数启动。

图表显示了以下工作流：

1. 开发者账户`CrossAccount-SM-Read-Role`中的假设是 PROD 账户`SM-Read-Role`中的。
2. 该`SM-Read-Role`角色使用附加的策略从 Secrets Manager 中检索密钥。

3. 这些证书用于访问亚马逊 Redshift 集群。

将数据上传到亚马逊 S3

下图显示了提取数据并将其上传到 Amazon S3 的跨账户读写过程。该工作流程由 Lambda 函数启动。这些模式将 A mazon Redshift 中的 IAM 角色链起来。来自 Amazon Redshift 集群的卸载命令假定为`CrossAccount-S3-Write-Role`，然后假定为。`S3-Write-Role`这种角色链允许亚马逊 Redshift 访问亚马逊 S3。

该工作流程包括以下步骤：

1. 开发者账户`CrossAccount-SM-Read-Role`中的假设是 PROD 账户`SM-Read-Role`中的。
2. 从 Secrets Manager 中`SM-Read-Role`检索亚马逊 Redshift 凭证。
3. Lambda 函数连接到 Amazon Redshift 集群并发送查询。
4. Amazon Redshift 集群假设. `CrossAccount-S3-Write-Role`
5. `CrossAccount-S3-Write-Role`假设`S3-Write-Role`在 DEV 账户中。
6. 查询结果将卸载到 DEV 账户中的 S3 存储桶。

工具

AWS 服务

- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥以帮助保护您的数据。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Redshift 是一项在 Amazon Web Services Cloud 中托管的 PB 级数据仓库服务。
- AWS Secrets Manager 有助于您通过对 Secrets Manager 的 API 调用来替换代码中的硬编码凭证（包括密码），以编程方式检索密钥。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

代码存储库

此模式的代码可在 GitHub unload-redshift-to-s3-python 存储库中找到。

最佳实践

安全免责声明

在实施此解决方案之前，请考虑以下重要的安全建议：

- 请记住，连接开发和生产帐户可能会扩大范围，降低整体安全状况。我们建议仅临时部署此解决方案，提取所需的数据部分，然后立即销毁已部署的资源。要销毁资源，您应删除 Lambda 函数，删除为此解决方案创建的任何 IAM 角色和策略，并撤消在账户之间授予的任何网络访问权限。
- 在将任何数据从生产环境复制到开发环境之前，请咨询您的安全和合规团队。通常不应以这种方式复制个人身份信息 (PII)、受保护的健康信息 (PHI) 以及其他机密或受监管的数据。仅复制公开可用的非机密信息（例如，商店前端的公开库存数据）。尽可能考虑对数据进行标记化或匿名化处理，或者生成合成测试数据，而不是使用生产数据。AWS 安全原则之一是让人们远离数据。换句话说，开发者不应在生产账户中执行操作。
- 限制对开发账户中的 Lambda 函数的访问，因为它可以在生产环境中从 Amazon Redshift 集群中读取数据。
- 为避免中断生产环境，请实施以下建议：
  - 使用单独的专用开发帐户进行测试和开发活动。
  - 实施严格的网络访问控制，将账户之间的流量限制在必要的范围内。
  - 监控和审核对生产环境和数据源的访问权限。
  - 对所有涉及的资源和服务实施最低权限访问控制。
  - 定期审查和轮换证书，例如 AWS Secrets Manager 密钥和 IAM 角色访问密钥。
- 有关本文中使用的服务，请参阅以下安全文档：
  - AWS Lambda 安全
  - 亚马逊 Redshift 安全
  - 亚马逊 S3 安全
  - AWS Secrets Manager 安全
  - IAM 安全最佳实践

访问生产数据和资源时，安全性是重中之重。始终遵循最佳实践，实施最低权限访问控制，并定期审查和更新您的安全措施。

操作说明

查询来自亚马逊 Redshift 的数据

Task	描述	所需技能
为亚马逊 Redshift 集群创建密钥。	要为 Amazon Redshift 集群创建密钥，请执行以下操作： 1. 在 PROD 账户中，登录并打开 Secrets Manager 控制台 AWS Management Console，网址为 https://console.aws.amazon.com/secretsmanager/。 2. 选择 “存储新密钥”。 3. 选择 Amazon Redshift 数据仓库的凭证。 4. 在用户名和密码中，输入您的实例的值，然后确认或选择加密密钥的值。 5. 选择您的密钥将访问的 Amazon Redshift 数据仓库。 6. 输入`Redshift-Creds-Secret` 机密名称。 7. 使用默认选项完成剩余的创建步骤，然后选择存储。 8. 查看您的密钥，并记下为识别密钥而生成的密钥 ARN 值。	DevOps 工程师
创建一个角色来访问 Secrets Manager。	要创建角色，请执行以下操作：	DevOps 工程师

Task	描述	所需技能
	1. 在 PROD 账户中，打开 IAM 控制台，网址为 https://console.aws.amazon.com/iam/。 2. 选择策略。 3. 选择创建策略。 4. 选择 JSON 选项卡，然后输入如下所示的 IAM 策略： `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "secretsmanager:GetResourcePolicy", "secretsmanager:GetSecretValue", "secretsmanager:DescribeSecret", "secretsmanager:ListSecretVersionIds" ], "Resource": [ "<Redshift-Creds-Secret-ARN>" ] }, {`	

Task	描述	所需技能
	<pre>"Effect": "Allow", "Action": "secretsmanager:ListSecrets", "Resource": "*" }] }</pre> Redshift-Creds-Secret-ARN 替换为 Secrets Manager 密钥的亚马逊资源名称 (ARN)，该密钥包含亚马逊 Redshift 集群的信息和证书。	

将数据上载到 Amazon S3

Task	描述	所需技能
创建用于访问 S3 存储桶的角色。	要创建用于访问 S3 存储桶的角色，请执行以下操作： 1. 在开发者账户中，打开 IAM 控制台。 2. 选择策略。 3. 选择创建策略。 4. 选择 JSON 选项卡，然后输入如下所示的 IAM 策略： <pre>{ "Version": "2012-10-17", "Statement": [</pre>	DevOps 工程师

Task	描述	所需技能

	<pre>{
	  "Sid": "kmsstmt"
	,
	  "Effect":
	 "Allow",
	  "Action": [
	  "kms:Decr
	ypt",
	  "kms:Encr
	ypt",
	  "kms:Gene
	rateDataKey"
	  ],
	  "Resource": [
	  "<kms-key-
	arn>"
	  ]
	 },
	 {
	  "Sid":
	 "s3stmt",
	  "Effect":
	 "Allow",
	  "Action": [
	  "s3:PutOb
	ject",
	  "s3:Get*",
	  "s3:List*"
	  ],
	  "Resource": [
	  "arn:aws:
	s3:::mybucket",
	  "arn:aws:
	s3:::mybucket/*"
	  ]
	 }
	]
	}</pre> mybucket替换为您要访问的 S3 存储桶的名称。此外	

Task	描述	所需技能
	，如果 S3 存储桶已加密，请`kms-key-arn` 替换为用于加密 S3 存储桶的 AWS Key Management Service (AWS KMS) 密钥的 ARN。否则，您就不需要政策中的 AWS KMS 章节了。 5. 选择查看策略，输入`S3-Write-Policy` 作为策略名称，然后选择创建策略。 6. 在导航窗格中，选择角色。 7. 选择 Create role（创建角色）。 8. 对于可信实体角色，请选择自定义信任策略。 9. 选择“下一步：权限”，然后选择您创建的 S3-Write-Policy 策略。 10.输入`S3-Write-Role` 角色名称，然后选择创建角色。	

Task	描述	所需技能
创建 Amazon Redshift 角色。	要创建 Amazon Redshift 角色，请执行以下操作： 1. 在 PROD 账户中，打开 IAM 控制台。 2. 选择策略。 3. 选择创建策略。 4. 选择 JSON 选项卡，然后输入如下所示的 IAM 策略： `{ "Version": "2012-10-17", "Statement": [ { "Sid": "CrossAccountPolicy", "Effect": "Allow", "Action": "sts:AssumeRole", "Resource": "S3-Write-Role-ARN" } ] }` 将 `S3-Write-Role-ARN` 替换为 DEV 账户 `S3-Write-Role` 中的 ARN。 5. 选择查看策略，输入 `S3-Write-Role-Assume-Policy` 作为策略名称，然后选择创建策略。	DevOps 工程师

Task	描述	所需技能
	6. 在导航窗格中，选择角色，然后选择创建角色。 7. 选择 AWS 服务作为您的可信实体类型，然后选择 Redshift、Redshift-可自定义。 8. 选择“下一步：权限”，然后选择您创建的 S3-Write-Role-Assume- Policy 策略。 9. 输入`CrossAccount-S3-Write-Role` 角色名称，然后选择创建角色。 10. 将 IAM 角色与您的亚马逊 Redshift 集群关联。	

部署 Lambda 函数

Task	描述	所需技能
部署 Lambda 函数。	要在对等互连 VPC 中部署 Lambda 函数，请执行以下操作： 1. 打开 Lambda 控制台，网址为。 https://console.aws.amazon.com/lambda/ 2. 选择 函数。 3. 选择 Create function（创建函数）。 4. 在基本信息下的函数名称中输入函数的名称。	DevOps 工程师

Task	描述	所需技能
	5. 对于运行时，选择 Python 3.8 。 6. 展开 “更改默认执行角色”，然后执行以下操作： a. 选择使用现有角色。 b. 对于现有角色，请选择您之前创建的 CrossAccount-rm-read-role Lambda 角色。 7. 展开 “高级设置”，然后执行以下操作： a. 选中 “启用 VPC” 复选框。 b. 对于 VPC，请在开发者账户中选择对等 VPC。 c. 对于子网，请选择私有子网。 d. 对于 Security groups (安全组)，选择默认安全组。 8. 选择 Create function (创建函数)。 9. 将 psycopg2 库作为一个层添加到 Lambda 函数中。 **Note** 你可以使用 psycopg2-lambda-layer 存储库中已经部署的层。请务必使用基于你 AWS 区域 和	

Task	描述	所需技能
	Python 运行时的网址。	

测试架构

Task	描述	所需技能

导入所需的资源。	要导入所需的资源，请运行以下命令：
	`import ast`
	`import boto3`
	`import psycopg2`
	`import base64`
	`from botocore.exceptions import ClientError`
		应用程序开发人员

运行 Lambda 处理程序函数。	Lambda 函数使用 AWS Security Token Service (AWS STS) 进行跨账户访问和临时凭证管理。该函数使用 AssumeRole API 操作来临时承担 `sm_read_role` IAM 角色的权限。
	要运行 Lambda 函数，请使用以下示例代码：
	`def lambda_handler(event, context):`
	`  sts_client = boto3.client('sts')`
	`  # Secrets Manager Configurations`
		应用程序开发人员

Task	描述	所需技能

	secret_name = "redshift_creds"
	sm_region = "eu-west-1"
	sm_read_role = "arn:aws:iam::PROD_ACCOUNT_NUMBER:role/SM-Read-Role"
	
	# S3 Bucket Configurations
	s3_bucket_path = "s3://mybucket/"
	s3_bucket_region = "eu-west-1"
	s3_write_role = "arn:aws:iam::DEV_ACCOUNT_NUMBER:role/S3-Write-Role"
	
	# Redshift Configurations
	sql_query = "select * from category"
	redshift_db = "dev"
	redshift_s3_write_role = "arn:aws:iam::PROD_ACCOUNT_NUMBER:role/CrossAccount-S3-Write-Role"
	
	chained_s3_write_role = "%s,%s" % (redshift_s3_write_role, s3_write_role)
	
	assumed_role_object = sts_client.assume_role(
	RoleArn=sm_read_role,	

Task	描述	所需技能

	RoleSessi
	onName="CrossAccou
	ntRoleAssumption",
	ExternalI
	d="YOUR_EXTERNAL_ID",
	)
	credentials =
	assumed_role_objec
	t['Credentials']
	
	secret_dict =
	ast.literal_eval(g
	et_secret(credenti
	als, secret_name,
	sm_region))
	execute_query(secr
	et_dict, sql_query
	, s3_bucket_path,
	chained_s3_write_r
	ole, s3_bucket_region,
	redshift_db)
	
	return {
	'statusCode':
	200
	}	

Task	描述	所需技能

明白秘密。	要获取 Amazon Redshift 密钥，请使用以下示例代码：
	
	`def get_secret(credentials, secret_name, sm_region):`
	`  # Create a Secrets Manager client`
	`  session = boto3.session.Session()`
	`  sm_client = session.client(`
	`  service_name='secretsmanager',`
	`  aws_access_key_id=credentials['AccessKeyId'],`
	`  aws_secret_access_key=credentials['SecretAccessKey'],`
	`  aws_session_token=credentials['SessionToken'],`
	`  region_name=sm_region`
	`  )`
	
	`  try:`
	`  get_secret_value_response = sm_client.get_secret_value(`
	`  SecretId=secret_name`
	`  )`
	`  except ClientError as e:`
	`  print(e)`
	`  raise e`
	`  else:`
		应用程序开发人员

Task	描述	所需技能

	<pre> if 'SecretSt
	ring' in get_secre
	t_value_response:
	            return
	 get_secret_value_r
	esponse['SecretStr
	ing']
	        else:
	            return
	 base64.b64decode(g
	et_secret_value_re
	sponse['SecretBina
	ry'])</pre>	

Task	描述	所需技能

运行卸载命令。	要将数据卸载到 S3 存储桶，请使用以下示例代码。
	
	`def execute_query(secret_dict, sql_query, s3_bucket_path, chained_s3_write_role, s3_bucket_region, redshift_db):`
	`  conn_string = "dbname='%s' port='%s' user='%s' password='%s' host='%s'" \`
	`  %(redshift_db,`
	`  secret_dict["port"],`
	`  secret_dict["username"],`
	`  secret_dict["password"],`
	`  secret_dict["host"])`
	`  con = psycopg2.connect(conn_string)`
	`  unload_command = "UNLOAD ('{}') TO '{}' IAM_ROLE '{}' DELIMITER '|' REGION '{}';" \`
	`  .format(sql_query,`
	`  s3_bucket_path + str(datetime.datetime.now()) + ".csv",`
		应用程序开发人员

Task	描述	所需技能

	<pre>chained_s3_write_role,
	
	s3_bucket_region)
	
	# Opening a cursor and run query
	cur = con.cursor()
	cur.execute(unload_command)
	
	print(cur.fetchone())
	cur.close()
	con.close()</pre>	

清理

Task	描述	所需技能
删除 Lambda 函数。	为避免产生计划外成本，请移除资源以及 DEV 和 PROD 账户之间的连接。 要删除 Lambda 函数，请执行以下操作： 1. 打开 AWS Lambda 控制台，网址为 https://console.aws.amazon.com/lambda/。 2. 找到并选择您创建的 Lambda 函数。 3. 选择操作，然后选择删除。 4. 确认删除操作。	DevOps 工程师

Task	描述	所需技能
移除 IAM 角色和策略。	从 DEV 和 PROD 账户中移除 IAM 角色和策略。 在 DEV 账户中，执行以下操作： 1. 打开 IAM 管理控制台。 2. 删除以下角色： • `S3-Write-Role` • `CrossAccount-RM-Read-Role`（Lambda 角色） 3. 删除关联的策略： • `S3-Write-Policy` • 担任 PROD 账户角色的 CrossAccount 政策 在 PROD 账户中，执行以下操作： 1. 打开 IAM 管理控制台。 2. 删除以下角色： • `SM-Read-Role` • `CrossAccount-S3-Write-Role` 3. 删除关联的策略： • 访问 Secrets Manager 的 CrossAccount 政策 • `S3-Write-Role-Assume-Policy`	DevOps 工程师

Task	描述	所需技能
在 Secrets Manager 中删除密钥。	要删除密钥，请执行以下操作： 1. 在 PROD 账户中，打开 Secrets Manager 控制台。 2. 找到并选择名为的密钥`Redshift-Creds-Secret` 。 3. 选择操作，然后选择删除密钥。 4. 确认删除操作。	DevOps 工程师
移除 VPC 对等互连和安全组规则。	要删除 VPC 对等互连和安全组规则，请执行以下操作： 1. 在 PROD 账户中，打开亚马逊 EC2 控制台，网址为 https://console.aws.amazon.com/ec2/。 2. 导航到安全组。 3. 查找 Amazon Redshift 集群使用的安全组。 4. 编辑入站规则，并删除允许从开发者账户的 Lambda VPC 建立连接的规则。 5. 导航到 VPC 对等连接，然后删除对等连接。	DevOps 工程师

Task	描述	所需技能
从 S3 存储桶中移除数据。	要从 Amazon S3 中删除数据，请执行以下操作： 1. 在开发者账户中，打开 Amazon S3 控制台，网址为 https://console.aws.amazon.com/s3/。 2. 找到您用于存储数据的存储桶。 3. 删除存储桶中的对象，或者如果不再需要整个存储桶，则将其删除。	DevOps 工程师
清理 AWS KMS 钥匙。	如果您创建了任何用于加密的自定义 AWS KMS 密钥，请执行以下操作： 1. 打开 AWS KMS 控制台，网址为 https://console.aws.amazon.com/kms/。 2. 找到为此模式创建的所有密钥。 3. 安排删除密钥。（删除密钥需要等待一段时间）。	DevOps 工程师

Task	描述	所需技能
查看并删除 Amazon CloudWatch 日志。	要删除日 CloudWatch 志，请执行以下操作： 1. 打开 CloudWatch 控制台，网址为 https://console.aws.amazon.com/cloudwatch/。 2. 检查是否有任何由您的 Lambda 函数或 Amazon Redshift 集群创建的日志组。 3. 如果不再需要这些日志组，请将其删除。	DevOps 工程师

相关资源

- 亚马逊 CloudWatch 文档
- IAM 文档
- Lambda 文档
- Amazon Redshift 文档
- Amazon S3 文档
- AWS Secrets Manager 文档
- AWS 安全原则

其他信息

将数据从 Amazon Redshift 卸载到亚马逊 S3 后，您可以使用亚马逊 Athena 对其进行分析。

Amazon Athena 是一项大数据查询服务，在您需要访问大量数据时非常有用。您无需配置服务器或数据库即可使用 Athena。Athena 支持复杂查询，你可以在不同的对象上运行它。

与大多数人一样 AWS 服务，使用 Athena 的主要好处是，它在运行查询时提供了极大的灵活性，而不会增加复杂性。使用 Athena 时，无需更改数据类型即可在 Amazon S3 中查询不同的数据类型，例

如 CSV 和 JSON。您可以查询来自各种来源的数据，包括外部数据 AWS。Athena 降低了复杂性，因为您不必管理服务器。在您运行查询之前，Athena 直接从 Amazon S3 读取数据，无需加载或更改数据。

按照工作负载分类的数据库迁移模式

主题

- IBM
- Microsoft
- 不适用
- 开源
- Oracle
- SAP

IBM

- 使用 AWS DMS 将 Db2 数据库从亚马逊迁移 EC2 到兼容 Aurora MySQL
- 使用日志传送将适用于 LUW 的 Db2 迁移 EC2 到 Amazon 以减少中断时间
- 将适用于 LUW 的 Db2 迁移到 EC2 具有高可用性灾难恢复功能的 Amazon
- 使用 AWS DMS 和 AWS SCT 从亚马逊上的 IBM Db2 迁移 EC2 到兼容 Aurora PostgreSQL
- 在 Amazon 上从 IBM WebSphere 应用程序服务器迁移到 Apache Tomcat EC2
- 将数据从 IBM Db2、SAP、Sybase 和其他数据库流式传输到 MongoDB Atlas AWS

Microsoft

- 加快 Microsoft 工作负载的发现和迁移到 AWS
- 使用链接服务器从亚马逊 EC2 上的微软 SQL Server 访问本地微软 SQL Server 表
- 评测将 SQL Server 数据库迁移至 MongoDB Atlas on AWS 的查询性能
- 使用 AWS Lambda 任务计划程序在亚马逊 EC2 上运行的 SQL Server Express 版中自动执行数据库任务
- 更改 Python 和 Perl 应用程序以支持数据库从Microsoft SQL Server 迁移至兼容 Amazon Aurora PostgreSQL 的版本
- 在 AWS 上的 SQL Server 的“始终打开”可用性组中配置只读路由
- 使用微软 Excel 和 Python 为 AWS DMS 任务创建 AWS CloudFormation 模板
- 使用 AWS DMS 将 Microsoft SQL Server 数据库导出至 Amazon S3
- 使用 AWS DMS 将 Amazon RDS for SQL Server 表导出至 S3 存储桶
- 从 SQL Server 迁移到 PostgreSQL 时，对 PII 数据实施 SHA1 哈希处理
- 将 EC2 Windows 实例提取并迁移到 AWS Managed Services 账户
- 将消息队列从 Microsoft Azure 服务总线迁移到 Amazon SQS
- 使用 AWS DMS 将微软 SQL Server 数据库从亚马逊迁移 EC2 到亚马逊 DocumentDB
- 使用 AWS DMS 和 AWS SCT 将 Microsoft SQL Server 数据库迁移到 Aurora MySQL
- 将 .NET 应用程序从 Microsoft Azure 应用服务迁移到 AWS Elastic Beanstalk
- 将本地微软 SQL Server 数据库迁移到亚马逊 EC2
- 将本地 Microsoft SQL Server 数据库迁移至 Amazon RDS for SQL Server
- 使用链接服务器将本地 Microsoft SQL Server 数据库迁移至 Amazon RDS for SQL Server
- 使用本机备份和还原将本地 Microsoft SQL Server 数据库迁移到 Amazon RDS for SQL Server。
- 使用 AWS DMS 将本地 Microsoft SQL Server 数据库迁移至 Amazon Redshift
- 使用 AWS SCT 数据提取代理将本地 Microsoft SQL Server 数据库迁移至 Amazon Redshift
- 将本地微软 SQL Server 数据库迁移到 EC2 运行 Linux 的亚马逊上的微软 SQL Server
- 使用 Rclone 将数据从 Microsoft Azure Blob 迁移至 Amazon S3
- 将关系数据库迁移到 MongoDB Atlas AWS
- 使用分布式可用性组将 SQL Server 迁移至 AWS
- 使用 ACM 将 Windows SSL 凭证迁移到应用程序负载均衡器
- 在 Amazon Web Services Cloud 中重新托管本地工作负载：迁移核对清单

- 解决将 Microsoft SQL Server 迁移至 Amazon Web Services Cloud 后出现的连接错误
- 使用本地 SMTP 服务器和数据库邮件发送 Amazon RDS for SQL Server 数据库实例通知
- 使用 Terraform 为数据库迁移设置 CI/CD 管道
- 使用 Amazon 为 SQL Server Always On FCI 设置多可用区基础架构 FSx

不适用

- 在重新托管迁移到的过程中，为防火墙请求创建批准流程 AWS
- 加密现有 Amazon RDS for PostgreSQL 数据库实例
- 估算 Amazon DynamoDB 表的存储成本
- 通过 AWS DMS 和 Amazon Aurora 实施跨区域灾难恢复

开源

- 使用 Python 应用程序为亚马逊 DynamoDB 自动生成 PynamoDB 模型和 CRUD 函数
- 在 pgAdmin 中使用 SSH 隧道进行连接
- 在 Aurora PostgreSQL 兼容中创建应用程序用户和角色
- 在 Amazon RDS 中为 PostgreSQL 数据库实例启用加密连接
- 使用并使用和，将 Amazon RDS for Oracle 迁移到适用于 PostgreSQL AWS SCT 的 Amazon RDS AWS DMSAWS CLIAWS CloudFormation
- 使用原生工具将本地 MariaDB 数据库迁移至 Amazon RDS for MariaDB
- 将本地 MySQL 数据库迁移到亚马逊 EC2
- 将本地 MySQL 数据库迁移至 Amazon RDS for MySQL
- 将本地 MySQL 数据库迁移至 Aurora MySQL
- 将本地 PostgreSQL 数据库迁移到 Aurora PostgreSQL
- 将 Couchbase 服务器数据库迁移到亚马逊 EC2
- 使用 Auto Scaling 从 IBM WebSphere 应用程序服务器迁移到亚马逊 EC2 上的 Apache Tomcat
- 使用和 AWS DMS 从 Oracle 8i 或 9i 迁移到适用于 Oracle 的 Amazon RD SharePlex S
- 从 Oracle 迁移 GlassFish 到 AWS Elastic Beanstalk
- 使用 pglogical 从 EC2 亚马逊上的 PostgreSQL 迁移到适用于 PostgreSQL 的亚马逊 RDS
- SageMaker 使用 AWS 开发人员工具将 ML 构建、训练和部署工作负载迁移到 Amazon
- 使用 AWS App2Container 将本地 Java 应用程序迁移到 AWS
- 使用 Percona、A XtraBackup mazon EFS 和 Amazon S3 将本地 MySQL 数据库迁移到 Aurora MySQL
- 将 Oracle 外部表迁移到 Amazon Aurora PostgreSQL-Compatible
- 将含有 100 多个参数的 Oracle 函数和过程迁移到 PostgreSQL
- 将 Redis 工作负载迁移至 Redis Enterprise Cloud on AWS
- 监控 Amazon Aurora 以查找未加密的实例
- 重启 RHEL 源服务器 SELinux 后，无需禁用 AWS 复制代理即可自动重启 AWS 复制代理
- 使用 Lambda 和 Secrets Manager 计划适用于 Amazon RDS for PostgreSQL 和 Aurora PostgreSQL 的任务
- EC2 使用 GTID 在亚马逊 RDS for MySQL 和亚马逊上的 MySQL 之间设置数据复制
- 使用 pg_transport 在两个 Amazon RDS 数据库实例之间传输 PostgreSQL 数据库

- 跨账户将数据从 Amazon Redshift 集群卸载到亚马逊 S3

Oracle

- 使用只读副本 PeopleSoft 在 Amazon RDS Custom 上将 HA 添加到 Oracle
- 将 JSON Oracle 查询转换至 PostgreSQL 数据库 SQL
- 将 Oracle 的 VARCHAR2 (1) 种数据类型转换为亚马逊 Aurora PostgreSQL 的布尔数据类型
- 通过使用兼容 PostgreSQL 的 Aurora 全局数据库来模拟 Oracle 灾难恢复
- 使用 Aurora PostgreSQL 中的自定义端点模拟 Oracle RAC 工作负载
- 使用 AWR 报告估计 Oracle 数据库的 Amazon RDS 引擎大小
- 在 Aurora PostgreSQL 中处理动态 SQL 语句中的匿名块
- 在 Aurora PostgreSQL 兼容中处理重载的 Oracle 函数
- 使用 Oracle SQL Developer 和 AWS SCT 以增量方式从 Amazon RDS for Oracle 迁移至 Amazon RDS for PostgreSQL
- 使用兼容 Aurora PostgreSQL 的文件编码将 BLOB 文件加载至文本中
- 将 Amazon RDS for Oracle 数据库实例迁移到使用 AMS 的其他账户
- 使用 AWS DMS 在 SSL 模式下将 Amazon RDS for Oracle 迁移到 Amazon RDS for PostgreSQL
- 将 Amazon RDS for Oracle 数据库迁移到另一个 AWS 区域 数据库 AWS 账户 并 AWS DMS 用于持续复制
- 将 Amazon RDS for Oracle 数据库实例迁移至另一个 VPC
- 使用 Oracle 数据泵将本地 Oracle 数据库迁移到亚马逊 EC2
- 使用 Logstash 将本地 Oracle 数据库迁移到亚马逊 OpenSearch 服务
- 使用 AWS DMS 和 AWS SCT 将本地 Oracle 数据库迁移至 Amazon RDS for MySQL
- 将本地 Oracle 数据库迁移到 Amazon RDS for Oracle
- 使用通过数据库链接直接导入 Oracle Data Pump 将本地 Oracle 数据库迁移到 Amazon RDS for Oracle
- 使用 Oracle 数据泵将本地 Oracle 数据库迁移到 Amazon RDS for Oracle
- 使用 Oracle Bystander 和 AWS DMS 将本地 Oracle 数据库迁移到 Amazon RDS for PostgreSQL
- 将本地 Oracle 数据库迁移到亚马逊上的 Oracle EC2
- 使用 AWS DMS 和 AWS SCT 将 Oracle 数据库从亚马逊迁移 EC2 到适用于 MariaDB 的亚马逊 RDS
- 使用 AWS DMS 将 Oracle 数据库从亚马逊迁移 EC2 到 Amazon RDS for Oracle
- 使用 AWS DMS 将 Oracle 数据库迁移至 Amazon DynamoDB
- 使用 Oracle GoldenGate 平面文件适配器将 Oracle 数据库迁移到 Amazon RDS for Oracle

- 使用 AWS DMS 和 AWS SCT 将 Oracle 数据库迁移至 Amazon Redshift
- 使用 AWS DMS 和 AWS SCT 将 Oracle 数据库迁移至 Aurora PostgreSQL
- 使用 Oracle 数据泵和 AWS DMS 将 Oracle JD Edwards EnterpriseOne 数据库迁移到 AWS
- 使用 AWS DMS 将 Oracle 分区表迁移到 PostgreSQL
- 使用 AWS DMS 将 Oracle PeopleSoft 数据库迁移到 AWS
- 将数据从本地 Oracle 数据库迁移到 Aurora PostgreSQL
- 从 Amazon RDS for Oracle 迁移到 Amazon RDS for MySQL
- 使用实体化视图和 AWS DMS 从 Oracle 8i 或 9i 迁移至 Amazon RDS for PostgreSQL
- 使用和 AWS DMS 从 Oracle 8i 或 9i 迁移到适用于 SharePlex PostgreSQL 的亚马逊 RDS
- 使用 Oracle 从 Oracle 数据库迁移到 Amazon RDS for PostgreSQL GoldenGate
- 使用 AWS DMS 和 AWS SCT 从亚马逊上的 Oracle 迁移 EC2 到适用于 MySQL 的亚马逊 RDS
- 使用 AWS DMS 从 Oracle 迁移至 Amazon DocumentDB
- 在 Amazon ECS 上从 Oracle 迁移 WebLogic 到 Apache Tomcat (ToMee)
- 将基于函数的索引从 Oracle 迁移到 PostgreSQL
- 将遗留应用程序从 Oracle Pro*C 迁移到 ECPG
- 将 Oracle CLOB 值迁移到 AWS 上 PostgreSQL 中的单独的行
- 将 Oracle 数据库错误代码迁移到 Amazon Aurora PostgreSQL-Compatible 数据库
- 将 Oracle 电子商务套件迁移到 Amazon RDS Custom
- 使用扩展将 Oracle 原生函数迁移到 PostgreSQL
- 将 Oracle OUT 绑定变量迁移到 PostgreSQL 数据库
- 将 Oracle 迁移 PeopleSoft 到亚马逊 RDS 定制版
- 将 Oracle ROWID 功能迁移到 AWS 上的 PostgreSQL
- 将 Oracle SERIALLY_REUSABLE pragma 包迁移至 PostgreSQL
- 将虚拟生成的列从 Oracle 迁移至 PostgreSQL
- 使用亚马逊监控 Oracle GoldenGate 日志 CloudWatch
- 从 Oracle Database Enterprise Edition 更换平台到 Amazon RDS for Oracle 上的 Standard Edition 2。
- 使用有效备用数据库为 Amazon RDS Custom 上的 Oracle 电子商务套件设置 HA/DR 架构
- 在 Aurora PostgreSQL-Compatible 上设置 Oracle UTL_FILE 功能
- 在 Amazon RDS 上为 Oracle PeopleSoft 应用程序过渡角色适用于 Oracle 定制
- 从 Oracle 迁移至 Amazon Aurora PostgreSQL 后验证数据库对象

SAP

- 使用 Systems Manager 自动备份 SAP HANA 数据库和 EventBridge
- 将本地 SAP ASE 数据库迁移到亚马逊 EC2
- 使用 AWS DMS 从 SAP ASE 迁移至 Amazon RDS for SQL Server
- 使用 AWS SCT 和 AWS DMS EC2 将亚马逊上的 SAP ASE 迁移到兼容 Aurora PostgreSQL 的亚马逊 Aurora
- 使用具有相同主机名的 SAP HSR 将 SAP HANA 迁移至 AWS
- 使用 Application Migration Service 缩短同构 SAP 迁移割接时间
- 在 IBM Db2 on AWS 上为 SAP 设置灾难恢复

更多模式

- 从亚马逊 EKS 容器访问亚马逊 Neptune 数据库
- 使用 Athena 访问、查询和联接 Amazon DynamoDB 表
- 在 Amazon DynamoDB 中聚合数据，以便在 Athena 中进行 ML 预测
- 允许 EC2 实例在 AMS 账户中写入对 S3 存储桶的访问权限
- 使用亚马逊 Athena 和亚马逊分析和可视化嵌套的 JSON 数据 QuickSight
- EC2 使用 AWS Directory Service 对亚马逊上的微软 SQL Server 进行身份验证
- 使用 AWS Batch 自动备份 Amazon RDS for PostgreSQL 数据库实例
- 使用 DynamoDB TTL 自动将项目归档到 Amazon S3
- 自动修复未加密的 Amazon RDS 数据库实例和集群
- 使用 AWS Systems Manager 维护窗口自动停止和启动 Amazon RDS 数据库实例
- 使用 DevOps 实践和 AWS Cloud9 构建具有微服务的松散耦合架构
- 使用和构建 COBOL Db2 程序 AWS Mainframe ModernizationAWS CodeBuild
- 使用 Amazon 构建企业数据网格 DataZone AWS CDK，以及 AWS CloudFormation
- 更改 Python 和 Perl 应用程序以支持数据库从Microsoft SQL Server 迁移至兼容 Amazon Aurora PostgreSQL 的版本
- 使用 Python 在 AWS 上将 EBCDIC 数据转换并解压为 ASCII
- 将 Teradata 标准化时态功能转换为 Amazon Redshift SQL
- 将 Teradata RESET WHEN 功能转换为 Amazon Redshift SQL
- 将 Oracle 的 VARCHAR2 (1) 种数据类型转换为亚马逊 Aurora PostgreSQL 的布尔数据类型
- 在 Aurora PostgreSQL 兼容中创建应用程序用户和角色
- 使用微软 Excel 和 Python 为 AWS DMS 任务创建 AWS CloudFormation 模板
- 使用 Kinesis Data Streams 和 Firehose 将 DynamoDB 记录传送到亚马逊 S3 AWS CDK
- 使用私有静态在 Amazon EC2 上部署 Cassandra 集群 IPs 以避免重新平衡
- 使用 RAG 和提示开发基于 AI 聊天的高级生成式 AI 助手 ReAct
- 通过使用兼容 PostgreSQL 的 Aurora 全局数据库来模拟 Oracle 灾难恢复
- 在 Amazon RDS for SQL Server 中启用透明数据加密
- 使用 AWS DMS 将 Microsoft SQL Server 数据库导出至 Amazon S3
- 从 SQL Server 迁移到 PostgreSQL 时，对 PII 数据实施 SHA1 哈希处理

- 使用 Oracle SQL Developer 和 AWS SCT 以增量方式从 Amazon RDS for Oracle 迁移至 Amazon RDS for PostgreSQL
- 使用兼容 Aurora PostgreSQL 的文件编码将 BLOB 文件加载至文本中
- 使用 AWS Secrets Manager 管理凭证
- 使用 AWS DMS 将 Db2 数据库从亚马逊迁移 EC2 到兼容 Aurora MySQL
- 使用 AWS DMS 将微软 SQL Server 数据库从亚马逊迁移 EC2 到亚马逊 DocumentDB
- 使用 AWS DMS 和 AWS SCT 将 Microsoft SQL Server 数据库迁移到 Aurora MySQL
- 将自托管 MongoDB 环境迁移到 MongoDB Atlas AWS
- 使用 AWS SCT 数据提取代理将 Teradata 数据库迁移到 Amazon Redshift
- 使用 AWS DMS 在 SSL 模式下将 Amazon RDS for Oracle 迁移到 Amazon RDS for PostgreSQL
- 使用并使用和，将 Amazon RDS for Oracle 迁移到适用于 PostgreSQL AWS SCT 的 Amazon RDS AWS DMSAWS CLIAWS CloudFormation
- 将 Amazon RDS 数据库实例迁移到另一个 VPC 或账户
- 将 Amazon RDS for Oracle 数据库迁移到另一个 AWS 区域 数据库 AWS 账户 并 AWS DMS 用于持续复制
- 将 Amazon RDS for Oracle 数据库实例迁移至另一个 VPC
- 将 Amazon Redshift 集群迁移至中国的 AWS 区域
- 使用原生工具将本地 MariaDB 数据库迁移至 Amazon RDS for MariaDB
- 将本地微软 SQL Server 数据库迁移到亚马逊 EC2
- 将本地 Microsoft SQL Server 数据库迁移至 Amazon RDS for SQL Server
- 使用链接服务器将本地 Microsoft SQL Server 数据库迁移至 Amazon RDS for SQL Server
- 使用本机备份和还原将本地 Microsoft SQL Server 数据库迁移到 Amazon RDS for SQL Server。
- 使用 AWS DMS 将本地 Microsoft SQL Server 数据库迁移至 Amazon Redshift
- 使用 AWS SCT 数据提取代理将本地 Microsoft SQL Server 数据库迁移至 Amazon Redshift
- 将本地微软 SQL Server 数据库迁移到 EC2 运行 Linux 的亚马逊上的微软 SQL Server
- 将本地 MySQL 数据库迁移到亚马逊 EC2
- 将本地 MySQL 数据库迁移至 Amazon RDS for MySQL
- 将本地 MySQL 数据库迁移至 Aurora MySQL
- 使用 Oracle 数据泵将本地 Oracle 数据库迁移到亚马逊 EC2
- 使用 Logstash 将本地 Oracle 数据库迁移到亚马逊 OpenSearch 服务
- 使用 AWS DMS 和 AWS SCT 将本地 Oracle 数据库迁移至 Amazon RDS for MySQL

- 将本地 Oracle 数据库迁移到 Amazon RDS for Oracle
- 使用通过数据库链接直接导入 Oracle Data Pump 将本地 Oracle 数据库迁移到 Amazon RDS for Oracle
- 使用 Oracle 数据泵将本地 Oracle 数据库迁移到 Amazon RDS for Oracle
- 使用 Oracle Bystander 和 AWS DMS 将本地 Oracle 数据库迁移到 Amazon RDS for PostgreSQL
- 将本地 Oracle 数据库迁移到亚马逊上的 Oracle EC2
- 将本地 PostgreSQL 数据库迁移到 Aurora PostgreSQL
- 将本地 SAP ASE 数据库迁移到亚马逊 EC2
- 将本地 ThoughtSpot Falcon 数据库迁移到亚马逊 Redshift
- 使用 AWS SCT 数据提取代理将本地 Vertica 数据库迁移至 Amazon Redshift
- 使用 AWS DMS 和 AWS SCT 将 Oracle 数据库从亚马逊迁移 EC2 到适用于 MariaDB 的亚马逊 RDS
- 使用 AWS DMS 将 Oracle 数据库从亚马逊迁移 EC2 到 Amazon RDS for Oracle
- 使用 AWS DMS 将 Oracle 数据库迁移至 Amazon DynamoDB
- 使用 Oracle GoldenGate 平面文件适配器将 Oracle 数据库迁移到 Amazon RDS for Oracle
- 使用 AWS DMS 和 AWS SCT 将 Oracle 数据库迁移至 Amazon Redshift
- 使用 AWS DMS 和 AWS SCT 将 Oracle 数据库迁移至 Aurora PostgreSQL
- 使用 Oracle 数据泵和 AWS DMS 将 Oracle JD Edwards EnterpriseOne 数据库迁移到 AWS
- 使用 AWS DMS 将 Oracle 分区表迁移到 PostgreSQL
- 使用 AWS DMS 将 Oracle PeopleSoft 数据库迁移到 AWS
- 将 Couchbase 服务器数据库迁移到亚马逊 EC2
- 将数据从本地 Oracle 数据库迁移到 Aurora PostgreSQL
- 使用 Starburs AWS Cloud t 将数据迁移到
- 使用日志传送将适用于 LUW 的 Db2 迁移 EC2 到 Amazon 以减少中断时间
- 将适用于 LUW 的 Db2 迁移到 EC2 具有高可用性灾难恢复功能的 Amazon
- 从 Amazon RDS for Oracle 迁移到 Amazon RDS for MySQL
- 从 Couchbase Server 迁移至 AWS 上的 Couchbase Capella
- 使用 AWS DMS 和 AWS SCT 从亚马逊上的 IBM Db2 迁移 EC2 到兼容 Aurora PostgreSQL
- 使用实体化视图和 AWS DMS 从 Oracle 8i 或 9i 迁移至 Amazon RDS for PostgreSQL
- 使用和 AWS DMS 从 Oracle 8i 或 9i 迁移到适用于 SharePlex PostgreSQL 的亚马逊 RDS
- 使用 Oracle 从 Oracle 数据库迁移到 Amazon RDS for PostgreSQL GoldenGate

- 使用 AWS DMS 和 AWS SCT 从亚马逊上的 Oracle 迁移 EC2 到适用于 MySQL 的亚马逊 RDS
- 使用 AWS DMS 从 Oracle 迁移至 Amazon DocumentDB
- 使用 pglogical 从 EC2 亚马逊上的 PostgreSQL 迁移到适用于 PostgreSQL 的亚马逊 RDS
- 使用 AWS DMS 从 SAP ASE 迁移至 Amazon RDS for SQL Server
- 将基于函数的索引从 Oracle 迁移到 PostgreSQL
- 将遗留应用程序从 Oracle Pro*C 迁移到 ECPG
- 将本地 Cloudera 工作负载迁移到 Cloudera Data Platform on AWS
- 使用 Percona、A XtraBackup mazon EFS 和 Amazon S3 将本地 MySQL 数据库迁移到 Aurora MySQL
- 将 Oracle 商业智能 12c 从本地服务器迁移到 Amazon Web Services Cloud
- 将 Oracle CLOB 值迁移到 AWS 上 PostgreSQL 中的单独的行
- 将 Oracle 数据库错误代码迁移到 Amazon Aurora PostgreSQL-Compatible 数据库
- 将 Oracle 电子商务套件迁移到 Amazon RDS Custom
- 将 Oracle 外部表迁移到 Amazon Aurora PostgreSQL-Compatible
- 使用扩展将 Oracle 原生函数迁移到 PostgreSQL
- 将 Oracle 迁移 PeopleSoft 到亚马逊 RDS 定制版
- 将 Oracle ROWID 功能迁移到 AWS 上的 PostgreSQL
- 将 Oracle SERIALLY_REUSABLE pragma 包迁移至 PostgreSQL
- 将 Redis 工作负载迁移至 Redis Enterprise Cloud on AWS
- 将关系数据库迁移到 MongoDB Atlas AWS
- 使用 AWS SCT 和 AWS DMS EC2 将亚马逊上的 SAP ASE 迁移到兼容 Aurora PostgreSQL 的亚马逊 Aurora
- 将虚拟生成的列从 Oracle 迁移至 PostgreSQL
- 设置最小可行数据空间，以便在组织之间共享数据
- 监控 Amazon ElastiCache 集群的静态加密
- 监控 ElastiCache 集群中的安全组
- 使用 Amazon Athena 使用 SQL 查询亚马逊 DynamoDB 表
- 使用 Application Migration Service 缩短同构 SAP 迁移割接时间
- 在不重启容器的情况下轮换数据库凭证
- 使用 AWS Fargate 大规模运行消息驱动型工作负载
- 使用可信上下文在 AWS 上的 Db2 联合身份验证数据库中保护和简化用户访问

- 在 AWS 上设置高度可用的 PeopleSoft 架构
- 在 Aurora PostgreSQL-Compatible 上设置 Oracle UTL_FILE 功能
- 将数据从 IBM Db2、SAP、Sybase 和其他数据库流式传输到 MongoDB Atlas AWS
- 使用 PGO 简化 PostgreSQL 在 Amazon EKS 上的部署
- 以 CSV 文件形式将大规模 Db2 z/OS 数据传输到 Amazon S3
- 使用 pg_transport 在两个 Amazon RDS 数据库实例之间传输 PostgreSQL 数据库
- 从 Oracle 迁移至 Amazon Aurora PostgreSQL 后验证数据库对象
- 验证新 Amazon Redshift 集群是否在 VPC 中启动

存储和备份

主题

- 允许 EC2 实例在 AMS 账户中写入对 S3 存储桶的访问权限
- 使用 Snowflake Snowpipe、亚马逊 S3、亚马逊 SNS 和亚马逊 Data Firehose 自动将数据流摄入 Snowflake 数据库
- 自动加密现有和新 Amazon EBS 卷
- 在 Amazon Web Services Cloud 上的 Stromasys Charon-SSP 仿真器中备份 Sun SPARC 服务器
- 使用 Veeam Backup & Replication 备份数据并将其存档至 Amazon S3
- 在 AWS 上配置 Veritas NetBackup VMware 云版
- 使用 AWS CLI 将数据从 S3 存储桶复制到其他账户和区域
- 使用 S3 Batch Replication 将数据从 S3 存储桶复制到另一个账户和区域
- 使用 PrivateLink 适用于 Amazon S3 的 DistCp AWS 将数据从本地 Hadoop 环境迁移到 Amazon S3
- 更多模式

允许 EC2 实例在 AMS 账户中写入对 S3 存储桶的访问权限

由 Mansi Suratwala (AWS) 编写

摘要

AWS Managed Services (AMS) 可帮助您更高效、更安全地运营 AWS 基础设施。AMS 账户有安全防护栏，可对您的 AWS 资源进行标准化管理。一个保护措施是，默认的亚马逊弹性计算云 (Amazon EC2) 实例配置文件不允许对亚马逊简单存储服务 (Amazon S3) Simple Service 存储桶进行写入访问。但是，您的组织可能有多个 S3 存储桶，因此需要对 EC2 实例的访问进行更多控制。例如，您可能希望将 EC2 实例中的数据库备份存储在 S3 存储桶中。

此模式说明了如何使用更改请求 (RFCs) 来允许您的 EC2 实例对您的 AMS 账户中的 S3 存储桶进行写入权限。RFC 是由您或 AMS 创建的用于在托管环境中进行更改的请求，其中包括特定操作的更改类型 (CT) ID。

先决条件和限制

先决条件

- AMS 高级账户。有关这方面的更多信息，请参阅 AMS 文档中的 AMS 运营计划。
- 访问要提交的 AWS Identity and Access Management (IAM) `customer-mc-user-role` 角色 RFCs。
- AWS Command Line Interface (AWS CLI)，使用您的 AMS 账户中的 EC2 实例进行安装和配置。
- 了解如何在 AMS RFCs 中创建和提交。有关这方面的更多信息，请参阅什么是 AMS 变更类型？ 在 AMS 文档中。
- 了解手动和自动变更类型 (CTs)。有关这方面的更多信息，请参阅 AMS 文档 CTs中的自动和手动。

架构

技术堆栈

- AMS
- AWS CLI
- Amazon EC2
- Amazon S3
- IAM

工具

- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行外壳中的命令进行交互。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Managed Services (AMS) 可帮助您更高效、更安全地运行 AWS 基础设施。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- 亚马逊弹性计算云 (Amazon EC2) 在中提供可扩展的计算容量 AWS Cloud。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。

操作说明

使用 RFC 创建 S3 存储桶

Task	描述	所需技能
使用自动 RFC 创建 S3 存储桶。	1. 登录您的 AMS 账户，选择“选择更改类型”页面，选择 RFCs，然后选择“创建 RFC”。 2. 提交“创建 S3 存储桶”自动 RFC。 **Note** 请务必记录 S3 存储桶的名称。	AWS 系统管理员、AWS 开发人员

创建 IAM 实例配置文件并将其与 EC2 实例关联

Task	描述	所需技能
提交手动 RFC 以创建 IAM 角色。	加入 AMS 账户后，将创建一个名为的默认 IAM 实例配置文件，并将其与您`customer-mc-ec2-instance-profile` 的 AMS 账户中的每个 EC2 实例相关联。但是，实例配置文件对您的 S3 存储桶没有写入权限。 要添加写入权限，请提交创建 IAM 资源手册 RFC 以创建具有以下三个策略的 IAM 角色：`customer_ec2_instance_` `customer_deny_policy` 、和`customer_ec2_s3_integration_policy` 。 **Important** `customer_ec2_instance_` 和`customer_deny_policy` 政策已存在于您的 AMS 账户中。但是，您需要使用以下示例策略`customer_ec2_s3_integration_policy` 进行创建：	AWS 系统管理员、AWS 开发人员

Task	描述	所需技能

	<pre>{
	  "Version": "2012-10-17",
	   "Statement": [
	    {
	      "Sid": "",
	       "Effect": "Allow",
	       "Principal": {
	         "Service": "ec2.amazonaws.com"
	      },
	       "Action": "sts:AssumeRole"
	    }
	  ]
	}
	
	Role Permissions:
	{
	     "Version": "2012-10-17",
	     "Statement": [
	        {
	             "Action": [
	
	"s3:ListBucket",
	
	"s3:GetBucketLocation"
	            ],
	             "Resource": "arn:aws:s3:::",
	             "Effect": "Allow"
	        },
	        {
	             "Action": [
	
	"s3:GetObject",</pre>	

Task	描述	所需技能

	<pre>"s3:PutObject",
	
	"s3:ListMultipartUploadParts",
	
	"s3:AbortMultipartUpload"
	],
	"Resource": "arn:aws:s3:::/*",
	"Effect": "Allow"
	}
	]
	}</pre>	

提交手动 RFC 以替换 IAM 实例配置文件。	提交手动 RFC，将目标 EC2 实例与新的 IAM 实例配置文件相关联。	AWS 系统管理员、AWS 开发人员
测试对 S3 存储桶的复制操作。	通过在中运行以下命令来测试对 S3 存储桶的复制操作 AWS CLI： <pre>aws s3 cp test.txt s3://<S3 bucket>/test2.txt</pre>	AWS 系统管理员、AWS 开发人员

相关资源

- 为您的 Amazon 实例创建 IAM EC2 实例配置文件
- 创建 S3 存储桶（使用 Amazon S3 控制台 AWS SDKs、或 AWS CLI）

使用 Snowflake Snowpipe、亚马逊 S3、亚马逊 SNS 和亚马逊 Data Firehose 自动将数据流摄入 Snowflake 数据库

由 Bikash Chandra Rout (AWS) 创建

摘要

此模式描述了如何使用 Amazon Web Services (AWS) 云上的服务来处理连续的数据流并将其加载到 Snowflake 数据库中。该模式使用 Amazon Data Firehose 将数据传输到亚马逊简单存储服务 (Amazon S3)，使用亚马逊简单通知服务 (Amazon SNS) Simple Notification 在收到新数据时发送通知，使用 Snowflake Snowpipe 将数据加载到 Snowflake 数据库中。

通过遵循这种模式，您可以在几秒钟内持续生成的数据可供分析，避免使用多个手动`COPY`命令，并且完全支持加载时的半结构化数据。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- 一种持续向 Firehose 传输流发送数据的数据源。
- 接收来自 Firehose 传输流的数据的现有 S3 存储桶。
- 活跃 Snowflake 账户。

限制

- Snowflake Snowpipe 无法直接连接到 Firehose。

架构

技术堆栈

- Amazon Data Firehose
- Amazon SNS
- Amazon S3

- Snowflake Snowpipe
- Snowflake 数据库

工具

- Amazon Data Firehose 是一项完全托管的服务，用于向亚马逊 S3、亚马逊 Redshift、 OpenSearch 亚马逊服务、Splunk 等目的地以及受支持的第三方服务提供商拥有的任何自定义 HTTP 终端节点或 HTTP 终端节点提供实时流式传输数据。
- 亚马逊简单存储服务 (Amazon S3) S ervice 是互联网存储。
- Amazon Simple Notification Service (Amazon SNS) 可协调和管理向订阅端点或客户端传送或发送消息的过程。
- Snowflake — Snowflake 是一个以（SaaS）形式提供的分析数据仓库。 Software-as-a-Service
- Snowflake Snowpipe – 在 Snowflake 阶段，一旦文件可用，Snowpipe 将立即加载文件中的数据。

操作说明

设置 Snowflake Snowpipe

Task	描述	所需技能
在 Snowflake 中创建 CSV 格式文件。	登录 Snowflake 并运行`CREATE FILE FORMAT`命令以创建具有指定字段分隔符的 CSV 文件。有关此命令和其他 Snowflake 命令的更多信息，请参阅“其他信息”部分。	开发人员
创建外部 Snowflake 阶段。	运行`CREATE STAGE`命令创建一个引用您之前创建的 CSV 文件的外部 Snowflake 舞台。重要：您将需要 S3 存储桶的 URL、 AWS 访问密钥和私有访问 AWS 密钥。运行`SHOW STAGES`命令以验证 Snowflake 舞台是否已创建。	开发人员

Task	描述	所需技能
创建 Snowflake 目标表。	运行`CREATE TABLE`命令创建 Snowflake 表。	开发人员
创建管道。	运行`CREATE PIPE`命令；确保命令中有`auto_ingest=true`该命令。运行`SHOW PIPES`命令以验证管道是否已创建。复制并保存`notification_channel`列值。此值可用于配置 Amazon S3 事件通知。	开发人员

配置 S3 存储桶

Task	描述	所需技能
为 S3 存储桶创建 30 天的生命周期策略。	登录 AWS Management Console 并打开 Amazon S3 控制台。选择包含来自 Firehose 的数据的 S3 存储桶。然后在 S3 存储桶中选择 “管理” 选项卡，然后选择 “添加生命周期规则”。在生命周期规则对话框内输入规则名称，并为存储桶配置 30 天生命周期规则。要获取有关此操作和其他操作的帮助，请参阅相关资源部分。	系统管理员、开发人员
为 S3 存储桶创建 IAM policy。	打开 AWS Identity and Access Management (IAM) 控制台并选择策略。选择 Create policy（创建策略），然后选择 JSON 选项卡。将策略从 “其他信息” 部分复制并粘贴	系统管理员、开发人员

Task	描述	所需技能
	到 JSON 字段中。此策略将授予`PutObject` 和`DeleteObject` 权限`GetObject` ，以及`GetObjectVersion` 、和`ListBucket` 权限。选择查看策略，输入策略名称，然后选择创建策略。	
将该策略分配至 IAM 角色。	打开 IAM 控制台，选择角色，然后选择创建角色。选择其他 AWS 账户作为可信实体。输入您的 AWS 账户 ID，然后选择“需要外部身份证”。输入占位符 ID，稍后将对其进行更改。选择 “下一步”，然后分配您之前创建的 IAM 策略。然后创建 IAM 角色。	系统管理员、开发人员
复制IAM 角色的 Amazon 资源名称（ARN）。	打开 IAM 控制台，然后选择角色。选择您之前创建的 IAM 角色，然后复制并存储角色 ARN。	系统管理员、开发人员

在 Snowflake 中设置存储集成

Task	描述	所需技能
在 Snowflake 创建存储集成。	登录 Snowflake 并运行命令。`CREATE STORAGE INTEGRATION` 这将修改信任关系，授予 Snowflake 访问权限，并为您的 Snowflake 阶段提供外部 ID。	系统管理员、开发人员

Task	描述	所需技能
为您的 Snowflake 账户检索 IAM 角色。	运行`DESC INTEGRATION` 命令检索 IAM 角色的 ARN。 **Important** `<integration_name>`是您之前创建的 Snowflake 存储集成的名称。	系统管理员、开发人员
记录两列的值。	复制并保存`storage_aws_iam_user_arn` 和`storage_aws_external_id` 列的值。	系统管理员、开发人员

允许 Snowflake Snowpipe 访问 S3 存储桶

Task	描述	所需技能
修改 IAM 角色策略。	打开 IAM 控制台，选择角色。选择您之前创建的 IAM 角色并选择信任关系选项卡。选择编辑信任关系。`snowflake_external_id` 用您之前复制的`storage_aws_external_id` 值替换。`snowflake_user_arn` 用您之前复制的`storage_aws_iam_user_arn` 值替换。然后选择“更新信任策略”。	系统管理员、开发人员

为 S3 存储桶开启并配置 SNS 通知

Task	描述	所需技能
打开 S3 存储桶事件通知。	打开 Amazon S3 控制台并选择存储桶。选择 “属性”，然后在 “高级设置” 下选择 “事件”。选择 “添加通知”，然后输入此事件的名称。如果未输入名称，则使用全局唯一标识符 (GUID)。	系统管理员、开发人员
为 S3 存储桶配置 Amazon SNS 通知。	在 “事件” 下，选择 ObjectCreate （全部），然后在 “发送至” 下拉列表中选择 SQS 队列。在 SNS 列表中，选择添加 SQS 队列 ARN，然后粘贴之前复制的`notification_channel` 值。然后选择保存。	系统管理员、开发人员
为 Snowflake SQS 队列订阅 SNS 主题。	为 Snowflake SQS 队列订阅您创建的 SNS 主题。有关此步骤的帮助，请参阅相关资源部分。	系统管理员、开发人员

查看 Snowflake 阶段集成

Task	描述	所需技能
检查并测试 Snowpipe。	登录 Snowflake 并打开 Snowflake 阶段。将文件拖放至 S3 存储桶，然后检查 Snowflake 表是否已加载这些文件。当 S3 存储桶中显示新对象时，Amazon S3 将向 Snowpipe 发送 SNS 通知。	系统管理员、开发人员

相关资源

- 管理存储生命周期
- 为 Snowflake SQS 队列订阅 Amazon SNS 主题。

其他信息

创建文件格式：

```
CREATE FILE FORMAT <name>
TYPE = 'CSV'
FIELD_DELIMITER = '|'
SKIP_HEADER = 1;
```

创建外部阶段：

```
externalStageParams (for Amazon S3) ::=
  URL = 's3://[//]'

  [ { STORAGE_INTEGRATION =  } | { CREDENTIALS = ( {  { AWS_KEY_ID = `` AWS_SECRET_KEY
 = `` [ AWS_TOKEN = `` ] } | AWS_ROLE = ``  } ) ) }` ]
  [ ENCRYPTION = ( [ TYPE = 'AWS_CSE' ] [ MASTER_KEY = '' ] |
                   [ TYPE = 'AWS_SSE_S3' ] |
                   [ TYPE = 'AWS_SSE_KMS' [ KMS_KEY_ID = '' ] |
                   [ TYPE = NONE ] )
```

创建表：

```
CREATE [ OR REPLACE ] [ { [ LOCAL | GLOBAL ] TEMP[ORARY] | VOLATILE } | TRANSIENT ]
 TABLE [ IF NOT EXISTS ]
  <table_name>
    ( <col_name> <col_type> [ { DEFAULT <expr>
                                | { AUTOINCREMENT | IDENTITY } [ ( <start_num> ,
 <step_num> ) | START <num> INCREMENT <num> ] } ]
                                 /* AUTOINCREMENT / IDENTITY supported only for numeric
 data types (NUMBER, INT, etc.) */
                             [ inlineConstraint ]
      [ , <col_name> <col_type> ... ]
      [ , outoflineConstraint ]
      [ , ... ] )
  [ CLUSTER BY ( <expr> [ , <expr> , ... ] ) ]
```

```
  [ STAGE_FILE_FORMAT = ( { FORMAT_NAME = '<file_format_name>'
                            | TYPE = { CSV | JSON | AVRO | ORC | PARQUET | XML }
 [ formatTypeOptions ] } ) ]
  [ STAGE_COPY_OPTIONS = ( copyOptions ) ]
  [ DATA_RETENTION_TIME_IN_DAYS = <num> ]
  [ COPY GRANTS ]
  [ COMMENT = '<string_literal>' ]
```

显示阶段：

```
SHOW STAGES;
```

创建管道：

```
CREATE [ OR REPLACE ] PIPE [ IF NOT EXISTS ]
  [ AUTO_INGEST = [ TRUE | FALSE ] ]
  [ AWS_SNS_TOPIC =  ]
  [ INTEGRATION = '' ]
  [ COMMENT = '' ]
  AS
```

显示管道：

```
SHOW PIPES [ LIKE '<pattern>' ]
           [ IN { ACCOUNT | [ DATABASE ] <db_name> | [ SCHEMA ] <schema_name> } ]
```

创建存储集成：

```
CREATE STORAGE INTEGRATION <integration_name>
  TYPE = EXTERNAL_STAGE
  STORAGE_PROVIDER = S3
  ENABLED = TRUE
  STORAGE_AWS_ROLE_ARN = '<iam_role>'
  STORAGE_ALLOWED_LOCATIONS = ('s3://<bucket>/<path>/', 's3://<bucket>/<path>/')
  [ STORAGE_BLOCKED_LOCATIONS = ('s3://<bucket>/<path>/', 's3://<bucket>/<path>/') ]
```

示例：

```
create storage integration s3_int
  type = external_stage
```

```
  storage_provider = s3
  enabled = true
  storage_aws_role_arn = 'arn:aws:iam::001234567890:role/myrole'
  storage_allowed_locations = ('s3://amzn-s3-demo-bucket1/mypath1/', 's3://amzn-s3-
demo-bucket2/mypath2/')
  storage_blocked_locations = ('s3://amzn-s3-demo-bucket1/mypath1/sensitivedata/',
 's3://amzn-s3-demo-bucket2/mypath2/sensitivedata/');
```

有关此步骤的更多信息，请参阅 Snowflake 文档中的配置 Snowflake 存储集成以访问 Amazon S3。

描述集成：

```
DESC INTEGRATION <integration_name>;
```

S3 存储桶策略：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "s3:PutObject",
              "s3:GetObject",
              "s3:GetObjectVersion",
              "s3:DeleteObject",
              "s3:DeleteObjectVersion"
            ],
            "Resource": "arn:aws:s3:::/*"
        },
        {
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "/*"
                    ]
                }
            }
        }
    ]
```

```
}
```

自动加密现有和新 Amazon EBS 卷

由 Tony DeMarco (AWS) 和 Josh Joy (AWS) 创作

摘要

Amazon Elastic Block Store (Amazon EBS) 卷加密对企业的数据保护策略非常重要。这是建立良好架构环境的重要一步。虽然没有直接的方法可以加密现有的未加密 EBS 卷或快照，但您可以通过创建新卷或快照来加密它们。有关更多信息，请参阅 Amazon EC2 文档中的加密 EBS 资源。此模式为加密新增和现有 EBS 卷提供了预防性控制和侦测性控制。在此模式中，您可以配置账户设置、创建自动修复流程以及实施访问控制。

先决条件和限制

先决条件

- 活跃 Amazon Web Services (AWS) 账户
- AWS 命令行界面（AWS CLI），已在 macOS、Linux 或 Windows 上安装并配置。
- jq，已在 macOS、Linux 或 Windows 上安装并配置
- AWS Identity and Access Management (IAM) 权限已配置为拥有对 AWS CloudFormation、亚马逊弹性计算云 (亚马逊 EC2)、AWS Systems Manager、AWS Config 和 AWS 密钥管理服务 (AWS KMS) 的读写权限
- 已配置已启用所有功能的 AWS Organizations，这是服务控制策略的要求
- AWS Config 已在目标账户中启用

限制

- 在目标 Amazon Web Services account 中，不得存在名为 encrypted-volumes 的 AWS Config 规则。此解决方案部署具有此名称的规则。使用具有此名称的预先存在的规则可能会导致部署失败，并产生与多次处理同一规则相关的不必要费用。
- 此解决方案采用同一 AWS KMS 密钥对所有 EBS 卷加密。
- 如果为账户启用了 EBS 卷加密，则此设置为区域特定。如果为某个 AWS 区域启用了它，则无法为该区域中单独的卷或快照禁用。有关更多信息，请参阅 Amazon EC2 文档中的默认加密。
- 修复现有未加密的 EBS 卷时，请确保该 EC2 实例未在使用中。此自动化将关闭实例，以便分离未加密卷并附加加密卷。修复过程中会出现停机。如果这是您组织的关键基础架构，请确保手动或自动的高可用性配置已到位，以免影响该实例上运行的任何应用程序的可用性。建议仅在标准维护时段修复关键资源。

架构

自动化工作流程

1. AWS Config 检测到未加密 EBS 卷。
2. 管理员通过 AWS Config 向 Systems Manager 发送补救命令。
3. Systems Manager 自动生成未加密 EBS 卷的快照。
4. Systems Manager 自动使用 AWS KMS 创建快照的加密副本。
5. Systems Manager 自动化执行以下操作：
   a. 如果受影响的 EC2 实例正在运行，则将其停止
   b. 将卷的新加密副本附加到实 EC2 例
   c. 将 EC2 实例恢复到其原始状态

工具

Amazon Web Services

- AWS CLI — AWS 命令行接口 (AWS CLI) Line Interface 提供对 AWS 服务的公共应用程序编程接口 APIs () 的直接访问。您可以使用 AWS CLI 探索服务的功能，并开发 Shell 脚本来管理资源。除了低级 API 等效命令，多项 Amazon Web Services 亦为 AWS CLI 提供了自定义项。自定义项可能包括更高级别的命令，可简化具有复杂 API 的服务的使用。
- AWS CloudFormation — AWS CloudFormation 是一项可帮助您建模和设置 AWS 资源的服务。您可以创建一个描述所需所有 AWS 资源（例如 Amazon EC2 实例）的模板，并 CloudFormation 为您预置和配置这些资源。
- AWS Config - AWS Config 提供 Amazon Web Services account 中 AWS 资源配置的详细视图。这些信息包括资源之间的关联方式以及资源以前的配置方式，让您了解资源的配置和关系如何随着的时间的推移而更改。
- 亚马逊 EC2 — Amazon Elastic Compute Cloud (Amazon EC2) 是一项网络服务，可提供可调整的计算容量，供您构建和托管软件系统。
- AWS KMS – AWS Key Management Service (AWS KMS) 是一项扩展到云的加密和密钥管理服务。AWS KMS 密钥和功能可用于其他 Amazon Web Services，您可以使用它们保护 AWS 环境中的数据。

- AWS Organizations – AWS Organizations 是一项账户管理服务，可让您将多个 Amazon Web Services account 整合到您创建并集中管理的组织中。
- AWS Systems Manager Aut omation — Systems Manager Automation 简化了亚马逊 EC2 实例和其他 AWS 资源的常见维护和部署任务。

其他服务

- jq – jq 是一个轻量级且灵活的命令行 JSON 处理器。您使用此工具从 AWS CLI 输出中提取关键信息。

代码

- 此模式的代码可在使用客户 KMS 密钥 GitHub 自动修复未加密的 EBS 卷存储库中找到。

操作说明

自动修复未加密卷

Task	描述	所需技能
下载脚本和 CloudFormation 模板。	从使用客户 KMS 密钥存储库 GitHub 自动修复未加密的 EBS 卷中下载 shell 脚本、JSON 文件和 CloudFormation 模板。	AWS 管理员、常规 AWS
确定 AWS KMS 密钥管理员。	1. 登录 AWS 管理控制台并在上打开 IAM 控制台https://console.aws.amazon.com/iam/。 2. 确定将成为 AWS KMS 密钥管理员的用户或者角色。如果需要为此目的创建新用户或者角色，请立即创建。有关更多信息，请参阅 IAM 文档中的 IAM Identities。此	AWS 管理员、常规 AWS

Task	描述	所需技能
	自动化功能将创建新 AWS KMS 密钥。 3. 一经确定即复制用户或角色的 Amazon 资源名称（ARN）。有关更多信息，请参阅 IAM 文档 ARNs中的 IAM。在下一步骤中，您将使用此 ARN。	
部署 Stack1 模板 CloudFormation 。	1. 打开 AWS CloudFormation 控制台，网址为https://console.aws.amazon.com/cloudformation/。 2. 在 CloudFormation中，部署 `Stack1.yaml` 模板。请注意以下部署详细信息： • 为堆栈赋予清晰的描述性名称。请注意堆栈名称，因为您需要在下一步骤中使用此值。 • 将密钥管理员的 ARN 粘贴至 Stack1 中的唯一参数字段。此用户或角色将成为堆栈创建的 AWS KMS 密钥的管理员。 有关部署 CloudFormation 模板的更多信息，请参阅 CloudFormation 文档中的使用 AWS CloudFormation 模板。	AWS 管理员、常规 AWS

Task	描述	所需技能
部署 Stack2 模板 CloudFormation 。	在 CloudFormation中，部署 `Stack2.yaml` 模板。请注意以下部署详细信息： • 为堆栈赋予清晰的描述性名称。 • 对于 Stack2 的唯一参数，请输入您在上一步骤中创建的堆栈名称。这允许 Stack2 引用堆栈在上一步骤中部署的新 AWS KMS 密钥和角色。	AWS 管理员、常规 AWS
创建测试用未加密卷。	使用未加密 EBS 卷创建 EC2 实例。有关说明，请参阅亚马逊 EC2 文档中的创建 Amazon EBS 卷。实例类型并不重要，无需访问该实例。您可以创建 t2.micro 实例，以保留在免费套餐中，且无需创建密钥对。	AWS 管理员、常规 AWS

Task	描述	所需技能
测试 AWS Config 规则。	1. 打开 AWS Config 控制台，网址为https://console.aws.amazon.com/config/。在规则页面上，选择加密卷规则。 2. 确认您的新未加密测试实例出现在不合规资源列表中。如果卷未立即显示，请等待几分钟后刷新结果。AWS Config 规则将在创建实例和卷后不久检测到资源变化。 3. 选择资源，然后选择修复。 您可以在 Systems Manager 中查看如下修复进度与状态： 1. 打开 AWS Systems Manager 控制台，网址为https://console.aws.amazon.com/systems-manager/。 2. 在导航窗格中，选择 自动化。 3. 选择执行 ID 链接以查看步骤和状态。	AWS 管理员、常规 AWS
配置其他账户或者 AWS 区域。	根据您的用例需要，对任何其他账户或 AWS 区域重复此操作	AWS 管理员、常规 AWS

启用 EBS 卷的账户级加密

Task	描述	所需技能
运行启用脚本。	1. 在 bash Shell 中，使用 cd 命令以导航至已克隆存储库。 2. 输入以下命令运行 enable-ebs-encryption-for-account 脚本。 `./Bash/enable-ebs-encryption-for-account.sh`	AWS 管理员、常规 AWS、bash
确认设置已更新。	1. 打开亚马逊 EC2 控制台，网址为https://console.aws.amazon.com/ec2/。 2. 在屏幕右侧的 “设置” 下，选择 “数据保护和安全”。 3. 在 EBS 加密部分下，确认始终加密新 EBS 卷已开启，并且默认加密密钥已设置为您之前指定的 ARN。 **Note** 如果 “始终加密新 EBS 卷” 设置已关闭或密钥仍设置为 alias/aws/ebs，请确认您登录的是运行 shell 脚本的同一账户和 AWS 区域，并检查 shell 中是否有错误消息。	AWS 管理员、常规 AWS

Task	描述	所需技能
配置其他账户或者 AWS 区域。	根据您的用例需要，对任何其他账户或 AWS 区域重复此操作	AWS 管理员、常规 AWS

避免创建未加密实例

Task	描述	所需技能
创建服务控制策略。	1. 在 https://console.aws.amazon.com/organizations/v2/ 上打开 AWS Organizations 控制台。 2. 创建新服务控制策略。有关更多信息，请参阅 AWS Organizations 文档中的创建服务控制策略。 3. 将 `DenyUnencryptedEC2.json` 的内容添加至策略并保存。你在第一部长篇故事中从 GitHub 存储库中下载了这个 JSON 文件。 4. 将此政策附加到组织根目录或任何必要的组织单位 (OUs)。有关更多信息，请参阅 AWS Organizations 文档中的附加和分离服务控制策略。	AWS 管理员、常规 AWS

相关资源

Amazon Web Services 文档

- AWS CLI
- AWS Config
- AWS CloudFormation
- Amazon EC2
- AWS KMS
- AWS Organizations
- AWS Systems Manager Automation

其他资源

- jq 手册（jq 网站）
- jq 下载 () GitHub

在 Amazon Web Services Cloud 上的 Stromasys Charon-SSP 仿真器中备份 Sun SPARC 服务器

由 Kevin Yung (AWS)、Luis Ramos (Stromasys) 和 Rohit Darji (AWS) 编写

摘要

此模式提供了四种选项，用于在从本地环境迁移到 Amazon Web Services (AWS) 云后备份 Sun Microsystems SPARC 服务器。这些备份选项可帮助您实施满足组织的恢复点目标（RPO）和恢复时间目标（RTO）的备份计划，使用自动化方法并降低总体运营成本。该模式概述了四个备份选项以及实现它们的步骤。

如果您使用作为访客托管在 Stromasys Charon-SSP 仿真器上的 Sun SPARC 服务器，则可以使用以下三个备份选项之一：

- 备份选项 1：Stromasys 虚拟磁带 – 使用 Charon-SSP 虚拟磁带功能在 Sun SPARC 服务器中设置备份设施，并使用 AWS Systems Manager Automation 将备份文件存档到 Amazon Simple Storage Service (Amazon S3) 和 Amazon Simple Storage Service Glacier。
- 备份选项 2：Stromasys 快照 – 使用 Charon-SSP 快照功能在 Charon-SSP 中为 Sun SPARC 客户机服务器设置备份。
- 备份选项 3：亚马逊弹性区块存储 (Amazon EBS) Elastic Block Store 卷快照 — 如果您在亚马逊弹性计算云（亚马逊）上托管 Charon-SSP 模拟器，则可以使用 EC2亚马逊 EBS 卷快照为 Sun SPARC 文件系统创建备份。

如果您在硬件上使用以访客身份托管的 Sun SPARC 服务器，在 A EC2 mazon 上使用 Charon-SSP，则可以使用以下备份选项：

- 备份选项 4：AWS Storage Gateway 虚拟磁带库 (VTL) – 使用带有 Storage Gateway VTL Tape Gateway 的备份应用程序备份 Sun SPARC 服务器。

如果您使用作为 Sun SPARC 服务器中的标记区域托管的 Sun SPARC 服务器，则可以使用备份选项 1、2 和 4。

Stromasys 提供软件和服务来模拟传统的 SPARC、Alpha、VAX 以及 PA-RISC 关键系统。有关使用 Stromasys 仿真迁移到 Amazon Web Services Cloud 的更多信息，请参阅 AWS Blog 上的使用 Stromasys 将 SPARC、Alpha 或其他遗留系统重新托管到 AWS。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 现有 Sun SPARC 服务器。
- Charon-SSP 现有许可证。Charon-SSP 的许可证可从 Amazon Web Services Marketplace 获取，Stromasys 虚拟环境 (VE) 的许可证可从 Stromasys 获取。有关更多信息，请联系 Stromasys 销售。
- 熟悉 Sun SPARC 服务器与 Linux 备份。
- 熟悉 Charon-SSP 仿真技术。有关这方面的更多信息，请参阅 Stromasys 文档中的 Stromasys 遗留服务器仿真。
- 如果要对 Sun SPARC 服务器文件系统使用虚拟磁带工具或备份应用程序，则必须为 Sun SPARC 服务器文件系统创建和配置备份工具。
- 了解 RPO 和 RTO。有关这方面的更多信息，请参阅 AWS Well-Architected Framework 文档中的可靠性支柱白皮书中的灾难恢复目标。
- 要使用备份选项 4，必须具备以下条件：
  - 基于软件的备份应用程序，支持 Storage Gateway VTL 磁带网关。有关这方面的更多信息，请参阅 AWS Storage Gateway 文档中的使用 VTL 设备。
  - Bacula Director 或类似的备份应用程序已安装并配置。有关这方面的更多信息，请参阅 Bacula Director 文档。

下表提供了有关此模式中的四个备份选项的信息。

备份选项	实现崩溃一致性？	实现应用程序一致性？	虚拟备份设备解决方案？	典型用例
选项 1 – Stromasys 虚拟磁带	是 您可自动执行 Sun SPARC 文件系统快照以备份虚拟磁带中的数据。例如，您	是 此备份选项需要一个自动脚本来刷新正在进行的事务、在文件系统快照期间配置只读或临时脱机	是	Sun SPARC 服务器文件系统使用 .tar 或 .zip 文件进行备份 应用程序数据备份

	可使用 UFS 或 ZFS 快照。	模式，或者进行应用程序数据转储。您可能还需要应用程序停机或者只读模式。		
选项 2 – Stromasys 快照	是 必须配置 Charon-SSP 管理器，或使用命令行启动参数才能启用此功能。 您还必须运行 Linux 命令，使 Charon-SSP 仿真器将 Sun SPARC 客户机服务器状态保存到快照文件中。 **Important** 您必须关闭 Sun SPARC 访客服务器。	是 此备份选项创建模拟来宾服务器的快照，包括其虚拟磁盘和内存转储。 **Important** 在快照期间必须关闭 Sun SPARC 客户机服务器。	否	Sun SPARC 服务器快照 应用程序数据备份

选项 3 – Amazon EBS 卷快照	是 您可使用 AWS Backup 自动生成 Amazon EBS 快照。	是 此备份选项需要自动脚本来刷新正在进行的交易，并在 Amazon EBS 卷快照期间配置 EC2 实例的只读或临时停止。 **Important** 此备份选项可能需要应用程序停机或只读模式才能实现应用程序一致性。	否	Sun SPARC 服务器文件系统快照 应用程序数据备份

选项 4 – AWS Storage Gateway VTL	是 您可使用备份代理将 Sun SPARC 文件系统备份数据自动备份到 VTL。	是 此备份选项需要自动脚本来刷新正在进行的事务，并在文件系统快照或应用程序数据转储期间配置只读或临时脱机模式。 **Important** 此备份选项可能需要应用程序停机或只读模式。	是	大量 Sun SPARC 服务器文件系统备份 应用程序数据备份

限制

- 您可使用此模式的方法来备份单个 Sun SPARC 服务器，但如果您有在集群中运行的应用程序，也可以使用这些备份选项来存储共享数据。

工具

备份选项 1：Stromasys 虚拟磁带

- Stromasys Charon-SSP 仿真器 – Charon-SSP 仿真器在标准 64 位 x86 兼容计算机系统内创建原始 SPARC 硬件的虚拟副本。它运行原始的 SPARC 二进制代码，包括 SunOS 或 Solaris 等操作系统 (OSs)、它们的分层产品和应用程序。
- 亚马逊 EC2 — Amazon Elastic Compute Cloud (Amazon EC2) 是一项网络服务，可提供可调整的计算容量，供您构建和托管软件系统。
- Amazon EFS — Amazon Elastic File System（Amazon EFS）提供了一个简单、无服务器的 set-and-forget弹性文件系统，可用于 AWS 云服务和本地资源。

- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项面向互联网的存储服务。
- Amazon S3 Glacier – Amazon Simple Storage Service Glacier 是一项安全、持久且成本极低的 Amazon S3 存储类，适用于数据存档和长期备份。
- AWS Systems Manager Automation — Automation 是 AWS Systems Manager 的一项功能，它简化了 EC2 实例和其他 AWS 资源的常见维护和部署任务。

备份选项 2：Stromasys 快照

- Stromasys Charon-SSP 仿真器 – Charon-SSP 仿真器在标准 64 位 x86 兼容计算机系统内创建原始 SPARC 硬件的虚拟副本。它运行原始的 SPARC 二进制代码， OSs 包括 SunOS 或 Solaris、它们的分层产品和应用程序。
- 亚马逊 EC2 — Amazon Elastic Compute Cloud (Amazon EC2) 是一项网络服务，可提供可调整的计算容量，供您构建和托管软件系统。
- Amazon EFS — Amazon Elastic File System（Amazon EFS）提供了一个简单、无服务器的 set-and-forget弹性文件系统，可用于 AWS 云服务和本地资源。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项面向互联网的存储服务。
- Amazon S3 Glacier – Amazon Simple Storage Service Glacier 是一项安全、持久且成本极低的 Amazon S3 存储类，适用于数据存档和长期备份。
- AWS Systems Manager Automation — Automation 是 AWS Systems Manager 的一项功能，它简化了 EC2 实例和其他 AWS 资源的常见维护和部署任务。

备份选项 3：Amazon EBS 卷快照

- Stromasys Charon-SSP 仿真器 – Charon-SSP 仿真器在标准 64 位 x86 兼容计算机系统内创建原始 SPARC 硬件的虚拟副本。它运行原始的 SPARC 二进制代码， OSs 包括 SunOS 或 Solaris、它们的分层产品和应用程序。
- AWS Backup – AWS Backup 是一项完全托管的数据保护服务，可在云中以及本地方便地集中管理和自动执行各种 Amazon Web Services。
- 亚马逊 EBS — Amazon Elastic Block Store (Amazon EBS) 提供用于实例的块级存储卷。 EC2
- 亚马逊 EC2 — Amazon Elastic Compute Cloud (Amazon EC2) 是一项网络服务，可提供可调整的计算容量，供您构建和托管软件系统。

备份选项 4：AWS Storage Gateway VTL

- Stromasys Charon-SSP 仿真器 – Charon-SSP 仿真器在标准 64 位 x86 兼容计算机系统内创建原始 SPARC 硬件的虚拟副本。它运行原始的 SPARC 二进制代码， OSs 包括 SunOS 或 Solaris、它们的分层产品和应用程序。
- Bacula – Bacula 是一个开源企业级计算机备份系统。有关现有备份应用程序是否支持磁带网关的更多信息，请参阅 AWS Storage Gateway 文档中的磁带网关支持的第三方备份应用程序。
- 亚马逊 EC2 — Amazon Elastic Compute Cloud (Amazon EC2) 是一项网络服务，可提供可调整的计算容量，供您构建和托管软件系统。
- Amazon RDS for MySQL – Amazon Relational Database Service (Amazon RDS) 支持运行多个版本 MySQL 的数据库实例。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项面向互联网的存储服务。
- Amazon S3 Glacier – Amazon Simple Storage Service Glacier 是一项安全、持久且成本极低的 Amazon S3 存储类，适用于数据存档和长期备份。
- AWS Storage Gateway – Storage Gateway 将本地软件设备与基于云的存储相连接，从而在本地 IT 环境与 AWS 存储基础设施间提供具备数据安全功能的无缝集成。

操作说明

备份选项 1 – 创建 Stromasys 虚拟磁带备份

Task	描述	所需技能
为虚拟磁带文件存储创建 Amazon EFS 共享文件系统。	登录 AWS 管理控制台 或使用 AWS CLI 创建 Amazon EFS 文件系统。 有关这方面的更多信息，请参阅 Amazon EFS 文档中的创建 Amazon EFS 文件系统。	云架构师
将 Linux 主机配置为挂载共享文件系统。	在亚马逊 EC2 Linux 实例上安装 Amazon EFS 驱动程序，将 Linux 操作系统配置为在启动期	DevOps 工程师

Task	描述	所需技能
	间挂载 Amazon EFS 共享文件系统。 有关这方面的更多信息，请参阅 Amazon EFS 文档中的使用 EFS 挂载助手挂载文件系统。	
安装 Charon-SSP 仿真器。	在亚马 EC2 逊 Linux 实例上安装 Charon-SSP 模拟器。 有关这方面的更多信息，请参阅 Stromasys 文档中的为 Charon-SSP 设置 Amazon Web Services Cloud 实例。	DevOps 工程师
在共享文件系统中为每台 Sun SPARC 客户机服务器创建虚拟磁带文件容器。	运行 `touch <vtape-container-name>` 命令，在共享文件系统中为部署在 Charon-SSP 仿真器中的每台 Sun SPARC 客户机服务器创建一个虚拟磁带文件容器。	DevOps 工程师
配置 Charon-SSP Manager 为 Sun SPARC 来宾服务器创建虚拟磁带设备。	登录 Charon-SSP Manager，创建虚拟磁带设备，并将其配置为使用每个 Sun SPARC 来宾服务器的虚拟磁带容器文件。 有关这方面的更多信息，请参阅 Stromasys 文档中的 Charon-SSP 5.2 for Linux 用户指南。	DevOps 工程师

Task	描述	所需技能
验证虚拟磁带设备在 Sun SPARC 来宾服务器中可用。	登录到每台 Sun SPARC 客户机服务器并运行 `mt -f /dev/rmt/1` 命令以验证操作系统中是否配置了虚拟磁带设备。	DevOps 工程师
开发 Systems Manager Automation 运行手册和自动化功能。	开发 Systems Manager Automation 运行手册，在 Systems Manager 中设置维护窗口以及关联以安排备份过程。 有关这方面的更多信息，请参阅 AWS Systems Manager 文档中的自动化演练和设置维护时段。	云架构师
配置 Systems Manager Automation 以归档轮换的虚拟磁带容器文件。	使用其他信息部分的返回选项 1 中的代码示例来开发 Systems Manager Automation 运行手册，将旋转后的虚拟磁带容器文件存档到 Amazon S3 和 Amazon S3 Glacier。	云架构师
部署 Systems Manager Automation 运行手册以进行归档和调度。	部署 Systems Manager Automation 运行手册并安排其在 Systems Manager 中自动运行。 有关这方面的更多信息，请参阅 Systems Manager 文档中的自动化演练。	云架构师

备份选项 2 – 创建 Stromasys 快照

Task	描述	所需技能
为虚拟磁带文件存储创建 Amazon EFS 共享文件系统。	登录 AWS 管理控制台 或使用 AWS CLI 创建 Amazon EFS 文件系统。 有关这方面的更多信息，请参阅 Amazon EFS 文档中的创建您的 Amazon EFS 文件系统。	云架构师
将 Linux 主机配置为挂载共享文件系统。	在亚马逊 EC2 Linux 实例中安装 Amazon EFS 驱动程序，将 Linux 操作系统配置为在启动期间挂载 Amazon EFS 共享文件系统。 有关这方面的更多信息，请参阅 Amazon EFS 文档中的使用 EFS 挂载助手挂载文件系统。	DevOps 工程师
安装 Charon-SSP 仿真器。	在亚马 EC2 逊 Linux 实例上安装 Charon-SSP 模拟器。 有关这方面的更多信息，请参阅 Stromasys 文档中的为 Charon-SSP 设置 Amazon Web Services Cloud 实例。	DevOps 工程师
配置 Sun SPARC 来宾服务器以使用快照选项启动。	使用 Charon-SSP Manager 为每个 Sun SPARC 来宾服务器设置快照选项。 有关这方面的更多信息，请参阅 Stromasys 文档中的 Charon-SSP 5.2 for Linux 用户指南。	DevOps 工程师

Task	描述	所需技能
开发 Systems Manager Automation 运行手册。	使用其他信息部分的备份选项 2 中的代码示例开发 Systems Manager Automation 运行手册，以便在维护时段内在 Sun SPARC 客户机服务器上远程运行快照命令。	云架构师
部署 Systems Manager 自动化运行手册并设置与 Amazon EC2 Linux 主机的关联。	部署 Systems Manager Automation 运行手册，在 Systems Manager 中设置维护窗口和关联以安排备份过程。 有关这方面的更多信息，请参阅 AWS Systems Manager 文档中的自动化演练和设置维护时段。	云架构师
将快照存档到长期存储中。	使用其他信息部分中的运行手册示例代码来开发 Systems Manager Automation 运行手册，将快照文件存档到 Amazon S3 和 Amazon S3 Glacier。	云架构师

备份选项 3 – 创建 Amazon EBS 卷快照

Task	描述	所需技能
安装 Charon-SSP 仿真器。	在亚马 EC2 逊 Linux 实例上安装 Charon-SSP 模拟器。 有关这方面的更多信息，请参阅 Stromasys 文档中的为 Charon-SSP 设置 Amazon Web Services Cloud 实例。	DevOps 工程师

Task	描述	所需技能
为 Sun SPRAC 客户机服务器创建 EBS 卷。	登录 AWS 管理控制台，打开 Amazon EBS 控制台，然后为 Sun SPRAC 客户机服务器创建 EBS 卷。 有关这方面的更多信息，请参阅 Stromasys 文档中的为 Charon-SSP 设置 Amazon Web Services Cloud 实例。	云架构师
将 EBS 卷连接到 Amazon EC2 Linux 实例。	在亚马逊 EC2 控制台上，将 EBS 卷附加到亚马逊 EC2 Linux 实例。 有关这方面的更多信息，请参阅亚马逊 EC2 文档中的将 Amazon EBS 卷附加到实例。	AWS DevOps
在 Charon-SSP 仿真器中将 EBS 卷映射至 SCSI 驱动器。	配置 Charon-SSP 管理器以将 EBS 卷映射为 Sun SPARC 来宾服务器中的 SCSI 驱动器。 有关这方面的更多信息，请参阅 Stromasys 文档中的 Charon-SSP V5.2 for Linux 指南的 SCSI 存储配置部分。	AWS DevOps
配置用于对 EBS 卷进行快照的 AWS Backup 计划。	设置 AWS Backup 策略和计划以对 EBS 卷进行快照。 有关这方面的更多信息，请参阅 AWS Developer Center 文档中的使用 AWS Backup 进行 Amazon EBS 备份和恢复。	AWS DevOps

备份选项 4 – 创建 AWS Storage Gateway VTL

Task	描述	所需技能
创建磁带网关设备。	登录 AWS 管理控制台，打开 AWS Storage Gateway 控制台，然后在 VPC 中创建磁带网关设备。 有关这方面的更多信息，请参阅 AWS Storage Gateway 文档中的创建网关。	云架构师
为 Bacula 目录创建 Amazon RDS 数据库实例。	打开 Amazon RDS 控制台，然后创建一个 Amazon RDS for MySQL 数据库实例。 有关这方面的更多信息，请参阅 Amazon RDS 文档中的创建 MySQL 数据库实例并连接到 MySQL 数据库实例上的数据库。	云架构师
在 VPC 部署备份应用程序控制器。	在 EC2 实例上安装 Bacula，部署备份应用程序控制器，然后将备份存储配置为与磁带网关设备连接。您可以在 `Bacula-storage-daemon-config.txt` 文件（附件）中使用示例 Bacula Director 存储进程守护程序配置。 有关这方面的更多信息，请参阅 Bacula 文档。	AWS DevOps
在 Sun SPARC 客户机服务器上设置备份应用程序。	使用 `SUN-SPARC-Guest-Bacula-Config.txt` 文件（附件）中的 Bacula 配置示	DevOps 工程师

Task	描述	所需技能
	例，设置第二个客户端，以便在 Sun SPARC 客户机服务器上安装和设置备份应用程序。	
设置备份配置与时间表。	使用 `Bacula-Directory-Config.txt` 文件（附件）中的 Bacula Director 配置示例，在备份应用程序控制器中设置备份配置和计划。 有关这方面的更多信息，请参阅 Bacula 文档。	DevOps 工程师
验证备份配置与时间表是否正确。	按照 Bacula 文档中的说明，在 Sun SPARC 客户机服务器中对您的设置进行验证和备份测试。 例如，您可使用以下命令验证配置文件： • `bacula-dir -t -c bacula-dir.conf` • `bacula-fd -t -c bacula-fd.conf` • `bacula-sd -t -c bacula-sd.conf`	DevOps 工程师

相关资源

- 带 VE 许可的 Charon 虚拟 SPARC
- Charon 虚拟 SPARC
- 在 Bacula Enterprise Edition 中使用云服务和对象存储
- 灾难恢复 (DR) 目标
- Charon 遗留系统仿真解决方案

其他信息

备份选项 1 – 创建 Stromasys 虚拟磁带

您可以使用以下示例 Systems Manager Automation 运行手册代码自动启动备份，然后交换磁带：

```
...
# example backup script saved in SUN SPARC Server
 #!/usr/bin/bash
 mt -f  rewind
 tar -cvf
 mt -f  offline
...
         mainSteps:
         - action: aws:runShellScript
           name:
           inputs:
             onFailure: Abort
             timeoutSeconds: "1200"
             runCommand:
             - |
               # Validate tape backup container file exists
               if [ ! -f {{TapeBackupContainerFile}} ]; then
                 logger -s -p local3.warning "Tape backup container file is not exists
 - {{TapeBackupContainerFile}}, create a new one"
                 touch {{TapeBackupContainerFile}}
               fi
         - action: aws:runShellScript
           name: startBackup
           inputs:
             onFailure: Abort
             timeoutSeconds: "1200"
             runCommand:
             - |
               user={{BACKUP_USER}}
               keypair={{KEYPAIR_PATH}}
               server={{SUN_SPARC_IP}}
               backup_script={{BACKUP_SCRIPT}}
               ssh -i $keypair $user@$server -c "/usr/bin/bash $backup_script"
         - action: aws:runShellScript
           name: swapVirtualDiskContainer
           inputs:
             onFailure: Abort
             timeoutSeconds: "1200"
```

```
            runCommand:
            - |
              mv {{TapeBackupContainerFile}} {{TapeBackupContainerFile}}.$(date +%s)
              touch {{TapeBackupContainerFile}}
        - action: aws:runShellScript
          name: uploadBackupArchiveToS3
          inputs:
            onFailure: Abort
            timeoutSeconds: "1200"
            runCommand:
            - |
              aws s3 cp {{TapeBackupContainerFile}} s3://{{BACKUP_BUCKET}}/
{{SUN_SPARC_IP}}/$(date '+%Y-%m-%d')/
 ...
```

备份选项 2 – Stromasys 快照

您可使用以下 Systems Manager Automation 运行手册示例代码来自动执行备份过程：

```
     ...

        mainSteps:
        - action: aws:runShellScript
          name: startSnapshot
          inputs:
            onFailure: Abort
            timeoutSeconds: "1200"
            runCommand:
            - |
              # You may consider some graceful stop of the application before taking a
 snapshot
              # Query SSP PID by configuration file
              # Example: ps ax | grep ssp-4 | grep Solaris10.cfg | awk '{print $1"
"$5}' | grep ssp4 | cut -f1 -d" "
              pid=`ps ax | grep ssp-4 | grep {{SSP_GUEST_CONFIG_FILE}} | awk '{print
$1" "$5}' | grep ssp4 | cut -f1 -d" "`
              if [ -n "${pid}" ]; then
                kill -SIGTSTP ${pid}
              else
                echo "No PID found for SPARC guest with config
{{SSP_GUEST_CONFIG_FILE}}"
                exit 1
              fi
        - action: aws:runShellScript
```

```
            name: startBackup
            inputs:
              onFailure: Abort
              timeoutSeconds: "1200"
              runCommand:
              - |
                # upload snapshot and virtual disk files into S3
                aws s3 sync {{SNAPSHOT_FOLDER}} s3://{{BACKUP_BUCKET}}/$(date '+%Y-%m-
%d')/
                aws s3 cp {{VIRTUAL_DISK_FILE}} s3://{{BACKUP_BUCKET}}/$(date '+%Y-%m-
%d')/
          - action: aws:runShellScript
            name: restratSPARCGuest
            inputs:
              onFailure: Abort
              timeoutSeconds: "1200"
              runCommand:
              - |
                /opt/charon-ssp/ssp-4u/ssp4u -f {{SSP_GUEST_CONFIG_FILE}} -d -a
 {{SPARC_GUEST_NAME}} --snapshot {{SNAPSHOT_FOLDER}}
 ...
```

备份选项 4 – AWS Storage Gateway VTL

如果您使用 Solaris 非全局区域来运行虚拟化的传统 Sun SPARC 服务器，则备份应用程序方法可以应用于 Sun SPARC 服务器中运行的非全局区域（例如，备份客户机可以在非全局区域内运行）。但是，备份客户机也可以在 Solaris 主机上运行并拍摄非全局快照。然后可以将快照备份至磁带。

以下示例配置将托管 Solaris 非全局区域的文件系统添加至 Solaris 主机的备份配置中：

```
FileSet {
   Name = "Branded Zones"
   Include {
     Options {
       signature = MD5
     }
     File = /zones
   }
 }
```

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 Veeam Backup & Replication 备份数据并将其存档至 Amazon S3

由珍娜·詹姆斯（AWS）、安东尼·菲奥雷（AWS）（AWS）和威廉·奎格利（AWS）创作

摘要

此模式详细介绍了使用 Veeam 横向扩展备份存储库功能将 Veeam Backup & Replication 创建的备份发送至支持的 Amazon Simple Storage Service (Amazon S3) 对象存储类的过程。

Veeam 支持多个 Amazon S3 存储类，以更好地满足您的特定需求。您可以根据备份或存档数据的数据访问、弹性和成本要求选择存储类型。例如，您可以将 30 天或更长时间内不打算访问的数据存储为 Amazon S3 不经常访问 (IA) 存储类型，从而降低成本。如果您计划将数据存档 90 天或更长时间，则可以使用 Amazon Simple Storage Service Glacier (Amazon S3) Glacier Flexible Retrieval 或 S3 Glacier Deep Archive (带 Veeam 存档层)。您还可以使用 S3 对象锁来制作在 Amazon S3 中不可变的备份。

此模式不包括如何使用磁带网关设置 Veeam Backup & Replication。 AWS Storage Gateway有关该主题的信息，请参阅 Veeam 网站上的使用 AWS VTL 网关的 Veeam Backup & Replication的部署指南

> **Warning**
>
> 这种情况需要 AWS Identity and Access Management (IAM) 用户具有编程访问权限和长期证书，这会带来安全风险。为帮助减轻这种风险，我们建议仅向这些用户提供执行任务所需的权限，并在不再需要这些用户时将其移除。必要时可以更新访问密钥。有关更多信息，请参阅《IAM 用户指南》中的 更新访问密钥。

先决条件和限制

先决条件

- Veeam Backup & Replication，包括 Veeam Availability Suite 或 Veeam Backup Essentials，已安装（您可以注册免费试用 ）
- 具有 Enterprise 或 Enterprise Plus 功能的 Veeam Backup & Replication 许可证，其中包含 Veeam Universal License (VUL)
- 有权访问 Amazon S3 存储桶的活跃 IAM 用户
- 有权访问亚马逊弹性计算云 (Amazon EC2) 和亚马逊虚拟私有云 (Amazon VPC) 的活跃 IAM 用户（如果使用存档层）

- 通过公共 Internet 连接或 AWS Direct Connect 公共虚拟接口 (VIF) 实现从本地到 AWS 服务 具有可用带宽的网络连接，用于备份和恢复流量
- 打开以下网络端口和端点，以确保与对象存储库通信正常：
  - Amazon S3 存储 – TCP – 端口 443：用于与 Amazon S3 存储进行通信。
  - `*.amazonaws.com`适用于 AWS 区域 中国区域的 Amazon S3 存储（云端点） AWS GovCloud (US) Regions，或`*.amazonaws.com.cn`用于中国区域：用于与 Amazon S3 存储进行通信。有关连接终端节点的完整列表，请参阅 AWS 文档中的 Amazon S3 终端节点。
  - Amazon S3 存储 – TCP HTTP – 端口 80：用于验证证书状态。请考虑证书验证终端节点（证书吊销列表 (CRL) URLs 和在线证书状态协议 (OCSP) 服务器）可能会发生变化。实际地址列表请参见证书本身。
  - Amazon S3 存储 — 证书验证终端节点 —`*.amazontrust.com`：用于验证证书状态。请考虑证书验证端点（CRL URLs 和 OCSP 服务器）可能会发生变化。实际地址列表请参见证书本身。

限制

- 对于用作 Veeam 对象存储库的任何 S3 存储桶，Veeam 均不支持其 S3 生命周期策略。其中包括带有 Amazon S3 存储类转换和 S3 生命周期过期规则的策略。Veeam 必须 是管理这些对象的唯一实体。启用 S3 生命周期策略可能会导致异常结果，包括数据丢失。

产品版本

- Veeam Backup & Replication v9.5 Update 4 或更高版本（仅备份或容量层）
- Veeam Backup & Replication v10 或更高版本(仅备份或容量层和 S3 对象锁定)
- Veeam Backup & Replication v11 或更高版本（备份或容量层、存档或存档层以及 S3 对象锁定）
- Veeam Backup & Replication v12 或更高版本(性能层、备份或容量层、存档或存档层以及 S3 对象锁定)
- S3 标准
- S3 标准 - IA
- S3 单区 - IA
- S3 Glacier Flexible Retrieval（仅 v11 及更高版本）
- S3 Glacier Deep Archive（仅 v11 及更高版本）
- S3 Glacier Instant Retrieval (仅 v12 及更高版本)

架构

源技术堆栈

- 从 Veeam 备份服务器或 Veeam 网关服务器连接至 Amazon S3 的本地 Veeam Backup & Replication 安装

目标技术堆栈

- Amazon S3
- 亚马逊 VPC 和亚马逊 EC2 （如果使用存档层）

目标架构：SOBR

下图显示了横向扩展备份存储库 (SOBR) 的架构。

Veeam Backup and Replication 软件可保护数据免受逻辑错误（例如系统故障、应用程序错误或意外删除）的影响。在此图中，备份首先在本地运行，然后将辅助副本直接发送至 Amazon S3。备份代表数据的 point-in-time副本。

工作流程包括将三个主要组件（分层或复制备份至 Amazon S3 所需组件）和一个选件：

- Veeam Backup & Replication (1) – 是负责协调、控制和管理备份基础设施、设置、任务、恢复任务和其他过程的备份服务器。
- Veeam 网关服务器（未在图中显示）— 当 Veeam 备份服务器未与 Amazon S3 出站连接时需要使用的可选本地网关服务器。
- 横向扩展备份存储库 (2) – 支持多层数据存储水平扩展的存储库系统。横向扩展备份存储库由一个或多个备份存储库组成，可快速访问数据，并可使用 Amazon S3 对象存储库扩展，以便用于长期存储(容量层)和存档(存档层)。Veeam 使用横向扩展备份存储库在本地（性能层）与 Amazon S3 对象存储（容量层和存档层）之间自动分层数据。

> **Note**
>
> 从 Veeam Backup & Replication v12.2 开始，直接到 S3 Glacier 功能使 S3 容量层成为可选的。可以将 SOBR 配置为性能层和 S3 Glacier 存档层。对于在容量层的本地（本地）存

储上投入了大量资金并且只需要在云中长期保留存档的用户，此配置非常有用。有关更多信息，请参阅 Veeam Backup & Replication 文档。

- Amazon S3 (3) — 提供可扩展性、数据可用性、安全性和性能的 AWS 对象存储服务。

目标架构：DTO

下图显示了 direct-to-object (DTO) 架构。

在此图中，备份数据被直接发送至 Amazon S3，无需先存储于本地。辅助副本可以存储至 S3 Glacier。

自动化和扩缩

您可以使用存储库中提供的 AWS CloudFormation 模板自动创建 IAM 资源和 S3 VeeamHub GitHub 存储桶。模板包括标准选项和不可变选项。

工具

工具和 AWS 服务

- Veeam Backup & Replication 是用于保护、备份、复制和恢复虚拟和物理工作负载的 Veeam 解决方案。
- AWS CloudFormation帮助您对资源进行建模和设置，快速一致地配置 AWS 资源，并在资源的整个生命周期中对其进行管理。您可以使用模板来描述资源及其依赖关系，然后将它们作为堆栈一起启动和配置，而不必单独管理资源。您可以跨多个 AWS 账户 和 AWS 区域管理和配置堆栈。
- 亚马逊弹性计算云 (Amazon EC2) 在中提供可扩展的计算容量 AWS Cloud。您可以根据需要使用 Amazon EC2 启动任意数量或数量的虚拟服务器，也可以进行横向扩展或扩展。
- AWS Identity and Access Management (IAM) 是一项用于安全控制访问的 Web 服务 AWS 服务。借助 IAM，您可以集中管理用户、安全证书（例如访问密钥）以及控制用户和应用程序可以访问哪些 AWS 资源的权限。
- Amazon Simple Storage Service (Amazon S3) 是一项对象存储服务。您可以通过 Amazon S3 随时在 Web 上的任何位置存储和检索的任意大小的数据。
- Amazon S3 Glacier (S3 Glacier) 是一项安全、持久且成本较低的存储服务，适用于数据存档和长期备份。

- Amazon Virtual Private Cloud（Amazon VPC）提供了一个逻辑上隔离的部分，您可以在 AWS Cloud 其中启动您定义的虚拟网络中的 AWS 资源。这个虚拟网络与您在数据中心中运行的传统网络极其相似，并会为您提供使用 AWS的可扩展基础设施的优势。

代码

使用VeeamHub GitHub 存储库中提供的 CloudFormation 模板自动为此模式创建 IAM 资源和 S3 存储桶。如果更喜欢手动创建资源，请按操作说明部分中的步骤操作。

最佳实践

- 根据 IAM 最佳实践，我们强烈建议您定期轮转长期 IAM 用户凭证，例如您用于将 Veeam Backup & Replication 备份写入 Amazon S3 的 IAM 用户。有关更多信息，请参阅 IAM 文档中的 Security best practices。

操作说明

在您的账户中配置 Amazon S3 存储桶

Task	描述	所需技能
创建 IAM 用户。	按照 IAM 文档中的说明创建 IAM 用户。此用户不应具有 AWS 控制台访问权限，您需要为此用户创建访问密钥。Veeam 使用此实体进行身份验证 AWS 以读取和写入您的 S3 存储桶。您必须授予最低权限（即，只授予执行任务所需权限），这样用户拥有的权限就不会超出所需。有关要附加到 Veeam IAM 用户的 IAM policy 示例，请参阅其他信息 部分。	AWS 管理员

Task	描述	所需技能
	Note 或者，您可以使用VeeamHub GitHub存储库中提供的 CloudFormation 模板为此模式创建 IAM 用户和 S3 存储桶。	
创建 S3 存储桶。	1. 登录 AWS Management Console 并打开 Amazon S3 控制台。 2. 如果您还没有现有 S3 存储桶可用作目标存储，请选择创建存储桶，然后指定存储桶名称 AWS 区域、和存储桶设置。 • 建议为 S3 存储桶启用阻止公共访问选项，并设置访问和用户权限策略，以满足组织的要求。有关示例，请参阅 Amazon S3 文档。 • 建议启用 S3 对象锁定，即便您不打算立即使用。此设置仅能在创建 S3 存储桶时启用。 有关更多信息，请参阅 Amazon S3 文档中的创建存储桶。	AWS 管理员

将 Amazon S3 和 S3 Glacier Flexible Retrieval（或 S3 Glacier Deep Archive）添加至 Veeam Backup & Replication

Task	描述	所需技能
启动新对象存储库向导。	在 Veeam 中设置对象存储和横向扩展备份存储库之前，必须添加要用于容量和存档层的 Amazon S3 和 S3 Glacier 存储库。在下一个操作说明中，将学习如何将此类存储库连接至横向扩展备份存储库。 1. 在 Veeam 控制台上，打开备份基础设施视图。 2. 在库存窗格，选择备份存储库节点，然后选择添加存储库。 3. 在添加备份存储库对话框中，选择对象存储、Amazon S3。	AWS 管理员、应用程序所有者
为容量层添加 Amazon S3 存储。	1. 在 Amazon Cloud Storage Services 对话框中，选择 Amazon S3。 2. 在向导的命名步骤中，指定对象存储名称和简短描述，例如创建者和创建日期。 3. 在向导的账户步骤中，指定对象存储账户。 • 对于凭证，请选择您在第一操作创建的 IAM 用户，以访问您的 Amazon S3 对象存储。	AWS 管理员、应用程序所有者

Task	描述	所需技能
	• 对于 AWS 区域， AWS 区域 选择 S3 存储桶所在的位置。 4. 在向导的存储桶步骤中，指定对象存储设置。 • 对于数据中心区域，选择 S AWS 区域 3 存储桶所在的位置。 • 对于存储桶，请选择第一个操作中创建的 S3 存储桶。 • 对于文件夹，请创建或选择要将对象存储库映射到的目标云文件夹。 • 如果想启用不可变性，请选择使最近备份在 X 天内不可变，然后设置应锁定备份的时间段。请注意，鉴于 Veeam 对 Amazon S3 的 API 调用次数的增加导致了启用的不可变性，继而导致了成本的增加。 5. 在向导的摘要步骤中，查看配置信息，然后选择完成。	

Task	描述	所需技能
为存档层添加 S3 Glacier 存储。	如果想创建存档层，请使用其他信息部分中详述的 IAM 权限。 1. 如前所述，启动新对象存储库向导。 2. 在 Amazon Cloud Storage Services 对话框中，选择 Amazon S3 Glacier。 3. 在向导的命名步骤中，指定对象存储名称和简短描述，例如创建者和创建日期。 4. 在向导的账户步骤中，指定对象存储账户。 • 对于证书，请选择您在第一个长篇故事中创建的 IAM 用户来访问您的 S3 Glacier 对象存储。 • 对于 AWS 区域， AWS 区域 选择 S3 存储桶所在的位置。 5. 在向导的存储桶步骤中，指定对象存储设置。 • 对于数据中心区域，选择 AWS 区域。 • 对于存储桶，请选择用于存储备份数据的 S3 存储桶。这与容量层所用存储桶相同。 • 对于文件夹，请创建或选择要将对象存储库映射到的目标云文件夹。	AWS 管理员、应用程序所有者

Task	描述	所需技能
	• 如果想启用不可变性，请选择使最近备份在整个保留策略期间不可变。请注意，鉴于 Veeam 对 Amazon S3 的 API 调用次数的增加导致了启用的不可变性，继而导致了成本的增加。 • 如果想使用 S3 Glacier Deep Archive 作为存档存储类，请选择使用 Deep Archive 存储类。 6. 在向导的代理设备步骤，配置用于将数据从 Amazon S3 传输到 S3 Glacier 的辅助实例。您可以使用默认设置或手动配置每个设置。若要手动配置设置： • 选择 Customize (自定义)。 • 对于EC2 实例类型，请根据将备份文件传输到横向扩展备份存储库的存档层的速度和成本要求，选择代理设备的实例类型。 • 对于 Amazon VPC，请选择目标实例的 VPC。 • 对于子网，请选择代理设备的子网。 • 对于安全组，请选择与选项关联的 VPC 安全组。 • 对于重定向器端口，请指定用于在代理设备和备份	

Task	描述	所需技能
	基础设施组件之间执行路由请求的 TCP 端口。 • 选择 确认 确认设置。 7. 在向导的摘要步骤中，查看配置信息，然后选择完成。	

添加横向扩展备份存储库

Task	描述	所需技能
启动新横向扩展备份存储库向导。	1. 在 Veeam 控制台上，打开备份基础设施视图。 2. 在库存窗格，选择横向扩展存储库，然后选择添加横向扩展存储库。	应用程序所有者、AWS 系统管理员
添加横向扩展备份存储库，并配置容量和存档层。	1. 在向导的命名步骤中，指定扩展备份存储库的名称和简要描述。 2. 如果需要，则添加性能范围。您也可以使用现有的 Veeam 本地备份存储库作为性能等级。 从 Veeam 版本 12 开始，您可以绕过本地性能层添加 S3 存储桶作为 direct-to-object (DTO) 备份的性能范围。 3. 选择高级，然后为横向扩展备份存储库指定其他选项。 • 选择使用每台计算机的备份文件，以为每台计算机创建单独的备份文件，并将这些文件同时以多个流	应用程序所有者、AWS 系统管理员

Task	描述	所需技能
	的形式写入备份存储库。建议使用此选项改善存储和计算资源利用率。 • 选择在所需数据区处于脱机状态时执行完全备份以创建完整备份文件，以防包含增量备份还原点的数据区脱机。此选项需要通过横向扩展备份存储库中的可用空间托管完整备份文件。 4. 在向导的策略步骤中，为存储库指定备份放置策略。 • 选择数据局部性，以将属于同一链的完整备份文件和增量备份文件存储至相同的性能范围。您可以将属于新备份链的文件按相同的性能范围或不同的性能范围存储(除非您以重复存储设备作为性能范围)。 • 选择性能以将完整备份文件和增量备份文件按不同性能范围存储。此选项需要快速可靠的网络连接。如果选择性能，则可在每个性能范围内限制要存储的备份文件类型。例如，您可以在一个数据区存储完全备份文件，并在其他数据区上存储增量备份文件。若要选择文件类型： • 选择 Customize (自定义)。	

Task	描述	所需技能
	• 在备份放置设置对话框中，选择性能范围，然后选择编辑。 • 选择要在此范围内存储的备份文件类型。 5. 在向导的容量层步骤中，配置要连接至横向扩展备份存储库的长期存储层。 • 选择使用对象存储扩展横向扩展备份存储库容量。对于对象存储库，请为您在上一操作中添加的容量层选择 Amazon S3 Glacier 存储。 • 选择窗口 以选择用于移动或复制数据的时间窗口。 • 选择创建备份后立即将其复制到对象存储，以将所有或仅最近创建的备份文件复制到容量范围。 • 选择在备份过期时将其移至对象存储，以将非活动备份链转移至容量范围。在移动 X 天以上的备份文件字段中，指定卸载备份文件的持续时间。（如果要在非活动备份链创建之日即卸载，请指定 0 天。）如果横向扩展备份存储库已达到指定阈值，您还可以选择覆盖以更快地移动备份文件。	

Task	描述	所需技能
	• 选择加密上传至对象存储的数据，并指定密码来加密所有数据及其元数据，以进行分载。选择添加或管理密码，以指定新密码。 6. 在向导的存档层步骤中，配置要连接至横向扩展备份存储库的长期存档层。（如果您跳过添加 Amazon S3 Glacier 存储，则不会出现此步骤。） • 选择将 GFS 完整备份存档至对象存储。对于对象存储库，请选择您在上一操作中添加的 Amazon S3 Glacier 存储。 • 对于超过 N 天的存档 GFS 备份，请选择将文件移至存档区的时间窗口。（如果要在非活动备份链创建之日即存档，则指定 0 天。） 7. 在向导的摘要步骤中，查看横向扩展备份储存库的配置信息，然后选择完成。	

相关资源

- 在你的 AWS 账户（IAM 文档）中创建 IAM 用户
- 创建存储桶（Amazon S3 文档）
- 阻止对 Amazon S3 存储的公有访问（Amazon S3 文档）
- 使用 S3 对象锁定（Amazon S3 文档）

- Veeam 技术文档
- 如何为 S3 对象存储连接创建安全 IAM policy（Veeam 文档）

其他信息

以下各节提供 IAM policy 示例，您可以在此模式的操作说明部分创建 IAM 用户时使用。

容量层的 IAM policy

> **Note**
>
> 将示例策略中 S3 存储桶的名称从更改<yourbucketname>为要用于 Veeam 容量层备份的 S3 存储桶的名称。另请注意，该策略应仅限于 Veeam 使用的特定资源（如以下政策中的Resource规范所示），并且该策略的第一部分禁用客户端加密，如 AWS 博客文章《防止意外加密 Amazon S3 对象》中所述。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RestrictSSECObjectUploads",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::<your-bucket-name>/*",
            "Condition": {
                "Null": {
                    "s3:x-amz-server-side-encryption-customer-algorithm": "false"
                }
            }
        },
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:GetObjectVersion",
                "s3:ListBucketVersions",
                "s3:ListBucket",
                "s3:PutObjectLegalHold",
                "s3:GetBucketVersioning",
                "s3:GetObjectLegalHold",
```

```
                "s3:GetBucketObjectLockConfiguration",
                "s3:PutObject*",
                "s3:GetObject*",
                "s3:GetEncryptionConfiguration",
                "s3:PutObjectRetention",
                "s3:PutBucketObjectLockConfiguration",
                "s3:DeleteObject*",
                "s3:DeleteObjectVersion",
                "s3:GetBucketLocation"

            ],
            "Resource": [
                "arn:aws:s3:::<yourbucketname>",
                "arn:aws:s3:::<yourbucketname>/*"
            ]
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets",
                "s3:ListBucket"
            ],
            "Resource": "arn:aws:s3:::*"
        }
    ]
}
```

存档层的 IAM policy

> **Note**
>
> 将示例策略中 S3 存储桶的名称从更改<yourbucketname>为要用于 Veeam 存档层备份的 S3 存储桶的名称。

若要使用现有 VPC、子网以及安全组：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "S3Permissions",
            "Effect": "Allow",
            "Action": [
                "s3:DeleteObject",
                "s3:PutObject",
                "s3:GetObject",
                "s3:RestoreObject",
                "s3:ListBucket",
                "s3:AbortMultipartUpload",
                "s3:GetBucketVersioning",
                "s3:ListAllMyBuckets",
                "s3:GetBucketLocation",
                "s3:GetBucketObjectLockConfiguration",
                "s3:PutObjectRetention",
                "s3:GetObjectVersion",
                "s3:PutObjectLegalHold",
                "s3:GetObjectRetention",
                "s3:DeleteObjectVersion",
                "s3:ListBucketVersions"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket-name>",
                "arn:aws:s3:::<bucket-name>/*"
            ]
        }
    ]
}

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:CreateKeyPair",
                "ec2:DescribeKeyPairs",
                "ec2:RunInstances",
                "ec2:DeleteKeyPair",
                "ec2:DescribeVpcAttribute",
                "ec2:CreateTags",
                "ec2:DescribeSubnets",
                "ec2:TerminateInstances",
```

```
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeImages",
                "ec2:DescribeVpcs"
            ],
            "Resource": "arn:aws:ec2:<region>:<account-id>:*"
        }
    ]
}
```

若要创建新 VPC、子网和安全组：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3Permissions",
            "Effect": "Allow",
            "Action": [
                "s3:DeleteObject",
                "s3:PutObject",
                "s3:GetObject",
                "s3:RestoreObject",
                "s3:ListBucket",
                "s3:AbortMultipartUpload",
                "s3:GetBucketVersioning",
                "s3:ListAllMyBuckets",
                "s3:GetBucketLocation",
                "s3:GetBucketObjectLockConfiguration",
                "s3:PutObjectRetention",
                "s3:GetObjectVersion",
                "s3:PutObjectLegalHold",
                "s3:GetObjectRetention",
                "s3:DeleteObjectVersion",
                "s3:ListBucketVersions"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket-name>",
                "arn:aws:s3:::<bucket-name>/*"
            ]
        }
    ]
}
```

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:CreateKeyPair",
                "ec2:DescribeKeyPairs",
                "ec2:RunInstances",
                "ec2:DeleteKeyPair",
                "ec2:DescribeVpcAttribute",
                "ec2:CreateTags",
                "ec2:DescribeSubnets",
                "ec2:TerminateInstances",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeImages",
                "ec2:DescribeVpcs",
                "ec2:CreateVpc",
                "ec2:CreateSubnet",
                "ec2:DescribeAvailabilityZones",
                "ec2:CreateRoute",
                "ec2:CreateInternetGateway",
                "ec2:AttachInternetGateway",
                "ec2:ModifyVpcAttribute",
                "ec2:CreateSecurityGroup",
                "ec2:DeleteSecurityGroup",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:AuthorizeSecurityGroupEgress",
                "ec2:DescribeRouteTables",
                "ec2:DescribeInstanceTypes"
            ],
            "Resource": "*"
        }
    ]
}
```

在 AWS 上配置 Veritas NetBackup VMware 云版

由 Shubham Salani (AWS) 创建

摘要

注意：自 2024 年 4 月 30 日起，AWS 或其渠道合作伙伴不再转售 VMware Cloud on AWS。该服务将继续通过博通提供。我们鼓励您联系您的 AWS 代表了解详情。

许多企业使用 Veritas NetBackup 作为其基于 vSphere 的本地 VMware vSphere 工作负载的备份和恢复解决方案。一旦企业将其工作负载迁移到 Amazon Web Services (AWS) VMware 云基础设施中的软件定义数据中心 (SDDCs)，就没有明确的 lift-and-shift集成程序。 NetBackup此模式描述了如何在您的 AWS 账户 NetBackup 中设置 Veritas 并将其配置为备份您的 VMware SDDCs账户中的工作负载。

此模式不包括迁移工作负载的说明。有关更多信息，请参阅使用 VMware HCX 将 VMware SDDC 迁移到 AWS 上的 VMware 云端。将工作负载设置为 VMware Cloud on AWS 时，请使用延伸集群（VMware 文档）。在此配置中，集群跨越单个区域内的两个 AWS 可用区。这可在其中一个可用区不可用时提供高可用性和弹性。Elastic DRS 和 vSAN 见证主机（VMware 文档）将数据无缝复制到第三个可用区（称为故障域）。这种奇偶校验解决方案可以帮助您在发生故障时恢复数据。由于此方法需要三个可用区，因此在为您的 VMware 云环境选择 AWS 区域时，请确保该区域有三个或更多可用区。有关更多信息，请参阅区域和可用区。

在这种模式下，每个 SDDC 都有一个备份主机，即代理服务器。使用亚马逊弹性计算云 (Amazon EC2) 实例，您可以在单独的虚拟私有云 (VPC) 中设置 NetBackup 主服务器和媒体服务器，每个 SDDC 各一个。由于弹性网络接口提供高带宽和低延迟，因此您可以使用它们来配置备份主机与其相应 NetBackup 的主服务器和媒体服务器之间的连接。这些 EC2 实例将备份定向到亚马逊弹性块存储 (Amazon EBS) 卷，这是备份的第一点。您可以使用 AWS DataSync 保持 EBS 卷的 SDDCs 同步。

您还可以使用 AWS Transit Gateway 和 接口 VPC 端点将 EBS 卷连接到其他存储服务，例如 Amazon Simple Storage Service（Amazon S3）。根据保留策略，您可以使用 S3 智能分层 S3 Glacier 存储类来优化存储成本。有关更多信息，请参阅使用 Amazon S3 存储类（Amazon S3 文档）。

先决条件和限制

先决条件

- 您的 VMware Cloud on AWS 环境使用跨越两个可用区的延伸集群。

- 备份主机必须位于 VMware 云端 AWS SDDC 上，该云端可以访问部署 VMware 虚拟机磁盘文件 (VMDK) 文件的数据存储。
- HotAdd 必须在 NetBackup 客户端上启用传输模式才能备份和恢复虚拟机 (VMs)，并且必须允许从用户指导的文件和文件夹中进行恢复。

限制

- NetBackup 主服务器必须使用 DNS 解析来解析到 SDDC 中 vCenter 备份主机的私有 IP 地址。
- 主服务器和备份 NetBackup 主机上的主机文件应包含以下内容：
  - 主服务器的私有 IP 地址和私有 DNS 名称
  - 备份主机的私有 IP 地址和私有 DNS 名称
- 如果您要将接口 VPC 端点配置到 S3 存储桶，则必须将 SDDC 计算网关防火墙配置为允许来自无类别域间路由 (CIDR) 区块源的 HTTPS。有关更多信息，请参阅使用 S3 端点访问 S3 存储桶（VMware 文档）。
- VMware Cloud on AWS 不支持以下功能 NetBackup：
  - 备份或恢复虚拟机模板
  - 使用 NetBackup vSphere 客户端（插件）HTML5
  - VMs 用于备份或恢复的锁定和解锁
  - 备份无法存储在 vSAN 数据存储中
  - 网络块设备 (NBD)、NBDSSL 和 SAN 传输模式

产品版本

- VMware AWS 上的 Cloud SDDC 版本 1.0 或更高版本
- Veritas 8.1.2 或 NetBackup 更高版本
- Linux 版本 6.8 或更高版本
- VMware vSphere 版本 6.0 或更高版本

架构

下图显示了 VMware Cloud on AWS 的 NetBackup 配置。 NetBackup 主服务器和媒体服务器部署在单独的 VPC 中，并 SDDCs 通过弹性网络接口连接到备份主机。 NetBackup 主服务器和媒体服务器将备份存储在 Amazon EBS 卷中。您可以选择使用 AWS Transit Gateway 和 AWS PrivateLink 接口 VPC 终端节点在 Amazon S3 存储桶中配置额外的存储空间。

工具

Amazon Web Services 和工具

- 亚马逊弹性区块存储 (Amazon EBS) E lastic Block Store 提供块级存储卷，用于亚马逊弹性计算云 (Amazon) 实例。 EC2
- AWS PrivateLink 可帮助您创建从您的虚拟私有云 (VPCs) 到 VPC 外部服务的单向私有连接。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。

其他服务

- VMware Cloud on AWS 是由亚马逊网络服务 (AWS) 和联合开发的集成云产品 VMware。
- NetBackup 用于 VMware备份和恢复在 VMware ESXi 主机上运行的 VMware 虚拟机。

操作说明

配置 NetBackup 服务器

Task	描述	所需技能
更新防火墙规则。	更新防火墙规则，在 AWS SDDC 上的 VMware 云与 NetBackup 主服务器和媒体服务器之间建立连接。执行以下操作： 1. 登录 AWS 上的 VMware Cloud，网址为 https://vmc.vmware.com/ 2. 在网络和安全选项卡上，选择网关防火墙。	网络管理员、云管理员

Task	描述	所需技能
	3. 在网关防火墙页面上，选择计算网关。 4. 选择添加规则，然后使用必要的防火墙端口设置创建新规则。有关更多信息，请参阅NetBackup 防火墙端口要求（Veritas 文档）。	
启动 NetBackup 主服务器和媒体服务器。	1. 登录 AWS 管理控制台，然后通过以下网址打开 Amazon EC2 控制台 https://console.aws.amazon.com/ec2/ 2. 启动 EC2 实例（Amazon EC2 文档），并使用以下配置详细信息： a. 对于 NetBackup 主服务器和媒体服务器，请选择 `NBU-Linux-GA-8-1-2-Setup-f032d23e-881b-4dee-ba70-b9ca3e915910-ami-072509a7ffc156938.4` Amazon 系统映像 (AMI)。此预配置的 AMI 可通过 AWS Marketplace 获得。 b. 选择实例类型。NetBackup 推荐`m5.2xlarge` 用于主服务器和媒体服务器。	云管理员、备份管理员

Task	描述	所需技能
为配置备份主机 NetBackup。	1. 登录 AWS 上的 VMware Cloud，网址为 https://vmc.vmware.com/ 2. 选择 SDDC。 3. 选择打开 VCENTER 选项卡。这将打开 SDDC vCenter。 4. 记下备份主机的完全限定域名（FQDN）。 5. 登录到 NetBackup 管理控制台。有关更多信息，请参阅登录 NetBackup 管理控制台（Veritas 文档）。 6. 选择主服务器和媒体服务器，然后选择 A VMware ccess Hosts。 7. 添加备份主机的 FQDN。 8. 选择应用，然后选择确定。	云管理员、备份管理员

（可选）设置 Amazon S3 存储

Task	描述	所需技能
在 Amazon S3 中配置存储。	1. 查看 Amazon S3 云存储选项（Veritas 文档），然后根据要求选择合适的存储类别。 2. NetBackup 按照配置云存储中的说明 NetBackup（Veritas 文档），配置为使用 Amazon S3 进行云存储。	云管理员、常规 AWS

相关资源

AWS 文档

- 创建接口 VPC 终端节点（AWS PrivateLink 文档）

Veritas 文档

- NetBackup 防火墙端口要求

VMware 文档

- 从内容库中的 OVF 模板部署虚拟机
- VMware Cloud on AWS 数据传输费用：它是如何运作的？（VMware 博客文章）
- VMware AWS 上的云：延伸集群

使用 AWS CLI 将数据从 S3 存储桶复制到其他账户和区域

由 Appasaheb Bagali (AWS) 和 Purushotham G K (AWS) 编写

摘要

此示例介绍了以下操作：将数据从 AWS 源账户的 Amazon Simple Storage Service (Amazon S3) 存储桶迁移至其他 Amazon Web Services account（相同或不同 AWS 区域中）中的目标 S3 存储桶。

源 S3 存储桶通过使用附加资源策略，可允许 AWS Identity and Access Management (IAM) 访问。目标账户中的用户必须代入拥有源存储桶 `PutObject` 和 `GetObject` 权限的角色。最后，运行 `copy` 和 `sync` 命令，将数据从源 S3 存储桶传输至目标 S3 存储桶。

账户拥有他们上传至 S3 存储桶的对象。如果您跨账户和地区复制对象，则将复制对象的权限授予目标账户。若要更改对象所有权，您可将其访问控制列表（ACL）更改为 `bucket-owner-full-control`。但是，我们建议您向目标账户授予编程跨账户权限，因为 ACLs 可能很难管理多个对象。

> **Warning**
>
> 这种情况需要具有编程访问权限和长期证书的 IAM 用户，这会带来安全风险。为帮助减轻这种风险，我们建议仅向这些用户提供执行任务所需的权限，并在不再需要这些用户时将其移除。必要时可以更新访问密钥。有关更多信息，请参阅《IAM 用户指南》中的更新访问密钥。

此模式涵盖一次性迁移。对于需要将新对象从源存储桶持续自动迁移到目标存储桶的场景，您可以改用 S3 Batch Replication，如使用 S3 Batch Replic ation 将数据从 S3 存储桶复制到另一个账户和区域模式中所述。

先决条件和限制

- 位于相同或跨不同 AWS 区域中的两个活跃 Amazon Web Services account。
- 源账户中的现有 S3 存储桶。
- 如果您的源存储桶或目标 Amazon S3 存储桶启用了默认加密，则必须修改 AWS Key Management Service (AWS KMS) 密钥权限。有关更多信息，请参阅有关此主题的AWS re:Post 文章。
- 熟悉跨账户权限。

架构

工具

- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。

最佳实践

- IAM 中的安全最佳实践（IAM 文档）
- 应用最低权限权限（IAM 文档）

操作说明

在目标 Amazon Web Services account 中创建 IAM 用户与角色

Task	描述	所需技能
创建 IAM 用户并获取访问密钥。	1. 登录 AWS 管理控制台，创建具有编程访问权限的 IAM 用户。有关详细步骤，请参阅 IAM 文档中的创建 IAM 用户。无需为此用户附加任何策略。 2. 为该用户生成访问密钥与私有密钥。有关说明，请参阅 AWS 文档中的 Amazon Web Services account 和访问密钥。	AWS DevOps

Task	描述	所需技能
创建基于 IAM 身份的策略。	使用以下权限，创建名为 `S3MigrationPolicy` 的基于 IAM 身份的策略。有关详细步骤，请参阅 IAM 文档中的创建 IAM policy。 `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "s3:ListBucket", "s3:GetObject", "s3:GetObjectTagging", "s3:GetObjectVersion", "s3:GetObjectVersionTagging" ], "Resource": [ "arn:aws:s3:::amazon-s3-demo-source-bucket", "arn:aws:s3:::amazon-s3-demo-source-bucket/*" ] },`	AWS DevOps

Task	描述	所需技能

	{
	"Effect": "Allow",
	"Action": [
	"s3:ListBucket",
	"s3:PutObject",
	"s3:PutObjectAcl",
	"s3:PutObjectTagging",
	"s3:GetObjectTagging",
	"s3:GetObjectVersion",
	"s3:GetObjectVersionTagging"
	],
	"Resource": [
	"arn:aws:s3:::amazon-s3-demo-destination-bucket",
	"arn:aws:s3:::amazon-s3-demo-destination-bucket/*"
	]
	}
	]
	}	

Task	描述	所需技能
	Note 根据您的用例修改源存储桶和目标存储桶名称。 这种基于身份的策略允许此角色用户访问源存储桶和目标存储桶。	

Task	描述	所需技能
创建一个 IAM 角色。	使用以下信任策略，创建名为 `S3MigrationRole` 的IAM 角色，然后附加之前创建的 `S3MigrationPolicy` 。有关详细步骤，请参阅 IAM 中的创建向 IAM 用户委派权限的角色。 `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "AWS": "arn:aws:iam::<destination_account>:user/<user_name>" }, "Action": "sts:AssumeRole", "Condition": {} } ] }` **Note** 根据您的用例，修改信任策略中目标 IAM 角色或用户名的亚马逊资源名称 (ARN)。	AWS DevOps

Task	描述	所需技能
	此信任策略允许新创建 IAM 用户代入 S3MigrationRole 。	

在源账户中创建 S3 存储桶策略，并附加 S3 存储桶策略

Task	描述	所需技能
创建并附加 S3 存储桶策略。	登录源账户的 AWS 管理控制台 并打开 Amazon S3 控制台。选择源 S3 存储桶，然后选择权限。在存储桶策略下，选择编辑，然后粘贴以下存储桶策略。选择保存。 { "Version": "2012-10-17", "Statement": [{ "Sid": "DelegateS3Access", "Effect": "Allow", "Principal": {"AWS": "arn:aws:iam::<destination_account>:role/<RoleName>"}, "Action": ["s3:ListBucket", "s3:GetObject", "s3:GetObjectTagging",	云管理员

Task	描述	所需技能

```
  "s3:GetObjectVersi
on",

  "s3:GetObjectVersi
onTagging"

 ],
           "Resource":
 [

 "arn:aws:s3:::amazon-
s3-demo-source-bucket/
*",

 "arn:aws:s3:::amazon-
s3-demo-source-bucket"
           ]
        }
    ]
}
```

> **Note**
>
> 请确保包含目标账户的 AWS 账户 ID，并根据您的要求配置存储桶策略模板。

此基于资源的策略允许目标角色 `S3MigrationRole` 访问源账户中的 S3 对象。

配置目标 S3 存储桶

Task	描述	所需技能
创建目标 S3 存储桶。	登录您的目标账户的 AWS 管理控制台，打开 Amazon S3 控制台，然后选择创建存储桶。根据您的要求创建 S3 存储桶。有关更多信息，请参阅 Amazon S3 文档中的创建存储桶。	云管理员

将数据复制至目标 S3 存储桶

Task	描述	所需技能
使用新创建的用户凭证配置 AWS CLI。	1. 安装最新版本 AWS CLI。有关说明，请参阅 AWS CLI 文档中安装或更新最新版本的 AWS CLI。 2. 使用您创建的用户的 AWS 访问密钥运行 `$ aws configure` 并更新 CLI。有关更多信息，请参阅 AWS CLI 文档中的《用户指南》中的配置和凭证文件设置。	AWS DevOps
代入 S3 迁移角色。	1. 使用 AWS CLI 代入 `S3MigrationRole` ： `aws sts assume-role \ --role-arn "arn:aws:iam::<destination_account>:`	AWS 管理员

Task	描述	所需技能
	role/S3MigrationRo le" \ --role-session- name AWSCLI-Session 此命令输出多条信息。在凭证数据块中，您需要 AccessKey Id 、SecretAcc essKey 和 SessionTo ken 。此示例使用环境变量 RoleAcces sKeyID 、RoleSecre tKey 和 RoleSessi onToken 。请注意，过期字段的时间戳参见 UTC 时区。时间戳指示 IAM 角色的临时凭证的过期时间。如果临时凭证过期，则必须再次调用 sts:AssumeRole API。 2. 创建三个环境变量以代入 IAM 角色。这些环境变量通过以下输出填充： # Linux export AWS_ACCES S_KEY_ID=RoleAcces sKeyID export AWS_SECRE T_ACCESS_KEY=RoleS ecretKey export AWS_SESSI ON_TOKEN=RoleSessi onToken # Windows	

Task	描述	所需技能

	`set AWS_ACCES S_KEY_ID=RoleAcces sKeyID`
	`set AWS_SECRE T_ACCESS_KEY=RoleS ecretKey`
	`set AWS_SESSI ON_TOKEN=RoleSessi onToken`
	3. 通过运行以下命令，验证自己是否代入了 IAM 角色：
	`aws sts get-caller-identity`
	有关更多信息，请参阅 AWS Knowledge Center。	

Task	描述	所需技能

将数据从源 S3 存储桶复制并同步至目标 S3 存储桶。	当您代入 S3MigrationRole 角色后，您可使用复制 (cp) 或同步 (sync) 命令复制数据。
	
	复制（详情请参阅 AWS CLI 命令参考）：
	
	`aws s3 cp s3://amazon-s3-demo-source-bucket/ \ s3://amazon-s3-demo-destination-bucket/ \ --recursive --source-region SOURCE-REGION-NAME --region DESTINATION-REGION-NAME`
	
	同步（详情请参阅 AWS CLI 命令参考）：
	
	`aws s3 sync s3://amazon-s3-demo-source-bucket/ \ s3://amazon-s3-demo-destination-bucket/ \ --source-region SOURCE-REGION-NAME --region DESTINATION-REGION-NAME`
		云管理员

故障排除

事务	解决方案
调用 `ListObjects` 操作时发生错误 (`AccessDenied`)：访问被拒绝	• 确保您已担任该角色`S3MigrationRole`。 • 运行 `aws sts get-caller-identity` 以检查使用的角色。如果输出未显示 `S3MigrationRole` 的 ARN，请再次代入该角色并重试。

相关资源

- 创建 S3 存储桶（Amazon S3 文档）
- Amazon S3 存储桶策略与用户策略（Amazon S3 文档）
- IAM 身份（用户、组和角色）（IAM 文档）
- cp 命令（AWS CLI 文档）
- sync 命令（AWS CLI 文档）

使用 S3 Batch Replication 将数据从 S3 存储桶复制到另一个账户和区域

由 Appasaheb Bagali (AWS)、Lakshmikanth B D (AWS)、Purushotham G K (AWS)、Shubham Harsora (AWS) 和 Suman Rajotia (AWS) 创建

摘要

此模式说明了在设置存储桶后，如何使用亚马逊简单存储服务 (Amazon S3) Simple Storage Batch Replication 将一个 S3 存储桶的内容自动复制到另一个 S3 存储桶，无需任何手动干预。源存储桶和目标存储桶可以位于相同 AWS 账户 或不同区域中。

S3 Batch Replication 为您提供了一种复制配置到位之前存在的 Amazon S3 对象、之前复制过的对象以及复制失败的对象的方法。此方法使用 S3 Batch Operations 作业。作业完成后，您将收到一份完成报告。

在需要将新对象从源存储桶持续自动迁移到目标存储桶的场景中，您可以使用 S3 Batch Replication。对于一次性迁移，您可以改用 AWS Command Line Interface (AWS CLI)，如模式中所述，使用将数据从 S3 存储桶复制到另一个账户和区域 AWS CLI。

先决条件和限制

- 消息来源 AWS 账户。
- 目的地 AWS 账户。
- 源账户中包含几个对象（文件或文件夹）的 S3 存储桶。
- 目标账户中的一个或多个 S3 存储桶。
- 在源@@ 存储桶和目标存储桶上启用了 S3 版本控制。
- AWS Identity and Access Management (IAM) 在源账户和目标账户上创建 IAM 策略、IAM 角色和 S3 存储桶策略的权限。
- 当 S3 批量复制任务处于活动状态时，Amazon S3 生命周期规则被禁用。这样可以确保源存储桶和目标存储桶之间的均衡性。否则，目标存储桶可能不是源存储桶的精确副本。

架构

工具

AWS 服务

- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

最佳实践

以下来自 re AWS : Invent 2022 的视频讨论了使用 Amazon S3 复制实现监管合规、数据保护和提高应用程序性能的最佳实践。

https://www.youtube-nocookie.com/embed/hr JEb ISBL04？ 控件=0

操作说明

为源账户中的跨账户复制创建 IAM 策略和角色

Task	描述	所需技能
为跨账户复制创建 IAM 策略。	在 AWS 来源账户中： 1. 打开 IAM 管理控制台。 2. 创建新的 IAM 策略。 3. 在策略编辑器部分，选择 JSON，然后粘贴以下代码。 `{ "Version": "2012-10-17", "Statement": [ { "Sid": "GetSourceBucketConfiguration", "Effect": "Allow", "Action": [ "s3:ListBucket",`	云管理员、AWS 管理员

Task	描述	所需技能
	"s3:GetBucketLocation", "s3:GetBucketAcl", "s3:GetReplicationConfiguration", "s3:GetObjectVersionForReplication", "s3:GetObjectVersionAcl", "s3:GetObjectVersionTagging"], "Resource": ["arn:aws:s3:::source-bucket-name", "arn:aws:s3:::source-bucket-name/*"] }, { "Sid": "ReplicateToDestinationBuckets", "Effect": "Allow", "Action": ["s3:List*", "s3:*Object",	

Task	描述	所需技能
	"s3:ReplicateObject", "s3:ReplicateDelete", "s3:ReplicateTags"], "Resource": ["arn:aws:s3:::destination-bucket-name*", "arn:aws:s3:::destination-bucket-name/*"] }, { "Sid": "PermissionToOverrideBucketOwner", "Effect": "Allow", "Action": ["s3:ObjectOwnerOverrideToBucketOwner"], "Resource": ["arn:aws:s3:::destination-bucket-name*", "arn:aws:s3:::dest	

Task	描述	所需技能

	<pre>ination-bucket-nam
	e/*"
	]
	}
	]
	}</pre>
	本政策包括三项声明：
	• GetSourceBucketConfiguration 提供对复制配置和对象版本的访问权限，以便在源存储桶上进行复制。
	• ReplicateToDestinationBuckets 提供复制到目标存储桶的权限。您可以在数组中指定多个目标存储桶。
	• PermissionToOverrideBucketOwner 提供对的访问权限，ObjectOwnerOverrideToBucketOwner 以便目标存储桶可以拥有目标账户中从源账户复制的对象。
	4. 选择“下一步”，提供策略名称（如）cross-account-bucket-replication-policy ，然后选择“创建策略”。	

Task	描述	所需技能
	有关更多信息，请参阅 IAM 文档中的创建 IAM 策略。	
为跨账户复制创建 IAM 角色。	在 AWS 来源账户中： 1. 在 IAM 控制台上，使用以下信息创建一个 IAM 角色： a. 在可信实体类型中选择 Amazon Web Services。 b. 要获得服务，请选择 S3。 c. 对于用例，请选择 S3 Batch Operation s。 d. 选择您在上一步中创建的策略。 2. 提供角色名称，例如 cross-account-bucket-replication-role，然后选择创建角色。 有关更多信息，请参阅 IAM 文档中的创建 IAM 角色。	云管理员、AWS 管理员

在源账户中创建复制规则

Task	描述	所需技能
针对源账户中的源存储桶创建复制规则。	在 AWS 来源账户中： 1. 打开 Amazon S3 控制台。 2. 导航到源存储桶，然后选择“管理”选项卡。 3. 使用以下配置创建复制规则：	AWS 管理员、云管理员

Task	描述	所需技能
	a. 提供规则名称，例如`s3-replication-rule` 。 b. 对于 Status（状态），选择 Enabled（已启用）。 c. 对于规则范围，请选择应用于存储桶中的所有对象。 d. 对于目标，选择在其他账户中指定存储桶，然后输入目标 AWS 账户 编号和存储桶名称。 e. 选择将对象所有权更改为目标存储桶所有者的选项。 f. 对于 IAM 角色，请选择您之前在源账户中创建的角色。 g. 对于其他复制选项，请选择所有可用选项。它们提供了快速复制内容、通过 Amazon CloudWatch 指标监控复制进度、复制删除标记和复制元数据更改的能力。 h. 选择保存。 4. 如果您有多个目标存储桶，请创建其他复制规则。 有关更多信息，请参阅 Amazon S3 文档中的在源存储	

Task	描述	所需技能
	桶和目标存储桶由不同账户拥有时配置复制。	

将存储桶策略应用于目标存储桶

Task	描述	所需技能
将存储桶策略应用于目标存储桶。	必须对目标账户中的 AWS 每个目标存储桶单独执行此步骤。 在 AWS 目标账户中： 1. 打开 Amazon S3 控制台，导航到目标存储桶，然后选择 “权限” 选项卡。 2. 通过提供以下 JSON 代码编辑存储桶策略，然后保存策略： `{ "Version": "2012-10-17", "Id": "PolicyForDestinationBucket", "Statement": [ { "Sid": "Permissions on objects and buckets", "Effect": "Allow", "Principal": { "AWS": "arn:aws:iam::SourceAWSAccountNumber`	AWS 管理员、AWS 系统管理员、云管理员

Task	描述	所需技能

	:role/IAM-Role-cre
	ated-in-step1-in-s
	ource-account"
	},
	"Action": [
	
	"s3:List*",
	
	"s3:GetBucketVersi
	oning",
	
	"s3:PutBucketVersi
	oning",
	
	"s3:ReplicateDelete",
	
	"s3:ReplicateObject"
	],
	"Resource":
	[
	
	"arn:aws:s3:::dest
	ination-bucket",
	
	"arn:aws:s3:::dest
	ination-bucket/*"
	]
	},
	{
	"Sid":
	"Permission to
	override bucket owner",
	"Effect":
	"Allow",
	"Principa
	l": {
	"AWS":
	"arn:aws:iam::Sour
	ceAWSAccountNumber
	:role/IAM-Role-cre
	ated-in-step1-in-s
	ource-account"	

Task	描述	所需技能

	<pre> },
	            "Action":
	"s3:ObjectOwnerOve
	rrideToBucketOwner",
	            "Resource
	": "arn:aws:s3:::dest
	ination-bucket/*"
	        }
	    ]
	}</pre>
	本政策包括两项声明：
	• Permissions on objects and buckets表示目标存储桶可以根据源账户中定义的角色复制内容。该角色提供对源存储桶的权限。
	• Permission to override bucket owner表示目标存储桶有权覆盖源账户的所有权。	

测试 Amazon S3 跨账户复制

Task	描述	所需技能
验证复制工作是否正常。	1. 向源存储桶添加对象。 2. 验证新对象是否出现在目标账户的 S3 存储桶中。 3. 查看 CloudWatch 指标： a. 在源存储桶中，选择 “指标” 选项卡。	AWS 管理员、云管理员

Task	描述	所需技能
	b. 在 “复制指标” 部分，选择复制规则。 c. 请选择 Display charts (显示图表)。这些图表通过显示待复制的操作、复制延迟和待复制的字节来反映复制的状态。 有关更多信息，请参阅 Amazon S3 文档 CloudWatch 中的使用亚马逊监控指标。	

相关资源

- 我什么时候使用 IAM？（IAM 文档）
- IAM 的工作原理（IAM 文档）
- 创建 IAM 角色（IAM 文档）
- 创建 IAM policy（IAM 文档）
- 访问管理概述：权限和策略（IAM 文档）
- 创建、配置和使用亚马逊 S3 存储桶（亚马逊 S3 文档）
- 在 Amazon S3 中上传、下载和处理对象（亚马逊 S3 文档）
- 复制对象（Amazon S3 文档）

使用 PrivateLink 适用于 Amazon S3 的 DistCp AWS 将数据从本地 Hadoop 环境迁移到 Amazon S3

由杰森·欧文斯（AWS）、安德烈斯·坎托（AWS）、杰夫·克洛普芬斯坦（AWS）、布鲁诺·罗查·奥利维拉（AWS）和塞缪尔·施密特（AWS）创作

摘要

此模式演示了如何使用适用于亚马逊简单存储服务 (Amazon S3) 的 Apache 开源工具，将几乎任意数量的数据从本地 Apache Hadoop 环境迁移到 PrivateLink 亚马逊网络服务 (DistCpAWS) 云。您可以使用 AWS PrivateLink for Amazon S3 通过本地数据中心和亚马逊虚拟私有云 (Amazon VPC) 之间的私有网络连接将数据迁移到 Amazon S3，而不必使用公共互联网或代理解决方案迁移数据。如果您在 Amazon Route 53 中使用 DNS 条目或在本地 Hadoop 集群的所有节点的 /etc/hosts 文件中添加条目，则系统会自动将您定向到正确的接口端点。

本指南提供了使用 DistCp 将数据迁移到 AWS 云的说明。 DistCp 是最常用的工具，但还有其他迁移工具可用。例如，您可以使用离线 AWS 工具，例如 AWS Snowb all 或 AWS Snowmobile，也可以使用 AWS St orage Gateway 或 AWS 等在线 AWS 工具。 DataSync此外，你可以使用其他开源工具，比如 Apache NiFi。

先决条件和限制

先决条件

- 在您的本地数据中心与 Amazon Web Services Cloud 之间建立私网连接的有效的 Amazon Web Services account
- Hadoop，安装在本地使用 DistCp
- 有权访问 Hadoop Distributed File System (HDFS) 中的迁移数据的 Hadoop 用户
- AWS 命令行界面（AWS CLI），已安装并配置
- 用于将对象放入 S3 存储桶的权限

限制

虚拟私有云 (VPC) 限制适用于 AWS f PrivateLink or Amazon S3。有关更多信息，请参阅接口终端节点属性和限制以及 AWS PrivateLink 配额（AWS PrivateLink 文档）。

AWS PrivateLink for Amazon S3 不支持以下内容：

- 美国联邦信息处理标准 (FIPS) 端点
- 网站端点
- 传统全局端点

架构

源技术堆栈

- 已安装的 Hadoop 集群 DistCp

目标技术堆栈

- Amazon S3
- Amazon VPC

目标架构

该图显示了 Hadoop 管理员如何使用 DistCp 私有网络连接（例如 AWS Direct Connect）将数据从本地环境复制到 Amazon S3，通过 Amazon S3 接口终端节点将数据复制到亚马逊 S3。

工具

Amazon Web Services

- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。

其他工具

- Apache Hadoop DistCp（分布式副本）是一种用于复制大型集群间和集群内部的工具。 DistCp 使用 Apache MapReduce 进行分发、错误处理和恢复以及报告。

操作说明

将数据迁移到 AWS Cloud

Task	描述	所需技能
为 AWS 创建适用于 Amazon S3 PrivateLink 的终端节点。	1. 登录 AWS 管理控制台，打开 Amazon VPC 控制台。 2. 在导航窗格中，选择端点，然后选择创建端点。 3. 对于服务类别，选择 Amazon Web Services。 4. 在搜索框中，输入 s3，然后按 Enter。 5. 在搜索结果中，选择 com.amazonaws。 < your-aws-region >.s3 服务名称，其中 “类型” 列中的值为 “接口”。 6. 对于 VPC，选择您的 VPC。对于子网，选择您的子网。 7. 对于安全组，选择或创建一个允许使用 TCP 443 的安全组。 8. 根据要求添加标签，然后选择创建端点。	AWS 管理员
验证端点并找到 DNS 条目。	1. 打开 Amazon VPC 控制台，选择端点，然后选择您之前创建的端点。 2. 在详细信息选项卡中，找到 DNS 名称对应的第一个 DNS 条目。这是区域 DNS 条目。当您使用此 DNS 名	AWS 管理员

Task	描述	所需技能
	称时，请求会在特定于可用区的 DNS 条目之间交替。 3. 选择子网选项卡。您可以在每个可用区中找到端点弹性网络接口的地址。	

Task	描述	所需技能

检查防火墙规则与路由配置。	要确认您的防火墙规则已打开并且网络配置已正确设置，请使用 Telnet 测试端口 443 上的端点。例如：
	
	`$ telnet vpce-<your-VPC-endpoint-ID>.s3.us-east-2.vpce.amazonaws.com 443`
	
	`Trying 10.104.88.6...`
	
	`Connected to vpce-<your-VPC-endpoint-ID>.s3.us-east-2.vpce.amazonaws.com.`
	
	`...`
	
	`$ telnet vpce-<your-VPC-endpoint-ID>.s3.us-east-2.vpce.amazonaws.com 443`
	
	`Trying 10.104.71.141...`
	
	`Connected to vpce-<your-VPC-endpoint-ID>.s3.us-east-2.vpce.amazonaws.com.`
	
	Note
	如果您使用区域条目，则成功测试表明 DNS 在您在 Amazon VPC 控制台中选定终端节点的子网选项卡上看到的
		网络管理员、AWS 管理员

Task	描述	所需技能
	两个 IP 地址之间交替出现。	

Task	描述	所需技能
配置名称解析。	您必须配置名称解析以允许 Hadoop 访问 Amazon S3 接口端点。不能使用端点名称本身。相反，您必须解决 `<your-bucket-name>.s3.<your-aws-region>.amazonaws.com` 或 `*.s3.<your-aws-region>.amazonaws.com` 。有关此命名限制的更多信息，请参阅 Hadoop S3A 客户端简介（Hadoop 网站）。 选择以下配置选项之一： • 使用本地 DNS 解析端点的私有 IP 地址。您可覆盖所有存储桶或选定存储桶的行为。有关更多信息，请参阅使用 AWS 安全混合访问亚马逊 S3 中的“选项 2：使用域名系统响应策略区域 (DNS RPZ) 访问 Amazon S3” PrivateLink（AWS 博客文章）。 • 将本地 DNS 配置为有条件地将流量转发到 VPC 中的解析器入站端点。流量被转发至 Route 53。有关更多信息，请参阅使用 AWS 安全混合访问亚马逊 S3 中的“选项 3：使用 Amazon Route 53 Resolver 入站终端节点从本地转发 DNS 请求”（AWS PrivateLink 博客文章）。	AWS 管理员

Task	描述	所需技能
	• 编辑 Hadoop 集群中所有节点上的 /etc/hosts 文件。这是临时测试解决方案，不建议用于生产。要编辑 /etc/hosts 文件，请为 `<your-bucket-name>.s3.<your-aws-region>.amazonaws.com` 或 `s3.<your-aws-region>.amazonaws.com` 添加一个条目。/etc/hosts 文件不能为一个条目设置多个 IP 地址。您必须从可用区之一选择单个 IP 地址，这将成为单点故障。	

Task	描述	所需技能

为 Amazon S3 配置身份验证。	要通过 Hadoop 对 Amazon S3 进行身份验证，我们建议您将临时角色凭证导出到 Hadoop 环境。有关更多信息，请参阅使用 S3 执行身份验证（Hadoop 网站）。对于长时间运行的作业，您可创建用户并分配仅有权将数据放入 S3 存储桶的策略。访问密钥和密钥可以存储在 Hadoop 上，只有 DistCp 任务本身和 Hadoop 管理员才能访问。有关存储密钥的更多信息，请参阅使用 Hadoop 凭证提供程序存储机密（Hadoop 网站）。有关其他身份验证方法的更多信息，请参阅 AWS IAM Identity Center 文档中的如何获取 IAM 角色的凭证以用于对 Amazon Web Services account 的 CLI 访问（AWS Single Sign-On 后续任务）。
	
	要使用临时凭证，请将临时凭证添加到凭证文件中，或运行以下命令将凭证导出到您的环境中：
	
	`export AWS_SESSION_TOKEN=SECRET-SESSION-TOKEN`
	`export AWS_ACCESS_KEY_ID=SESSION-ACCESS-KEY`
		AWS 管理员

Task	描述	所需技能

	`export AWS_SECRE T_ACCESS_KEY=SESSION-SECRET-KEY`
	
	如您使用传统的访问密钥和私有密钥组合，请运行以下命令：
	
	`export AWS_ACCES S_KEY_ID=my.aws.key`
	`export AWS_SECRE T_ACCESS_KEY=my.se cret.key`
	
	Note
	如果您使用访问密钥和私有密钥组合，请将 DistCp 命令中的凭证提供程序从更改"`org.apac he.hadoop .fs.s3a.T emporaryA WSCredent ialsProvi der`" 为"`org.apac he.hadoop .fs.s3a.S impleAWSC redential sProvider`" 。	

Task	描述	所需技能
使用传输数据 DistCp。	DistCp 要使用传输数据，请运行以下命令： `hadoop distcp -Dfs.s3a.aws.credentials.provider=\"org.apache.hadoop.fs.s3a.TemporaryAWSCredentialsProvider" \ -Dfs.s3a.access.key="${AWS_ACCESS_KEY_ID}" \ -Dfs.s3a.secret.key="${AWS_SECRET_ACCESS_KEY}" \ -Dfs.s3a.session.token="${AWS_SESSION_TOKEN}" \ -Dfs.s3a.path.style.access=true \ -Dfs.s3a.connection.ssl.enabled=true \ -Dfs.s3a.endpoint=s3.<your-aws-region>.amazonaws.com \ hdfs:///user/root/ s3a://<your-bucket-name>` **Note** 当您在 AWS for Amazon S3 中使用 DistCp 命令时，不会自动发现终端节点 PrivateLink 的 AWS 区域。Hadoop 3.3.2	迁移工程师、AWS 管理员

Task	描述	所需技能
	及更高版本通过启用显式设置 S3 存储桶的 AWS 区域的选项来解决此问题。有关更多信息，请参阅 S3A 添加选项 fs.s3a.endpoint.region 设置 AWS 区域（Hadoop 网站）。 有关其他 S3A 提供商的更多信息，请参阅常规 S3A 客户端配置（Hadoop 网站）。例如，如果您使用加密，则可根据您的加密类型将以下选项添加到上述一系列命令中： `-Dfs.s3a.server-side-encryption-algorithm=AES-256 [or SSE-C or SSE-KMS]` **Note** 要将接口终端节点与 S3A 配合使用，必须为接口终端节点的 S3 区域名称（例如`s3.<your-aws-region>.amazonaws.com`）创建 DNS 别名条目。有关说明，请参阅为 Amazon S3 配置身份验证部	

Task	描述	所需技能
	分。Hadoop 3.3.2 及以前版本需要使用此解决方法。未来版本的 S3A 将不需要这种解决方法。 如果您在 Amazon S3 上遇到签名问题，请添加使用签名版本 4（SigV4）签名选项： `-Dmapreduce.map.java.opts="-Dcom.amazonaws.services.s3.enableV4=true"`	

更多模式

- 通过安装， AWS 服务 从 IBM z/OS 进行访问 AWS CLI
- 使用和事件自动将事件驱动的备份从 Amazon S3 备份 CodeCommit 到 Amazon S CodeBuild 3 CloudWatch
- 使用 DynamoDB TTL 自动将项目归档到 Amazon S3
- 使用 Systems Manager 自动备份 SAP HANA 数据库和 EventBridge
- 使用 BMC AMI 云数据将大型机数据备份并存档到 Amazon S3
- 使用 AWS Glue 构建 ETL 服务管道以增量方式将数据从 Amazon S3 加载到 Amazon Redshift
- 使用在 Amazon Bedrock 中配置模型调用日志 AWS CloudFormation
- 使用 Python 在 AWS 上将 EBCDIC 数据转换并解压为 ASCII
- 将 Oracle 的 VARCHAR2 (1) 种数据类型转换为亚马逊 Aurora PostgreSQL 的布尔数据类型
- 使用 Amazon EFS 创建 Amazon ECS 任务定义并在 EC2 实例上挂载文件系统
- 使用 Kinesis Data Streams 和 Firehose 将 DynamoDB 记录传送到亚马逊 S3 AWS CDK
- 使用 Terraform 和 DRA 部署 Lustre 文件系统以实现高性能数据处理
- 估算 Amazon DynamoDB 表的存储成本
- 使用 Security Hub 识别 AWS Organizations 其中的公有 Amazon S3 存储桶
- 将 Amazon RDS for Oracle 数据库实例迁移到使用 AMS 的其他账户
- 将本地 SFTP 服务器迁移到使用 AWSAWS Transfer for SFTP
- 使用 AWS DMS 将 Oracle 分区表迁移到 PostgreSQL
- 使用 Rclone 将数据从 Microsoft Azure Blob 迁移至 Amazon S3
- 将 Oracle CLOB 值迁移到 AWS 上 PostgreSQL 中的单独的行
- 在 AWS 大规模迁移中迁移共享文件系统
- 使用 AWS SFTP 将小型数据集从本地迁移至 Amazon S3
- 监控 Amazon Aurora 以查找未加密的实例
- 使用 Transfer Family 将大型机文件直接移动到 Amazon S3
- 使用 AWS Lambda 自动化功能 AWS 账户 从 AWS Managed Microsoft AD 中移除 Amazon EC2 条目
- 使用带 AWS Fargate 的 Amazon EFS on Amazon EKS，运行带持久数据存储的有状态工作负载
- 成功导入 S3 存储桶作为 AWS CloudFormation 堆栈
- 使用 AWS 在不同 AWS 区域的 Amazon EFS 文件系统之间同步数据 DataSync

- 查看您的 Amazon Web Services account 或组织 EBS 快照详情

开发人员工具

主题

- DevOps
- 基础设施
- 网络和移动应用程序

DevOps

主题

- 使用 Amazon Bedrock 实现 AWS 基础设施运营自动化
- 使用 Terraform 在负载平衡器端点发生变化时自动 CloudFront 更新
- 使用 GitHub 操作在亚马逊自动 CodeGuru 审查 AWS CDK Python 应用程序
- 自动执行 AWS 资源评测
- 使用开源工具自动安装 SAP 系统
- 使用 AWS CDK 自动部署 AWS Service Catalog 产品组合与产品
- 使用和事件自动将事件驱动的备份从 Amazon S3 备份 CodeCommit 到 Amazon S CodeBuild 3 CloudWatch
- 自动删除 AWS CloudFormation 堆栈和相关资源
- 使用和自动进行动态管道管理，以便在 Gitflow 环境中部署修补程序解决方案 AWS Service CatalogAWS CodePipeline
- 使用 Terraform 在亚马逊托管 Grafana 上自动提取和可视化亚马逊 MWAA 自定义指标
- 使用自动化工作流程简化 Amazon Lex 机器人的开发和部署
- 使用 AWS CodePipeline 和 AWS 自动部署堆栈集 CodeBuild
- 使用云托管人和 AWS CDK 将适用于 Systems Manager 的 AWS 托管策略自动附加到 EC2 实例配置文件
- 自动使用 AWS CDK 为微服务构建 CI/CD 管道与 Amazon ECS 集群
- 使用 DevOps 实践和 AWS Cloud9 构建具有微服务的松散耦合架构
- 使用 GitHub Actions 和 Terraform 构建 Docker 镜像并将其推送到 Amazon ECR
- 使用 AWS CodeCommit、AWS 和 AWS Device Farm 构建和测试 iOS 应用程序 CodePipeline
- 使用 cdk-nag 规则包查看 AWS CDK 应用程序或 CloudFormation 模板以了解最佳实践
- 为在 Amazon EKS 上运行的应用程序配置双向 TLS 身份验证
- 使用 AWS 自动创建 AppStream 2.0 资源 CloudFormation
- 使用 Firelens 日志路由器为 Amazon ECS 创建自定义日志解析器
- 使用和 P HashiCorp acker 创建管道 CodePipeline 和 AMI
- 使用创建管道并将项目更新部署到本地 EC2 实例 CodePipeline
- 自动为 Java 和 Python 项目创建动态 CI 管道
- 使用 Terraf CloudWatch orm 部署 Synthetics 加那利群岛

- 在 Amazon ECS 上部署 Java 微服务 CI/CD 管道
- 部署 ChatOps 解决方案来管理 SAST 扫描结果，方法是在聊天应用程序中使用 Amazon Q Developer 自定义操作和 AWS CloudFormation
- 使用 AWS Network Firewall 和 AWS Transit Gateway 部署防火墙
- 使用 AWS CodePipeline CI/CD 管道部署 AWS Glue 作业
- 使用实例配置文件从 AWS Cloud9 部署 Amazon EKS EC2 集群
- 使用 AWS CodePipeline、AWS 和 AWS 在多个 AWS CodeCommit 区域部署代码 CodeBuild
- 使用 Terraform 执行 Amazon Redshift SQL 查询
- 将 AWS Organizations 中整个组织的 AWS Backup 报告导出为 CSV 文件
- 将亚马逊 EC2 实例列表的标签导出到 CSV 文件
- 使用 Troposphere 生成包含 AWS Config 托管规则的 AWS CloudFormation 模板
- 为 SageMaker 笔记本实例提供对另一个 AWS 账户中 CodeCommit 存储库的临时访问权限
- 为多 DevOps 账户环境实施 GitHub Flow 分支策略
- 为多账户环境实施 Gitflow 分支策略 DevOps
- 为多 DevOps 账户环境实施中继分支策略
- 在部署 AWS 基础设施之前，实施集中式自定义 Checkov 扫描以强制执行策略
- 自动检测变化并为 monorepo 启动不同的 CodePipeline 管道 CodeCommit
- 使用 AWS 将 Bitbucket 存储库与 AWS Amplify 集成 CloudFormation
- 使用 Step Functions 和 Lambda 代理函数在 AWS 账户上启动 CodeBuild 项目
- 使用应用程序恢复控制器管理 EMR 集群的多可用区故障转移
- 使用 AWS 代码服务和 AWS KMS 多区域密钥，管理微服务到多个账户和区域的蓝/绿部署
- 使用 AWS 和 AWS CloudFormation Config 监控亚马逊 ECR 存储库的通配符权限
- 使用 AWS CDK 和 GitHub 操作工作流程优化多账户无服务器部署
- 从 AWS CodeCommit 事件中执行自定义操作
- 使用 GitHub 操作基于 AWS CloudFormation 模板配置 AWS Service Catalog 产品
- 通过部署角色自动售货机解决方案来配置权限最低的 IAM 角色
- 将亚马逊 CloudWatch 指标发布到 CSV 文件
- 使用 AWS Lambda 自动化功能 AWS 账户 从 AWS Managed Microsoft AD 正面移除 Amazon EC2 条目
- 使用 AWS Lambda 自动化功能 AWS 账户 从 AWS Managed Microsoft AD 中移除 Amazon EC2 条目

- AWS Glue 使用 pytest 框架对 Python ETL 作业运行单元测试
- 在 Amazon S3 中设置 Helm v3 图表存储库
- 使用 AWS 和 AW CodePipeline S CDK 设置 CI/CD 管道
- 使用证书管理器和 “让我们加 end-to-end密” 为 Amazon EKS 上的应用程序设置加密
- 使用 Flux 简化 Amazon EKS 多租户应用程序部署
- 使用自定义资源将多个电子邮件端点订阅 SNS 主题
- 使用 AWS Fargate WaitCondition 挂钩构造协调资源依赖关系和任务执行
- 在 AWS 中使用第三方 Git 源存储库 CodePipeline
- 使用 AWS 创建 CI/CD 管道以验证 Terraform 的配置 CodePipeline
- 更多模式

使用 Amazon Bedrock 实现 AWS 基础设施运营自动化

由 Ishwar Chauthaiwale (AWS) 和 Anand Bukkapatnam Tirumala (AWS) 创作

摘要

在云原生解决方案中，自动化常见基础架构操作在维护高效、安全和具有成本效益的环境方面起着至关重要的作用。手动处理操作非常耗时，而且容易出现人为错误。此外，具有不同 AWS 专业知识水平的团队成员需要在执行这些任务的同时确保遵守安全协议。此模式演示了如何使用 Amazon Bedrock 通过自然语言处理 (NLP) 自动执行常见 AWS 基础设施操作。

这种模式可以帮助组织开发可重复使用、模块化和安全的代码，用于在多个环境中部署基于人工智能的生成基础架构。通过专注于基础设施即代码 (IaC) 和自动化，它提供了关键 DevOps 优势，包括版本控制、一致的部署、减少错误、更快的配置和改进的协作。

该模式实现了一个安全的架构，使团队能够管理与密钥相关的操作， AWS 服务 包括：

- 亚马逊简单存储服务 (Amazon S3) Semple Service 存储桶版本控制管理
- 创建亚马逊关系数据库服务 (Amazon RDS) 快照
- 亚马逊弹性计算云 (Amazon EC2) 实例管理

该架构使用亚马逊虚拟私有云（ Amazon VPC ）终端节点和私有网络进行安全通信，其 AWS Lambda 功能在私有子网中充当任务执行者。Amazon S3 提供数据管理并实施全面的 AWS Identity and Access Management (IAM) 角色和权限，以确保适当的访问控制。此解决方案不包括聊天记录功能，也不会存储聊天记录。

先决条件和限制

- 活跃 AWS 账户的.
- 应采取适当的访问控制措施，以帮助保护和控制访问权限。访问控制的示例包括使用基础模型访问权限 AWS Systems Manager、用于部署的 IAM 角色和基于服务的角色、禁用对 Amazon S3 存储桶的公开访问以及设置死信队列。
- AWS Key Management Service (AWS KMS) 客户管理的密钥。
- AWS Command Line Interface (AWS CLI) 版本 2 或更高版本，已在部署环境中安装和配置。
- 安装并配置了 Terraform P AWS rovider 版本 4 或更高版本。
- 安装并配置了 Terraform 版本 1.5.7 或更高版本。

- 在 Amazon Bedrock 中查看和定义代理操作组的 OpenAPI 架构，以帮助防止未经授权的访问并维护数据完整性。
- 在你 AWS 账户 中@@ 启用了所需的 Amazon Titan Text Embeddings v2 以及 Claude 3.5 Sonnet 或 Claude 3 Haiku 基础模型的访问权限。为避免部署失败，请确认您的目标部署 AWS 区域 支持所需的模型。
- 遵循架构完善的框架最佳实践的配置虚拟私AWS 有云 (VPC)。
- 已完成对 Amazon 责任人工智能政策的审查。

产品版本

- Amazon Titan 文本嵌入 v2
- Anthropic Claude 3.5 十四行诗或克劳德 3 俳句
- Terraform P AWS rovider 版本 4 或更高版本
- Terraform 版本 1.5.7 或更高版本

架构

下图显示了此模式的工作流和体系结构组件。

解决方案架构由多个层组成，它们协同工作以处理自然语言请求并执行相应的 AWS 操作：

1. 用户通过 Amazon Bedrock 聊天控制台提出操作请求。
2. 聊天机器人使用 Amazon Bedrock 知识库来处理请求。它实现了用于自然语言处理的 Amazon Titan 文本嵌入 v2 模型。
3. 如果用户提示中包含操作请求，则 Amazon Bedrock 操作组使用 Anthropic Claude 3 Haiku 或 Claude 3.5 Sonnet 模型（取决于你的选择）作为执行逻辑，并通过 OpenAPI 架构定义操作。
4. 操作组使用访问 AWS PrivateLink 用于安全服务通信的 Amazon VPC 终端节点。
5. 该 AWS Lambda 功能可通过亚马逊 Bedrock 服务的亚马逊 VPC 终端节点访问。
6. Lambda 函数是主要的执行引擎。根据请求，Lambda 函数调用 API 以对执行操作。 AWS 服务 Lambda 函数还处理操作的路由和执行。
7. 从 Lambda 函数 AWS 服务 获取 API 请求并执行相应的操作。
8. Lambda 函数计算 Amazon Bedrock 可以理解的输出有效负载。

9. 此有效载荷通过使用 PrivateLink 安全服务通信发送到 Amazon Bedrock。Amazon Bedrock 使用的大型语言模型 (LLM) 可以理解此有效负载，并将其转换为人类可理解的格式。
10. 然后在 Amazon Bedrock 聊天控制台上向用户显示输出。

该解决方案支持以下主要操作：

- Amazon S3-启用存储桶版本控制以进行版本控制。
- Amazon RDS — 创建用于备份的数据库快照。
- Amazon EC2 — 列出实例并控制实例的启动和停止。

工具

AWS 服务

- Amazon Bedrock 是一项完全托管的服务，它通过统一的 API 提供来自领先的人工智能初创公司和亚马逊的高性能基础模型 (FMs) 供您使用。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行外壳中的命令进行交互。
- 亚马逊弹性计算云 (Amazon EC2) 在中提供可扩展的计算容量 AWS Cloud。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon OpenSearch Server less 是亚马逊 OpenSearch 服务的按需无服务器配置。
- AWS PrivateLink帮助您创建从您的虚拟私有云 (VPCs) 到 VPC 外部服务的单向私有连接。
- Amazon Relational Database Service（Amazon RDS）可帮助您在中设置、操作和扩展关系数据库 AWS Cloud。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Systems Manager 可帮助您管理在 AWS Cloud中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。
- Amazon Virtual Private Cloud（亚马逊 VPC）可帮助您将 AWS 资源启动到您定义的虚拟网络中。该虚拟网络类似于您在数据中心中运行的传统网络，并具有使用 AWS的可扩展基础设施的优势。

其他工具

- Git 是一个开源的分布式版本控制系统。
- Terraform 是一款基础设施即代码 (IaC) 工具 HashiCorp ，可帮助您创建和管理云和本地资源。

代码存储库

此模式的代码可在 GitHub aws-sam infra-ops-orchestrator ples/ 存储库中找到。

最佳实践

- 定期监控 Lambda 执行日志。有关更多信息，请参阅 Lambda 函数监控和故障排除。有关最佳实践的更多信息，请参阅使用 AWS Lambda 函数的最佳实践。
- 定期审查安全配置，确保符合贵组织的要求。有关更多信息，请参阅安全最佳实践。
- 遵循最低权限原则，授予执行任务所需的最低权限。有关更多信息，请参阅 IAM 文档中的授予最低权限和安全最佳实践。

操作说明

部署解决方案

Task	描述	所需技能

克隆存储库。	要在本地计算机上克隆存储库，请运行以下命令：
	`git clone "git@github.com:aws-samples/infra-ops-orchestrator.git"`
	`cd infra-ops-orchestrator`
		AWS DevOps， DevOps 工程师

编辑环境变量。	在克隆`terraform.tfvars` 存储库的根目录中编辑文件。查看所指示的占位符`[XXXXX]`，并根据您的环境对其进行更新。	AWS DevOps， DevOps 工程师

Task	描述	所需技能

创建基础架构。	要创建基础架构，请运行以下命令：
	`terraform init`
	`terraform plan`
	仔细审查执行计划。如果计划的更改可以接受，则运行以下命令：
	`terraform apply --auto-approve`
		AWS DevOps， DevOps 工程师

访问解决方案

Task	描述	所需技能
访问解决方案。	成功部署后，请按照以下步骤使用基于聊天的界面： 1. 要访问基础设施 Orchestrator Assistrator 助手，请 AWS Management Console 使用具有 A mazon Bedrock 权限的 IAM 角色登录，然后通过以下网址打开 Amazon Bedrock 控制台。https://console.aws.amazon.com/bedrock/从左侧导航窗格中选择代理。然后，在 “代理” 部分选择 “基础架构协调器助手”。	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	2. 请确保您的 AWS 环境中存在针对以下建议的目标资源，然后尝试以下示例操作： • 通过询问：“创建 RDS 实例 [实例名称] 的快照” 来创建 Amazon RDS 实例的快照备份 • 询问：“为存储桶 [存储桶名称] 启用版本控制”，即可在 Amazon S3 存储桶上启用版本控制 • 通过询问 “列出所有 EC2 EC2 实例” 来列出 Amazon 实例 • 通过询问：“启动 EC2 实例 [instance-id]” 或 “停止 EC2 实例 [instance-id]” 来启动或停止 Amazon EC2 实例 注意：将方括号中的值替换为实际资源名称或您的 AWS 环境 IDs 中的值。	

清理资源

Task	描述	所需技能
删除已创建的资源。	要删除由此模式创建的所有基础架构，请运行以下命令： `terraform plan -destroy`	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	仔细审查销毁计划。如果计划的删除是可以接受的，则运行以下命令： `terraform destroy` 注意：此命令将永久删除由此模式创建的所有资源。在删除任何资源之前，该命令将提示您进行确认。	

故障排除

事务	解决方案
代理行为	有关此问题的信息，请参阅 Amazon Bedrock 文档中的测试代理行为并对其进行故障排除。
Lambda 网络问题	有关这些问题的信息，请参阅 Lambda 文档中的 Lambda 中的网络问题疑难解答。
IAM 权限	有关这些问题的信息，请参阅 IAM 文档中的 IAM 疑难解答。

相关资源

- 为 Amazon RDS 的单可用区数据库实例创建数据库快照
- 在 Amazon Bedrock 中为代理的操作组定义 OpenAPI 架构
- 在存储桶上启用版本控制
- 亚马逊 Bedrock Agents 的工作原理
- 使用 Amazon Bedrock 知识库检索数据并生成 AI 响应
- 通过安全访问服务 AWS PrivateLink
- 停止并启动 Amazon EC2 实例

- 使用操作组定义代理要执行的操作

使用 Terraform 在负载平衡器端点发生变化时自动 CloudFront 更新

由 Tamilselvan P (AWS)、Mohan Annam (AWS) 和 Naveen Suthar (AWS) 创作

摘要

当亚马逊 Elastic Kubernetes Service（Amazon EKS）的用户通过 Helm 图表删除并重新安装其入口配置时，就会创建一个新的应用程序负载均衡器 (ALB)。这就造成了问题，因为Amazon CloudFront 继续引用旧ALB的DNS记录。因此，将无法访问发往此端点的服务。（有关此问题工作流程的更多详细信息，请参阅其他信息。）

为了解决这个问题，此模式描述了如何使用使用 Python 开发的自定义 AWS Lambda 函数。此 Lambda 函数会自动检测何时通过亚马逊规则创建新 ALB。 EventBridge 然后 适用于 Python (Boto3) 的 AWS SDK，该函数使用新 ALB 的 DNS 地址更新 CloudFront 配置，确保流量路由到正确的终端节点。

这种自动化解决方案可在不增加路由或延迟的情况下保持服务连续性。该过程有助于确保即使底层基础设施发生变化，也 CloudFront 始终引用正确的 ALB DNS 端点。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- 使用 Helm 在 Amazon EKS 上部署的用于测试和验证的示例 Web 应用程序。有关更多信息，请参阅亚马逊 EKS 文档中的使用 Helm 在 Amazon EKS 上部署应用程序。
- 配置 CloudFront 为将呼叫路由到由 Helm 入口控制器创建的 ALB。有关更多信息，请参阅 Amazon EKS 文档中的使用 Helm 安装 AWS 负载均衡器控制器和文档中的限制对应用程序负载均衡器的访问权限。 CloudFront
- 在本地工作区中安装和配置 Terraform。

限制

- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅按地区划分的AWS 服务。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

产品版本

- Terraform 版本 1.0.0 或更高版本

- Terraform P AWS rovider 版本 4.20 或更高版本

架构

下图显示了此模式的工作流和体系结构组件。

此解决方案执行以下步骤：

1. 每当重启或部署 Helm 时，Amazon EKS 入口控制器都会创建一个新的应用程序负载均衡器 (ALB)。
2. EventBridge 查找 ALB 创建事件。
3. ALB 创建事件会触发 Lambda 函数。
4. Lambda 函数是基于 python 3.9 部署的，并使用 boto3 API 进行调用。 AWS 服务 Lambda 函数使用最新的负载均衡器 DNS 名称更新 CloudFront 条目，该名称是从创建负载均衡器事件中收到的。

工具

AWS 服务

- Amazon 通过全球数据中心网络交付您的网页内容，从而降低延迟并提高性能，从而 CloudFront加快网络内容的分发。
- 亚马逊 Elastic Kubernetes Service（亚马逊 EKS）可帮助你在上面运行 AWS Kubernetes，而无需安装或维护自己的 Kubernetes 控制平面或节点。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如， AWS Lambda 函数、使用 API 目的地的 HTTP 调用端点或其他 AWS 账户目的地的事件总线。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- 适用于 Python (Boto3) 的 AWS SDK是一个软件开发套件，可帮助您将 Python 应用程序、库或脚本与集成 AWS 服务。

其他工具

- Python 是通用的计算机编程语言。

- Terraform 是一款基础设施即代码 (IaC) 工具 HashiCorp ，可帮助您创建和管理云和本地资源。

代码存储库

此模式的代码可在 GitHub aws-cloudfront-automation-terraform- samples 存储库中找到。

操作说明

设置本地工作站

Task	描述	所需技能
设置和配置 Git CLI。	要在本地工作站上安装和配置 Git 命令行界面 (CLI)，请按照 Git 文档中的入门-安装 G it 说明进行操作。	DevOps 工程师
创建项目文件夹并添加文件。	1. 转到模式的GitHub 存储库，然后选择 “代码” 按钮。 2. 在 “克隆” 对话框中，选择 HTTPS 选项卡。在使用网址克隆中，复制显示的网址。 3. 在本地计算机上创建文件夹。用你的项目名称来命名。 4. 在本地计算机上打开终端，然后导航到该文件夹。 5. 要克隆此模式的 git 存储库，请运行以下命令：`git clone https://github.com/aws-samples/aws-cloudfront-automation-terraform-samples` 6. 克隆存储库后，使用以下命令转到克隆的目录：cd	DevOps 工程师

Task	描述	所需技能
	`<directory name>/cloudfront-update` 在您选择的集成开发环境 (IDE) 中打开此项目。	

使用 Terraform 配置配置目标架构

Task	描述	所需技能

部署解决方案。	要在目标中部署资源 AWS 账户，请使用以下步骤：
	
	1. 转到 `cloudfront-update` 文件夹。
	2. 使用更新`terraform.tfvars` 文件`cloudfront_distribution_id` 。
	3. 要 AWS 区域 为您的 AWS 个人资料设置，请运行以下命令：
	
	`export AWS_REGION={{ REGION }}`
	
	4. 要初始化 Terraform，请运行以下命令：
	
	`terraform init`
	
	5. 要验证 Terraform，请运行以下命令：
	
	`terraform validate`
		DevOps 工程师

Task	描述	所需技能

	6. 要创建 Terraform 执行计划，请运行以下命令：
	`terraform plan`
	7. 要应用中的操作`terraform plan`，请运行以下命令：
	`terraform apply`	

验证部署

Task	描述	所需技能
验证部署。	1. 登录 AWS Management Console 并在 https://console.aws.amazon.com/cloudfront/v4/ hom CloudFront e 上打开亚马逊主机。 2. 在左侧导航窗格中，选择 “分配”，然后打开 CloudFront 分配。 3. 在 O rigins 选项卡上，验证源名称和起源映射是否具有更新的 ALB DNS 记录。	DevOps 工程师

清理基础设施

Task	描述	所需技能
清理基础架构。	要清理您之前创建的基础架构，请使用以下步骤：	DevOps 工程师

Task	描述	所需技能
	1. 运行以下命令：`terraform destroy` 2. 要确认销毁命令，请输入`yes`。	

故障排除

事务	解决方案
验证提供商凭证时出错	在本地计算机上运行 Terraform `apply` 或`destroy`命令时，可能会遇到类似以下内容的错误： `Error: configuring Terraform AWS Provider: error validating provider credentials: error calling sts:GetCallerIdentity: operation error STS: GetCallerIdentity, https response error StatusCode: 403, RequestID: 123456a9-fbc1-40ed-b8d8-513d0133ba7f, api error InvalidClientTokenId: The security token included in the request is invalid.` 此错误是由本地计算机配置中使用的凭证安全令牌过期引起的。 要解决此错误，请参阅 AWS Command Line Interface (AWS CLI) 文档中的设置和查看配置设置。

相关资源

AWS resources

- 限制对应用程序负载均衡器的访问权限

- 使用 Loa AWS d Balancer 控制器路由互联网流量

Terraform 文档

- AWS Provider
- 安装 Terraform
- 远程状态

其他信息

有问题的工作流程

图表显示了以下工作流：

1. 当用户访问应用程序时，呼叫将转到。 CloudFront
2. CloudFront 将呼叫路由到相应的 Application Load Balancer (ALB)。
3. ALB 包括目标 IP 地址，即应用程序 Pod 的 IP 地址。然后，ALB 向用户提供预期的结果。

但是，此工作流程存在问题。应用程序部署是通过 Helm 图表进行的。每当有部署或者有人重启 Helm 时，也会重新创建相应的入口。因此，外部负载平衡器控制器会重新创建 ALB。此外，在每次重新创建期间，系统都会使用不同的 DNS 名称重新创建 ALB。因此， CloudFront 将在原始设置中有一个陈旧的条目。由于此条目过时，用户将无法访问该应用程序。此问题会导致用户停机。

替代解决方案

另一种可能的解决方案是为 ALB 创建外部 DNS，然后将其指向中的 CloudFront Amazon Route 53 私有托管区域终端节点。但是，这种方法在应用程序流中又增加了一个跳跃，这可能会导致应用程序延迟。这种模式的 Lambda 函数解决方案不会中断当前流。

使用 GitHub 操作在亚马逊自动 CodeGuru 审查 AWS CDK Python 应用程序

由 Vanitha Dontireddy (AWS) 和 Sarat Chandra Pothula (AWS) 创作

摘要

这种模式展示了 Amazon 对 AWS Cloud Development Kit (AWS CDK) Python 应用程序的 CodeGuru 自动代码审查的集成，这些审查是通过操作精心编排的。 GitHub 该解决方案部署了在 Python AWS CDK 中定义的无服务器架构。通过在开发管道中自动执行专家代码分析，此方法可以为 AWS CDK Python 项目执行以下操作：

- 提高代码质量。
- 简化工作流程。
- 最大限度地发挥无服务器计算的优势。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- AWS Command Line Interface (AWS CLI) 版本 2.9.11 或更高版本，已安装并配置。
- 具有读写工作流程权限的活跃 GitHub 账户和 GitHub 存储库，并通过 GitHub Actions 创建拉取请求 (PR)，以确保 PR 工作流程正常运行。
- 在 “ GitHub 操作” 中扮演一个 OpenID Connect (OIDC) 角色，用于在中部署解决方案。 AWS 账户要创建角色，请使用AWS CDK 构造。

限制

- Amazon CodeGuru Profiler 支持用所有 Java 虚拟机 (JVM) 语言（例如 Scala 和 Kotlin）、运行时和 Python 3.6 或更高版本编写的应用程序。
- Amazon CodeGuru Reviewer 仅支持与来自以下来源提供商的 Java 和 Python 代码存储库的关联：AWS CodeCommit、Bitbucket GitHub、、 GitHub 企业云和 GitHub 企业服务器。此外，只有通过操作才能支持 GitHub 亚马逊简单存储服务 (Amazon S3) Service 存储库。
- 在持续集成和持续部署 (CI/CD) 管道中，没有一种自动打印结果的方法。相反，这种模式使用 GitHub 操作作为处理和显示发现结果的替代方法。

- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅按地区划分的 AWS 服务。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

架构

下图显示了此解决方案的架构。

如图所示，当开发者创建拉取请求 (PR) 以供审核时，Acti GitHub ons 会触发以下步骤：

1. IAM 角色假设 — 管道使用 GitHub 密钥中指定的 IAM 角色来执行部署任务。
2. 代码分析
   - CodeGuru 审阅者分析存储在 Amazon S3 存储桶中的代码。它可以识别缺陷并提供修复和优化建议。
   - CodeGuru 对违反政策的行为和漏洞进行安全扫描。
3. 调查结果回顾
   - 管道会在控制台输出中打印指向调查结果仪表板的链接。
   - 如果检测到关键发现，管道会立即失效。
   - 对于严重程度高、正常或低严重性的发现，管道会继续执行下一步。
4. 公关批准
   - 审阅者必须手动批准 PR。
   - 如果 PR 被拒绝，则管道将失败并停止进一步的部署步骤。
5. CDK 部署 — 在 PR 批准后，CDK 部署过程开始。它设置了以下内容 AWS 服务 和资源：
   - CodeGuru Profiler
   - AWS Lambda 函数
   - Amazon Simple Queue Service (Amazon SQS) 队列
6. 生成分析数据-要为 P CodeGuru rofiler 生成足够的分析数据，请执行以下操作：
   - 该管道通过定期向 Amazon SQS 队列发送消息来多次调用 Lambda 函数。

工具

AWS 服务

- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，可帮助您在代码中定义和配置 AWS Cloud 基础架构。
- CDK Toolkit 是一个命令行云开发套件，可帮助您与 AWS CDK 应用程序进行交互。
- Amazon CodeGuru Profiler 会从您的实时应用程序收集运行时性能数据，并提供建议，以帮助您微调应用程序性能。
- Amazon CodeGuru Reviewer 使用程序分析和机器学习来检测开发人员难以发现的潜在缺陷。然后，P CodeGuru rofiler 会提供改进你的 Java 和 Python 代码的建议。
- Amazon CodeGuru Security 是一款静态应用程序安全工具，它使用机器学习来检测违反安全策略和漏洞。它提供解决安全风险的建议并生成指标，以便您可以跟踪应用程序的安全状况。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Queue Service (Amazon SQS) 提供了一个安全、持久且可用的托管队列，它可帮助您集成和分离分布式软件系统与组件。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

其他工具

- GitHub A@@ c tions 是一个持续集成和持续交付 (CI/CD) 平台，与 GitHub 存储库紧密集成。您可以使用 GitHub Actions 来自动执行构建、测试和部署管道。

代码存储库

此模式的代码可在 GitHub amazon-codeguru-suite-cdk-python 存储库中找到。

最佳实践

- 遵循开发和部署云基础架构的最佳实践 AWS CDK。
- 在 GitHub 操作工作流程中使用时，请遵循 IAM AWS 服务 中的安全最佳实践，包括：
  - 请勿在存储库代码中存储凭据。
  - 假设 IAM 角色以接收临时证书，并在可能的情况下使用临时证书。
  - 向 GitHub 操作工作流程中使用的 IAM 角色@@ 授予最低权限。仅授予在操作工作流程中执行操作所需的权限。 GitHub

- 监控 GitHub 操作工作流程中使用的 IAM 角色的活动。
- 定期轮换您使用的任何长期证书。

操作说明

设置您的环境

Task	描述	所需技能

设置 AWS 凭据。	要导出定义堆栈 AWS 账户 以及部署堆栈 AWS 区域 位置的变量，请运行以下命令：
	`export CDK_DEFAULT_ACCOUNT=<12-digit AWS account number>`
	`export CDK_DEFAULT_REGION=<AWS Region>`
	的 AWS 凭证通过环境变量提供。 AWS CDK
		AWS DevOps， DevOps 工程师

克隆存储库。	要在本地计算机上克隆存储库，请运行以下命令： `git clone https://github.com/aws-samples/amazon-codeguru-suite-cdk-python.git`	AWS DevOps， DevOps 工程师
安装 CDK 工具包。	要确认 CDK 工具包已安装并检查版本，请运行以下命令： `cdk --version`	AWS DevOps， DevOps 工程师

Task	描述	所需技能

	如果 CDK Toolkit 版本低于 2.27.0，请输入以下命令将其更新到 2.27.0 版本：
	`npm install -g aws-cdk@2.27.0`
	如果未安装 CDK 工具包，请运行以下命令进行安装：
	`npm install -g aws-cdk@2.27.0 --force`	

安装所需的依赖项。	要安装所需的项目依赖项，请运行以下命令：
	`python -m pip install --upgrade pip`
	`pip install -r requirements.txt`
		AWS DevOps， DevOps 工程师

Task	描述	所需技能

引导 CDK 环境。	要引导 AWS CDK 环境，请运行以下命令：
	`npm install`
	`npm run cdk bootstrap "aws://${ACCOUNT_NUMBER}/${AWS_REGION}"`
	成功引导环境后，应显示以下输出：
	`# Bootstrapping environment aws://{account}/{region}...`
	`# Environment aws://{account}/{region} bootstrapped`
		AWS DevOps， DevOps 工程师

部署 CDK 应用程序

Task	描述	所需技能
合成 AWS CDK 应用程序。	要合成 AWS CDK 应用程序，请运行以下命令： `cdk synth` 有关此命令的更多信息，请参阅文档中的 cdk synthesiz e。AWS CDK	AWS DevOps， DevOps 工程师
部署资源	要部署资源，请运行以下命令：	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	`cdk deploy --require-approval never` **Note** 该`--require-approval never`标志表示 CDK 将自动批准并执行所有更改。这包括 CDK 通常会标记为需要手动审查的更改（例如 IAM 政策变更或资源移除）。在生产环境中使用该`--require-approval never`标志之前，请确保您的 CDK 代码和 CI/CD 管道经过充分测试且安全。	

创建 GitHub 机密和个人访问令牌

Task	描述	所需技能
在中创建所需的密钥 GitHub。	要允许 A GitHub ctions 工作流程在不暴露仓库代码中的敏感信息的情况下安全地访问 AWS 资源，请创建密钥。要在、和中 GitHub 创建密钥 `ROLE_TO_ASSUME` `CodeGuruReviewArtifactBucketName` `AWS_ACCOUNT_ID` ，	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	请按照 GitHub 操作文档中为存储库创建密钥中的说明进行操作。 以下是有关变量的更多信息： • AWS_ACCOUNT_ID -执行管道的 AWS 账户 ID。 • CodeGuruReviewArtifactBucketName — 存储 CodeGuru Reviewer 工件的 S3 存储桶的名称。此模式使用存储桶名称codeguru-reviewer-build-artifacts-<ACCOUNT_ID>-<REGION> 。 • AWS_REGION — 资源 AWS 区域 所在的位置。 • ROLE_TO_ASSUME — 管道担任的 IAM 角色的名称。此模式使用角色名称githubActionsDeployRole 。	

Task	描述	所需技能
创建 GitHub 个人访问令牌。	要为您 GitHub 的 Actions 工作流程设置一种安全的身份验证和交互方式 GitHub，请执行以下操作： 1. 要创建对仓库具有读写权限的 GitHub 个人访问令牌，请按照 GitHub 文档中管理您的个人访问令牌中的说明进行操作。 2. 要将此令牌另存为 Actions 的存储库密钥，请按照 GitHub 操作文档中为仓库创建密钥中的说明进行 GitHub 操作。	AWS DevOps， DevOps 工程师

清理

Task	描述	所需技能
清理资源。	要清理你的 AWS CDK Python 应用程序，请运行以下命令： `cdk destroy --all`	DevOps 工程师

故障排除

事务	解决方案
显示指向仪表板调查结果的链接。	在 CI/CD 管道中无法打印调查结果。相反，这种模式使用 GitHub 操作作为处理和显示发现结果的替代方法。

相关资源

AWS resources

- AWS Cloud 开发套件
- 亚马逊 CodeGuru 文档
- Amazon S3
- AWS Identity and Access Management
- Amazon Simple Queue Service
- 什么是 AWS Lambda？

GitHub 文档

- 在亚马逊 Web Services 中配置 OpenID Connect
- GitHub 操作
- 重复使用工作流程
- 触发工作流程

自动执行 AWS 资源评测

由 Naveen Suthar (AWS)、Arun Bagal (AWS)、Manish Garg (AWS) 和 Sandeep Gawande (AWS) 创建

摘要

此示例介绍了一种使用 AWS Cloud Development Kit (AWS CDK) 设置资源评测功能的自动方法。通过使用这种模式，运营团队可以自动收集资源审计详细信息，并在单个控制面板上查看部署在 Amazon Web Services account 中的所有资源的详细信息。这在以下用例中很有用：

- 识别基础设施即代码 (IaC) 工具，隔离由不同 IaC 解决方案（例如 Ter HashiCorp raform、AWS CloudFormation、AWS CDK 和 AW S 命令行界面 (AWS CLI)）创建的资源
- 获取资源审计信息

该解决方案还将帮助领导团队通过单独的控制面板深入了解 Amazon Web Services account 中的资源和活动。

> **Note**
>
> Amazon QuickSight 是一项付费服务。在运行它来分析数据和创建控制面板之前，请先查看 Amazon 的 QuickSight 定价。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- AWS Identity and Acccess Management（IAM）角色和权限，具有预配置资源访问权限
- 创建的允许访问亚马逊简单存储服务 (Amazon S3) 和亚马逊 A thena 的亚马逊 QuickSight 账户
- 已安装 2.55.1 或更高版本的 AWS CDK。
- 已安装 3.9 或更高版本的 Python。

限制

- 此解决方案部署至单个 Amazon Web Services account。
- 除非已设置 AWS 并将数据存储在 S3 存储桶中，否则 CloudTrail 该解决方案不会跟踪部署前发生的事件。

产品版本

- AWS CDK 版本 2.55.1 或更高版本
- Python 版本 3.9 或更高版本

架构

目标技术堆栈

- Amazon Athena
- AWS CloudTrail
- AWS Glue
- AWS Lambda
- Amazon QuickSight
- Amazon S3

目标架构

AWS CDK 代码将部署所有资源，这些资源是在 Amazon Web Services account 中设置资源评测功能的必要条件。下图显示了向 AWS Glue、Amazon Athena 和发送 CloudTrail 日志的过程。 QuickSight

1. CloudTrail 将日志发送到 S3 存储桶进行存储。
2. 事件通知会调用 Lambda 函数，以处理日志并生成经过筛选数据。
3. 筛选后的数据存储至另一个 S3 存储桶。
4. 在 S3 存储桶中的筛选数据上设置 AWS Glue 爬网程序，以便在 AWS Glue Data Catalog 表创建架构。
5. 筛选后的数据已就绪，可供 Amazon Athena 查询。
6. 通过访问查询的数据 QuickSight 进行可视化。

自动化和扩缩

- 如果 AWS Organizations 中存在组织范围的 CloudTrail 跟踪，则该解决方案可以从一个 AWS 账户扩展到多个 AWS 账户。通过在组织 CloudTrail 级别部署，您还可以使用此解决方案来获取所有所需资源的资源审计详细信息。
- 此模式使用 AWS 无服务器资源部署解决方案。

工具

Amazon Web Services

- Amazon Athena 是一种交互式查询服务，可帮助您使用标准 SQL 直接在 Amazon S3 中分析数据。
- AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，可帮助您在代码中定义和预调配 Amazon Web Services Cloud 基础设施。
- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和 AWS 区域的整个生命周期中对其进行管理。
- AWS CloudTrail 可帮助您审计 AWS 账户的治理、合规和运营风险。
- AWS Glue 是一项完全托管的提取、转换、加载（ETL）服务。它可以帮助您在数据存储和数据流之间对数据进行可靠地分类、清理、扩充和移动。此模式使用 AWS Glue 爬网程序与 AWS Glue Data Catalog 表。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon QuickSight 是一项云规模的商业智能 (BI) 服务，可帮助您在单个控制面板中可视化、分析和报告数据。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

代码存储库

此模式的代码可在 GitHub infrastructure-assessment-iac-automation存储库中找到。

代码存储库包含以下文件和文件夹：

- `lib` 文件夹 — 用于创建 AWS 资源的 AWS CDK 构造 Python 文件
- `src/lambda_code` — 在 Lambda 函数中运行的 Python 代码
- `requirements.txt` — 必须安装的所有 Python 依赖项列表

- `cdk.json` — 用于提供启动资源所需值的输入文件

最佳实践

为 Lambda 函数设置监控和警报。有关更多信息，请参阅 Lambda 函数监控和故障排除。有关使用 Lambda 函数时的一般最佳实践标准，请参阅 AWS 文档

操作说明

设置您的环境

Task	描述	所需技能
在本地机器克隆存储库。	要克隆存储库，请运行 `git clone https://github.com/aws-samples/infrastructure-assessment-iac-automation.git` 命令。	AWS DevOps， DevOps 工程师

设置 Python 虚拟环境以及安装所需依赖项。	要设置 Python 虚拟环境，请运行以下命令。
	`cd infrastructure-assessment-iac-automation`
	`python3 -m venv .venv`
	`source .venv/bin/activate`
	要设置所需依赖项，请运行命令 `pip install -r requirements.txt` 。
		AWS DevOps， DevOps 工程师

设置 AWS CDK 环境和合成 AWS CDK 代码。	1. 若要在您的 Amazon Web Services account 中设置 AWS CDK 环境，请运行命令 `cdk bootstrap`	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	`aws://ACCOUNT-NUMBER/REGION` 。 2. 要将代码转换为 AWS CloudFormation 堆栈配置，请运行命令`cdk synth`。	

在本地机器上设置 AWS 凭证

Task	描述	所需技能

为待部署堆栈的账户和区域导出变量。	要使用环境变量为 AWS CDK 提供 AWS 凭证，请运行以下命令。
	`export CDK_DEFAULT_ACCOUNT=<12 Digit AWS Account Number>`
	`export CDK_DEFAULT_REGION=<region>`
		AWS DevOps， DevOps 工程师

设置 AWS CLI 配置文件。	要为账户设置 AWS CLI 配置文件，请按 AWS 文档中的说明进行操作。	AWS DevOps， DevOps 工程师

配置和部署资源评测工具

Task	描述	所需技能
在账户中部署资源。	若要使用 AWS CDK 在 Amazon Web Services account 中部署资源，请执行以下操作：	AWS DevOps

Task	描述	所需技能
	1. 在克隆存储库根目录中，在 `cdk.json` 文件中为以下参数提供输入： • `s3_context` • `ct_context` • `kms_context` • `lambda_context` • `glue_context` • `qs_context` 这些值定义了资源配置和命名方法。默认值已设置，可根据需要进行更改。 **Note** 为避免出错称 S3 存储桶已存在，请确保在`ct` 和`output`部分`s3_context` 中为提供唯一的名称。 2. 要部署资源，请运行命令 `cdk deploy`。 该`cdk deploy`命令创建一个 CloudTrail 资源来记录事件并将日志文件保存在输入 S3 存储桶中。追踪日志文件将由 Lambda 函数处理。筛选后的结果存储在输出 S3 存储桶中，可供亚马逊 Athena 和亚马逊使用。QuickSight	

Task	描述	所需技能
运行 AWS Glue 爬网程序，并创建数据目录表。	AWS Glue 爬网程序用于保持数据架构的动态性。该解决方案通过按照 AWS Glue 爬网程序调度程序的定义定期运行爬网程序来创建和更新 AWS Glue Data Catalog 表中的分区。在输出 S3 存储桶中有数据后，使用以下步骤运行 AWS Glue 爬网程序，并创建用于测试的数据目录表架构： 1. 登录 AWS 管理控制台，然后导航到 AWS Glue 控制台。 2. 在导航窗格中，在数据目录下，选择爬网程序。 3. 选择 `iac-tool-qa-resource-iac-json-crawler` 爬网程序。 4. 运行爬网程序。 5. 爬网程序成功运行后，它会创建 AWS Glue Data Catalog 表。AWS QuickSight 将使用该表对数据进行可视化。 **Note** AWS CDK 代码将 AWS Glue 爬网程序配置为在特定时间运行，但您也可以按需运行。	AWS DevOps， DevOps 工程师

Task	描述	所需技能
部署 QuickSight 构造。	1. 要部署 QuickSight 构造，请取消注释介于`#QuickSight setup - start`和`#QuickSight setup – ends` 中的`resource_iac_tool_stack.py` 代码。 2. 取消注释后，运行`cdk deploy`命令在 QuickSight 账户`QuickSight DataSet` 中创建`QuickSight DataSource` 和。	AWS DevOps， DevOps 工程师

Task	描述	所需技能
创建 QuickSight 仪表板。	要创建示例 QuickSight 仪表板和分析，请执行以下操作： 1. 导航到 QuickSight 控制台并选择部署资源的 AWS 区域。 2. 在导航窗格中，选择数据集，然后验证是否`ct-operations-iac-ds` 已在 Amazon 数据集中创建了一个名为 QuickSight 的数据集。 如果您看不到数据集，请重新部署 QuickSight 构造。 3. 选择 `ct-operations-iac-ds` 数据集，然后选择在分析中使用。 4. 选择默认工作表。 5. 从左侧的字段列表中选择相应列。 6. 选择所需列后，选择适当的视觉类型以查看数据。 有关更多信息，请参阅在 Amazon 中启动分析 QuickSight和在 Amazon 中启动可视化类型 QuickSight。	AWS DevOps， DevOps 工程师

清理解决方案中的所有 AWS 资源

Task	描述	所需技能
移除 AWS 资源。	1. 若要移除解决方案部署的 AWS 资源，请运行命令 `cdk destroy`。 2. 从两个 S3 存储桶中删除所有对象，然后移除所有存储桶。 有关更多信息，请参阅删除存储桶。	AWS DevOps， DevOps 工程师

基于 AWS 资源评测工具自动化设置其他功能

Task	描述	所需技能
监控和清理手动创建资源。	（可选）如果您的组织有使用 IaC 工具创建资源的合规性要求，则您可以使用 AWS 资源评测工具，自动获取手动配置的资源，从而实现合规。您也可以使用此工具将资源导入至 IaC 工具或重新创建这些资源。若要监控手动配置的资源，请执行以下概要任务： 1. 自动部署 AWS 资源评测工具。 2. 设置 Lambda 函数，以每天查询 Athena 表，查找有关手动配置资源的相关数据，然后将其导出至逗号分隔值 (CSV) 文件。	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	3. Lambda 函数运行后，可以向相应利益相关者发送包含所需数据的通知。 4. 为了延长保留期，可以将 .csv 文件存储至 S3 存储桶。 5. 根据.csv 文件中的信息，删除手动创建的资源，或将其导入现有的 IaC 解决方案。	

故障排除

事务	解决方案
AWS CDK 会返回错误信息。	要获得有关 AWS CDK 问题的帮助，请参阅常见 AWS CDK 问题疑难解答。

相关资源

- 使用 Python 构建 Lambda 函数
- AWS CDK 入门
- 在 Python 中使用 AWS CDK
- 创建 CloudTrail 日志跟踪
- 开始使用亚马逊 QuickSight

其他信息

多个账户

若要为多个账户设置 AWS CLI 凭证，请使用 AWS 配置文件。有关更多信息，请参阅设置 AWS CLI 中的配置多项配置文件部分。

AWS CDK 命令

使用 AWS CDK 时，切记以下有用的命令：

- 列出应用程序的所有堆栈

```
cdk ls
```

- 发出合成的 AWS 模板 CloudFormation

```
cdk synth
```

- 将堆栈部署至您的默认 Amazon Web Services account 和区域

```
cdk deploy
```

- 将已部署的堆栈与当前状态比较

```
cdk diff
```

- 打开 AWS CDK 文档

```
cdk docs
```

使用开源工具自动安装 SAP 系统

由 Guilherme Sesterheim (AWS) 编写

摘要

此模式介绍了如何使用开源工具创建以下资源，以自动安装 SAP 系统：

- SAP S/4HANA 1909 数据库
- 一个 SAP ABAP 中央服务 (ASCS) 实例
- 一个 SAP 主应用程序服务器 (PAS) 实例

HashiCorp Terraform 创建 SAP 系统的基础架构，Ansible 配置操作系统 (OS) 并安装 SAP 应用程序。Jenkins 运行安装。

这种设置将 SAP 系统的安装变成了可重复的过程，有助于提高部署效率和质量。

> **Note**
>
> 此模式中提供的示例代码适用于高可用性 (HA) 系统和非 HA 系统。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 包含所有 SAP 媒体文件的 Amazon Simple Storage Service (Amazon S3) 存储桶
- 具有访问密钥和私有密钥、并有以下权限的 AWS Identity and Access Management (IAM) 主体：
  - 只读权限：Amazon Route 53、AWS Key Management Service (AWS KMS)
  - 读写权限：亚马逊 S3、亚马逊弹性计算云（亚马逊）、亚马逊弹性文件系统（亚马逊 EFS EC2）、IAM、亚马逊、亚马逊 CloudWatch、亚马逊 DynamoDB
- Route 53 私有托管区域
- 在 Amazon Marketplace 中订阅 Red Hat Enterprise Linux for SAP with HA and Update Services 8.2 亚马逊机器映像（AMI）
- AWS KMS 客户托管密钥

- Secure Shell (SSH) 密钥对
- 一个 Amazon EC2 安全组，允许从你安装 Jenkins 的主机名在端口 22 上进行 SSH 连接（主机名很可能是本地主机）
- HashiCorp 已安装和@@ 配置的 Vagrant
- VirtualBox由 Oracle 安装和配置
- 熟悉 Git、Terraform、Ansible 以及 Jenkins

限制

- 仅 SAP S/4HANA 1909 针对此特定场景进行了全面测试。如果您使用其他版本的 SAP HANA，则需要修改此模式中的示例 Ansible 代码。
- 此模式中的示例程序适用于 Mac OS 和 Linux 操作系统。部分命令只能在基于 Unix 的终端中运行。但是，您可通过使用不同的命令和 Windows 操作系统，以获得类似的结果。

产品版本

- SAP S/4HANA 1909
- Red Hat Enterprise Linux (RHEL) 8.2 或更高版本

架构

下图介绍了使用开源工具在 Amazon Web Services account 中自动安装 SAP 系统的示例工作流：

图表显示了以下工作流：

1. Jenkins 通过运行 Terraform 和 Ansible 代码编排 SAP 系统安装的运行。
2. Terraform 代码构建 SAP 系统基础设施。
3. Ansible 代码配置操作系统和安装 SAP 应用程序。
4. Amazon 实例上安装了 SAP S/4HANA 1909 数据库、ASCS 实例和包含所有已定义先决条件的 PAS 实例。 EC2

> **Note**
>
> 此模式中的示例设置会自动在您的 AWS 账户中创建一个 Amazon S3 存储桶来存储 Terraform 状态文件。

技术堆栈

- Terraform
- Ansible
- Jenkins
- SAP S/4HANA 1909 数据库
- SAP ASCS 实例
- SAP PAS 实例
- 亚马逊 EC2

工具

Amazon Web Services

- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥以保护您的数据。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。

其他工具

- HashiCorp Terraform 是一款命令行界面应用程序，可帮助您使用代码来配置和管理云基础架构和资源。
- Ansible 是一款开源配置即代码 (CaC) 工具，可帮助实现应用程序、配置和 IT 基础设施的自动化。
- Jenkins 是一款开源自动化服务器，它使开发人员能够构建、测试和部署其软件。

代码

此模式的代码可在 GitHub aws-install-sap-with-jenkins- ansible 存储库中找到。

操作说明

配置先决条件

Task	描述	所需技能
将您的 SAP 媒体文件添加至 Amazon S3 存储桶。	创建包含所有 SAP 媒体文件的 Amazon S3 存储桶。 **Important** 请务必按照 Launch Wizard 文档中针对 S/4HANA 的 AWS La unch Wizard 文件夹层次结构进行操作。	云管理员
安装 VirtualBox。	VirtualBox由 Oracle 安装和配置。	DevOps 工程师
安装 Vagrant。	通过以下方式安装和配置 Vagrant。 HashiCorp	DevOps 工程师
配置 Amazon Web Services account。	1. 确认您的 IAM 主体拥有访问密钥和私有密钥，并具有以下访问权限： • 只读权限：Amazon Route 53、AWS Key Management Service (AWS KMS) • 读写权限：亚马逊 S3、亚马逊弹性计算云（亚马逊）、亚马逊弹性文件系统（亚马逊 EFS	常规 AWS

Task	描述	所需技能
	EC2）、IAM、亚马逊、亚马逊 CloudWatch、亚马逊 DynamoDB 2. 保存 IAM 主体访问密钥和私有密钥以供日后参考。 3. 如果您还没有 Route 53 私有托管区域，请创建一个。保存区域名称（例如 sapteam.net）以供日后参考。 4. 在Amazon Marketplace 上订阅 Red Hat Enterprise Linux for SAP with HA and Update Services 8.2. 保存 AMI ID（例如 ami-00000 00）以供日后参考。 5. 创建 AWS KMS 客户托管密钥。保存 KMS 密钥的 Amazon 资源名称（ARN），以备日后参考。 **Note** 以下是 AWS KMS 客户托管密钥 ARN 的示例：arn: aws: kms: us-east-1 :123412341234: key/uui d 6. 创建 SSH 密钥对。保存密钥对的名称和 .pem 文件，以供日后参考。	

Task	描述	所需技能
	7. 创建一个 Amazon EC2 安全组，允许从你安装 Jenkins 的主机名在端口 22 上进行 SSH 连接。保存安全组 ID，以供日后参考。 **Note** 主机名很可能是本地主机。	

构建和运行您的 SAP 安装

Task	描述	所需技能
从中克隆代码存储库 GitHub。	在上克隆 aws-install-sap-with-jenkins-an sible 存储库。GitHub	DevOps 工程师
启动 Jenkins 服务。	打开 Linux 终端。然后，导航至包含克隆代码存储库文件夹的本地文件夹，并运行以下命令： `sudo vagrant up` **Note** Jenkins 的初创公司大约需要 20 分钟。成功后，该命令会返回服务已启动并正在运行消息。	DevOps 工程师

Task	描述	所需技能
在 Web 浏览器中打开并登录 Jenkins。	1. 在 Web 浏览器中输入 http://localhost:5555。打开 Jenkins。 2. 登录 Jenkins 时，使用 admin 作为用户名，使用 my_secret_pass_from_vault 作为密码。	DevOps 工程师

Task	描述	所需技能
配置 SAP 系统安装参数。	1. 在 Jenkins 中，选择管理 Jenkins。然后，选择管理凭证。将显示您可配置的凭证变量列表。 2. 配置以下所有凭证变量： • 对于 AWS_ACCOUNT_CREDENTIALS，请输入您的 IAM 委托人的访问密钥 ID 和私有访问密钥 ID。 • 对于 AMI_ID，请输入 Red Hat Enterprise Linux for SAP with HA and Update Services 8.2 AMI 的 AMI ID。 • 对于 KMS_KEY_ARN，请输入您的 AWS KMS 客户托管密钥 ARN。 • 于 SSH_KEYPAIR_NAME，请输入 SSH 密钥对名称，而不必输入 .pem 文件类型。 • 对于 SSH_KEYPAIR_FILE，请输入密钥对的 .pem 文件（例如mykeypair.pem）的全名。请务必将密钥对的 .pem 文件上传至 Jenkins。 • 对于 S3_ROOT_FOLDER_INSTALL_FILES，输入包含 SAP 媒体文件的 Amazon S3 存储桶的名称和文件夹（如果适用）（例如	AWS 系统管理员、 DevOps 工程师

Task	描述	所需技能
	s3: //S4H1909 ）。my-media-bucket • 对于 PRIVATE_DNS_ZONE_NAME，请输入 Route 53 私有托管区的名称（例如 myprivatecompanyurl.net ）。 • 对于 VPC_ID，请输入您要在其中创建 SAP 资源的 Amazon VPC 的 VPC ID（例如 vpc-12345 ）。 • 对于 SUBNET_IDS， IDs 如果您在测试环境中工作，请输入两个公有子网（用于未来的 HA 功能）。如果您在生产环境中工作，最佳实践是在堡垒主机上使用两个私有子网。 • 在 SECURITY_GROUP_ID 中，输入允许从你安装 Jenkins 的主机名在端口 22 上进行 SSH 连接的亚马逊 EC2 安全组的 ID。 **Note** 您可以根据自己的用例根据需要配置其他非必需的参数。例如，您可以更改实例的 SAP 系统 ID (SID)、SAP 系统的默认密码、名称和标	

Task	描述	所需技能
	签。所有必需变量的名称开头都有（必填）。	
运行您的 SAP 系统安装。	1. 在 Jenkins 中，选择 Jenkins Home。然后选择 SAP hana+ascs+Pas 3 实例 。 2. 选择启动并安装。然后，选择 IAM。 3. 选择立即构建。 有关管道操作步骤的信息，请参阅 AWS Blog 上的使用开源工具自动化 SAP 安装中的了解管道操作步骤部分。 **Note** 如果出现错误，请将光标移到出现的红色错误框上，然后选择日志。显示错误管道操作步骤日志。大多数错误的原因是参数设置不正确。	DevOps 工程师，AWS 系统管理员

相关资源

- DevOps 适用于 SAP — SAP 安装：从 2 个月到 2 小时（DevOps 企业峰会视频库）

使用 AWS CDK 自动部署 AWS Service Catalog 产品组合与产品

由 Sandeep Gawande (AWS)、RAJNEESH TYAGI (AWS) 和 Viyoma Sachdeva (AWS) 创作

摘要

AWS Service Catalog 可帮助您集中管理获准在组织 AWS 环境中使用的 IT 服务或产品目录。一系列产品称为产品组合，产品组合还包含配置信息。利用 AWS Service Catalog，您可以为组织内的每类用户创建一个自定义产品组合，然后授予对适当产品组合的访问权限。然后，这些用户可在产品组合中快速部署他们需要的任何产品。

如果您拥有复杂的网络基础架构（例如多区域和多账户架构），建议您在单个中央账户中创建和管理服务目录组合。此模式介绍如何使用 AWS Cloud Development Kit (AWS CDK) 在中央账户中自动创建服务目录组合，向最终用户授予访问权限，然后选择向一个或多个目标 Amazon Web Services account 配置产品。此 ready-to-use解决方案在源账户中创建 Service Catalog 产品组合。它还可以选择使用 AWS CloudFormation 堆栈在目标账户中配置产品，并帮助您 TagOptions 为产品进行配置：

- AWS CloudFormation StackSets — 您可以使用 StackSets 在多个 AWS 区域和账户中启动 Service Catalog 产品。在此解决方案中，您可选择在部署此解决方案时自动配置产品。有关更多信息，请参阅使用 AWS CloudFormation StackSets（S ervice Catalog 文档）和StackSets 概念（CloudFormation 文档）。
- TagOption 库-您可以使用 TagOption 库管理已配置产品的标签。A TagOption是在 AWS Service Catalog 中管理的键值对。它不是 AWS 标签，但它可以用作基于创建 AWS 标签的模板 TagOption。有关更多信息，请参阅TagOption 库（Service Catalog 文档）。

先决条件和限制

先决条件

- 用作管理 Service Catalog 产品组合的源账户的有效 Amazon Web Services account。
- 如果您使用此解决方案在一个或多个目标客户中配置产品，则目标账户必须已经存在并且处于活动状态。
- AWS Identity and Access Management (IAM) 权限，用于访问 AWS Service Catalog CloudFormation、AWS 和 AWS IAM。

产品版本

- AWS CDK 版本 2.27.0

架构

目标技术堆栈

- 集中式 Amazon Web Services account 中的 Service Catalog 产品组合
- 部署至目标账户的 Service Catalog 产品

目标架构

1. 在投资组合 (或源) 账户中，您可使用您的用例的 Amazon Web Services account、AWS 区域、IAM 角色、产品组合和产品信息更新 config.json 文件。
2. 您部署 AWS CDK 应用程序。
3. AWS CDK 应用程序扮演部署 IAM 角色并创建 config.json 文件中定义的 Service Catalog 产品组合和产品。

   如果您配置 StackSets 为在目标账户中部署产品，则该过程将继续进行。如果您未配置 StackSets 为配置任何产品，则该过程已完成。
4. AWS CDK 应用程序扮演StackSet 管理员角色并部署您在 c onfig. json 文件中定义的 AWS CloudFormation 堆栈集。
5. 在目标账户中， StackSets 担任StackSet 执行角色并配置产品。

工具

Amazon Web Services

- AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，可帮助您在代码中定义和预调配 Amazon Web Services Cloud 基础设施。
- AWS CDK Toolkit 是一个命令行云开发套件，可帮助您与 AWS CDK 应用程序进行交互。
- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Service Catalog 可帮助您集中管理获准在 AWS 上使用的 IT 服务目录。最终用户可在遵循组织设定约束的情况下快速部署他们所需已获得批准的 IT 服务。

代码存储库

此模式的代码可在 GitHubaws-cdk-servicecatalog-automation存储库中找到。代码存储库包含以下文件和文件夹：

- cdk-sevicecatalog-app— 此文件夹包含此解决方案的 AWS CDK 应用程序。
- config — 此文件夹包含 config.json 文件和用于在 Service Catalog 产品组合中部署产品的 CloudFormation 模板。
- config/config.json — 该文件包含所有配置信息。您可更新此文件，以针对您的用例自定义此解决方案。
- config/tem plates — 此文件夹包含服务中心产品的 CloudFormation 模板。
- setup.sh — 此脚本部署解决方案。
- uninstall.sh — 此脚本删除部署此解决方案时所创建的堆栈和所有 AWS 资源。

若要使用示例代码，请按照操作部分的说明执行。

最佳实践

- 用于部署此解决方案的 IAM 角色应遵守最低权限原则（IAM 文档）。
- 遵守使用 AWS CDK 开发云应用程序的最佳实践（AWS Blog 文章）。
- 遵守 A WS CloudFormation 最佳实践（CloudFormation 文档）。

操作说明

设置您的环境

Task	描述	所需技能
安装 AWS CDK Toolkit。	确保您已安装 AWS CDK Toolkit。输入以下命令，以确认是否已安装并检查版本。 `cdk --version`	AWS DevOps， DevOps 工程师

Task	描述	所需技能

	如果未安装 AWS CDK Toolkit，请输入以下命令以进行安装。
	`npm install -g aws-cdk@2.27.0`
	如果 AWS CDK Toolkit 版本低于 2.27.0，则输入以下命令，以将其更新至 2.27.0 版本。
	`npm install -g aws-cdk@2.27.0 --force`	

克隆存储库。	输入以下命令。在其他信息部分的克隆存储库中，您可以复制包含存储库 URL 的完整命令。这将从中克隆aws-cdk-servicecatalog-automation存储库。 GitHub
	`git clone <repository-URL>.git`
	这将在目标目录中创建 `cd aws-cdk-servicecatalog-automation` 文件夹。输入以下命令以导航至此文件夹。
	`cd aws-cdk-servicecatalog-automation`
		AWS DevOps， DevOps 工程师

Task	描述	所需技能

设置 AWS 凭证。	输入以下命令。它们会导出以下变量，这些变量定义您要部署堆栈的 Amazon Web Services account 和区域。
	`export CDK_DEFAULT_ACCOUNT=<12-digit AWS account number>`
	`export CDK_DEFAULT_REGION=<AWS Region>`
	AWS CDK 的 AWS 凭证是通过环境变量提供的。
		AWS DevOps， DevOps 工程师

为最终用户 IAM 角色配置权限。	如果您要使用 IAM 角色授予对产品组合及其中的产品的访问权限，则这些角色必须具有由 servicecatalog.amazonaws.com 服务主体担任的权限。有关如何授予这些权限的说明，请参阅使用 Service Catalog 启用可信访问(AWS Organizations 文档)。	AWS DevOps， DevOps 工程师

Task	描述	所需技能
配置所需的 IAM 角色 StackSets。	如果您使用 StackSets 在目标账户中自动配置产品，则需要配置管理和运行堆栈集的 IAM 角色。 1. 在源账户中，确认 `AWSCloudFormationStackSetAdministrationRole` 是否已存在。在目标账户中，确认 `AWSCloudFormationStackSetExecutionRole` 是否已经存在。如果这些角色已经存在，您可以跳至下一操作指南。 2. 按照授予自我托管权限（IAM 文档）中的说明，在产品组合账户中创建堆栈集管理角色，并在每个目标账户中创建执行角色。	AWS DevOps， DevOps 工程师

自定义与部署解决方案

Task	描述	所需技能
创建 CloudFormation 模板。	在该 `config/templates` 文件夹中，为要包含在产品组合中的任何产品创建 CloudFormation 模板。有关更多信息，请参阅使用 AWS CloudFormation 模板（CloudFormation 文档）。	应用程序开发人员、AWS DevOps、 DevOps 工程师

Task	描述	所需技能
自定义配置文件。	在 `config` 文件夹中，打开 config.json 文件。并根据您的用例定义相应的参数。若非另有说明，以下参数为必需参数： 在 `portfolios` 部分中，定义以下参数，以创建一个或多个 Service Catalog 产品组合： • `portfolioName` – 产品组合的名称。 • `providerName` – 管理投资组合的个人、团队或组织的名称。 • `description` – 投资组合的简要说明。 • `roles` – （可选）应有权访问此产品组合的任何 IAM 角色的名称。具有此角色的用户可访问此产品组合中的产品。 • `users` – （可选）应有权访问此产品组合及其产品的任何 IAM 用户的名称。 • `groups` – （可选）应有权访问此产品组合及其产品的任何 IAM 用户组的名称。 **Warning** IAM 用户具有长期凭证，这会带来安全风险。为帮助减轻这种风	应用程序开发人员、 DevOps 工程师、AWS DevOps

Task	描述	所需技能

	险，我们建议仅向这些用户提供执行任务所需的权限，并在不再需要这些用户时将其移除。
	
	Important
	`roles`、`users`、和`groups`都是可选参数，但是如果您未定义其中一个参数，则任何人都无法在 Service Catalog 控制台中查看产品组合产品。至少定义以下参数之一。有关更多信息，请参阅向 Service Catalog 最终用户授予权限（Service Catalog 文档）。
	
	• （可选）在该`tagOption`部分中，TagOptions 为产品定义：
	  • `key`— TagOption 密钥的名称
	  • `value`— 允许的字符串值 TagOption
	
	有关更多信息，请参阅TagOption 库（Service Catalog 文档）。	

Task	描述	所需技能

	• 在 products 部分，为产品定义以下参数：
	  • portfolioName – 要向其添加产品的产品组合的名称。只能指定一种产品组合。
	  • productName – 产品的名称。
	  • owner – 产品的所有者。
	  • productVersionName – 以字符串值表示的产品版本名称，例如 v1。
	  • templatePath — 产品 CloudFormation 模板的文件路径。
	  • deployWithStackSets —（可选）指定一个或多个账户和区域，StackSets 用于自动配置产品组合中的产品。如果您使用此部署选项，则需要使用以下参数。
	  • accounts – 目标账户。
	  • regions – 目标区域。
	  • stackSetAdministrationRoleName — 用于管理 StackSets 配置的 IAM 角色的名称。请勿更改此值。此角色必须具有此确切名称。	

Task	描述	所需技能
	• `stackSetExecutionRoleName` — 部署堆栈实例的目标账户中的 IAM 角色的名称。请勿更改此值。此角色必须具有此确切名称。 有关已完成的配置文件的示例，请参阅其他信息部分的示例配置文件。	
部署解决方案。	输入以下命令。这将部署 AWS CDK 应用程序，并按照 config.json 文件中指定的方式配置 Service Catalog 产品组合和产品。 `sh +x setup.sh`	应用程序开发人员、 DevOps 工程师、AWS DevOps

Task	描述	所需技能
验证部署。	通过执行以下操作，验证部署是否成功： 1. 使用可访问配置文件中定义的一个或多个投资组合的凭证登录 AWS 管理控制台。 2. 打开 Service Catalog 控制台，网址为https://console.aws.amazon.com/servicecatalog/。 3. 在导航窗格中的配置下，选择产品。验证您能否能看到为产品组合指定的产品列表。 4. 按照启动产品（Service Catalog 文档）中的说明启动其中一个可用产品。确认可用的产品版本和标签与您在配置文件中提供的值是否相匹配。 5. 如果您选择使用在一个或多个目标账户中自动配置产品 StackSets，请执行以下操作： a. 使用授予您查看目标账户之一中所配置产品权限的凭证登录。 b. 在 Service Catalog 控制台的导航窗格，在配置下，选择已配置产品。 c. 确认预期产品在列表中显示。	常规 AWS

Task	描述	所需技能
（可选）更新产品组合与产品。	如果您想使用此解决方案更新产品组合、产品或配置新产品： 1. 在 config.json 文件中进行所需更改。 2. 根据需要在`config/template` 文件夹中添加或修改任何 CloudFormation 模板。 3. 重新部署解决方案。 例如，您可添加其他产品组合或预配置更多资源。AWS CDK 应用程序仅实施更改。如果先前部署的产品组合或产品无变化，则重新部署不会影响它们。	应用程序开发人员、 DevOps 工程师、通用 AWS

清理解决方案

Task	描述	所需技能
（可选）移除此解决方案所部署的 AWS 资源。	如果要删除预配置产品，请按删除预配置产品（Service Catalog 文档）中的说明进行操作。 如果您想删除此解决方案所创建的所有资源，请输入以下命令。 `sh uninstall.sh`	AWS DevOps， DevOps 工程师，应用程序开发人员

相关资源

- AWS Service Catalog Construct Library（AWS API 参考）
- StackSets 概念（CloudFormation 文档）
- AWS Service Catalog（AWS Marketing）
- 在 AWS CDK 中使用 Service Catalog（AWS 研讨会）

其他信息

克隆存储库

输入以下命令以从中克隆存储库 GitHub。

```
git clone https://github.com/aws-samples/aws-cdk-servicecatalog-automation.git
```

示例配置文件

以下是带示例值的 config.json 文件示例。

```
{
    "portfolios": [
        {
            "displayName": "EC2 Product Portfolio",
            "providerName": "User1",
            "description": "Test1",
            "roles": [
                "<Names of IAM roles that can access the products>"
            ],
            "users": [
                "<Names of IAM users who can access the products>"
            ],
            "groups": [
                "<Names of IAM user groups that can access the products>"
            ]
        },
        {
            "displayName": "Autoscaling Product Portfolio",
            "providerName": "User2",
            "description": "Test2",
            "roles": [
                "<Name of IAM role>"
```

```
            ]
        }
    ],
    "tagOption": [
        {
            "key": "Group",
            "value": [
                "finance",
                "engineering",
                "marketing",
                "research"
            ]
        },
        {
            "key": "CostCenter",
            "value": [
                "01",
                "02",
                "03",
                "04"
            ]
        },
        {
            "key": "Environment",
            "value": [
                "dev",
                "prod",
                "stage"
            ]
        }
    ],
    "products": [
        {
            "portfolioName": "EC2 Product Profile",
            "productName": "Ec2",
            "owner": "owner1",
            "productVersionName": "v1",
            "templatePath": "../../config/templates/template1.json"
        },
        {
            "portfolioName": "Autoscaling Product Profile",
            "productName": "autoscaling",
            "owner": "owner1",
            "productVersionName": "v1",
```

```
            "templatePath": "../../config/templates/template2.json",
            "deployWithStackSets": {
                "accounts": [
                    "012345678901",
                ],
                "regions": [
                    "us-west-2"
                ],
                "stackSetAdministrationRoleName":
 "AWSCloudFormationStackSetAdministrationRole",
                "stackSetExecutionRoleName": "AWSCloudFormationStackSetExecutionRole"
            }
        }
    ]
}
```

使用和事件自动将事件驱动的备份从 Amazon S3 备份 CodeCommit 到 Amazon S CodeBuild 3 CloudWatch

由 Kirankumar Chandrashekar (AWS) 创建

摘要

在亚马逊网络服务 (AWS) 云上，您可以使用 AWS CodeCommit 托管基于 Git 的安全存储库。CodeCommit 是一项完全托管的源代码控制服务。但是，如果 CodeCommit 存储库被意外删除，其内容也会被删除并且无法恢复。

此模式描述了在对 CodeCommit 存储库进行更改后，如何自动将存储库备份到亚马逊简单存储服务 (Amazon S3) 存储桶。如果稍后删除了 CodeCommit 存储库，则此备份策略将为您提供 point-in-time 恢复选项。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 现有 CodeCommit 存储库，可根据您的要求配置用户访问权限。有关更多信息，请参阅 CodeCommit 文档 CodeCommit中的 AWS 设置。
- 用于上传 CodeCommit 备份的 S3 存储桶。

限制

- 这种模式会自动备份您的所有 CodeCommit 存储库。如果您想备份单个 CodeCommit 存储库，则必须修改 Amazon Event CloudWatch s 规则。

架构

下图说明了此模式的工作流。

工作流程由以下步骤组成：

1. 代码被推送到 CodeCommit 存储库。

2. CodeCommit 存储库会将存储库更改通知 CloudWatch 事件（例如，`git push`命令）。
3. CloudWatch 事件调用 AWS CodeBuild 并向其发送 CodeCommit 存储库信息。
4. CodeBuild 克隆整个 CodeCommit 存储库并将其打包成.zip 文件。
5. CodeBuild 将.zip 文件上传到 S3 存储桶。

技术堆栈

- CloudWatch 活动
- CodeBuild
- CodeCommit
- Amazon S3

工具

- Amazon CloudWatch Ev CloudWatch ents — Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。
- AWS CodeBuild — CodeBuild 是一项完全托管的持续集成服务，可编译源代码、运行测试并生成可随时部署的软件包。
- AWS CodeCommit — CodeCommit 是一项完全托管的源代码控制服务，可托管基于 Git 的安全存储库。
- AWS Identity and Access Management (IAM) – IAM 是一项 Web 服务，可帮助您安全地控制对 AWS 资源的访问。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项面向互联网的存储服务。

操作说明

创建 CodeBuild 项目

Task	描述	所需技能
创建 CodeBuild 服务角色。	登录 AWS 管理控制台，并打开 IAM 控制台。选择 Role（角色），然后选择	云管理员

Task	描述	所需技能
	Create role（创建角色）。为创建服务角色 CodeBuild 以克隆 CodeCommit 存储库、将文件上传到 S3 存储桶以及向 Amazon 发送日志 CloudWatch。有关更多信息，请参阅 CodeBuild 文档中的创建 CodeBuild 服务角色。	
创建 CodeBuild 项目。	在 CodeBuild 控制台上，选择创建 CodeBuild 项目。使用 “其他信息” 部分中的`buildspec.yml` 模板创建 CodeBuild 项目。有关本故事的帮助，请参阅 CodeBuild 文档中的创建构建项目。	云管理员

创建和配置 CloudWatch 事件规则

Task	描述	所需技能
为 CloudWatch 事件创建 IAM 角色。	在 IAM 控制台上，选择角色并为 CloudWatch 事件创建一个 IAM 角色。有关这方面的更多信息，请参阅 IAM 文档中的 CloudWatch 事件 IAM 角色。 **Important** 您必须向 IAM 角色添加 CloudWatch 事件`codebuild:StartBuild` 权限。	云管理员

Task	描述	所需技能
创建 CloudWatch 事件规则。	1. 在 CloudWatch 控制台上，选择 “事件”，然后选择 “规则”。选择创建规则，然后使用其他信息部分中的 CloudWatch 事件规则。这将创建一条规则，用于监听 CodeCommit 存储库中的事件更改（例如`git push`或`git commit`命令）。有关更多信息，请参阅 AWS CodePipeline 文档中的为 CodeCommit 来源创建 CloudWatch 事件规则。 2. 选择目标，选择主题，然后选择配置输入。选择输入转换器，然后使用其他信息部分中的输入路径和输入模板。这样可以确保您的 CodeCommit 存储库详细信息被解析并作为环境变量发送到 CodeBuild 项目。有关更多信息，请参阅 CloudWatch 文档中的输入变压器教程。 3. 选择配置详细信息，输入规则的名称和描述。选择 Create rule(创建规则)。 **⚠ Important** 此 CloudWatch 事件规则描述了所有 CodeCommit 仓库中的更改。如果要备份	云管理员

Task	描述	所需技能
	单个 CodeCommit 存储库或使用单独的 S3 存储桶进行不同的存储库备份，则必须修改 CloudWatch 事件规则。	

相关资源

创建 CodeBuild 项目

- 创建 CodeBuild 服务角色
- 创建 CodeBuild 项目
- Git 客户端命令的必需权限

创建和配置 CloudWatch 事件规则

- 为 CodeCommit 源创建 CloudWatch 事件规则
- 使用输入转换器自定义要传递给事件目标的内容
- 创建在 CloudWatch 事件上启动的事件规则
- 创建 CloudWatch 活动 IAM 角色

其他信息

CodeBuild buildspec.yml 模板

```
version: 0.2
phases:
  install:
    commands:
      - pip install git-remote-codecommit
  build:
    commands:
       - env
```

```
      - git clone -b $REFERENCE_NAME codecommit::$REPO_REGION://$REPOSITORY_NAME
      - dt=$(date '+%d-%m-%Y-%H:%M:%S');
      - echo "$dt"
      - zip -yr $dt-$REPOSITORY_NAME-backup.zip ./
      - aws s3 cp $dt-$REPOSITORY_NAME-backup.zip s3:// #substitute a valid S3 Bucket
 Name here
```

CloudWatch 赛事规则

```
{
  "source": [
    "aws.codecommit"
  ],
  "detail-type": [
    "CodeCommit Repository State Change"
  ],
  "detail": {
    "event": [
      "referenceCreated",
      "referenceUpdated"
    ]
  }
}
```

CloudWatch 事件规则目标的示例输入转换器

输入路径：

```
{"referenceType":"$.detail.referenceType","region":"$.region","repositoryName":"$.detail.reposi
```

输入模板（请根据需要填写值）：

```
{
    "environmentVariablesOverride": [
        {
            "name": "REFERENCE_NAME",
            "value": ""
        },
        {
            "name": "REFERENCE_TYPE",
            "value": ""
        },
```

```
        {
            "name": "REPOSITORY_NAME",
            "value": ""
        },
        {
            "name": "REPO_REGION",
            "value": ""
        },
        {
            "name": "ACCOUNT_ID",
            "value": ""
        }
    ]
}
```

自动删除 AWS CloudFormation 堆栈和相关资源

由 SANDEEP SINGH (AWS) 和詹姆斯·雅各布 (AWS) 创作

摘要

AWS CloudFormation是一项广泛使用的服务，用于管理云基础架构即代码 (IaC)。使用时 CloudFormation，您可以将相关资源作为一个称为堆栈的单元进行管理。您可通过创建、更新和删除堆栈来创建、更新和删除一组资源。

有时，您不再需要 CloudFormation 堆栈中的资源。根据资源及其配置，删除堆栈及其关联资源可能很复杂。在现实世界的生产系统中，由于相互冲突的条件或 CloudFormation 无法超越的限制，删除有时会失败或需要很长时间。可能需要仔细的计划和执行，以确保以高效、一致的方式正确删除所有资源。此模式描述了如何设置一个框架，以帮助您管理涉及以下复杂性的 CloudFormation 堆栈的删除：

- 具有删除保护的资源-某些资源可能已启用删除保护。常见的示例是亚马逊 DynamoDB 表和亚马逊简单存储服务 (Amazon S3) Service 存储桶。删除保护可防止自动删除，例如通过删除 CloudFormation。如果要删除这些资源，则必须手动或以编程方式覆盖或暂时禁用删除保护。在继续操作之前，您应该仔细考虑删除这些资源的影响。
- 具有保留策略的资源 — 某些资源，例如 AWS Key Management Service (AWS KMS) 密钥和 Amazon S3 存储桶，可能有保留策略，规定在请求删除后应保留多长时间。您应在清理策略中考虑这些政策，以保持对组织策略和监管要求的合规性。
- 延迟删除附加到 VPC 的 Lambda 函数 — 删除附加到虚拟私有云 (VPC) 的AWS Lambda函数可能需要 5-40 分钟，具体取决于流程中涉及的多个相互关联的依赖关系。如果您在删除堆栈之前将函数与 VPC 分离，则可以将延迟减少到 1 分钟以内。
- 非直接创建的资源 CloudFormation — 在某些应用程序设计中，资源可能由应用程序本身或通过 CloudFormation 堆栈置备的资源在原始堆栈之外创建。下面是两个示例：
  - CloudFormation 可能会预配置一个运行用户数据脚本的亚马逊弹性计算云 (Amazon EC2) 实例。然后，此脚本可能会创建一个AWS Systems Manager参数来存储与应用程序相关的数据。此参数不是通过管理的 CloudFormation。
  - CloudFormation 可能会配置一个 Lambda 函数，该函数会自动生成用于存储 CloudWatch 日志的 Amazon 日志组。此日志组不是通过管理的 CloudFormation。

  尽管这些资源不是由直接管理的 CloudFormation，但在删除堆栈时通常需要清理它们。如果不加以管理，他们可能会成为孤儿，导致不必要的资源消耗。

尽管这些护栏可能导致复杂性，但它们是故意和关键的。 CloudFormation 在许多情况下，允许超越所有限制和不分青红皂白地删除资源可能会导致不利和不可预见的后果。但是，作为 DevOps 负责管理环境的云工程师，有时可能需要克服这些限制，尤其是在开发、测试或暂存环境中。

目标业务成果

通过实施此框架，您可以获得以下好处：

- 成本管理 — 定期高效地清理临时环境，例如 end-to-end或用户验收测试环境，有助于防止资源的运行时间超过必要的时间。这可以显著降低成本。
- 安全 — 自动清理过时或未使用的资源可减少攻击面并有助于维护安全的 AWS 环境。
- 运营效率 — 定期和自动清理可以带来以下运营好处：
  - 删除旧日志组或空的 Amazon S3 存储桶的自动脚本可以通过保持环境的清洁和可管理来提高运营效率。
  - 快速删除和重新创建堆栈支持设计和实现的快速迭代，从而实现更强大、更具弹性的架构。
  - 定期删除和重建环境可以帮助您识别和修复潜在的问题。这可以帮助您确保基础架构能够承受现实场景。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- Python 3.6 或更高版本，已安装
- AWS Command Line Interface (AWS CLI)，已安装并配置

限制

- 命名约定用于标识应删除的资源。此模式中的示例代码使用资源名称的前缀，但您可以定义自己的命名约定。不使用此命名约定的资源将不会被识别或随后删除。

架构

下图显示了该框架如何识别目标 CloudFormation 堆栈以及与之相关的额外资源。

图表显示了以下工作流：

1. 收集资源 — 自动化框架使用命名约定返回所有相关的 CloudFormation 堆栈、Amazon Elastic Container Registry (Amazon ECR) Registry 存储库、DynamoDB 表和 Amazon S3 存储桶。

> **Note**
>
> 此阶段的函数使用分页器，这是 Boto3 中的一项功能，它抽象了对截断的 API 结果集进行迭代的过程。这样可以确保所有资源都得到处理。要进一步优化性能，请考虑应用服务器端筛选或考虑使用 JMESPath 来执行客户端筛选。

2. 预处理 — 自动化框架识别并解决了必须覆盖的服务限制， CloudFormation 才能允许删除资源。例如，它将 DynamoDB 表的`DeletionProtectionEnabled`设置更改为。`False`在命令行界面中，对于每种资源，您都会收到一条提示，询问您是否要覆盖约束。
3. 删除堆栈-自动化框架删除 CloudFormation 堆栈。在命令行界面中，您会收到一条提示，询问您是否要删除堆栈。
4. 后处理 — 自动化框架会删除未 CloudFormation 作为堆栈一部分直接配置的所有相关资源。这些资源类型的示例包括 Systems Manager 参数和 CloudWatch 日志组。单独的函数收集这些资源，对其进行预处理，然后将其删除。在命令行界面中，对于每种资源，您都会收到一条提示，询问您是否要删除该资源。

> **Note**
>
> 此阶段的函数使用分页器，这是 Boto3 中的一项功能，它抽象了对截断的 API 结果集进行迭代的过程。这样可以确保所有资源都得到处理。要进一步优化性能，请考虑应用服务器端筛选或考虑使用 JMESPath 来执行客户端筛选。

自动化和扩缩

如果您的 CloudFormation 堆栈包含示例代码中未包含的其他资源，或者堆栈存在此模式中未解决的约束条件，则可以根据自己的用例调整自动化框架。按照相同的方法收集资源、预处理、删除堆栈，然后进行后处理。

工具

AWS 服务

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。

- CloudFormation 命令行界面 (CFN-CLI) 是一款开源工具，可帮助您开发 AWS 和测试第三方扩展，然后注册它们以在中使用。 CloudFormation
- 适用于 Python (Boto3) 的 AWS SDK是一个软件开发套件，可帮助您将 Python 应用程序、库或脚本与集成 AWS 服务。

其他工具

- Cl@@ ic k 是一款 Python 工具，可帮助您创建命令行界面。
- P@@ o etry 是一款在 Python 中进行依赖管理和打包的工具。
- Pyenv 是一款可以帮助你管理和在 Python 版本之间切换的工具。
- Python 是通用的计算机编程语言。

代码存储库

此模式的代码可在 GitHub cloudformation-stack-cleanup存储库中找到。

最佳实践

- 为资源添加标签以便于识别 — 实施标记策略以识别为不同环境和目的创建的资源。标签可以帮助您根据标签筛选资源，从而简化清理过程。
- 设置资源生命周期-定义资源生命周期，以便在一段时间后自动删除资源。这种做法可以帮助您确保临时环境不会成为永久性的成本负担。

操作说明

安装工具

Task	描述	所需技能
克隆存储库。	1. 在虚拟环境中创建一个文件夹。用你的项目名称来命名。 2. 在本地计算机上打开终端，然后导航到该文件夹。	DevOps 工程师

Task	描述	所需技能
	3. 输入以下命令将cloudformation-stack-cleanup存储库克隆到您的项目目录： `git clone https://github.com/aws-samples/cloudformation-stack-cleanup.git`	
安装 Poetry。	按照说明（Poetry 文档）在目标虚拟环境中安装 Poetry。	DevOps 工程师

安装依赖项。	1. 输入以下命令导航到项目目录：
	`cd cloudformation-stack-cleanup`
	2. 输入以下命令：
	`poetry install`
	这将安装所有必需的依赖项，例如 Boto3、cl ick 和 CLI CloudFormation 的源代码。
		DevOps 工程师

（可选）安装 Pyenv。	按照说明 (GitHub) 安装 Pyenv。	DevOps 工程师

（可选）自定义框架

Task	描述	所需技能
创建用于收集、预处理和删除目标资源的函数。	1. 在克隆的存储库中，输入以下命令以导航到该`cli`目录： `cd cfncli/cli` 2. 打开 cleanup_enviornment.py 文件。 3. 创建一个新的 Python 函数，用于收集要修改的资源类型。有关示例，请参阅此文件中的`gather_ddb_tables` 函数。 4. 创建一个新的 Python 函数，该函数将覆盖目标资源的服务限制。有关示例，请参阅此文件中的`remove_ddb_deletion_protection` 函数。 5. 创建一个收集非托管目标资源的新 Python 函数。有关示例，请参阅此文件中的`gather_log_groups` 函数。 6. 创建一个用于删除非托管目标资源的新 Python 函数。有关示例，请参阅此文件中的`delete_log_group` 函数。 7. 保存并关闭 cleanup_enviornment.py 文件。	DevOps 工程师，Python

创建示例资源

Task	描述	所需技能
创建堆 CloudFormation 栈。	1. 导航到 项目目录。 2. 输入以下命令来创建用于配置 DynamoDB 表和安全组的 CloudFormation 堆栈。更新以下项的值<VPCID>： `aws cloudformation create-stack \ --stack-name sampleforcleanup-Stack \ --template-body file://samples/sample-cfn-stack.yaml \ --parameters ParameterKey=VpcId,ParameterValue=<VPCID> \ --region us-east-1`	AWS DevOps
创建 Systems Manager 参数。	输入以下命令来创建未通过 CloudFormation配置的 Systems Manager 参数： `aws ssm put-parameter \ --name "/samplefor cleanup/database/password" \ --value "your_db_password" \ --type "SecureString" \ --description "Database password for my app" \ --tier "Standard" \`	AWS DevOps

Task	描述	所需技能
	`--region "us-east-1"`	
创建 Amazon S3 存储桶。	输入以下命令来创建未通过 CloudFormation配置的 Amazon S3 存储桶： `aws s3api create-bucket \ --bucket samplesorcleanup-unmanagedbucket-<UniqueIdentifier> \ --region us-east-1 \ --create-bucket-configuration LocationConstraint=us-east-1`	AWS DevOps

删除示例资源

Task	描述	所需技能
删除 CloudFormation 堆栈。	1. 输入以下命令以删除您创建的示例 CloudFormation 堆栈、Systems Manager 参数和 Amazon S3 存储桶： `cfncli --region us-east-1 \ dev cleanup-env \ --prefix-list sampleforcleanup` 2. 出现提示时，Y请输入继续。	AWS DevOps
验证资源删除。	在输出中，确认所有示例资源均已删除。有关输出示例，请	AWS DevOps

Task	描述	所需技能
	参阅此模式的 “其他资源” 部分。	

相关资源

- 删除堆栈（CloudFormation 文档）
- 故障排除 CloudFormation（CloudFormation 文档）
- 授予 Lambda 函数访问亚马逊 VPC 中资源的权限（Lambda 文档）
- 如何删除停留在 DELETE_FAILED 状态的 AWS CloudFormation 堆栈？（AWS 知识中心）

其他信息

以下是该`cfncli`命令的输出示例：

```
cfncli --region aus-east-1  dev cleanup-env --prefix-list sampleforcleanup


https://sts.us-east-1.amazonaws.com
Cleaning up: ['sampleforcleanup'] in xxxxxxxxxx:us-east-1
Do you want to proceed? [Y/n]: Y
No S3 buckets
No ECR repositories
No Lambda functions in VPC
The following DynamoDB tables will have their deletion protection removed:
sampleforcleanup-MyDynamoDBTable
Do you want to proceed with removing deletion protection from these tables? [Y/n]: Y
Deletion protection disabled for DynamoDB table 'sampleforcleanup-MyDynamoDBTable'.
The following CloudFormation stacks will be deleted:
sampleforcleanup-Stack
Do you want to proceed with deleting these CloudFormation stacks? [Y/n]: Y
Initiated deletion of CloudFormation stack: `sampleforcleanup-Stack`
Waiting for stack `sampleforcleanup-Stack` to be deleted...
CloudFormation stack `sampleforcleanup-Stack` deleted successfully.
The following ssm_params will be deleted:
/sampleforcleanup/database/password
Do you want to proceed with deleting these ssm_params? [Y/n]: Y
Deleted SSM Parameter: /sampleforcleanup/database/password
```

```
Cleaned up: ['sampleforcleanup']
```

使用和自动进行动态管道管理，以便在 Gitflow 环境中部署修补程序解决方案 AWS Service CatalogAWS CodePipeline

由 Balaji Vedagiri (AWS)、Faisal Shahdad (AWS)、Shanmugam Shanker (AWS) 和 Vivek Thangamuthu (AWS) 创作

摘要

> **Note**
>
> AWS CodeCommit 不再向新客户提供。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

此模式解决了管理动态修补程序管道的情况，该管道专门用于将修补程序解决方案安全地部署到生产环境。该解决方案是通过使用产品 AWS Service Catalog 组合和产品来实施和管理的。Amazon EventBridge 规则用于事件自动化。通过使用 Service Catalog 产品组合约束和 AWS Identity and Access Management (IAM) 角色为开发人员强制执行限制。只有一个 AWS Lambda 函数可以启动由 EventBridge 规则触发的 Service Catalog 产品。此模式专为具有特定 Gitflow 设置的环境而设计，详见附加信息。

通常，部署修补程序是为了解决在实时环境（例如生产）中报告的关键问题或安全问题。修补程序应仅直接部署到暂存和生产环境。暂存和生产管道被广泛用于常规开发请求。这些管道不能用于部署修补程序，因为质量保证中有一些持续的功能无法升级到正式版。为了发布修补程序，此模式描述了具有以下安全功能的动态、短暂的管道：

- 自动创建-每当在存储库中创建修补程序分支时，都会自动创建修补程序管道。 AWS CodeCommit
- 访问限制-开发人员无权在修补程序流程之外创建此管道。
- 受控阶段 — 管道具有带有特殊访问令牌的受控阶段，确保拉取请求 (PR) 只能创建一次。
- 批准阶段 — 审批阶段包括在流程中，以获得相关利益相关者的必要批准。
- 自动删除-每当仓库中的分支与 PR 合并后，只要删除 CodeCommit 存储库中的`hotfix`分支，就会自动删除该修补程序管道。

先决条件和限制

先决条件

- 需要三个 AWS 账户 处于活动状态，如下所示：
  - 工具账户-用于持续集成和持续交付 (CI/CD) 设置。
  - 暂存账户-用于用户验收测试。
  - 生产账户-适用于企业最终用户。
  - （可选）添加一个 AWS 账户 以充当 QA 帐户。如果您想要同时设置主管道（包括 QA）和用于测试的修补程序管道解决方案，则需要此帐户。
- 具有可选条件的 AWS CloudFormation 堆栈，可根据需要使用主管道在 QA 账户中部署。通过创建和删除`hotfix`分支，仍然可以在没有主管道设置的情况下测试该模式。
- 亚马逊简单存储服务 (Amazon S3) 存储桶，用于存储 CloudFormation 用于创建服务目录产品的模板。
- 根据合规性要求为 CodeCommit 存储库创建 PR 批准规则（在创建存储库之后）。
- 限制开发人员和团队负责人的 IAM 权限以拒绝执行 prcreation-lambda Lambd a 函数，因为只能从管道中调用该函数。

限制

- 在部署阶段使用 CloudFormation 提供程序，使用 CloudFormation 更改集部署应用程序。如果要使用其他部署选项，请根据需要修改 CodePipeline 堆栈。
- 此模式使用 AWS CodeBuild 和其他配置文件来部署示例微服务。如果您有不同的工作负载类型（例如，无服务器工作负载），则必须更新所有相关配置。
- 这种模式将应用程序部署在单个 AWS 区域（例如，美国东部（弗吉尼亚北部）us-east-1）中。AWS 账户要跨多个区域进行部署，请更改命令和堆栈中的区域引用。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅按地区划分的 AWS 服务。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

架构

本节中的图表提供了创建生命周期事件和删除生命周期事件的工作流程。

上面创建生命周期事件的图表显示了以下内容：

1. 开发人员在 CodeCommit 存储库中创建一个`hotfix-*`分支来开发与 hotfix 相关的解决方案。

2. 分hotfix-*支创建事件是通过 EventBridge 规则捕获的。事件详细信息包括存储库名称和分支名称。
3. 该 EventBridge 规则调用该函数。 AWS Lambda hotfix-lambda-function该 EventBridge 规则将事件信息作为输入传递给 Lambda 函数。
4. Lambda 函数处理输入以检索存储库名称和分支名称。它使用从已处理的输入中检索到的值启动 Service Catalog 产品。
5. Service Catalog 产品包括将解决方案部署到舞台和生产环境的管道设置。管道区块包括源代码、构建和部署阶段。此外，还有一个手动批准阶段，可以促进生产环境的部署。
6. 源阶段从第一步中创建的存储库和hotfix-*分支中检索代码。该代码通过 Amazon S3 存储桶传递到构建阶段，用于存放工件。在构建阶段，将创建一个容器映像，其中包含在hotfix-*分支中开发并推送到亚马逊弹性容器注册表 (Amazon ECR) 中的修补程序。
7. 阶段环境的部署阶段使用包含此修补程序的最新容器映像更新亚马逊弹性容器服务 (Amazon ECS)。此修补程序是通过创建和执行 CloudFormation 更改集来部署的。
8. 在舞台环境中成功部署后，将调用 prcreation-lambda Lambda 函数。此 Lambda 函数创建从hotfix-*分支到存储库develop和分main支的 PR。Lambda 函数可确保hotfix-*分支中开发的修复程序被回溯并包含在后续部署中。
9. 手动批准阶段有助于确保必要的利益相关者审查修复并批准在生产环境中部署。
10. 生产环境的部署阶段使用包含此修补程序的最新容器映像更新 Amazon ECS。此修补程序是通过创建和执行 CloudFormation 更改集来部署的。

上面的删除生命周期事件的图表显示了以下内容：

1. 成功将修补程序部署到生产环境后，开发人员会删除该hotfix-*分支。
2. 分hotfix-*支删除事件是通过 EventBridge 规则捕获的。事件详细信息包括存储库名称和分支名称。
3. 该 EventBridge 规则调用 Lambda 函数。该 EventBridge 规则将事件信息作为输入传递给 Lambda 函数。
4. Lambda 函数处理输入以检索存储库名称和分支名称。Lambda 函数根据传递的输入确定相应的 Service Catalog 产品，然后终止该产品。
5. Service Catalog 预配置的产品终止会删除之前在该产品中创建的管道和相关资源。

自动化和扩缩

- 该模式包括一个 EventBridge 规则和一个 Lambda 函数，它们可以并行处理多个修补程序分支创建请求。Lambda 函数为匹配的事件规则预配置 Service Catalog 产品。
- 管道设置通过使用提供版本控制功能的 Service Catalog 产品来处理。该解决方案还可以自动扩展，以并行处理同一应用程序的多个修补程序开发。
- prcreation-lambda 函数可确保通过自动创建拉取请求，将这些修补程序更改合并回`main`和`develop`分支中。这种方法对于使和`develop`分支保持最新状态`main`并修复所有问题并避免潜在的代码回归至关重要。此过程有助于保持各分支之间的一致性，并通过确保所有长期存在的分支都有最新的修复程序来防止代码回归。

工具

AWS 服务

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。
- AWS CodeBuild是一项完全托管的生成服务，可帮助您编译源代码、运行单元测试和生成可随时部署的工件。
- AWS CodeCommit是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。 AWS CodeCommit 不再向新客户提供。的现有客户 AWS CodeCommit 可以继续照常使用该服务。有关更多信息，请参阅如何将 AWS CodeCommit 仓库迁移到其他 Git 提供商。
- AWS CodePipeline帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件更改所需的步骤。
- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- Amazon Elastic Container Service (Amazon ECS)是一项快速且可扩展的容器管理服务，可帮助运行、停止和管理集群上的容器。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥以帮助保护您的数据。
- AWS Service Catalog帮助您集中管理已获批准的 IT 服务目录。 AWS最终用户可在遵循组织设定约束的情况下快速部署他们所需已获得批准的 IT 服务。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

其他工具

- AWS CloudFormation Linter（cfn-lint）是一款根据资源规范检查 CloudFormation YAML 或 JSON 模板的 linter。CloudFormation 它还会执行其他检查，例如检查资源属性的有效值以及是否遵守最佳实践。
- cfn-nag 是一个开源工具，它通过搜索模式来识别 CloudFormation 模板中的潜在安全问题。
- Docker 是一组平台即服务（PaaS）产品，它们使用操作系统级别的虚拟化技术在容器中交付软件。该模式使用 Docker 在本地构建和测试容器映像。
- Git 是开源分布式版本控制系统。

代码存储库

此模式的代码可在 d GitHub ynamic_hotfix_ codepipeline 存储库中找到。

最佳实践

查看并调整您环境中的 IAM 角色和服务控制策略 (SCP)，以确保它们适当地限制访问权限。这对于防止任何可能超越此模式中包含的安全措施的行为至关重要。遵循最低权限原则，授予执行任务所需的最低权限。有关更多信息，请参阅 IAM 文档中的授予最低权限和安全最佳实践。

操作说明

设置工作环境

Task	描述	所需技能
克隆存储库。	要将示例存储库克隆到工作地点的新目录中，请运行以下命令： `git clone git@github.com:aws-samples/dynamic_hotfix_codepipeline.git`	AWS DevOps
导出用于 CloudFormation 堆栈部署的环境变量。	定义以下环境变量，这些变量将在此模式的后面用作 CloudFormation 堆栈的输入。 • `ApplicationName` — 此变量用于命名为应用	AWS DevOps

Task	描述	所需技能

	程序创建的资源，便于跟踪这些资源。使用以下命令，Applicationname 替换为实际的应用程序名称：
	
	`export ApplicationName=<Applicationname>`
	
	• BucketStartName — 此变量用于命名 Amazon S3 存储桶。S3 存储桶名称在所有存储桶中必须是全局唯一的 AWS 账户。使用以下命令，BucketName 替换为 S3 存储桶的唯一名称：
	
	`export BucketStartName=<BucketName>`
	
	• 账号和区域-这些变量存储不同环境和部署区域的账 AWS 账户 号。使用以下命令，将占位符（例如prodaccountnumber 和region）替换为实际 AWS 账户 数字和正在 AWS 区域 使用的数字。
	
	Note
	QAAccount 为可选项。如果要使用QAAccount ，请	

Task	描述	所需技能

	使用主管道堆栈的参数进行设置。
	
	`export ProdAccount=<prodaccountnumber>`
	`export StageAccount=<stage/preprodaccountnumber>`
	`export QAAccount=<qaccountnumber>`
	`export ToolsAccount=<toolsaccountnumber>`
	`export DepRegion=<region>`	

设置中必需的先决条件 AWS 账户

Task	描述	所需技能

在工具账户中创建 CI/CD 所需的资源。	要在工具账户中部署 CloudFormation 堆栈，请使用以下命令。（如果您未使用 QA 帐户进行设置，请删除该`QAAccount`参数。）
	
	`#InToolsAccount`
	`aws cloudformation deploy \`
	`--template-file pre-requisites/pre-reqs.yaml \`
	`--stack-name prereqs \`
		AWS DevOps

Task	描述	所需技能
	--parameter-overrides BucketStartName=${BucketStartName} \ ApplicationName=${ApplicationName} ProdAccount=${ProdAccount} \ StageAccount=${StageAccount} ToolsAccount=${ToolsAccount} \ QAAccount=${QAAccount} \ --capabilities CAPABILITY_IAM CAPABILITY_NAMED_IAM --region ${DepRegion} 记下 CodeCommit 存储库和 Amazon ECR 从前面的堆栈中创建的资源。在接下来的步骤中，这些参数是设置管道main分支所必需的。	

Task	描述	所需技能

在工作负载帐户中创建 CI/CD 所需的资源。	1. 要将 CloudFormation 模板打包到每个工作负载帐户（阶段、生产和可选 QA）中，请使用以下命令。在以下命令中，S3bucketpackage 替换为要用于打包的 Amazon S3 存储桶名称。
	
	`#InStageAccount`
	`aws cloudformation package \`
	`--template-file pre-requisites/infrastack.yaml \`
	`--s3-bucket <S3bucketpackage> \`
	`--s3-prefix infraStack \`
	`--region ${DepRegion} \`
	`--output-template-file pre-requisites/infrastructure_stage.template`
	
	`#InProdAccount`
	`aws cloudformation package \`
	`--template-file pre-requisites/infrastack.yaml \`
	`--s3-bucket <S3bucketpackage> \`
	`--s3-prefix infraStack \`
	`--region ${DepRegion} \`
		AWS DevOps

Task	描述	所需技能
	--output-template-file pre-requisites/infrastructure_prod.template 2. 要在每个工作负载帐户中部署 CloudFormation 模板，请使用以下命令： #InStageAccount aws cloudformation deploy --stack-name inframainstack \ --parameter-overrides ApplicationName=${ApplicationName} ToolsAccount=${ToolsAccount} DepRegion=${DepRegion} \ --template-file pre-requisites/infrastructure_stage.template --region ${DepRegion} --capabilities CAPABILITY_NAMED_IAM #InProdAccount aws cloudformation deploy --stack-name inframainstack \ --parameter-overrides ApplicationName=${ApplicationName} ToolsAccount=${ToolsAccount} DepRegion=${DepRegion} \	

Task	描述	所需技能
	`--template-file pre-requisites/infrastructure_prod.template --region ${DepRegion} --capabilities CAPABILITY_NAMED_IAM`	

Task	描述	所需技能

更新 S3 工件存储桶策略以允许工作负载帐户进行访问。	要更新工具帐户中的 CloudFormation 堆栈先决条件，请使用以下命令为阶段和生产工作负载帐户添加所有必需的权限。（如果您不使用该QAAccount 参数进行设置，请将其删除。）
	
	`#InToolsAccount`
	`aws cloudformation deploy \`
	`--template-file pre-requisites/pre-reqs.yaml \`
	`--stack-name prereqs \`
	`--parameter-overrides BucketStartName=${BucketStartName} \`
	`ApplicationName=${ApplicationName} ProdAccount=${ProdAccount} \`
	`StageAccount=${StageAccount} ToolsAccount=${ToolsAccount} \`
	`QAAccount=${QAAccount} PutPolicy=true \`
	`--capabilities CAPABILITY_IAM CAPABILITY_NAMED_IAM --region ${DepRegion}`
		AWS DevOps

在工具账户中设置 Lambda 函数和 Service Catalog 资源

Task	描述	所需技能
设置 Service Catalog 产品组合和产品。	要设置 Service Catalog 产品组合和产品，请执行以下操作： 1. 将模板 pipeline-main.yaml 和 pipeline-hotfix.yaml 从目录中的存储库上传到您要部署 CodePipeline 到的区域中的现有 Amazon S3 存储桶 `Bucketname` () 中 ()。`DepRegion` ``` #InToolsAccount aws s3 cp ./codepip eline/pipeline-mai n.yaml s3://<Buc ketname>/pipeline- main.yaml aws s3 cp ./codepip eline/pipeline-hot fix.yaml s3://<Buc ketname>/pipeline- hotfix.yaml ``` 2. 设置用于管理和`hotfix`分支机构管道的 Service Catalog 产品组合`main`和产品。记下来自以下堆栈的`Outputs`部分`MainProdu ctId` 和`MainProdu ctArtifactId` 来自以下堆栈的详细信息。在`main`分支的管道设置过程中，后续步骤中需要这些信息。	AWS DevOps

Task	描述	所需技能

	<pre>#InToolsAccount
	aws cloudformation
	 deploy \
	    --template-file
	 pre-requisites/ser
	vicecatalogsetup.y
	aml \
	    --stack-name
	 servicecatalogsetup
	 \
	    --parameter-
	overrides TemplateB
	ucket=<Bucketname> \
	    --capabilities
	 CAPABILITY_IAM
	 CAPABILITY_NAMED_I
	AM --region ${DepRegi
	on}</pre> 3. 为 IAM 角色提供访问权限，该角色将工具账户中的资源部署到 Service Catalog 产品组合的主管道产品组合。使用此产品组合通过`main`分支部署应用程序。有关如何提供访问权限的更多信息，请参阅 Service Catalog 文档中的向用户授予访问权限。	

Task	描述	所需技能
设置 Lambda 函数。	此解决方案使用以下 Lambda 函数来管理修补程序工作流程： • `hotfix-lambda-function` 在创建hotfix分支时处理 Service Catalog 产品配置。 • `hotfix-cleanup-lambda-function` 管理hotfix分支被删除时的产品终止。 • `prcreation-lambda`创建从分支到develop和hotfixmain分支的拉取请求。 要使 Lambda 函数能够在通过关联 EventBridge 规则创建或删除hotfix 分支时配置和终止 Service Catalog 产品，请使用以下步骤： 1. 要为 Lambda 函数创建 IAM 角色和权限，请使用以下命令部署堆栈： CloudFormation `#InToolsAccount aws cloudformation deploy \ --template-file pre-requisites/lambdasetup.yaml \ --stack-name prsclambdasetup \ --capabilities`	AWS DevOps

Task	描述	所需技能
	CAPABILITY_IAM CAPABILITY_NAMED_IAM --region ${DepRegion} 2. 堆栈部署完成后，使用授予对 Service Catalog 产品组合修补程序管道组合的`hotfix-lambda-execution-role` 访问权限。AWS Management Console此访问权限允许 Lambda 函数启动或终止修补程序分支的 Service Catalog 产品。	

为主分支创建管道并在工作负载帐户中部署应用程序

Task	描述	所需技能

为`main`分支设置管道。	要为主分支设置管道，请在工具帐户中运行以下命令。将`MainProductId` 和`MainProductArtifactId` 的参数替换为`servicecatalogsetup` 堆栈输出中的值。
	`#InToolsAccount`
	`aws servicecatalog provision-product \`
	`--product-id <MainProductId> \`
	`--provisioning-artifact-id <MainProductArtifactId> \`
		AWS DevOps

Task	描述	所需技能
	`--provisioned-product-name "${ApplicationName}-main-pipeline" \ --provisioning-parameters Key=CodeCommitRepoName,Value="${ApplicationName}-repository" Key=ECRRepository,Value="${ApplicationName}-app" \ --region=${DepRegion}`	

Task	描述	所需技能
使用main分支部署应用程序。	1. 要克隆在先决条件中创建的 CodeCommit 存储库，请使用命令。git clone有关更多信息，请参阅如 CodeCommit 文档中所述，CodeCommit 通过克隆存储库来连接存储库。 2. 将存储库中可用repotemplates 目录中的所有应用程序文件复制到本地存储库 clone (${ApplicationName}-repository) 中。修改以下文件以更新 ToolsAccount ID。在每个文件中，找到RegistryAccountid 参数并将其值设置为您的 ToolsAccount ID。将更改提交到 CodeCommit 存储库，然后将文件推送到main和develop分支。 • ecs-configuration-prod.yaml • ecs-configuration-qa.yaml • ecs-configuration-stage.yaml 3. 要验证应用程序部署，请使用监控 CodePipeline 执行情况 AWS Management Console。部署完成后，在舞台环境中使用 Application Load Balancer FQDN 访问	AWS DevOps

Task	描述	所需技能
	应用程序。确认应用程序按预期运行。 4. 要批准部署到生产环境，请使用 CodePipeline 控制台查找应用程序的管道。找到`ApprovalToStart` 舞台。审查更改，如果满意，则提供手动批准以继续进行生产部署。	

为 hotfix-* 分支创建管道并部署此修补程序

Task	描述	所需技能
创建`hotfix-*`分支并提交更改。	要为`hotfix-*`分支创建管道并将修补程序部署到工作负载帐户，请执行以下操作： 1. 使用以关键字开头的名称创建分支`hotfix`。例如，此模式使用 CodeCommit 应用程序存储库中的`hotfix-check1` 分支 (`${ApplicationName}-repository`)。有关更详细的步骤，请参阅 CodeCommit 文档中的 Connect 到 AWS CodeCommit 存储库和基本 Git 命令。 2. 验证是否已成功为`hotfix-check1` 分支机构动态配置 `Hotfix CICD Pipeline` Service Catalog 产品。预配置的产品名称以此修补	AWS DevOps

Task	描述	所需技能
	程序分支名称和应用程序 CodeCommit 存储库名称命名。 3. 在 index.html 文件中提交一些小改动，然后将其推送到 CodeCommit 存储库。 4. 验证在舞台环境中 CodePipeline 执行是否成功。要在生产环境中部署，请在中提供手动批准 CodePipeline。 5. 使用 Application Load Balancer 完全限定域名 (FQDN) 确认更改在应用程序主页上可见。FQDN 可在的`Outputs`部分中找到。`inframainstack-ALBStack-*`	
删除分`hotfix-check1` 支。	要删除之前创建的`hotfix-check1` 分支，请执行以下操作： 1. 删除 CodeCommit 应用程序存储库中的`hotfix-check1` 分支。 2. 验证为`hotfix-check1` 分支机构配置的 Service Catalog 产品是否成功终止。	AWS DevOps

清理资源

Task	描述	所需技能

清理已部署的资源。	要清理之前部署的资源，请执行以下操作：
	1. 要将 Amazon ECS 服务缩减到工作负载账户中的零副本，请使用以下命令：
	`aws ecs update-service --cluster ${ApplicationName}-Cluster --service ${ApplicationName}-Service-stage --desired-count 0 --region ${DepRegion}`
	`aws ecs update-service --cluster ${ApplicationName}-Cluster --service ${ApplicationName}-Service-prod --desired-count 0 --region ${DepRegion}`
	2. 终止为`main`分支机构配置的 Service Catalog 产品。
	3. 清理工具账户中在 Amazon S3 存储桶中创建的对象。删除注册表本身之前，请先删除 Amazon ECR 中的所有 Docker 镜像。
	4. 在删除 Service Catalog 产品组合之前，请移除 Service Catalog 产品组合的“已授予访问权限” 部分中的 IAM 角色。
		AWS DevOps

Task	描述	所需技能

	5. 删除部署在工具帐户和工作负载帐户中的 CloudFormation 堆栈。
	
	`##In Tools Account##`
	`aws cloudformation delete-stack --stack-name servicecatalogsetup --region ${DepRegion}`
	`aws cloudformation delete-stack --stack-name prlambdasetup --region ${DepRegion}`
	`aws cloudformation delete-stack --stack-name prereqs --region ${DepRegion}`
	
	`##In Workload Accounts##`
	`aws cloudformation delete-stack --stack-name inframainstack --region ${DepRegion}`
	
	有关更多信息，请参阅 Service Catalog 文档中的删除预配置产品。	

故障排除

事务	解决方案
您提交到 CodeCommit 存储库的更改尚未部署。	检查 CodeBuild 日志，查看 Docker 构建操作中是否存在错误。有关更多信息，请参阅 CodeBuild 文档。
未配置 Service Catalog 产品。	查看相关 CloudFormation 堆栈中是否存在失败的事件。有关更多信息，请参阅 CloudFormation 文档。

相关资源

- 基本 Git 命令
- 配置 IAM 策略以限制向分支的推送和合并
- Connect 连接到 AWS CodeCommit 存储库
- 向用户授予访问权限
- 将 Docker 镜像推送到亚马逊 ECR 私有存储库
- 故障排除 AWS CodeBuild
- 什么是 AWS CodePipeline？

其他信息

此模式专为具有 Gitflow 设置的环境而设计，该设置用于CI/CD process. The pipelines follow the deployment cycle that starts from development and moves through quality assurance (QA), stage, and production environments. The CI/CD设置中的开发工作流程，包括两个 git 分支，这些分支对环境进行了促销部署，如下所示：

- 分`develop`支部署到开发环境中。
- 该`main`分支部署到 QA、舞台和生产环境。

在这种设置中，在积极开发新功能的同时，要比通常的部署周期更快地应用修补程序或安全补丁是一项挑战。需要专门的流程来处理修补程序或安全请求，从而确保实时环境保持正常运行和安全。

但是，在以下情况下，您可以使用其他可用选项而无需专门的部署流程：

- 对于开发人员来说，对生产环境进行小修补的CI/CD process is well-equipped with automated testing, such as functional and end-to-end tests, which eliminate the need for manual testing and prevent delays in deployments to production. However, if automated testing isn't well integrated into the CI/CD过程可能会变得复杂而繁琐。这是因为 QA 环境中可能有新功能等待批准和签署。修补程序或安全补丁无法以简单的方式同时投入生产。
- 开发团队不断将新功能部署到生产环境中，将修补程序或安全补丁集成到每个新功能的计划部署中。换句话说，生产环境的下一个功能更新包括两个组件：添加新功能和包含修补程序或安全补丁。但是，如果部署周期不连续，则 QA 环境中可能有多个新功能正在等待批准。管理不同的版本并确保重新应用正确的更改会变得复杂且容易出错。

> **Note**
>
> 如果您使用的是版本 2，并在`hotfix`分支上设置了适当的触发器，则仍然需要一个专门的流程来处理计划外的请求。 AWS CodePipeline 在版本 2 中，您可以为推送或拉取请求设置触发器。执行将排队或立即执行，具体取决于管道的先前状态。但是，通过专用管道，修复程序会立即应用于生产环境，从而确保紧急问题毫不拖延地得到解决。

使用 Terraform 在亚马逊托管 Grafana 上自动提取和可视化亚马逊 MWAA 自定义指标

由 Faisal Abdullah (AWS) 和 Satya Vajrapu (AWS) 创作

摘要

此模式讨论如何使用亚马逊托管 Grafana 创建和监控适用于 Apache Airflow 的亚马逊托管工作流程（亚马逊 MWAA）采集的自定义指标。Amazon MWAA 充当工作流程的协调器，它使用用 Python 编写脚本的有向无环图 (DAGs)。这种模式以监控自定义指标为中心，包括过去一小时内 DAGs 运行的总数、 DAGs 每小时通过和失败的计数以及这些进程的平均持续时间。此分析显示了 Amazon Managed Grafana 如何与 Amazon MWAA 集成，从而能够全面监控和洞察该环境中的工作流程编排。

先决条件和限制

先决条件

- AWS 账户 具有创建和管理以下内容所需的用户权限的活跃用户 AWS 服务：
  - AWS Identity and Access Management (IAM) 角色和策略
  - AWS Lambda
  - Amazon Managed Grafana
  - Amazon Managed Workflows for Apache Airflow (Amazon MWAA)
  - Amazon Simple Storage Service（Amazon S3）
  - Amazon Timestream
- 访问外壳环境，该环境可以是本地计算机上的终端或 AWS CloudShell.
- 安装了 Git 并安装和配置了最新版本的 AWS Command Line Interface (AWS CLI) 的 shell 环境。有关更多信息，请参阅 AWS CLI 文档 AWS CLI中的安装或更新到最新版本的。
- 安装了以下 Terraform 版本：`required_version = ">= 1.6.1, < 2.0.0"`你可以使用 tfswitch 在不同版本的 T erraform 之间切换。
- 已在中 AWS IAM Identity Center 为您配置身份源 AWS 账户。有关更多信息，请参阅 IAM 身份中心文档中的在 IAM 身份中心确认您的身份来源。您可以从默认 IAM Identity Center 目录的 Active Directory 或外部身份提供者 (IdP) 中进行选择，例如 Okta。有关更多信息，请参阅相关资源。

限制

- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

产品版本

- Terraform `required_version = ">= 1.6.1, < 2.0.0"`
- 亚马逊 Managed Grafana 版本 9.4 或更高版本。此模式已在版本 9.4 上进行了测试。

架构

以下架构图突出显示了解决方案中 AWS 服务 使用的架构。

前面的图表逐步完成了以下工作流程：

1. Amazon MWAA 中的自定义指标源 DAGs 于在环境中执行的指标。这些指标以 CSV 文件格式上传到 Amazon S3 存储桶。以下内容 DAGs 使用 Amazon MWAA 的数据库查询功能：
   - `run-example-dag`— 此 DAG 包含定义一个或多个任务的示例 Python 代码。它每 7 分钟运行一次，并打印日期。打印日期后，DAG 会包含一项在特定持续时间内休眠或暂停执行的任务。
   - `other-sample-dag`— 此 DAG 每 10 分钟运行一次，并打印日期。打印日期后，DAG 会包含一项在特定持续时间内休眠或暂停执行的任务。
   - `data-extract`— 此 DAG 每小时运行一次，查询 Amazon MWAA 数据库并收集指标。收集指标后，此 DAG 会将其写入 Amazon S3 存储桶以供进一步处理和分析。
2. 为了简化数据处理，Lambda 函数在 Amazon S3 事件触发时运行，这便于将指标加载到 Timestream 中。
3. Timestream 作为数据源集成到亚马逊托管 Grafana 中，来自亚马逊 MWAA 的所有自定义指标都存储在那里。
4. 用户可以查询数据并构建自定义控制面板，以可视化关键绩效指标，深入了解 Amazon MWAA 中的工作流程编排。

工具

AWS 服务

- AWS IAM Identity Center帮助您集中管理对所有应用程序 AWS 账户 和云应用程序的单点登录 (SSO) 访问权限。

- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。在这种模式下， AWS Lambda 运行 Python 代码以响应 Amazon S3 事件并自动管理计算资源。
- Amazon Man aged Grafana 是一项完全托管的数据可视化服务，您可以使用它来查询、关联和可视化您的指标、日志和跟踪并发出警报。这种模式使用 Amazon Managed Grafana 创建用于指标可视化和提醒的控制面板。
- 适用于 Apache Airflow 的亚马逊托管工作流程（亚马逊 MWAA）是一项适用于 Apache Airflow 的托管编排服务，可用于在云中大规模设置和运营数据管道。Apache Airflow 是一种开源工具，用于以编程方式编写、安排和监视被称为工作流程的流程和任务序列。在这种模式中，示例 DAGs 和指标提取器 DAG 部署在 Amazon MWAA 中。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。在这种模式中，Amazon S3 用于以 CSV 格式存储 DAGs、脚本和自定义指标。
- Amazon Timestream for LiveAnalytics 是一个快速、可扩展、完全托管、专门构建的时间序列数据库，可轻松存储和分析每天数万亿个时间序列数据点。Timestream LiveAnalytics 还与用于数据收集、可视化和机器学习的常用服务集成。在这种模式中，它用于提取生成的 Amazon MWAA 自定义指标。

其他工具

- HashiCorp Terraform 是一款基础设施即代码 (IaC) 工具，可帮助您使用代码来配置和管理云基础架构和资源。此模式使用 Terraform 模块在中自动配置基础架构。 AWS

代码存储库

此模式的代码可在 visualize-amazon-mwaa-custom-metrics- grafana 存储库 GitHub 中找到。该`stacks/Infra`文件夹包含以下内容：

- 所有资源的 Terraform 配置文件 AWS
- 文件夹中的 Grafana 仪表板.json 文件 `grafana`
- 文件夹中包含 Apache Airflow 的亚马逊托管工作流程 DAGs `mwaa/dags`
- Lambda 代码，用于解析.csv 文件并将指标存储在 Timestream 数据库中的文件夹中 `src`
- IAM 策略文件夹中的.json 文件 `templates`

最佳实践

Terraform 必须存储有关您的托管基础设施和配置的状态，这样它才能将现实世界的资源映射到您的配置。默认情况下，Terraform 将状态存储在本地名为的文件中。`terraform.tfstate`确保 Terraform 状态文件的安全性和完整性至关重要，因为它可以保持基础架构的当前状态。有关更多信息，请参阅 Terraform 文档中的远程状态。

操作说明

使用 Terraform 部署基础架构

Task	描述	所需技能

部署基础设施。	要部署解决方案基础架构，请执行以下操作：
	1. 在本地计算机上或使用打开终端或命令提示符 AWS CloudShell。
	2. 导航到要克隆存储库的目录。
	3. 要克隆存储库，请运行以下命令：
	`git clone https://github.com/aws-samples/visualize-amazon-mwaa-custom-metrics-grafana`
	4. 克隆过程完成后，运行以下命令以导航到克隆的存储库目录：
	`cd visualize-amazon-mwaa-custom-metrics-grafana/stacks/infra`
	5. 要下载并初始化所需的提供程序，请运行以下命令：
		AWS DevOps

Task	描述	所需技能

	`terraform init`
	6. 要全面了解 Terraform 将创建的所有资源，请运行以下命令：
	`terraform plan`
	Terraform 提供了以下资源：
	• 亚马逊 Virtual Private Cloud（亚马逊 VPC）和相关的网络组件
	• Amazon S3 资源
	• AWS Lambda 函数
	• Amazon 托管 Grafana 资源（工作空间、控制面板、数据源）
	• 支持 IAM 资源（角色和策略）
	7. 要根据计划输出创建 AWS 资源，请运行以下命令：
	`terraform apply -auto-approve`
	基础设施配置将在大约 20 分钟内完成。
	8. 要根据 Terraform 文件中定义的配置创建指定的 AWS 资源，请运行以下命令：	

Task	描述	所需技能
	`terraform apply`	

验证已部署的基础架构资源

Task	描述	所需技能
验证 Amazon MWAA 环境。	要验证 Amazon MWAA 环境，请执行以下操作： 1. 登录 AWS Management Console，导航至 Amazon MWAA 控制面板控制台，然后选择打开 Air flow 用户界面。 2. 您应该看到以下三个处 DAGs 于 “活动” 状态： • 数据提取 • run-example-dag • other-sample-dag 3. 如果 DAG 未处于活动状态，则可以通过启用 DAG 名称旁边的切换开关将其激活。	AWS DevOps，数据工程师
验证 DAG 时间表。	要查看每个 DAG 时间表，请转到 A irflow 用户界面中的 “时间表” 选项卡。 以下 DAGs 每项都有预先配置的计划，该计划在 Amazon MWAA 环境中运行并生成自定义指标：	AWS 数据工程师 DevOps

Task	描述	所需技能
	• run-example-dag-每 7 分钟运行一次 • other-sample-dag-每 10 分钟运行一次 • 数据提取-每小时运行一次 您还可以在运行列下查看每个 DAG 的成功运行情况。	

配置亚马逊托管 Grafana 环境

Task	描述	所需技能
配置对亚马逊托管 Grafana 工作区的访问权限。	Terraform 脚本创建了所需的亚马逊托管 Grafana 工作空间、控制面板和指标页面。要配置访问权限以便您可以查看它们，请执行以下操作： 1. 打开 Amazon Managed Grafana 控制台。 2. 在 Workspaces 中，选择工作区`grafana-ws-dev` ，然后导航到下部窗格中的“身份验证”选项卡。 3. 选择“分配新用户或群组”按钮。 4. 在“群组”选项卡中添加您的群组或在“用户”选项卡中添加用户，然后选择“分配用户和群组”按钮。 5. 添加用户（或群组）后，将该用户（或群组）设为管理	AWS DevOps

Task	描述	所需技能
	员。在 “分配的用户组” 选项卡的 “分配的用户或组” 中选择用户，然后从下拉菜单中选择 “设为管理员”。有关更多信息，请参阅亚马逊托管 Grafana 文档中的在亚马逊托管 Grafana 工作空间中使用 AWS IAM Identity Center。 6. 导航到工作区，然后选择 Grafana 工作空间网址。要以管理员身份登录亚马逊托管 Grafana，请选择登录方式。 AWS IAM Identity Center	

Task	描述	所需技能
安装亚马逊 Timestream 插件。	Amazon MWAA 自定义指标已加载到 Timestream 数据库中。您可以使用 Timestream 插件通过 Amazon Managed Grafana 控制面板对指标进行可视化。 要安装 Timestream 插件，请执行以下操作： 1. 在 Amazon Managed Grafana 控制台中，展开左侧导航窗格中的菜单，然后转到管理、插件。 2. 搜索并安装最新版本的亚马逊 Timestream 插件。 3. 安装插件后，前往管理、数据源以查看 Timestream 数据源。如果未列出数据源，请刷新页面。 有关更多信息，请参阅 Amazon Managed Grafana 文档中的使用插件扩展您的工作空间。	AWS DevOps， DevOps 工程师

在亚马逊托管 Grafana 控制面板中可视化自定义指标

Task	描述	所需技能
查看亚马逊托管 Grafana 控制面板。	要查看采集到亚马逊托管 Grafana 工作区的指标，请执行以下操作：	AWS DevOps

Task	描述	所需技能
	1. 在 Amazon Managed Grafana 控制台中，选择左侧导航窗格中的控制面板。 2. 要查看指标，请选择 MWAA 事件仪表板，然后选择 mwaa _metrics。 仪表板指标页面显示以下信息： • 过去一小时内的 DAG 运行总数 • 在过去一小时内成功运行、失败和正在运行的 DAG 总数 • 所有 DAG 运行、成功运行和失败运行的平均持续时间	

Task	描述	所需技能
自定义亚马逊托管 Grafana 控制面板。	要自定义仪表板以备将来进一步增强，请执行以下操作： 1. 在亚马逊托管 Grafana `mwaa_metrics` 控制面板页面上，选择控制面板设置图标。 2. 要查看定义仪表板的数据结构，请选择 JSON 模型。您可以通过直接在控制台中编辑此 JSON 模型来自定义控制面板。 或者，此仪表板的源代码可在GitHub 存储库`stacks/infra/grafana` 文件夹中的`dashboard.json` 文件中找到。	AWS DevOps

清理 AWS 资源

Task	描述	所需技能
暂停 Amazon MWAA DAG 的运行。	要暂停 DAG 运行，请执行以下操作： 1. 在 Amazon MWAA 控制台中，导航到 A irflow 环境并选择 Ope n Airflow 用户界面。 2. 要暂停 DAG，请使用每个 DAG 旁边的切换开关。	AWS DevOps，数据工程师

Task	描述	所需技能
	3. 刷新 Airflow 用户界面页面，该页面应在 “已暂停” DAGs 部分中列出三个。	
删除 Amazon S3 存储桶中的对象。	要删除 Amazon S3 存储桶 mwaa-events-bucket-* 和 mwaa-metrics-bucket-*，请按照 Amazon S3 文档中删除存储桶中有关使用 Amazon S3 控制台的说明进行操作。	AWS DevOps

Task	描述	所需技能
摧毁 Terraform 创建的资源。	要销毁由 Terraform 创建的资源和相关的本地 Terraform 状态文件，请执行以下操作： 1. （可选）在删除资源之前，您可以预览 Terraform 将要进行的更改。要生成计划，请运行以下命令： `terraform plan -destroy` 命令输出显示该`destroy`命令将删除之前创建的所有 AWS 资源。 2. `terraform destroy -auto-approve` 此命令大约需要 20 分钟才能摧毁基础架构。 **Note** 要销毁 Terraform 管理的所有资源，请运行以下命令。：`-auto-approve`标签不会等待用户确认后才开始销毁资源。 3. 要删除本地 Terraform 状态文件，请运行以下命令：	AWS DevOps

Task	描述	所需技能

	`rm .terraform.lock.hcl`
	`rm -rf .terraform`
	`rm terraform.tfstate*`	

故障排除

事务	解决方案
`null_resource.plugin_mgmt (local-exec): aws: error: argument operation: Invalid choice, valid choices are:`	AWS CLI 将您的版本升级到最新版本。
加载数据源错误- `Fetch error: 404 Not Found Instantiating…`	错误是间歇性的。等待几分钟，然后刷新您的数据源以查看列出的 Timestream 数据源。

相关资源

AWS 文档

- 用于仪表板和可视化的 Amazon Managed Grafana
- 将亚马逊托管 Grafana 配置为使用 Okta
- AWS IAM Identity Center 与您的亚马逊托管 Grafana 工作区配合使用
- DAGs 在 Amazon 上使用 MWAA

AWS 视频

- 使用 Amazon Managed Grafana 配置 IAM 身份中心进行身份验证，如以下视频所示。

https://www.youtube-nocookie.com/embed/XX2Xcz-Ps9U？ 控件=0

- 如果 IAM 身份中心不可用，您还可以使用外部身份提供商 (IdP)（例如 Okta）来集成 Amazon Managed Grafana 身份验证，如以下视频所示。

https://www.youtube-nocookie.com/embed/Z4 JHxl2xp Og？ 控件=0

其他信息

您可以为您的 Amazon MWAA 环境创建全面的监控和警报解决方案，从而实现主动管理并对潜在问题或异常情况做出快速响应。Amazon Managed Grafana 包括以下功能：

提醒 — 您可以根据预定义的阈值或条件在 Amazon Managed Grafana 中配置警报。设置电子邮件通知，以便在某些指标超过或低于指定阈值时提醒相关利益相关者。有关更多信息，请参阅亚马逊托管 Grafana 文档中的 Grafana 提醒。

集成 — 您可以将 Amazon Managed Grafana 与各种第三方工具（例如 PagerDuty、或 OpsGenie Slack）集成，以增强通知功能。例如，您可以设置 webhook 或与集成， APIs 以便根据在 Amazon Managed Grafana 中生成的警报在这些平台上触发事件和通知。此外，此模式还提供了用于创建 AWS 资源的GitHub 存储库。您可以进一步将此代码与您的基础架构部署工作流程集成。

使用自动化工作流程简化 Amazon Lex 机器人的开发和部署

由 Balaji Panneerselvam (AWS)、Anand Jumnani (AWS)、Attila Dancso (AWS)、James O'Hara (AWS) 和 Pavan Dusanapudi (AWS) 创作

摘要

当您尝试管理多个功能、开发人员和环境时，开发和部署 Amazon Lex 对话机器人可能具有挑战性。使用基础设施即代码 (IaC) 原则的自动化工作流程可以帮助简化流程。这种模式可以通过以下方式帮助提高 Amazon Lex 开发人员的工作效率，并实现高效的机器人生命周期管理：

- 支持多项功能的并行开发-通过自动化工作流程，开发人员可以在不同的分支中并行开发不同的功能。然后，可以在不阻碍其他工作的情况下合并和部署更改。
- 使用 Amazon Lex 控制台用户界面 ——开发人员可以使用用户友好的 Amazon Lex 控制台来构建和测试机器人。然后在用于部署的基础设施代码中描述这些机器人。
- 跨环境推广机器人 ——该工作流程可自动将机器人版本从开发和测试等较低的环境提升到生产环境。这种方法降低了手动促销的风险和开销。
- 维护版本控制-在 Git 中管理机器人定义，而不仅仅是通过 Amazon Lex 服务，可以为您提供版本控制和审计跟踪。与仅使用 AWS Management Console 或修改存储在中的机器人不同，更改会跟踪 APIs 到各个开发者 AWS。

通过自动化 Amazon Lex 机器人发布流程，团队可以更快地交付功能，同时降低风险和工作量。机器人仍处于版本控制之下，而不是在 Amazon Lex 控制台中被隔离。

先决条件和限制

先决条件

- 该工作流程涉及多个 AWS 账户 不同的环境（开发、生产和 DevOps），这需要账户管理和跨账户访问配置。
- Python 3.9 可在您的部署环境或管道中使用。
- 在本地工作站上安装并配置 Git 以进行源代码控制。
- AWS Command Line Interface (AWS CLI) 已安装并配置为使用命令行或 Python 进行身份验证。

限制

- 存储库访问权限-该工作流程假设持续集成和持续交付 (CI/CD) 管道具有向源代码存储库提交更改所需的权限。
- 初始机器人版本 — 该工具要求使用 AWS CloudFormation 模板部署机器人的初始版本。在自动工作流程接管之前，您必须创建机器人的第一个迭代并将其提交到存储库。
- 合并冲突 — 尽管工作流程旨在实现并行开发，但在整合来自不同分支的更改时，仍有可能出现合并冲突。解决机器人配置中的冲突可能需要手动干预。

产品版本

- Python 3.9 或更高版本
- AWS CDK v2 2.12 4.0 或更高版本
- 适用于 Python (Boto3) 的 AWS SDK1.28 或更高版本

架构

下图显示了该解决方案的高级架构和关键组件。

关键组件包括以下内容：

- Lex 机器人存储库 — 一个 Git 存储库，用于存储 Amazon Lex 机器人的 IaC 定义。
- DevOps— AWS 账户 专门用于存放 CI/CD 管道以及用于开发和部署过程的相关资源。
- Pipelin@@ es — 自动执行机器人开发和部署生命周期各个阶段的 AWS CodePipeline 实例，例如创建新机器人、导出机器人定义、导入机器人定义和删除机器人。
- 票务机器人和主机器人 — Amazon Lex 机器人资源，其中工单机器人是由各个团队或开发人员开发的特定功能机器人，主机器人是集成了所有功能的基准机器人。

架构图说明了以下工作流程：

1. 基准主机器人 — 工作流程的起点是在开发（开发）环境中对主机器人进行基准测试。主机器人是 future 开发和新增功能的基础。
2. 创建工单机器人 — 当需要新功能或更改时，会创建工单机器人。票务机器人本质上是主机器人的副本或分支，开发人员可以在不影响主版本的情况下使用它。

3. 导出工单机器人-工单机器人的工作完成后，它将从 Amazon Lex 服务中导出。然后，包含票务机器人的分支将从主分支重新定位。此步骤可确保纳入票务机器人开发期间对主机器人所做的任何更改，从而减少潜在的冲突。
4. 导入 rebased ticket bot 并验证 — 重新基准的工单机器人被导回开发环境并进行验证，以确保其在主分支的最新更改下正常运行。如果验证成功，则会创建一个拉取请求 (PR)，将工单机器人更改合并到主分支中。
5. 删除工单机器人 — 将更改成功合并到主分支后，不再需要工单机器人。可以删除工单机器人，以保持环境的清洁和可管理。
6. 将@@ 主机器人部署到开发环境中并进行测试 — 更新后的主机器人（现在包括新功能或更改）已部署到开发环境中。在这里，它经过了全面的测试，以确保所有功能都能按预期运行。
7. 将@@ 主机器人部署到生产环境中 — 在开发环境中完成测试并成功后，将主机器人部署到生产环境中。此步骤是工作流程的最后阶段，最终用户可以使用新功能。

自动化和扩缩

自动化工作流程允许开发人员并行处理不同的功能，每个功能都在不同的分支中。这便于并行开发，使团队能够有效地协作并更快地交付功能。由于分支彼此隔离，因此可以合并和部署更改，而不会阻塞或干扰其他正在进行的工作。

该工作流程可自动在不同环境（例如开发、测试和生产）中部署和推广机器人版本。

将机器人定义存储在 Git 等版本控制系统中可提供全面的审计跟踪并实现高效协作。对每个开发人员进行变更跟踪，确保整个开发生命周期的透明度和问责制。这种方法还有助于代码审查，使团队能够在部署到生产环境之前识别和解决问题。

通过使用 AWS CodePipeline 和其他 AWS 服务，自动化工作流程可以扩展以适应不断增加的工作负载和团队规模。

工具

AWS 服务

- AWS Cloud Development Kit (AWS CDK)是一个开源软件开发框架，用于通过使用熟悉的编程语言在代码中定义 AWS Cloud 基础架构，并通过它进行配置 AWS CloudFormation。此模式中的示例实现使用 Python。
- AWS CDK 命令行界面 (AWS CDK CLI)- AWS CDK 工具包是与您的 AWS CDK 应用程序交互的主要工具。它执行您的应用程序，查询您定义的应用程序模型，并生成和部署 CDK 生成的 AWS CloudFormation 模板。

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。此模式 CloudFormation 用于使用基础设施即代码部署 Amazon Lex 机器人配置和相关资源。
- AWS CodeBuild是一项完全托管的生成服务，可帮助您编译源代码、运行单元测试和生成可随时部署的工件。此模式 CodeBuild 用于构建和打包部署工件。
- AWS CodePipeline帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件更改所需的步骤。这种模式 CodePipeline 用于协调持续交付管道。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您AWS 服务 通过命令行 shell 中的命令进行交互。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Lex V2 AWS 服务 用于为使用语音和文本的应用程序构建对话界面（机器人）。
- 适用于 Python (Boto3) 的 AWS SDK是一个软件开发套件，可帮助您将 Python 应用程序、库或脚本与集成 AWS 服务。

其他工具

- Git 是一个开源的分布式版本控制系统。

代码存储库

此模式的代码可在-ama GitHub management-framework-sample-forzon- lex 存储库中找到。代码存储库包含以下文件夹和文件：

- `prerequisite`文件夹-包含用于设置所需资源和环境的 CloudFormation 堆栈定义（使用 AWS CDK）。
- `prerequisite/lexmgmtworkflow`文件夹-Lex 管理工作流项目的主目录，包括堆栈定义和 Python 代码。
- `prerequisite/tests`— 包含单元测试。
- `src`文件夹-源代码目录，包括 Amazon Lex 机器人管理封装程序和实用工具。
- `src/dialogue_lambda`— 对话钩子 Lambda 函数的源代码目录，该函数在与 Amazon Lex 机器人对话期间拦截和处理用户输入。

最佳实践

- 将关注点分开
  - 在开发环境和生产环境之间保持明确的 DevOps责任分工。
  - AWS 账户 为每个环境单独使用以强制适当的隔离和安全边界。
  - 使用跨账户角色和最低权限访问原则，确保环境之间的访问受控。
- 基础设施即代码
  - 定期审查和更新基础架构代码，以适应最佳实践和不断变化的需求。
  - 为源代码存储库制定明确的分支和合并策略
- 测试和验证
  - 在管道的各个阶段实施自动测试，以便在开发周期的早期发现问题。
  - 在升级到更高的环境之前，使用 Amazon Lex 控制台或自动测试框架来验证机器人的配置和功能。
  - 考虑为部署到生产环境或关键环境实施手动批准门。
- 监控和记录
  - 为管道、部署和机器人交互设置监控和记录机制。
  - 监控管道事件、部署状态和机器人性能指标，以便及时发现和解决问题。
  - 使用 AWS 服务（例如 Amazon CloudWatch AWS CloudTrail、和） AWS X-Ray 进行集中记录和监控。
  - 定期审查和分析自动化工作流程的性能、效率和有效性。
- 安全与合规
  - 实施安全编码实践，并遵循 Amazon Lex 机器人开发和部署 AWS 的安全最佳实践。
  - 定期审查和更新 IAM 角色、策略和权限，使其符合最低权限原则。
  - 考虑将安全扫描和合规性检查集成到管道中。

操作说明

为 Amazon Lex 机器人管理设置 IaC

Task	描述	所需技能

设置本地 CDK 环境。	1. 要克隆此模式的存储库并导航到该prerequisite 目录，请运行以下命令：
	`git clone https://github.com/aws-samples/management-framework-sample-for-amazon-lex.git`
	`cd management-framework-sample-for-amazon-lex`
	2. 要安装和激活 Python 虚拟环境，请运行以下命令，将 CDK 的依赖项安装在本地项目文件夹中，而不是全局安装到项目文件夹：
	`pip install virtualenv`
	`python<version> -m venv .venv`
	`source .venv/bin/activate`
	`python -m pip install -r requirements.txt`
		AWS DevOps

在devops环境中创建跨账户角色。	该devops账户负责托CI/CD pipelines. To enable the CI/CD 管和管理与dev和prod环境交互的管道，运行以下命令在账户中创建跨账户角色。devops	AWS DevOps

Task	描述	所需技能

	`cdk bootstrap --profile =devops`
	
	`cdk deploy LexMgmtDe vopsRoleStack -c dev-account-id=222 2222222222 -c prod- account-id=33333333333 3 --profile=devops`	

在dev环境中创建跨账户角色。	在dev账户中创建具有必要权限的 IAM 角色以允许该devops账户担任此角色。CI/CD 管道使用此角色在dev账户中执行操作，例如部署和管理 Amazon Lex 机器人资源。
	
	要创建 IAM 角色，请运行以下命令：
	
	`cdk bootstrap --profile =dev`
	
	`cdk deploy LexMgmtCr ossaccountRoleStac k -c devops-account- id=1111111111111 -- profile=dev`
		AWS DevOps

Task	描述	所需技能

在prod环境中创建跨账户角色。	在prod账户中创建具有必要权限的 IAM 角色以允许该devops账户担任此角色。CI/CD 管道使用此角色在prod账户中执行操作，例如部署和管理 Amazon Lex 机器人资源。
	`cdk bootstrap --profile =prod`
	`cdk deploy LexMgmtCr ossaccountRoleStac k -c devops-account- id=1111111111111 -- profile=prod`
		AWS DevOps

在devops环境中创建管道。	要管理 Amazon Lex 机器人的开发工作流程，请运行以下命令在devops环境中设置管道。 `cdk deploy LexMgmtWo rkflowStack -c devops- account-id=111111111 1111 -c dev-account- id=2222222222222 -c prod-account-id=33 3333333333 --profile =devops`	AWS DevOps

为主机器人建立基准

Task	描述	所需技能
定义主机器人的初始版本。	要定义主机器人的初始版本，请触发BaselineB otPipeline 管道。	AWS DevOps

Task	描述	所需技能
	该管道部署 CloudFormation 模板中定义的基本机器人定义，将主机器人定义导出为.json 文件。并将主机器人代码存储在版本控制系统中。	

实现功能开发工作流程

Task	描述	所需技能
创建票务机器人来开发和测试功能。	`TicketBot` 是一个新的机器人实例，是从功能分支中现有的主机器人定义中导入的。这种方法可确保新机器人具有主机器人的所有当前功能和配置。 要定义工单机器人的初始版本，请触发`CreateTicketBotPipeline` 管道。 该管道在版本控制系统中创建了一个新的功能分支，并基于主机器人创建了一个新的工单机器人实例。	Lex Bot 开发者
开发和测试票务机器人功能。	要开发和测试该功能，请登录 AWS Management Console 并打开 Amazon Lex 控制台，网址为https://console.aws.amazon.com/lex/。有关更多信息，请参阅 Amazon Lex 文档中的使用控制台测试机器人。	Lex Bot 开发者

Task	描述	所需技能
	通过该`TicketBot` 实例，您现在可以添加、修改或扩展机器人的功能以实现新功能。例如，您可以创建或修改意图、话语、插槽和对话流。有关更多信息，请参阅 Amazon Lex 文档中的添加意图。	
导出工单机器人定义。	导出的机器人定义本质上是以 JSON 格式表示机器人的配置和功能。 要导出工单机器人定义，请触发`ExportTicketBotPipeline` 管道。 该管道将工单机器人定义导出为.json 文件，并将工单机器人代码存储在版本控制系统的功能分支中。	Lex Bot 开发者
从最新的主分支中重新设置功能分支。	在开发新功能的过程中，主分支可能会收到来自不同开发人员或团队的其他更改。 要将这些更改合并到功能分支中，请执行 Git `rebase` 操作。此操作本质上是在主分支的最新提交之上重播来自功能分支的提交，从而确保功能分支包含所有最新更改	Lex Bot 开发者

Task	描述	所需技能
导入并验证重新设置的工单机器人。	重新设置功能分支后，必须将其导入工单机器人实例。此导入会使用重新调整的分支中的最新更改来更新现有的票务机器人。 要导入重新调整的工单机器人，请触发`ImportTicketBotPipeline` 管道。 管道将版本控制系统功能分支中的工单机器人定义.json 文件导入到`TicketBot` 实例中。	Lex Bot 开发者
验证重新调整后的机器人定义。	导入重新调整后的机器人定义后，验证其功能至关重要。您要确保新功能按预期运行，并且不会与现有功能冲突。 此验证通常包括使用各种输入场景测试机器人、检查响应以及验证机器人的行为是否符合预期。您可以通过以下任一方式执行验证： • 使用 Amazon Lex 控制台手动测试该机器人。 • 使用测试框架和工具，使用自动化方法，这些框架和工具可以模拟用户交互并断言预期的响应。	Lex Bot 开发者

Task	描述	所需技能
将功能分支合并到主分支中。	在隔离TicketBot 实例中开发和测试新功能后，请执行以下操作： 1. 将更改提交到版本控制系统中的相应功能分支。 2. 要将功能分支合并到主分支，请创建拉取请求 (PR)。此 PR 用于请求审查更改并将其合并到主代码库中。	Lex Bot 开发者、仓库管理员
删除功能分支和工单机器人。	功能分支成功合并到主分支后，从源代码存储库中删除该功能分支和票务机器人。 要删除功能分支和工单机器人，请触发DeleteTicketBotPipeline 管道。 该管道会删除开发过程中创建的临时机器人资源（例如，工单机器人）。此操作有助于维护干净的存储库，并防止与 future 功能分支混淆或冲突。	Lex Bot 开发者

维护主机器人

Task	描述	所需技能
将最新的主机器人定义导入到dev环境中。	要将主分支中最新的主机器人定义导入到dev环境中，请触发DeployBotDevPipeline 管道。	AWS DevOps

Task	描述	所需技能
	该管道还会在批准后创建 git 标签。	
将最新的主机器人定义导入到prod环境中。	要将主分支中最新的机器人定义导入到prod环境中，请提供上一个任务中的标签引用作为参数并触发DeployBotProdPipeline 管道。 管道将最新的 bot 定义从特定标签导入到prod环境中。	AWS DevOps

故障排除

事务	解决方案
当您将 Amazon Lex 机器人部署到其他账户时 AWS 账户，工具服务必须具有访问这些账户中的资源的必要权限。	要授予跨账户访问权限，请使用 IAM 角色和策略。在目标账户中创建 IAM 角色，并将策略附加到授予所需权限的角色。然后，从部署 Amazon Lex 机器人的账户中扮演这些角色。 有关更多信息，请参阅 Amazon Lex 文档中的导入所需的 IAM 权限和在 Lex V2 中导出机器人所需的 IAM 权限。

相关资源

- 在 Amazon Lex V2 中导入机器人
- 在中启动管道 CodePipeline
- 使用 Amazon Lex V2 机器人

使用 AWS CodePipeline 和 AWS 自动部署堆栈集 CodeBuild

由 Thiyagarajan Mani (AWS)、Mihir Borkar (AWS) 和 Raghu Gowda (AWS) 编写

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

在持续集成和持续交付（满足此要求的CI/CD) processes, you might want to deploy applications automatically into all your existing AWS accounts and into new accounts that you add to your organization in AWS Organizations. When you architect a CI/CD解决方案）中，AWS 的委托堆栈集管理员功能 CloudFormation 非常有用，因为它通过限制对管理账户的访问来实现一层安全保护。但是，AWS CodePipeline 使用服务托管权限模型将应用程序部署到多个账户和区域。您必须使用 AWS Organizations 管理账户使用堆栈集进行部署，因为 AWS CodePipeline 不支持委托堆栈集管理员功能。

此模式介绍了如何解决此限制。该模式使用 AWS CodeBuild 和自定义脚本通过 AWS 自动部署堆栈集 CodePipeline。它可自动执行以下应用程序部署活动：

- 将应用程序作为堆栈集部署到现有组织单位中 (OUs)
- 将应用程序的部署扩展到其他 OUs 区域和区域
- 从所有或特定区域 OUs 或区域中移除已部署的应用程序

先决条件和限制

先决条件

在按照此模式中的步骤操作之前：

- 在 AWS Organizations 管理账户创建组织。有关说明，请参阅 AWS Organizations 文档。
- 在 AWS Organizations 之间启用可信访问权限并 CloudFormation 使用服务托管权限。有关说明，请参阅 CloudFormation 文档中的通过 AWS Organizations 启用可信访问。

限制

此模式提供的代码具有以下限制：

- 您只能为一个应用程序部署单个 CloudFormation 模板；目前不支持多个模板部署。
- 定制当前实施需要 DevOps 专业知识。
- 此模式不使用 AWS Key Management System (AWS KMS) 密钥。但是，您可以通过重新配置此模式中包含的 CloudFormation 模板来启用此功能。

架构

CI/CD 部署管道架构可处理以下内容：

- 通过将堆栈集部署责任委派给作为应用程序部署的堆栈集管理员的专用 CI/CD 账户，限制对管理账户的直接访问。
- 每当在 OU 下创建并映射新账户时，使用服务管理的权限模型自动部署应用程序。
- 确保环境级别所有账户的应用程序版本一致性。
- 在存储库和管道级别使用多个审核阶段，为已部署的应用程序提供额外的安全和治理层。
- 克服了当前的限制， CodePipeline 即在中使用自定义的部署脚本 CodeBuild 来自动部署或移除堆栈集和堆栈实例。有关自定义脚本实现的 API 调用的流量控制和层次结构的说明，请参阅其他信息部分。
- 为开发、测试和生产环境创建单独的堆栈集。此外，您还可以创建在每个阶段将多个 OUs 和区域组合在一起的堆栈集。例如，您可以在开发部署阶段将沙盒 OUs和开发结合起来。
- 支持将应用程序部署到账户子集或列表中或从中排除 OUs。

自动化和扩缩

您可以使用此模式提供的代码为您的应用程序创建 AWS CodeCommit 存储库和代码管道。然后，您可以将它们作为堆栈集部署至 OU 级别的多个账户中。该代码还自动执行组件，例如用于通知审批者的 Amazon Simple Notification Service (Amazon SNS) 主题、所需 AWS Identity and Access Management (IAM) 角色以及要在管理账户中应用的服务控制策略（SCP）。

工具

Amazon Web Services

- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。

- AWS CodeBuild 是一项完全托管的构建服务，可帮助您编译源代码、运行单元测试和生成可随时部署的项目。
- AWS CodeCommit 是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。
- AWS CodeDeploy 可自动部署到亚马逊弹性计算云 (Amazon EC2) 或本地实例、AWS Lambda 函数或亚马逊弹性容器服务 (Amazon ECS) Service 服务。
- AWS CodePipeline 可帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件变更所需的步骤。
- AWS Organizations 是一项账户管理服务，使您可将多个 Amazon Web Services account 整合到您所创建的组织中并进行集中管理。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。

代码存储库

此模式的代码可在 GitHub automated-code-pipeline-stackset- deployment 存储库中找到。有关文件夹结构和其他详细信息，请参见存储库的自述文件。

最佳实践

在 OU 级别部署应用程序时，此模式限制对管理账户的直接访问。向管道和存储库流程添加多个审批阶段有助于为使用此方法部署的应用程序和组件提供额外的安全性和治理。

操作说明

在 AWS Organizations 中配置账户

Task	描述	所需技能
启用管理账户中的所有功能。	按照 AWS Organizations 文档中的说明为您的组织启用管理账户中的所有功能。	AWS 管理员、平台管理员
创建 CI/CD 账户。	在 AWS Organizations，在您的组织中创建一个专用 CI/CD 账户，然后分配一个团队来拥有和控制该账户的访问权限。	AWS 管理员

Task	描述	所需技能
添加委托管理员。	在管理账户中，将您在上一步中创建的 CI/CD 账户注册至委派堆栈集管理员。有关说明，请参阅 A WS CloudFormation 文档。	AWS 管理员、平台管理员

创建应用程序存储库和 CI/CD 管道

Task	描述	所需技能
克隆代码存储库。	1. 将此模式提供的代码库克隆到您的计算机上： `git clone https://github.com/aws-samples/automated-code-pipeline-stackset-deployment.git` 2. 查看自述文件，以了解目录结构和其他细节。	AWS DevOps
创建 SNS 主题。	您可以使用 GitHub 存储库中提供的`sns-template.yaml` 模板来创建 SNS 主题和配置批准请求的订阅。 1. 在 Amazon Web Services Console，登录 CI/CD 账户。 2. 在 https://console.aws.amazon.com/cloudformation 上打开 CloudFormation 控制台。	AWS DevOps

Task	描述	所需技能
	3. 使用新资源创建新堆栈（标准选项）。 4. 对于“指定模板”，选择“上传模板文件”、“选择文件”，然后从克隆 GitHub 存储库的`templates` 文件夹中选择该`sns-template.yaml` 文件。选择下一步。 5. 提供有意义的应用程序堆栈名称。 6. 指定资源前缀 7. 选择下一步、下一步和提交。 8. 成功创建堆栈后，选择输出选项卡，并记下拉取请求、测试环境和生产环境的 SNS 主题的 Amazon 资源名称 (ARNs)。您将在后续步骤中使用该信息。	

Task	描述	所需技能
为 CI/CD 组件创建 IAM 角色。	您可以使用 GitHub 存储库中提供的`cicd-role-template.yaml` 模板创建 CI/CD 组件所需的 IAM 角色和策略。 1. 在 Amazon Web Services Console，登录 CI/CD 账户。 2. 在 https://console.aws.amazon.com/cloudformation 上打开 CloudFormation 控制台。 3. 使用新资源创建新堆栈（标准选项）。 4. 对于 “指定模板”，选择 “上传模板文件”、“选择文件”，然后从克隆 GitHub 存储库的`templates` 文件夹中选择该`cicd-role-template.yaml` 文件。选择下一步。 5. 提供有意义的应用程序堆栈名称。 6. 输入以下参数的值： • 权限边界策略的 ARN。您可以从 IAM 控制台上权限边界策略策略详情部分获取此 ARN。 • 您之前记下的 SNS 生产批准主题 ARN。 • 您之前记下的 SNS 测试批准主题 ARN。 • 模板创建的资源的前缀。	AWS DevOps

Task	描述	所需技能
	7. 选择下一步、下一步和提交。 8. 成功创建堆栈后，选择 Outputs 选项卡，并记下 ARNs 已创建的 IAM 角色。您将在后续步骤中使用该信息。	

Task	描述	所需技能
为您的应用程序创建 CodeCommit 存储库和代码管道。	您可以使用 GitHub 存储库中提供的`cicd-pipeline-template.yaml` 模板为应用程序创建 CodeCommit 存储库和代码管道。 1. 在 Amazon Web Services Console，登录 CI/CD 账户。 2. 在 https://console.aws.amazon.com/cloudformation 上打开 CloudFormation 控制台。 3. 使用新资源创建新堆栈（标准选项）。 4. 对于“指定模板”，选择“上传模板文件”、“选择文件”，然后从克隆 GitHub 存储库的`templates` 文件夹中选择该`cicd-pipeline-template.yaml` 文件。选择下一步。 5. 提供有意义的应用程序堆栈名称。 6. 输入以下参数的值： • AppRepositoryName— 将为应用程序创建的 CodeCommit 存储库的名称。 • AppRepositoryDescription— 将为应用程序创建的 CodeCommit 存储库的简要描述。	AWS DevOps

Task	描述	所需技能
	• ApplicationName— 您的应用程序的名称。此字符串用作 CodeCommit 存储库的名称和 CI/CD 管道的前缀。 • CloudWatchEventRoleARN — 上一个任务中 CloudWatch 事件角色的 ARN。 • CodeBuildProjectRoleARN — 上一个任务中 CodeBuild 项目角色的 ARN。 • CodePipelineRoleARN — 上一个任务中 CodePipeline 角色的 ARN。 • DeploymentConfigBucket— 亚马逊简单存储服务 (Amazon S3) 存储桶名称，用于存储部署配置文件和脚本.zip 文件。 • DeploymentConfigKey— 路径和.zip 文件名（亚马逊 S3 密钥）。 • PRApprovalSNSARN — 拉取请求通知的 SNS 主题的 ARN。 • ProdApprovalSNSARN — 用于生产批准的 SNS 主题的 ARN。 • TESTApprovalSNSARN — 用于测试批准的 SNS 主题的 ARN。	

Task	描述	所需技能
	• TemplateBucket— 将存储 CI/CD account where the CI/CD管道创建模板中 S3 存储桶的名称。 7. 选择下一步、下一步和提交。 8. 堆栈成功完成后，它将创建一个具有指定名称和默认目录结构的 CodeCommit 存储库、部署配置文件、脚本以及存储库的代码管道。	

部署堆栈集

Task	描述	所需技能

克隆应用程序存储库。	您之前使用的 CI/CD 管道模板创建了示例应用程序存储库和代码管道。若要克隆并验证存储库，请执行以下操作：
	1. 登录到 CI/CD 账户。
	2. 查找您在上一篇操作说明中创建的应用程序存储库和 CI/CD 管道。
	3. 复制存储库的 URL，然后使用 git 克隆命令在本地计算机上克隆存储库。
	4. 验证目录结构和文件是否与以下内容匹配：
	`root`
	`|- deploy_configs`
		应用程序开发人员、数据工程师

Task	描述	所需技能

	<pre>| |- deployment_config.json
	|- parameters
	| |- template-parameter-dev.json
	| |- template-parameter-test.json
	| |- template-parameter-prod.json
	|- templates
	| |- template.yml
	|- buildspec.yml</pre>
	其中deploy_configs 文件夹包含部署配置文件，templates 和parameters 文件夹包含默认文件，您将用自己的 CloudFormation 模板和参数文件替换这些文件。
	
	Important
	请勿自定义文件夹结构。
	
	5. 创建功能分支。	

Task	描述	所需技能
添加应用程序构件。	使用 CloudFormation 模板更新应用程序存储库。 **Note** 此解决方案仅支持部署单个 CloudFormation 模板。 1. 构建用于部署应用程序代码更改的 CloudFormation 模板，并命名该模板 `<application-name>.yaml` 。 2. 将应用程序存储库`templates` 文件夹中的`template.yml` 文件替换为在步骤 1 中创建的 CloudFormation 模板。 3. 为每个环境（开发、测试和生产）准备参数文件。 4. 使用 `<cloudformation-template-name>-parameter-<environment-name>.json` 格式命名参数文件。 5. 将 `parameters` 文件夹中的默认参数文件替换为步骤 4 中的文件。	应用程序开发人员、数据工程师

Task	描述	所需技能

更新部署配置文件。	更新 deployment_config.json 文件：
	1. 在应用程序存储库中，导航到 deploy_configs 文件夹。
	2. 打开文件 deployment_config.json ：
	{
	"deployment_action": "<deploy/delete>",
	"stack_set_name": "<stack set name>",
	"stack_set_desciption": "<stack set description>",
	"deployment_targets": {
	"dev": {
	"org_units": ["list of OUs"],
	"regions": ["list of regions"],
	"filter_accounts": ["list of accounts"],
	"filter_type":
		应用程序开发人员、数据工程师

Task	描述	所需技能

	"<DIFFERENCE/INTERSECTION/UNION>"
	},
	"test": {
	"org_units": ["list of OUs"],
	"regions": ["list of regions"],
	"filter_accounts": ["list of accounts"],
	"filter_type": "<DIFFERENCE/INTERSECTION/UNION>"
	},
	"prod": {
	"org_units": ["list of OUs"],
	"regions": ["list of regions"],
	"filter_accounts": ["list of accounts"],	

Task	描述	所需技能

	<pre>"filter_type": "<DIFFERENCE/INTERSECTION/UNION>"
	}
	},
	"cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
	"auto_deployement": "<True/False>",
	"retain_stacks_on_account_removal": "<True/False>",
	"region_deployment_concurrency": "<SEQUENTIAL/PARALLEL>"
	}</pre>	

	3. 更新部署操作、堆栈集名称、堆栈集描述以及部署目标的值。 例如，您可以将设置 `deployment_action` 为 `delete`，删除整个堆栈集及其关联堆栈实例。`deploy`用于创建新的堆栈集、更新现有堆栈集，或者为其他 OUs 或区域添加或移除堆栈实例。有关更	

Task	描述	所需技能
	多示例，请参阅其他信息部分。 此模式通过将环境名称添加到您在部署配置文件中提供的堆栈集名称来为每个环境创建单独的堆栈集。	

Task	描述	所需技能
提交更改和部署堆栈集。	提交您在应用程序模板中指定的更改，并将堆栈集逐步合并并部署到多个环境中： 1. 保存所有文件，并将更改提交到本地应用程序存储库的功能分支。 2. 将功能分支推送至远程存储库。 3. 创建拉取请求以将更改合并至主分支。 当拉取请求获得批准并且更改已合并到主分支后，CI/CD 管道将会启动。 4. 成功完成开发部署阶段后，请查看 CloudFormation 控制台的 “服务管理” 选项StackSets卡。 您会看到一个带有后缀 `dev` 的新堆栈集。 5. 检查开发部署阶段的 CodeBuild 日志中是否存在任何问题。 6. 通过要求审批者批准这些阶段的部署并重复步骤 5 和 6，将堆栈集部署到测试和生产环境中。测试和生产环境的堆栈集具有后缀 `test` 和 `prod`。	应用程序开发人员、数据工程师

故障排除

事务	解决方案
部署失败，以下情况除外： 将模板参数文件的名称更改为-parameter-.json，不允许使用默认名称 <application name><evn>	CloudFormation 模板参数文件必须遵循指定的命名约定。更新参数文件名并重试。
部署失败，以下情况除外： 将 CloudFormation 模板名称更改为.yml，不允许使用默认 template.yml 或 template.yaml <application name>	CloudFormation 模板名称必须遵循指定的命名约定。更新文件名和重试。
部署失败，以下情况除外： 找不到适用于 {环境名称} 环境的有效 CloudFormation 模板及其参数文件	检查 CloudFormation 模板的文件命名约定及其指定环境的参数文件。
部署失败，以下情况除外： 部署配置文件中提供的部署操作无效。有效的选项是部署和删除。	您在部署配置文件中为 `deployment_action` 参数指定了无效值。该参数有两个有效值：`deploy` 和 `delete`。`deploy` 用于创建和更新堆栈集及其关联堆栈实例。`delete` 仅当您想要移除整个堆栈集和关联堆栈实例时使用。

相关资源

- GitHub automated-code-pipeline-stackset-部署存储库
- 启用企业中的所有功能（AWS Organizations 文档）
- 注册委托管理员（AWS CloudFormation 文档）
- 服务托管堆栈集的账户级别目标（AWS CloudFormation 文档）

其他信息

流程图

以下流程图介绍了自定义脚本实现的 API 调用的流程控制和层次结构，以自动化堆栈集部署。

部署配置文件示例

创建新堆栈集

以下部署配置文件创建了一个新的堆栈集，该堆栈集`sample-stack-set`在三个 AWS 区域`us-east-1`中名为 OUs。

```
{
     "deployment_action": "deploy",
     "stack_set_name": "sample-stack-set",
     "stack_set_desciption": "this is a sample stack set",
    "deployment_targets": {
                             "dev": {
                                        "org_units": ["dev-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                             "test": {
                                        "org_units": ["test-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                             "prod": {
                                        "org_units": ["prod-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    }
                           },
     "cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
     "auto_deployement": "True",
     "retain_stacks_on_account_removal": "True",
     "region_deployment_concurrency": "PARALLEL"
 }
```

将现有堆栈集部署至其他 OU

如果您部署了上一个示例中所示的配置，并且想要将堆栈集部署到开发环境 dev-org-unit-2 中名为的其他 OU，则部署配置文件可能如下所示。

```
{
     "deployment_action": "deploy",
     "stack_set_name": "sample-stack-set",
     "stack_set_desciption": "this is a sample stack set",
    "deployment_targets": {
                              "dev": {
                                        "org_units": ["dev-org-unit-1", "dev-org-
unit-2"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                              "test": {
                                        "org_units": ["test-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                              "prod": {
                                        "org_units": ["prod-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    }
                           },
     "cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
     "auto_deployement": "True",
     "retain_stacks_on_account_removal": "True",
     "region_deployment_concurrency": "PARALLEL"
 }
```

将现有堆栈集部署至其他 AWS 区域

如果您部署了上一个示例中所示的配置，并且想要将堆栈集部署到开发环境中的其他 AWS 区域 (us-east-2) OUs（ dev-org-unit-1和dev-org-unit-2 ），则部署配置文件可能如下所示。

> **Note**
>
> CloudFormation 模板中的资源必须有效且特定于区域。

```
{
     "deployment_action": "deploy",
     "stack_set_name": "sample-stack-set",
     "stack_set_desciption": "this is a sample stack set",
    "deployment_targets": {
                             "dev": {
                                        "org_units": ["dev-org-unit-1", "dev-org-
unit-2"],
                                        "regions": ["us-east-1", "us-east-2"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                             "test": {
                                        "org_units": ["test-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                             "prod": {
                                        "org_units": ["prod-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    }
                           },
     "cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
     "auto_deployement": "True",
     "retain_stacks_on_account_removal": "True",
     "region_deployment_concurrency": "PARALLEL"
 }
```

从 OU 或 AWS 区域移除堆栈实例

假设已经部署了上一个示例中显示的部署配置。以下配置文件将堆栈实例从 OU dev-org-unit-2 的两个区域中移除。

```
{
     "deployment_action": "deploy",
     "stack_set_name": "sample-stack-set",
     "stack_set_desciption": "this is a sample stack set",
    "deployment_targets": {
                             "dev": {
                                        "org_units": ["dev-org-unit-1"],
```

```
                                        "regions": ["us-east-1", "us-east-2"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                                "test": {
                                        "org_units": ["test-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                                "prod": {
                                        "org_units": ["prod-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    }
                            },
    "cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
    "auto_deployement": "True",
    "retain_stacks_on_account_removal": "True",
    "region_deployment_concurrency": "PARALLEL"
 }
```

以下配置文件将两者的堆栈实例从开发环境 OUs 中的 AWS 区域us-east-1中删除。

```
{
     "deployment_action": "deploy",
     "stack_set_name": "sample-stack-set",
     "stack_set_desciption": "this is a sample stack set",
    "deployment_targets": {
                                "dev": {
                                        "org_units": ["dev-org-unit-1", "dev-org-
unit-2"],
                                        "regions": ["us-east-2"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                                "test": {
                                        "org_units": ["test-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
```

```
                                    },
                                "prod": {
                                        "org_units": ["prod-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    }
                            },
     "cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
     "auto_deployement": "True",
     "retain_stacks_on_account_removal": "True",
     "region_deployment_concurrency": "PARALLEL"
 }
```

删除整个堆栈集

以下部署配置文件将删除整个堆栈集及其所有关联的堆栈实例。

```
{
     "deployment_action": “delete”,
     "stack_set_name": "sample-stack-set",
     "stack_set_desciption": "this is a sample stack set",
    "deployment_targets": {
                                "dev": {
                                        "org_units": ["dev-org-unit-1", "dev-org-
unit-2"],
                                        "regions": ["us-east-2"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                                "test": {
                                        "org_units": ["test-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                                "prod": {
                                        "org_units": ["prod-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    }
                            },
     "cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
```

```
    "auto_deployement": "True",
    "retain_stacks_on_account_removal": "True",
    "region_deployment_concurrency": "PARALLEL"
}
```

将账户排除在部署之外

以下部署配置文件将作为 OU dev-org-unit-1 一部分的账户 111122223333 排除在部署之外。

```
{
    "deployment_action": "deploy",
    "stack_set_name": "sample-stack-set",
    "stack_set_desciption": "this is a sample stack set",
   "deployment_targets": {
                            "dev": {
                                        "org_units": ["dev-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": ["111122223333"],
                                        "filter_type": "DIFFERENCE"
                                    },
                            "test": {
                                        "org_units": ["test-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                            "prod": {
                                        "org_units": ["prod-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    }
                          },
    "cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
    "auto_deployement": "True",
    "retain_stacks_on_account_removal": "True",
    "region_deployment_concurrency": "PARALLEL"
}
```

将应用程序部署至 OU 中的一部分账户

以下部署配置文件仅将应用程序部署到 OU dev-org-unit-1 中的三个账户 (111122223333 444455556666、和 777788889999)。

```
{
    "deployment_action": "deploy",
    "stack_set_name": "sample-stack-set",
    "stack_set_desciption": "this is a sample stack set",
   "deployment_targets": {
                            "dev": {
                                        "org_units": ["dev-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": ["111122223333",
"444455556666", "777788889999"],
                                        "filter_type": "INTERSECTION"
                                    },
                            "test": {
                                        "org_units": ["test-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                            "prod": {
                                        "org_units": ["prod-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    }
                          },
    "cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
    "auto_deployement": "True",
    "retain_stacks_on_account_removal": "True",
    "region_deployment_concurrency": "PARALLEL"
}
```

使用云托管人和 AWS CDK 将适用于 Systems Manager 的 AWS 托管策略自动附加到 EC2 实例配置文件

创建者：Ali Asfour（AWS）和 Aaron Lennon（AWS）

摘要

您可以将亚马逊弹性计算云 (Amazon EC2) 实例与 AWS Systems Manager 集成，以自动执行操作任务并提供更多的可见性和控制力。要与 Systems Manager 集成， EC2 实例必须安装一个 AWS Systems Manager 代理 (SSM 代理)，并在其实例配置文件上附加一个 A AmazonSSMManagedInstanceCore WS 身份和访问管理 (IAM) 策略。

但是，如果您想确保所有 EC2 实例配置文件都附加了AmazonSSMManagedInstanceCore策略，则在更新没有 EC2 实例配置文件的新实例或 EC2 具有实例配置文件但没有AmazonSSMManagedInstanceCore策略的实例时可能会遇到困难。在多个 Amazon Web Services（AWS）账户和 AWS 区域 中添加此策略也可能很困难。

这种模式通过在 Amazon Web Services account 中部署三项云托管人策略来帮助解决这些困难：

- 第一个 Cloud Custodian 策略会检查是否有实例配置文件但没有该AmazonSSMManagedInstanceCore策略的现有 EC2 实例。然后附上 AmazonSSMManagedInstanceCore 策略。
- 第二个 Cloud Custodian 策略会检查没有 EC2 实例配置文件的现有实例，并添加附加了该AmazonSSMManagedInstanceCore策略的默认实例配置文件。
- 第三项云托管人策略在您的账户中创建 AWS Lambda 函数，以监控实例和实例配置 EC2 文件的创建。这样可以确保在创建 EC2 实例时自动附加AmazonSSMManagedInstanceCore策略。

这种模式使用 AWS DevOps 工具将云托管人策略持续大规模部署到多账户环境，无需配置单独的计算环境。

先决条件和限制

先决条件

- 两个或以上有效 Amazon Web Services account。一个账户是安全账户，其他账户是成员账户。
- 在安全账户中预调配 AWS 资源的权限。此模式使用管理员权限，但您应根据贵组织的要求和策略授予权限。

- 能够从安全账户向成员账户分派 IAM 角色并创建所需 IAM 角色。有关更多信息，请参阅 IAM 文档中的使用 IAM 角色跨 Amazon Web Services account 委派访问权限。
-

> **Important**
>
> 已安装和配置 AWS 命令行界面（AWS CLI）。出于测试目的，您可以使用 `aws configure` 命令或设置环境变量来配置 AWS CLI。：不建议在生产环境中使用此功能，我们建议仅向该账户授予最低访问权限。有关更多信息，请参阅 IAM 文档中的授予最低权限许可。

- 将 `devops-cdk-cloudcustodian.zip` 文件（附件）下载到本地计算机中。
- 熟悉 Python。
- 已安装并配置所需工具（Node.js、AWS Cloud Development Kit（AWS CDK）和 Git）。您可以使用 `install-prerequisites.sh` 文件中的 `devops-cdk-cloudcustodian.zip` 文件来安装这些工具。确保以根权限运行此文件。

限制

- 尽管这种模式可以在生产环境中使用，但请确保所有 IAM 角色和策略都符合贵组织的要求和政策。

软件包版本

- 云托管人版本 0.9 或更高版本
- TypeScript 版本 3.9.7 或更高版本
- Node.js 版本 14.15.4 或更高版本
- npm 版本 7.6.1 或更高版本
- AWS CDK 版本 1.96.0 或更高版本

架构

图表显示了以下工作流：

1. 云托管人策略被推送到安全账户中的 AWS CodeCommit 存储库。Amazon E CloudWatch vents 规则会自动启动 AWS CodePipeline 管道。

2. 该管道从中 CodeCommit 获取最新代码，并将其发送到 AWS 处理的持续集成和持续交付 (CI/CD) 管道的持续集成部分。 CodeBuild
3. CodeBuild 执行完整的 DevSecOps 操作，包括对 Cloud Custodian 策略进行策略语法验证，并在`--dryrun`模式下运行这些策略以检查识别了哪些资源。
4. 如果没有错误，则下一个任务会提醒管理员查看更改并批准向成员账户部署。

技术堆栈

- AWS CDK
- CodeBuild
- CodeCommit
- CodePipeline
- IAM
- Cloud Custodian

自动化和扩缩

除了使用 CodePipeline AWS 堆栈部署 AWS 资源外，AWS CDK pipelines 模块还提供了一个 CI/CD 管道 CodeBuild，该管道用于协调源代码的构建和测试。 CloudFormation 您可以对贵组织中的所有成员账户和区域使用此模式。您还可以扩展 `Roles creation` 堆栈以在成员账户中部署其他 IAM 角色。

工具

- AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，用于在代码中定义云基础设施并通过 AWS CloudFormation 进行配置。
- AWS 命令行界面（AWS CLI）是一种开源工具，它使您能够使用命令行 Shell 中的命令与 Amazon Web Services 交互。
- AWS CodeBuild 是一项完全托管的云端构建服务。
- AWS CodeCommit 是一项版本控制服务，您可以使用它来私下存储和管理资产。
- AWS CodePipeline 是一项持续交付服务，您可以使用它对发布软件所需的步骤进行建模、可视化和自动化。
- AWS Identity and Access Management 是一项 Web 服务，可帮助您安全地控制对 AWS 资源的访问。

- 云托管人是一种工具，可将大多数组织用于管理其公共云账户的数十种工具和脚本统一到一个开源工具中。
- Node.js 是一个基于谷歌浏览器 V8 JavaScript 引擎构建的 JavaScript 运行时。

代码

有关此模式中使用的模块、账户函数、文件和部署命令的详细列表，请参阅 `devops-cdk-cloudcustodian.zip` 文件（附件）中的 `README` 文件。

操作说明

使用 AWS CDK 设置管线

Task	描述	所需技能
设置 CodeCommit 存储库。	1. 将 `devops-cdk-cloudcustodian.zip` 文件（附件）解压缩到本地计算机上的工作目录中。 2. 登录您的安全账户的 AWS 管理控制台，打开 CodeCommit 控制台，然后创建新的`devops-cdk-cloudcustodian` 存储库。 3. 进入项目目录并将 CodeCommit 存储库设置为源，提交更改，然后通过运行以下命令将其推送到 origin 分支： • `cd devops-cdk-cloudcustodian` • `git init --initial-branch=main`	开发人员

Task	描述	所需技能
	• `git add . git commit -m 'initial commit'` • `git remote add origin https://git-codecommit.us-east-1.amazonaws.com/v1/devops-cdk-cloudcustodian` • `git push origin main` 有关这方面的更多信息，请参阅 AWS CodeCommit 文档中的创建 CodeCommit 存储库。	
安装所需工具。	使用该 `install-prerequisites.sh` 文件在 Amazon Linux 上安装所有必需的工具。这不包括 AWS CLI，因为它是预安装的。 有关这方面的更多信息，请参阅 AWS CDK 文档中的 AWS CDK 入门的先决条件部分。	开发人员

Task	描述	所需技能
安装所需 AWS CDK 软件包。	1. 在 AWS CLI 中运行以下命令来设置虚拟环境：`$ python3 -m venv .env` 2. 运行以下命令来设置虚拟环境：`$ source .env/bin/activate` 3. 激活虚拟环境后，通过运行以下命令来安装所需依赖项：`$ pip install -r requirements.txt` 4. 要添加其他依赖项（例如，其他 AWS CDK 库），请将它们添加到 `requirements.txt` 文件中，然后运行以下命令：`pip install -r requirements.txt` 以下软件包是 AWS CDK 的必需并且包含在 `requirements.txt` 文件中： • `aws-cdk.aws-cloudwatch` • `aws-cdk.aws-codebuild` • `aws-cdk.aws-codecommit` • `aws-cdk.aws-codedeploy` • `aws-cdk.aws-codepipeline`	开发人员

Task	描述	所需技能
	• aws-cdk.aws-codepipeline-actions • aws-cdk.aws-events • aws-cdk.aws-events-targets • aws-cdk.aws-iam • aws-cdk.aws-logs • aws-cdk.aws-s3 • aws-cdk.aws-sns • aws-cdk.aws-sns-subscriptions • aws-cdk.aws-sqs • aws-cdk.core	

配置环境

Task	描述	所需技能
更新所需变量。	打开 CodeCommit 存储库根文件夹中的vars.py文件并更新以下变量： • 使用要在其中部署管线的 AWS 区域 更新 var_deploy_region = ‘us-east-1’ 。 • var_codecommit_repo_name = “cdk-cloudcustodian” 使用 CodeCommit 存储库的名称进行更新。	开发人员

Task	描述	所需技能
	• var_codecommit_branch_name = “main” 使用 CodeCommit 分支名称进行更新。 • 使用批准更改的管理员的电子邮件地址更新 var_adminEmail=notifyadmin@email.com’ 。 • 使用用于在更改时发送云托管人通知的 Slack Webhook 更新 var_slackWebHookUrl = https://hooks.slack.com/services/T000000000/B00000000/XXXXXXXXXXXXXXXXXXXXXX ’。 • 使用贵组织 ID 更新 var_orgId = ‘o-yyyyyyyyyy’ 。 • 使用部署管线的账户的 Amazon Web Services account ID 更新 security_account = ‘123456789011’ 。 • 使用想要引导 AWS CDK 堆栈并部署所需 IAM 角色的成员账户更新 member_accounts = [‘111111111111’,’111111111112’,’111111111113’] 。	

Task	描述	所需技能
	• 如果您希望管线自动将 AWS CDK 引导到成员账户，请将 `cdk_bootstrap_member_accounts = True` 设置为 `True`。 如果设置为 `True`，则还需要成员账户中现有 IAM 角色的名称，该角色可以从安全账户中分派。此 IAM 角色还必须具有引导 AWS CDK 所需权限。 • **Note** 使用成员账户中现有 IAM 角色更新 `cdk_bootstrap_role = 'AWSControlTowerExecution'` ，该角色可以从安全账户中分派。此角色还必须具有引导 AWS CDK 所需权限。：这仅在设置`cdk_bootstrap_member_accounts` 为时适用`True`。	

Task	描述	所需技能

使用成员账户信息更新 account.yml 文件。	要对多个账户运行 c7n-org Cloud Custodian 工具，必须将 `accounts.yml` 配置文件放在存储库的根目录中。以下是适用于 AWS 的一个示例云托管人配置文件：
	`accounts:`
	`- account_id: '123123123123'`
	`  name: account-1`
	`  regions:`
	`  - us-east-1`
	`  - us-west-2`
	`  role: arn:aws:iam::123123123123:role/CloudCustodian`
	`  vars:`
	`  charge_code: xyz`
	`  tags:`
	`  - type:prod`
	`  - division:some division`
	`  - partition:us`
	`  - scope:pci`
		开发人员

引导 Amazon Web Services account

Task	描述	所需技能
引导安全账户。	运行以下命令，使用 `cloudcustodian_stack` 应用程序引导 `deploy_account` ： `cdk bootstrap -a 'python3`	开发人员

Task	描述	所需技能
	`cloudcustodian/cloudcustodian_stack.py`	
选项 1 – 自动引导成员账户。	如果在 `vars.py` 文件中将 `cdk_bootstrap_member_accounts` 变量设置为 `True`，则管线会自动引导 `member_accounts` 变量中指定的账户。 如果需要，您可以使用可从安全账户分派且具有引导 AWS CDK 所需权限的 IAM 角色更新 `*cdk_bootstrap_role*` 。 管线会自动引导添加到 `member_accounts` 变量中的新账户，以便可以部署所需角色。	开发人员

Task	描述	所需技能

选项 2 – 手动引导成员账户。	尽管我们不建议使用这种方法，但您可以将 `cdk_bootstrap_member_accounts` 的值设置为 `False`，然后通过运行以下命令手动执行此步骤：
	
	`$ cdk bootstrap -a 'python3 cloudcustodian/member_account_roles_stack.py' \`
	`--trust {security_account_id} \`
	`--context assume-role-credentials:writeIamRoleName={role_name} \`
	`--context assume-role-credentials:readIamRoleName={role_name} \`
	`--mode=ForWriting \`
	`--context bootstrap=true \`
	`--cloudformation-execution-policies arn:aws:iam::aws:policy/AdministratorAccess`
		开发人员

Task	描述	所需技能
	Important 请务必使用可从安全账户担任`{security_account_id}` 且具有引导 AWS CDK 所需权限的 IAM 角色的名称更新和`{role_name}` 值。 您也可以使用其他方法来引导成员账户，例如，使用 AWS CloudFormation。有关更多信息，请参阅 AWS CDK 文档中的引导。	

部署 AWS CDK 堆栈

Task	描述	所需技能
在成员账户中创建 IAM 角色。	运行以下命令以部署`member_account_roles_stack` 堆栈并在成员账户中创建 IAM 角色： `cdk deploy --all -a 'python3 cloudcustodian/member_account_roles_stack.py' --require-approval never`	开发人员
部署云托管人管线堆栈。	运行以下命令以创建部署到安全账户的云托管人`cloudcustodian_stack.py` 管线：	开发人员

Task	描述	所需技能
	`cdk deploy -a 'python3 cloudcustodian/cloudcustodian_stack.py'`	

相关资源

- AWS CDK 入门

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

自动使用 AWS CDK 为微服务构建 CI/CD 管道与 Amazon ECS 集群

由 Varsha Raju (AWS) 创建

摘要

此模式描述了如何自动创建持续集成和持续交付（CI/CD) pipelines and underlying infrastructure for building and deploying microservices on Amazon Elastic Container Service (Amazon ECS). You can use this approach if you want to set up proof-of-concept CI/CD pipelines to show your organization the benefits of CI/CD, microservices, and DevOps. You can also use this approach to create initial CI/CD管道），然后您可以根据组织的要求对其进行自定义或更改。

此示例方法构建了生产环境和非生产环境，每个环境都有一个虚拟私有云（VPC）和一个 Amazon ECS 集群（配置为在两个可用区中运行）。这些环境由您的所有微服务共享，然后您创建一个CI/CD pipeline for each microservice. These CI/CD管道，从 AWS 的源存储库提取更改 CodeCommit，自动生成更改，然后将其部署到您的生产和非生产环境中。当管道成功完成其所有阶段后，您可以使用 URLs 在生产和非生产环境中访问该微服务。

先决条件和限制

先决条件

- 活跃 Amazon Web Services (AWS) account
- 包含 `starter-code.zip` 文件（附件）的现有 Amazon Simple Storage Service (Amazon S3) 存储桶。
- AWS Cloud Development Kit (AWS CDK)，已在您的账户中安装并配置。有关这方面的更多信息，请参阅 AWS CDK 文档中的 AWS CDK 入门。
- Python 3 和 `pip`，已安装并配置。有关这方面的更多信息，请参阅 Python 文档。
- 熟悉 AWS CDK、AWS、AWS、AW CodePipeline S CodeBuild、 CodeCommit亚马逊弹性容器注册表 (Amazon ECR) Container Registry、Amazon ECS 和 AWS Fargate。
- 熟悉 Docker。
- 对 CI/CD 的理解以及. DevOps

限制

- 适用一般的 Amazon Web Services Account 限制。有关这方面的更多信息，请参阅 AWS 一般参考文档中的 AWS 服务限额。

产品版本

- 使用 Node.js 版本 16.13.0 和 AWS CDK 版本 1.132.0 测试此代码。

架构

图表显示了以下工作流：

1. 应用程序开发人员将代码提交到 CodeCommit 存储库。
2. 管道已启动。
3. CodeBuild 构建 Docker 镜像并将其推送到亚马逊 ECR 存储库
4. CodePipeline 将新映像部署到非生产 Amazon ECS 集群中的现有 Fargate 服务。
5. Amazon ECS 将映像从 Amazon ECR 存储库提取至非生产 Fargate 服务。
6. 通过非生产 URL 执行测试。
7. 发布经理批准生产部署。
8. CodePipeline 将新映像部署到生产 Amazon ECS 集群中的现有 Fargate 服务
9. Amazon ECS 将映像从 Amazon ECR 存储库提取至生产 Fargate 服务。
10. 生产用户通过生产 URL 访问功能。

技术堆栈

- AWS CDK
- CodeBuild
- CodeCommit
- CodePipeline
- Amazon ECR
- Amazon ECS
- Amazon VPC

自动化和扩缩

您可以使用此模式的方法为部署在共享 AWS CloudFormation 堆栈中的微服务创建管道。自动化功能可以在每个 VPC 中创建多个 Amazon ECS 集群，还可以为部署在共享 Amazon ECS 集群中的微服务创建管道。但是，这要求您提供新资源信息作为管道堆栈输入。

工具

- AWS CDK — AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，用于在代码中定义云基础设施并通过 AWS 进行配置。 CloudFormation
- AWS CodeBuild — AWS CodeBuild 是一项完全托管的云端构建服务。 CodeBuild 编译您的源代码，运行单元测试，并生成随时可以部署的工件。
- AWS CodeCommit — AWS CodeCommit 是一项版本控制服务，可让您在 AWS 云中私下存储和管理 Git 存储库。 CodeCommit 您无需管理自己的源代码控制系统或担心扩展其基础架构。
- AWS CodePipeline — AWS CodePipeline 是一项持续交付服务，可用于对发布软件所需的步骤进行建模、可视化和自动化。您可以快速建模和配置软件发布过程的不同阶段。 CodePipeline 自动执行持续发布软件更改所需的步骤。
- Amazon ECS – Amazon Elastic Container Service (Amazon ECS) 是一项高度可扩展的快速容器管理服务，可用于运行、停止和管理集群上的容器。您可以在由 AWS Fargate 管理的无服务器基础架构上运行任务和服务。或者，为了更好地控制您的基础设施，您可以在您管理的亚马逊弹性计算云 (Amazon EC2) 实例集群上运行任务和服务。
- Docker - Docker 有助于开发人员打包、交付和运行任何应用程序，将其作为轻量级、便携且自给自足的容器。

代码

此模式代码可在 `cicdstarter.zip` 和 `starter-code.zip` 文件（附件）中找到。

操作说明

设置您的环境

Task	描述	所需技能
为 AWS CDK 设置工作目录。	1. 在您的本地计算机上创建名为 `cicdproject` 的目录。 2. 将 `cicdstarter.zip` 文件（附件）下载至	AWS DevOps，云基础设施

Task	描述	所需技能
	cicdproject 目录中并解压缩。这将创建一个名为 cicdstarter 的文件夹。 3. 运行 cd <user-home>/cicdproject/cicdstarter 命令。 4. 通过运行 python3 -m venv .venv 命令设置 Python 虚拟环境。 5. 运行 source ./.venv/bin/activate 命令。 6. 通过运行 aws configure 命令或使用以下环境变量配置 AWS 环境： • AWS_ACCESS_KEY_ID • AWS_SECRET_ACCESS_KEY • AWS_DEFAULT_REGION	

创建共享基础设施

Task	描述	所需技能
创建共享基础设施。	1. 在工作目录中运行 cd cicdvpcecs 命令。 2. 运行 pip3 install -r requirements.txt 命令，以安装所有必需 Python 依赖项	AWS DevOps，云基础设施

Task	描述	所需技能
	3. 运行 `cdk bootstrap command`，以设置适用于 AWS CDK 的 AWS 环境。 4. 运行 `cdk synth --context aws_account=<aws_account_ID> --context aws_region=<aws-region>` 命令。 5. 运行 `cdk deploy --context aws_account=<aws_account_ID> --context aws_region=<aws-region>` 命令。 6. AWS CloudFormation 堆栈创建以下基础设施： • 名为 `cicd-vpc-ecs/cicd-vpc-nonprod` 的非生产 VPC • 名为 `cicd-vpc-ecs/cicd-vpc-prod` 的生产 VPC • 名为 `cicd-ecs-nonprod` 的非生产 Amazon ECS 集群 • 名为 `cicd-ecs-prod` 的生产 Amazon ECS 集群	

Task	描述	所需技能
监控 AWS CloudFormation 堆栈。	1. 登录 AWS 管理控制台，打开 AWS CloudFormation 控制台，然后从列表中选择`cicd-vpc-ecs` 堆栈。 2. 在堆栈详细信息窗格中，选择事件选项卡，并监控堆栈创建进度。	AWS DevOps，云基础设施
测试 AWS CloudFormation 堆栈。	1. 创建 `cicd-vpc-ecs` AWS CloudFormation 堆栈后，请确保`cicd-vpc-ecs/cicd-vpc-nonprod` 和`cicd-vpc-ecs/cicd-vpc-prod` VPCs 已创建。 2. 确保 `cicd-ecs-nonprod` 和 `cicd-ecs-prod` Amazon ECS 集群已创建。 **Important** 请务必记录两个安全组 IDs 的， VPCs 并记录两个安全组 IDs 中的默认安全组的安全组 VPCs。	AWS DevOps，云基础设施

为微服务创建 CI/CD 管道

Task	描述	所需技能
为微服务创建基础设施。	1. 命名微服务。例如，此模式使用 `myservice1` 作为微服务名称。 2. 在工作目录中运行 `cd <working-directory>/cdkpipeline` 命令。 3. 运行 `pip3 install -r requirements.txt` 命令。 4. 运行此模式的其他信息部分中提供的完整 `cdk synth` 命令。 5. 运行此模式的其他信息部分中提供的完整 `cdk deploy` 命令。 **Note** 您也可以使用目录中的`cdk.json`文件为这两个命令提供值。	AWS DevOps，云基础设施
监控 AWS CloudFormation 堆栈。	打开 AWS CloudFormation 控制台并监控`myservice1-cicd-stack` 堆栈的进度。最终状态将更改为 `CREATE_COMPLETE` 。	AWS DevOps，云基础设施
测试 AWS CloudFormation 堆栈。	1. 在 AWS CodeCommit 控制台上，确认名为的存储	

Task	描述	所需技能
	库myservice1 存在且包含入门代码。 2. 在 AWS CodeBuild 控制台上，验证名为的构建项目myservice1 是否存在。 3. 在 Amazon ECR 控制台上，验证名为 myservice1 的 Amazon ECR 存储库是否存在。 4. 在 Amazon ECS 控制台上，验证名为 myservice1 的 Fargate 服务是否存在于非生产和生产 Amazon ECS 集群。 5. 在亚马逊弹性计算云 (Amazon EC2) 控制台上，验证非生产和生产应用程序负载均衡器是否已创建。记录下的 DNS 名称 ALBs。 6. 在 AWS CodePipeline 控制台上，验证名为的管道myservice1 是否存在。其必须包含 Source、Build、Deploy-NonProd 和 Deploy-Prod 阶段。管道也应包含 in progress 状态。 7. 监控管道，直至所有阶段均已完成。 8. 手动批准以生产。 9. 在浏览器窗口中，输入的 DNS 名称 ALBs。	

Task	描述	所需技能
	10.该应用程序应显示`Hello World`在非生产和生产URLs环境中。	
使用管道。	1. 打开您之前创建的 CodeCommit 存储库并打开该`index.js`文件。 2. 将 `Hello World`替换为 `Hello CI/CD`。 3. 保存并提交更改至主分支。 4. 验证管道是否已启动以及更改是否经过 `Build`、`Deploy-NonProd` 和 `Deploy-Prod` 阶段。 5. 手动批准生产。 6. 生产版和非生产版现在都 URLs 应该显示*`Hello CICD`*了。	AWS DevOps，云基础设施
对每项微服务重复此操作说明。	重复此操作说明中的任务，对每项微服务创建 CI/CD 管道。	AWS DevOps，云基础设施

相关资源

- 使用带 AWS CDK 的 Python
- AWS CDK Python 参考
- 使用 AWS CDK 创建 AWS Fargate 服务

其他信息

`cdk synth` 命令

```
cdk synth --context aws_account=<aws_account_number> --context
 aws_region=<aws_region> --context vpc_nonprod_id=<id_of_non_production
 VPC> --context vpc_prod_id=<id_of_production_VPC> --context
 ecssg_nonprod_id=< default_security_group_id_of_non-production_VPC>
 --context ecssg_prod_id=<default_security_group_id_of_production_VPC>
 --context code_commit_s3_bucket_for_code=<S3 bucket name> --context
 code_commit_s3_object_key_for_code=<Object_key_of_starter_code> --context
 microservice_name=<name_of_microservice>
```

`cdk deploy` command

```
cdk deploy --context aws_account=<aws_account_number> --context
 aws_region=<aws_region> --context vpc_nonprod_id=<id_of_non_production_VPC>
 --context vpc_prod_id=<id_of_production_VPC> --context ecssg_nonprod_id=<
 default_security_group_id_of_non-production_VPC> --context
 ecssg_prod_id=<default_security_group_id_of_production_VPC> --
context code_commit_s3_bucket_for_code=<S3 bucket name> --context
 code_commit_s3_object_key_for_code=<Object_key_of_starter_code> --context
 microservice_name=<name_of_microservice>
```

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 DevOps 实践和 AWS Cloud9 构建具有微服务的松散耦合架构

创建者：Alexandre Nardi (AWS)

摘要

注意： AWS Cloud9 不再向新客户开放。的现有客户 AWS Cloud9 可以继续照常使用该服务。了解更多

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

此模式演示了如何在无服务器架构中开发典型的 Web 应用程序，适用于开始在 Amazon Web Services (AWS) 上测试 DevOps 实践的开发人员和开发主管。它构建了一个示例应用程序，创建用于浏览和购买书籍的店面和后端，并提供可独立开发的微服务。该模式使用AWS Cloud9作为开发环境，使用亚马逊DynamoDB数据库作为数据存储，使用AWS和AWS等AWS服务作为持续集成 CodePipeline 和持续部署 (CI/CD) 功能。 CodeBuild

此模式将引导您完成以下开发活动：

- 创建标准 AWS Cloud9 开发环境
- 使用 AWS CloudFormation 模板创建 Web 应用程序和图书微服务
- 使用 AWS Cloud9 修改前端、提交更改与测试更改
- 创建并测试微服务的 CI/CD 管线
- 自动化单元测试

此模式的代码在 GitHub AWS Worksho DevOps End-to-End p 存储库中提供。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 从 AWS DevOps End-to-End 研讨会下载到您的电脑上的文件

> **Important**
>
> 在您的 AWS 账户中构建此演示应用程序会创建和消耗 AWS 资源。您负责用于创建和运行应用程序的 Amazon Web Services 和资源的成本。完成工作后，请务必删除所有资源以避免持续产生费用。有关清理说明，请参阅操作说明部分。

限制

本演练仅用于演示和开发之目的。要在生产环境中使用它，请参阅 AWS Identity and Access Management (IAM) 文档中的安全最佳实践，并对 IAM 角色、Amazon DynamoDB 和其他服务进行必要更改。Web 应用程序源自 AWS 书店演示应用程序，有关其他注意事项，请参阅自述文件的已知限制部分。

架构

书店应用程序的架构在 AWS 书店演示应用程序的自述文件的架构部分进行了说明。

从部署的角度来看，Bookstore Demo App 使用单个 CloudFormation 模板在一个堆栈中部署所有服务和对象。此模式进行了一些更改，以演示特定开发人员或团队如何在特定产品（书籍）中工作，并独立于应用程序的其余部分进行更新。因此，此模式的代码将 Books 微服务的 AWS Lambda 函数和相关对象分离到 CloudFormation 第二个模板中，该模板创建了 Books 堆栈。这使得可以看到使用 CI/CD 实践更新微服务。在下图中，虚线边框标识了书籍微服务。

工具

工具

- 用于测试的 Jest 框架 JavaScript
- Python 3.9

代码

此模式的源代码和模板可在 AWS Works GitHub ho DevOps End-to-End p 存储库中找到。在按操作说明部分中的步骤进行操作之前，请将存储库中的所有文件下载到您的计算机上。

> **Note**
>
> Epics 部分提供了本演练的高级步骤，为您提供有关该过程的一般信息。要完成每个步骤，请参阅 AWS Worksho DevOps End-to-End p 存储库中的自述文件，了解详细说明。

AWS Worksho DevOps End-to-End p 存储库扩展了 AWS Bookstore 演示应用程序存储库，并使用 AWS Cloud9 Bootstrapping 代码的修改版本创建 AWS Cloud9 IDE。

最佳实践

使用书店应用程序非常简单。以下是部分推荐的最佳实践：

- 安装应用程序时，为方便起见，您可使用自己选择的项目名称或使用默认名称 (`demobookstore`)。
- 在应用程序启动并运行后，如果您想继续测试一天，最好关闭 Amazon Neptune 数据库，因为数据库实例可能会产生额外费用。但请注意，数据库将在 7 天后自动启动。
- 有关代码的详细信息，请参阅 AWS 书店演示应用程序存储库的文档。它描述了每个微服务和表。
- 有关其他最佳实践，请参阅 “如果有时间，请查看一些挑战... AWS Worksho DevOps End-to-End p 存储库中自述文件的一部分。我们建议您查看这些信息，深入了解其他安全功能并实践解耦服务。

操作说明

下载源代码

Task	描述	所需技能
从中下载源代码 GitHub。	此模式的源代码和模板可在 AWS Works GitHub ho DevOps End-to-End p 存储库中找到。在按操作说明部分中的后续步骤进行操作之前，请将存储库中的所有文件下载到您的计算机上。 **Note** Epics 部分提供了本演练的高级步骤，为您提	应用程序开发人员

Task	描述	所需技能
	供有关该过程的一般信息。要完成每个步骤，请参阅 AWS Worksho DevOps End-to-End p 存储库中的自述文件，了解详细说明。 AWS Worksho DevOps End-to-End p 存储库扩展了 AWS Bookstore 演示应用程序存储库，并使用 AWS Cloud9 Bootstrapping 代码的修改版本创建 AWS Cloud9 IDE。	

构建书店 Web 应用程序与书籍微服务

Task	描述	所需技能
为书店应用程序创建前端和 Lambda 函数。	1. 登录CloudFormation 控制台，部署DemoBooks toreMainTemplate.y ml 模板以创建 DemoBookStoreStack 堆栈。这将创建书籍微服务外部的前端和 Lambda 函数。 2. 在堆栈的输出选项卡中，记下WebApplication标签下的网站网址。	开发人员
创建书籍微服务。	在CloudFormation 控制台上，部署DemoBookstoreBooks ServiceTemplate.ym	开发人员

Task	描述	所需技能
	l 模板以创建 DemoBooks ServiceStack 堆栈。	
测试您的应用程序。	使用 DemoBookStoreStack 堆栈中的网站 URL 访问书店应用程序。	开发人员

使用 Cloud9 环境来维护您的应用程序

Task	描述	所需技能
创建 AWS Cloud9 IDE。	在CloudFormation 控制台上，部署C9EnvironmentTemplate.yml 模板以创建 AWS Cloud9 环境。	开发人员、开发人员主管
创建 CodeCommit 存储库。	1. 登录 AWS CodeCommit 控制台，确认您有demobookstore-WebAssets 存储库，其中包含前端应用程序的代码。 2. 为书籍微服务创建名为 demobookstore-BooksService 的存储库。 3. 使用 git clone 命令克隆 AWS Cloud9 中的两个存储库（demobookstore-WebAssets 和 demobookstore-BooksService ）。	开发人员
更改前端代码并检查管线。	1. 使用 AWS Cloud9 在网页上进行部分代码更改。这将更新 demobookstore-WebAssets 存储库。	开发人员

Task	描述	所需技能
	2. 在 AWS CodePipeline 控制台上，验证 DemobookStore-assets- Pipeline 是否正在运行。 3. 通过浏览器刷新 Web 应用程序来测试它（在 Firefox 上为 Ctrl+F5）。	

为书籍微服务实施 CI/CD 管线

Task	描述	所需技能
为内部版本和服务更新添加 YAML 文件。	1. 在 AWS Cloud9，上传 `buildspec.yml` 和 `DemoBookstoreBooksServiceUpdateTemplate.yml` 文件。 • `buildspec.yml` 有构建说明，还包括自动测试的测试说明。此时将对它们进行注释，稍后将使用它们。 • `DemoBookstoreBooksServiceUpdateTemplate.yml` 是 `DemoBookstoreBooksServiceTemplate.yml` 的一个更新版本，将在管线的部署阶段使用。 2. 提交并推送文件。	开发人员

Task	描述	所需技能
为构建管线创建 S3 存储桶。	要创建 S3 存储桶，请按照 Amazon S3 文档中的说明进行操作。 • 存储桶名称必须全局唯一；例如，`demobookstore-books-service-pipeline-bucket-<YYYYMMDDHHMM>` 。 • 清除阻止所有公有访问复选框，然后选中我确认...复选框。	开发人员
使用 IAM 创建用于 CloudFormation 部署的角色。	创建一个 `demobookstore-CloudFormation-role` 角色并附加到 `AdministratorAccess` 策略。在下一个操作说明中，您可以重新配置这个角色以获得最低权限。	开发人员
创建一个新管线，以自动构建和部署书籍微服务。	如自述文件中所述，创建包含提交、生成和部署阶段的BooksService管道（例如demobookstore-- Pipeline）。	开发人员
在 AWS Cloud9 中测试您的微服务。	更改ListBooks函数并查看管道是否正常运行。	开发人员
自动执行 ListBooks Lambda 函数的单元测试。	在 AWS Cloud9 IDE，启用内部版本以运行单元测试并检查测试结果。有关说明，请参阅自述文件。	开发人员

（可选）实现其他功能

Task	描述	所需技能
确保您的解决方案安全。	配置 `demobookstore-CloudFormation-role` 为具有最低权限，并同时检查其他使用的角色。	开发人员
消除 CloudFormation 模板中的依赖关系。	`DemoBookstoreMainTemplate.yml` 模板和 `DemoBookstoreBooksServiceTemplate.yml` 模板之间交换信息的方法基于输出和导入。在这两个模板之间传递值会增加依赖项。要消除依赖项，请考虑使用 AWS Systems Manager Parameter Store。	开发人员
创建购物车微服务。	以书籍微服务为例，从 `DemoBookstoreMainTemplate.yml` 模板中删除与购物车相关的功能并创建购物车微服务。	开发人员

清理

Task	描述	所需技能
删除 S3 存储桶。	在 Amazon S3 控制台上，删除与示例 Web 应用程序关联的以下存储桶： • 为 AWS 书店演示应用程序创建的两个存储桶。存储桶名称以您在创建前	开发人员

Task	描述	所需技能
	端 CloudFormation 时为 AWS 提供的堆栈名称开头；例如，。DemoBookStoreStack • <YYYYMMDDHHMM>一个存储桶用于生成管道；例如，demobookstore-books-service-pipeline-bucket-。	
删除堆栈。	在CloudFormation 控制台上，删除与示例 Web 应用程序关联的堆栈： • DemoBooksServiceStack • DemoBookStoreStack 移除可能需要 90 多分钟。如果删除失败，请再次将其删除，并根据通知删除所有手动资源（例如 VPC 或网络接口）。	开发人员
删除 IAM 角色。	在 IAM 控制台上，删除以下角色： • `demobookstore-Cloudformation-role` • `demobookstore-BooksService-BuildProject-service-role` 有关 step-by-step说明，请参阅 I AM 文档。	开发人员

相关资源

- AWS 书店演示应用程序
- AWS Cloud9 Bootstrapping 示例
- 在 AWS CloudFormation 控制台上创建堆栈（AWS CloudFormation 文档）
- 创建存储桶（Amazon S3 文档）

其他信息

有关详细 step-by-step说明，请参阅 AWS Worksho DevOps End-to-End p GitHub 存储库中的自述文件。

关于 2023 年 5 月的更新：此模式已更新为使用较新版本的 Node 和 Python。我们更新了源代码中的多个程序包，并删除了 Glyphicon，因为它不再是免费的。我们还删除了对 AWS 书店演示应用程序存储库的所有依赖关系，因此这两个存储库现在可以独立发展。

使用 GitHub Actions 和 Terraform 构建 Docker 镜像并将其推送到 Amazon ECR

由 Ruchika Modi (AWS) 创作

摘要

此模式说明了如何创建可重复使用 GitHub 的工作流程来构建 Dockerfile 并将生成的映像推送到亚马逊弹性容器注册表 (Amazon ECR) Container Registry (Amazon ECR)。该模式使用 Terraform 和 Actions 自动执行 Dockerfiles 的构建过程。 GitHub 这最大限度地减少了人为错误的可能性，并大大缩短了部署时间。

向 GitHub 存储库的主分支 GitHub 推送操作会启动资源的部署。该工作流程根据 GitHub 组织和存储库名称的组合创建唯一的 Amazon ECR 存储库。然后，它会将 Dockerfile 镜像推送到 Amazon ECR 存储库。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 一个活跃的 GitHub 账户。
- 存储GitHub 库。
- 已安装并配置 Terraform 版本 1 或更高版本。
- 用于 Terraform 后端的亚马逊简单存储服务 (Amazon S3) 存储桶。
- 用于 Terraform 状态锁定和一致性的亚马逊 DynamoDB 表。该表必须有一个名为的分区键`LockID`，其类型必须为`String`。如果未进行此配置，则状态锁定将被禁用。
- 一个 AWS 身份和访问管理 (IAM) 角色，有权为 Terraform 设置 Amazon S3 后端。有关配置说明，请参阅 Terraform 文档。

限制

此可重复使用的代码仅通过 GitHub 操作进行了测试。

架构

目标技术堆栈

- Amazon ECR 存储库
- GitHub 行动
- Terraform

目标架构

该图阐释了以下内容：

1. 用户将 Dockerfile 和 Terraform 模板添加到存储库中。 GitHub

2. 这些新增内容启动了 GitHub 操作工作流程。

3. 该工作流程会检查 Amazon ECR 存储库是否存在。否则，它将根据 GitHub 组织和存储库名称创建存储库。

4. 该工作流程构建 Dockerfile 并将映像推送到 Amazon ECR 存储库。

工具

Amazon 服务

- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展和可靠的托管容器注册服务。

其他工具

- GitHub 操作已集成到 GitHub 平台中，可帮助您在 GitHub 仓库中创建、共享和运行工作流程。您可以使用 GitHub Actions 来自动执行诸如构建、测试和部署代码之类的任务。
- Terraform 是一款基础设施即代码 (IaC) 工具 HashiCorp ，可帮助您创建和管理云和本地基础架构。

代码存储库

此模式的代码可在 GitHub Docker ECR 操作工作流存储库中找到。

- 创建 GitHub 操作时，Docker 工作流程文件将保存在此存储库的`/.github/workflows/`文件夹中。此解决方案的工作流程位于工作流.y aml 文件中。
- 该`e2e-test`文件夹提供了一个示例 Dockerfile 供参考和测试。

最佳实践

- 有关编写 Dockerfile 的最佳实践，请参阅 Docker 文档。
- 为 Amazon ECR 使用 VPC 终端节点。VPC 终端节点由 AWS 提供支持 PrivateLink，AWS 是一种允许您 APIs 通过私有 IP 地址私有访问 Amazon ECR 的技术。对于使用 Fargate 启动类型的 Amazon ECS 任务，VPC 终端节点允许任务从 Amazon ECR 提取私有镜像，而无需为任务分配公有 IP 地址。

操作说明

设置 OIDC 提供商和存储库 GitHub

Task	描述	所需技能
配置 OpenID Connect。	创建 OpenID Connect (OIDC) 提供商。您将在此操作中使用的 IAM 角色的信任策略中使用该提供商。有关说明，请参阅文档中的在亚马逊 Web Services 中配置 OpenID Connect。 GitHub	AWS 管理员、AWS DevOps、常规 AWS
克隆 GitHub 存储库。	将 GitHub Docker ECR 操作工作流存储库克隆到您的本地文件夹： `$git clone https://github.com/aws-samples/docker-ecr-actions-workflow`	DevOps 工程师

自定义 GitHub 可重复使用的工作流程并部署 Docker 镜像

Task	描述	所需技能
自定义启动 Docker 工作流程的事件。	此解决方案的工作流程在 workflow.yaml 中。此脚本当	DevOps 工程师

Task	描述	所需技能
	前配置为在收到workflow_dispatch 事件时部署资源。您可以通过将事件更改为另一个父工作流workflow_call 并从其他父工作流调用工作流程来自定义此配置。	

Task	描述	所需技能

自定义工作流程。	w orkflow.yaml 文件被配置为创建动态、可重复使用的工作流程。 GitHub 您可以编辑此文件以自定义默认配置，或者如果您使用`workflow_dispatch` 事件手动启动部署，则可以从 Actions 控制台传递输入值。 GitHub
	• 请务必指定正确的 AWS 账户 ID 和目标区域。
	• 创建 Amazon ECR 生命周期策略（参见示例策略）并相应地更新默认路径 (`e2e-test/policy.json`)。
	• 工作流程文件需要两个 IAM 角色作为输入：
	  • 一个 IAM 角色，有权为 Terraform 设置 Amazon S3 后端（参见 “先决条件” 部分）。您可以在中更新默认角色名称`workload-assumable-role` 。`yaml`相应地归档。
	  • 具有访问权限的 IAM 角色 GitHub。亚马逊 ECR 政策中也使用此角色来限制亚马逊 ECR 的操作。有关更多信息，请参阅 data.tf 文件。
		DevOps 工程师

Task	描述	所需技能
部署 Terraform 模板。	该工作流程会根据您配置的事件自动部署用于创建 Amazon ECR 存储库的 Terraform 模板。 GitHub 这些模板在 Github 存储库的根目录下以 `.tf` 文件形式提供。	AWS DevOps， DevOps 工程师

故障排除

事务	解决方案
将 Amazon S3 和 DynamoDB 配置为 Terraform 远程后端时出现问题或错误。	按照 Terraform 文档中的说明为远程后端配置在 Amazon S3 和 DynamoDB 资源上设置所需的权限。
无法使用该`workflow_dispatch` 事件运行或启动工作流程。	只有在主分支上配置工作流时，配置为从 `workflow_dispatch` 事件部署的工作流程才会起作用。

相关资源

- 重复使用工作流程（GitHub 文档）
- 触发工作流程（GitHub 文档）

使用 AWS CodeCommit、AWS 和 AWS Device Farm 构建和测试 iOS 应用程序 CodePipeline

由 Abdullahi Olaoye (AWS) 编写

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

此模式概述了创建持续集成和持续交付 (CI/CD) 管道的步骤，该管道使用 AWS 在 AWS CodePipeline 上的真实设备上构建和测试 iOS 应用程序。该模式使用 AWS CodeCommit 来存储应用程序代码，使用 Jenkins 开源工具来构建 iOS 应用程序，使用 AWS Device Farm 在真实设备上测试构建的应用程序。这三个阶段是使用 AWS 在管道中共同编排的。 CodePipeline

这种模式基于 AWS DevOps 博客上的 “使用 AW S 和移动服务构建和测试 iOS DevOps 和 iPadOS 应用程序” 一文。有关详细说明，请参阅此博客文章。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 一个 Apple 开发人员账户
- 编译服务器 (macOS)
- Xcode 版本 11.3（已在编译服务器上安装和设置）
- AWS 命令行界面（AWS CLI）已在工作站上安装并配置
- Git 基础知识

限制

- 应用程序编译服务器必须运行 macOS。
- 生成服务器必须具有公有 IP 地址，因此 CodePipeline 可以远程连接到该服务器以启动构建。

架构

源技术堆栈

- 本地 iOS 应用程序构建过程，涉及在物理设备上使用模拟器或手动测试

目标技术堆栈

- 用于 CodeCommit 存储应用程序源代码的 AWS 存储库
- 使用 Xcode 构建应用程序的 Jenkins 服务器
- 用于在真实设备上测试应用程序的 AWS Device Farm 设备池

目标架构

当用户向源存储库提交更改时，管道 (AWS CodePipeline) 会从源存储库获取代码，启动 Jenkins 构建，然后将应用程序代码传递给 Jenkins。构建完成后，管道会检索构建构件并启动 AWS Device Farm 作业，根据设备池测试应用程序。

工具

- AWS CodePipeline 是一项完全托管的持续交付服务，可帮助您自动执行发布管道，实现快速可靠的应用程序和基础设施更新。 CodePipeline 每次发生代码更改时，都会根据您定义的发布模型自动执行发布过程的构建、测试和部署阶段。
- AWS CodeCommit 是一项完全托管的源代码控制服务，可托管基于 Git 的安全存储库。它使团队可以轻松地在安全且高度可扩展的生态系统中就代码进行协作。 CodeCommit 无需操作自己的源代码控制系统或担心扩展其基础架构。
- AWS Device Farm 是一项应用程序测试服务，可让您通过在各种桌面浏览器和真实移动设备上测试网页和移动应用程序来提高其质量，而无需预置和管理任何测试基础设施。
- Jenkins 是一款开源自动化服务器，它使开发人员能够构建、测试和部署其软件。

操作说明

设置构建环境

Task	描述	所需技能
在运行 macOS 的编译服务器上安装 Jenkins。	Jenkins 将用于构建应用程序，因此您必须首先将其安装至编译服务器上。要获取有关此任	DevOps

Task	描述	所需技能
	务和后续任务的详细说明，请参阅本模式末尾的 “相关资源” 部分中的 AWS 博客文章 “使用 AWS DevOps 和移动服务以及其他资源构建和测试 iOS 和 iPadOS 应用程序”。	
配置 Jenkins。	按屏幕上的说明进行配置 Jenkins。	DevOps
安装适用于 Jenkins 的 AWS CodePipeline 插件。	为了让 Jenkins 与 AWS CodePipeline 服务进行交互，此插件必须安装在 Jenkins 服务器上。	DevOps
创建一个 Jenkins 自由式项目。	在 Jenkins 中创建一个自由式项目。配置项目以指定触发器和其他构建配置选项。	DevOps

配置 AWS Device Farm

Task	描述	所需技能
创建 Device Farm 项目。	打开 AWS Device Farm 控制台。创建一个项目和一个设备池进行测试。有关说明，请参见博客文章。	开发人员

配置源存储库

Task	描述	所需技能
创建 CodeCommit 存储库。	创建一个存储源代码的存储库。	DevOps

Task	描述	所需技能
将应用程序代码提交至存储库。	Connect 连接到您创建的 CodeCommit 存储库。将代码从本地计算机推送至存储库。	DevOps

配置管道

Task	描述	所需技能
在 AWS 中创建管道 CodePipeline。	打开 AWS CodePipeline 控制台，然后创建管道。该管道协调了 CI/CD 流程所有阶段。有关说明，请参阅 AWS 博客文章使用 AW S 和移动服务构建和测试 iOS DevOps 和 iPadOS 应用程序。	DevOps
向管道中添加测试阶段。	要添加测试阶段并将其与 AWS Device Farm 集成，请编辑管道。	DevOps
启动管道。	要启动管道和 CI/CD 流程，请选择 “发布更改”。	DevOps

查看应用程序测试结果

Task	描述	所需技能
查看测试结果。	在 AWS Device Farm 控制台，选择您创建的项目，然后查看测试结果。控制台将显示每项测试详细信息。	开发人员

相关资源

Step-by-step 此模式的说明

- 使用 AW@@ S 和移动服务构建和测试 iOS DevOps 和 iPadOS 应用程序（AWS DevOps 博客文章）

配置 AWS Device Farm

- AWS Device Farm 控制台

配置源存储库

- 创建 AWS CodeCommit 存储库
- 连接到 AWS CodeCommit 存储库

配置管道

- AWS CodePipeline 控制台

其他资源

- AWS CodePipeline 文档
- AWS CodeCommit 文档
- AWS Device Farm 文档
- Jenkins 文档
- 在 macOS 上安装 Jenkins
- 适用于 Jenkins 的 AWS CodePipeline 插件
- Xcode 安装
- AWS CLI 安装和配置
- Git 文档

使用 cdk-nag 规则包查看 AWS CDK 应用程序或 CloudFormation 模板以了解最佳实践

由 Arun Donti 编写

摘要

此模式说明了如何使用 cdk-nag 实用程序通过使用规则包组合来检查 AWS Cloud Development Kit (AWS CDK) 应用程序的最佳实践。cdk-nag是一个受 cfn_nag 启发的开源项目。它通过使用 AWS CDK Aspects 实施评估包中的规则，例如 AWS 解决方案库、健康保险流通和责任法案 (HIPAA) 以及美国国家标准与技术研究所 (NIST) 800-53。您可以使用这些包中的规则检查您的 AWS CDK 应用程序的最佳实践，根据最佳实践检测和修复代码，并抑制您不想在评估中使用的规则。

你也可以使用 cdk-nag 通过 cloud formation-include 模块来 CloudFormation 检查你的 AWS 模板。

有关所有可用包的信息，请参阅 cdk-nag 存储库的规则部分。评估包适用于：

- AWS 解决方案库
- HIPAA 安全
- NIST 800-53 第 4 版
- NIST 800-53 第 5 版
- 支付卡行业数据安全标准 (PCI DSS) 3.2.1

先决条件和限制

先决条件

- 使用 AWS CDK 的应用程序

工具

- AWS CDK — Cloud Development Kit (AWS CDK) 是一个软件开发框架，用于在代码中定义云基础设施并通过 AWS 进行配置。 CloudFormation
- AWS CloudFormation — AWS CloudFormation 可帮助您建模和设置 AWS 资源，快速一致地配置这些资源，并在资源的整个生命周期中对其进行管理。您可以使用模板来描述资源及其依赖关系，然后将它们作为堆栈一起启动和配置，而不必单独管理资源。您可以跨多个 Amazon Web Services account 和 AWS 区域管理和预置堆栈。

操作说明

将 cdk-nag 与 AWS CDK 应用程序集成

Task	描述	所需技能
了解 cdk-nag。	导航到 cdk-nag GitHub 存储库并通读文档。	应用程序开发人员
在您的 AWS CDK 应用程序安装 cdk-nag 软件包。	要在您的 AWS CDK 应用程序中使用 cdk-nag，您必须先安装它。 cdk-nag 可以从 PyPI、npm 和 Apache Maven 下载。 NuGet有关可用版本和下载位置的最新信息，请参见存储库中的自述文件。	应用程序开发人员
选择你的 NagPacks。	cdk-nag 有不同的规则包叫做 。NagPacks每个都 NagPack 包含符合特定标准的规则。例如，AWS 解决方案 NagPack 包含一般最佳实践，而 NIST 800-53 rev 5 NagPack 可以帮助实现合规性。您可以将多个包应用 NagPacks 于您的应用程序，也可以根据需要添加和删除包。有关可用包的列表，请参阅 GitHub 存储库中的自述文件。有关每个包中各个规则的信息，请参阅 GitHub 存储库的 “规则” 部分。	应用程序开发人员
将 cdk-nag 集成至您的 AWS CDK 应用程序中。	您可以在应用程序级别将 cdk-nag 集成至应用程序中，也可以将其集成到应用程序中的各个阶段或堆栈中。例如，要将 AWS 解决方案和 HIPAA 安全 NagPacks 集成到应用程序级	应用程序开发人员

Task	描述	所需技能

	别的 AWS CDK v2 TypeScript 应用程序中，您可以使用以下代码：
	
	`import { App, Aspects } from 'aws-cdk-lib';`
	`import { CdkTestStack } from '../lib/cdk-test-stack';`
	`import { AwsSolutionsChecks, HIPAASecurityChecks } from 'cdk-nag';`
	
	`const app = new App();`
	`new CdkTestStack(app, 'CdkNagDemo');`
	`// Simple rule informational messages`
	`Aspects.of(app).add(new AwsSolutionsChecks());`
	`// Additional explanations on the purpose of triggered rules`
	`Aspects.of(app).add(new HIPAASecurityChecks({ verbose: true }));`	

相关资源

- cdk-nag 代码存储库
- Construct Hub 的 cdk-nag

为在 Amazon EKS 上运行的应用程序配置双向 TLS 身份验证

由 Mahendra Siddappa (AWS) 编写

摘要

基于证书的传输层安全性协议（TLS）是一个可选的 TLS 组件，可在服务器和客户端之间提供双向对等身份验证。使用双向 TLS，客户端必须在会话协商过程中提供 X.509 证书。服务器使用此证书来识别和验证客户端。

双向 TLS 是物联网 (IoT) 应用的常见要求，可用于开放银行等 business-to-business应用或标准。

此模式描述如何使用 NGINX 入口控制器为运行在 Amazon Elastic Kubernetes Service（Amazon EKS）集群上的应用程序配置双向 TLS。您可以通过注释入口资源来为 NGINX 入口控制器启用内置的双向 TLS 功能。有关 NGINX 控制器上双向 TLS 注释的更多信息，请参阅 Kubernetes 文档中的客户端证书身份验证。

> **Important**
>
> 此模式使用自签名证书。我们建议您只在测试集群中使用此功能，而不要在生产环境中使用。如果您想在生产环境中使用此模式，则可以使用 AWS 私有证书颁发机构（AWS Private CA）或您现有的公有密钥基础设施（PKI）标准来颁发私有证书。

先决条件和限制

先决条件

- 一个 Amazon Web Services（AWS）有效账户。
- 现有 Amazon EKS 集群。
- AWS 命令行界面（AWS CLI）版本 1.7 或更高版本，已在 macOS、Linux 或 Windows 上安装并配置。
- 已安装并配置的 kubectl 命令行实用程序，以便访问 Amazon EKS 集群。有关这方面的更多信息，请参阅 Amazon EKS 文档中的安装 kubectl。
- 用于测试应用程序的现有域名系统（DNS）名称。

限制

- 此模式使用自签名证书。我们建议您只在测试集群中使用此功能，而不要在生产环境中使用。

架构

技术堆栈

- Amazon EKS
- Amazon Route 53
- Kubectl

工具

- Amazon Elastic Kubernetes Service (Amazon EKS) 可帮助您在 AWS 上运行 Kubernetes，而无需安装或维护您自己的 Kubernetes 控制面板或节点。
- Amazon Route 53 是一种可用性高、可扩展性强的 DNS Web 服务。
- Kubectl 是命令行实用程序，用于与 Amazon EKS 集群交互。

操作说明

生成自签名证书

Task	描述	所需技能
生成 CA 密钥和证书。	通过运行以下命令生成证书颁发机构（CA）密钥和证书。 `openssl req -x509 -sha256 -newkey rsa:4096 -keyout ca.key -out ca.crt -days 356 -nodes -subj '/CN=Test Cert Authority'`	DevOps 工程师

Task	描述	所需技能
生成服务器密钥和证书，并使用 CA 证书进行签名。	通过运行以下命令生成服务器密钥和证书，并使用 CA 证书进行签名。 `openssl req -new -newkey rsa:4096 -keyout server.key -out server.csr -nodes -subj '/CN= <your_domain_name> ' && openssl x509 -req -sha256 -days 365 -in server.csr -CA ca.crt -CAkey ca.key -set_serial 01 -out server.crt` **Important** 请务必使用现有域名`<your_domain_name>` 进行替换。	DevOps 工程师
生成客户端密钥和证书，并使用 CA 证书进行签名。	通过运行以下命令生成客户端密钥和证书，并使用 CA 证书进行签名。 `openssl req -new -newkey rsa:4096 -keyout client.key -out client.csr -nodes -subj '/CN=Test' && openssl x509 -req -sha256 -days 365 -in client.csr -CA ca.crt -CAkey ca.key -set_serial 02 -out client.crt`	DevOps 工程师

部署 NGINX 入口控制器

Task	描述	所需技能
将 NGINX 入口控制器部署到 Amazon EKS 集群中。	使用以下命令部署 NGINX 入口控制器。 `kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.7.0/deploy/static/provider/aws/deploy.yaml`	DevOps 工程师
验证 NGINX 入口控制器服务正在运行。	使用以下命令验证 NGINX 入口控制器服务正在运行。 `kubectl get svc -n ingress-nginx` **Important** 确保服务字段地址包含网络负载均衡器的域名。	DevOps 工程师

在 Amazon EKS 集群中创建命名空间来测试双向 TLS

Task	描述	所需技能
在 Amazon EKS 集群中创建命名空间。	通过运行以下命令在 Amazon EKS 集群中创建名为 `mtls` 的命名空间。	DevOps 工程师

Task	描述	所需技能
	`kubectl create ns mtls` 这将部署示例应用程序来测试双向 TLS。	

为示例应用程序创建部署和服务

Task	描述	所需技能

在 mtls 命名空间中创建 Kubernetes 部署和服务。	创建一个名为 `mtls.yaml` 的文件。将以下代码粘贴到该文件中。
	
	`kind: Deployment`
	`apiVersion: apps/v1`
	`metadata:`
	`  name: mtls-app`
	`  labels:`
	`  app: mtls`
	`spec:`
	`  replicas: 1`
	`  selector:`
	`  matchLabels:`
	`  app: mtls`
	`  template:`
	`  metadata:`
	`  labels:`
	`  app: mtls`
	`  spec:`
	`  containers:`
	`  - name: mtls-app`
	`  image: hashicorp/http-echo`
	`  args:`
	`  - "-text=mTLS is working"`
		DevOps 工程师

Task	描述	所需技能

	``
	--- 	
	
	kind: Service
	apiVersion: v1
	metadata:
	  name: mtls-service
	spec:
	  selector:
	    app: mtls
	  ports:
	    - port: 5678 #
	 Default port for image
	``
	
	通过运行以下命令在 `mtls` 命名空间中创建 Kubernetes 部署和服务。
	
	``
	kubectl create -f
	 mtls.yaml -n mtls
	``	

验证已创建 Kubernetes 部署。	运行以下命令验证部署是否已创建，且有一个容器组（pod）处于可用状态。 ``` kubectl get deploy -n mtls ```	DevOps 工程师
验证是否已创建 Kubernetes 服务。	通过运行以下命令验证是否已创建 Kubernetes 服务。 ``` kubectl get service -n mtls ```	DevOps 工程师

在 mtls 命名空间中创建密钥

Task	描述	所需技能
创建入口资源的密钥。	运行以下命令，使用您之前创建的证书为 NGINX 入口控制器创建密钥。 `kubectl create secret generic mtls-certs --from-file=tls.crt=server.crt --from-file=tls.key=server.key --from-file=ca.crt=ca.crt -n mtls` 您的密钥有一个服务器证书供客户端识别服务器，还有一个 CA 证书供服务器验证客户端证书。	DevOps 工程师

在 mtls 命名空间中创建入口资源

Task	描述	所需技能

在 mtls 命名空间中创建入口资源。	创建一个名为 `ingress.yaml` 的文件。将以下代码粘贴到文件中（将 `<your_domain_name>` 替换为现有域名）。
	`apiVersion: networking.k8s.io/v1`
	`kind: Ingress`
	`metadata:`
	`  annotations:`
		DevOps 工程师

Task	描述	所需技能

	<pre> nginx.ingress.kube
	rnetes.io/auth-tls-
	verify-client: "on"
	    nginx.ingress.kube
	rnetes.io/auth-tls-
	secret: mtls/mtls-certs
	  name: mtls-ingress
	spec:
	  ingressClassName:
	 nginx
	  rules:
	  - host: "*.<your_
	domain_name>"
	    http:
	      paths:
	      - path: /
	        pathType: Prefix
	        backend:
	          service:
	            name: mtls-
	service
	            port:
	              number:
	 5678
	  tls:
	  - hosts:
	    - "*.<your_
	domain_name>"
	    secretName: mtls-
	certs</pre>
	通过运行以下命令以在 `mtls` 命名空间中创建入口资源。
	<pre>kubectl create -f
	 ingress.yaml -n mtls</pre>	

Task	描述	所需技能
	这意味着 NGINX 入口控制器可以将流量路由到示例应用程序。	
验证入口资源是否已创建。	通过运行以下命令验证是否已创建入口资源。 `kubectl get ing -n mtls` **Important** 确保入口资源的地址显示为 NGINX 入口控制器创建的负载均衡器。	DevOps 工程师

配置 DNS 以将主机名指向负载均衡器

Task	描述	所需技能
创建指向 NGINX 入口控制器的负载均衡器的 CNAME 记录。	登录 AWS 管理控制台，打开 Amazon Route 53 控制台，然后创建将 `mtls.<your_domain_name>` 指向 NGINX 入口控制器的负载均衡器的规范名称（CNAME）记录。 有关更多信息，请参阅 Route 53 文档中的使用 Route 53 控制台创建记录。	DevOps 工程师

测试应用程序

Task	描述	所需技能
在没有证书的情况下测试双向 TLS 设置。	运行以下命令。 `curl -k https://mtls.<your_domain_name>` 您应该收到“400 未发送必需的 SSL 证书”错误响应。	DevOps 工程师
在有证书的情况下测试双向 TLS 设置。	运行以下命令。 `curl -k https://mtls.<your_domain_name> --cert client.crt --key client.key` 您应该收到“mTLS 正在运行”的响应。	DevOps 工程师

相关资源

- 通过使用 Amazon Route 53 控制台创建记录
- 在 Amazon EKS 上使用带有 NGINX 入口控制器的网络负载均衡器
- 客户证书认证

使用 AWS 自动创建 AppStream 2.0 资源 CloudFormation

创建者：Ram Kandaswamy (AWS)

摘要

此模式提供了使用 AWS CloudFormation 模板在亚马逊网络服务 (AWS) 云中自动创建 Amazon AppStream 2.0 资源的代码示例和步骤。该模式向您展示如何使用 AWS CloudFormation 堆栈自动创建 AppStream 2.0 应用程序资源，包括映像生成器、映像、队列实例和堆栈。您可以使用桌面或应用程序交付模式，在 HTML5兼容的浏览器上将 AppStream 2.0 应用程序流式传输给最终用户。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 接受 AppStream 2.0 条款和条件
- AppStream 资源基础知识，例如堆栈、舰队和图像生成器

限制

- 创建与 AppStream 2.0 实例关联的 AWS Identity and Access Management (IAM) 角色后，您无法修改该实例。
- 创建 AppStream 2.0 映像生成器实例后，您无法修改该实例的属性（例如子网或安全组）。

架构

下图向您展示了如何使用 AWS CloudFormation 模板自动创建 AppStream 2.0 资源。

图表显示了以下工作流：

1. 您可以根据此模式的 “其他信息” 部分中的 YAML 代码创建 AWS CloudFormation 模板。
2. AWS CloudFormation 模板创建了一个 AWS CloudFormation 测试堆栈。
   a. （可选）您可以使用 AppStream 2.0 创建映像生成器实例。
   b. （可选）您可以使用自定义软件创建 Windows 映像。
3. AWS CloudFormation 堆栈创建一个 AppStream 2.0 队列实例和堆栈。
4. 您可以在 HTML5兼容的浏览器上将 AppStream 2.0 资源部署给最终用户。

技术堆栈

- 亚马逊 AppStream 2.0
- AWS CloudFormation

工具

- Amazon AppStream 2.0 是一项完全托管的应用程序流服务，可让您随时随地即时访问您的桌面应用程序。 AppStream 2.0 管理托管和运行应用程序所需的 AWS 资源，自动扩展，并按需向用户提供访问权限。
- AWS CloudFormation 可帮助您建模和设置 AWS 资源，快速一致地配置这些资源，并在资源的整个生命周期中对其进行管理。您可以使用模板来描述资源及其依赖关系，然后将它们作为堆栈一起启动和配置，而不必单独管理资源。您可以跨多个 Amazon Web Services account 和 AWS 区域 管理和预置堆栈。

操作说明

（可选）创建 AppStream 2.0 镜像

Task	描述	所需技能
安装自定义软件和创建映像。	1. 安装您计划部署给用户的 AppStream 2.0 应用程序。 2. 使用 Photon 创建图像代理或 PowerShell 脚本为您的自定义软件创建新的 Windows 映像。 **Note** 考虑使用 Windows AppLocker 功能进一步锁定图像。	AWS DevOps，云架构师

部署 AWS CloudFormation 模板

Task	描述	所需技能
更新 AWS CloudFormation 模板。	1. 将此模式的其他信息部分中的代码保存为 YAML 文件。 2. 使用环境中参数所需值更新 YAML 文件。	AWS 系统管理员、云管理员、云架构师、常规 AWS、AWS 系统管理员
使用模板创建 AWS CloudFormation 堆栈。	1. 登录 AWS 管理控制台并打开 AWS CloudFormation 控制台。 2. 在导航窗格中，选择堆栈。 3. 选择创建堆栈，然后选择使用新资源(标准)。 4. 在先决条件 — 准备模板部分，请选择模板已就绪。 5. 在指定模板部分，选择上传模板 文件。 6. 选择 “选择文件”，然后选择更新后的 AWS CloudFormation 模板。 7. 完成向导中的剩余步骤创建您的堆栈。	应用程序所有者、AWS 系统管理员、Windows 工程师

相关资源

参考

- 开始使用 Amazon AppStream 2.0：使用示例应用程序进行设置
- 创建 AppStream 2.0 舰队和堆栈

教程和视频

- 亚马逊 AppStream 2.0 用户工作流程

- 如何将旧版 Windows Forms 应用程序迁移到亚马逊 AppStream 2.0
- AWS re: Invent 2018：使用亚马逊 AppStream 2.0 安全交付桌面应用程序 (01) BAP2

其他信息

以下代码是允许您自动创建 AppStream 2.0 资源的 AWS CloudFormation 模板示例。

```
AWSTemplateFormatVersion: 2010-09-09
Parameters:
  SubnetIds:
    Type: 'List<AWS::EC2::Subnet::Id>'
  testSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup::Id'
  ImageName:
    Type: String
Resources:

  AppStreamFleet:
    Type: 'AWS::AppStream::Fleet'
    Properties:
      ComputeCapacity:
        DesiredInstances: 5
      InstanceType: stream.standard.medium
      Name: appstream-test-fleet
      DisconnectTimeoutInSeconds: 1200
      FleetType: ON_DEMAND
      IdleDisconnectTimeoutInSeconds: 1200
      ImageName: !Ref ImageName
      MaxUserDurationInSeconds: 345600
      VpcConfig:
        SecurityGroupIds:
          - !Ref testSecurityGroup
        SubnetIds: !Ref SubnetIds
  AppStreamStack:
    Type: 'AWS::AppStream::Stack'
    Properties:
      Description: AppStream stack for test
      DisplayName: AppStream test Stack
      Name: appstream-test-stack
      StorageConnectors:
        - ConnectorType: HOMEFOLDERS
      UserSettings:
```

```
        - Action: CLIPBOARD_COPY_FROM_LOCAL_DEVICE
          Permission: ENABLED
        - Action: CLIPBOARD_COPY_TO_LOCAL_DEVICE
          Permission: ENABLED
        - Action: FILE_DOWNLOAD
          Permission: ENABLED
        - Action: PRINTING_TO_LOCAL_DEVICE
          Permission: ENABLED
  AppStreamFleetAssociation:
    Type: 'AWS::AppStream::StackFleetAssociation'
    Properties:
      FleetName: appstream-test-fleet
      StackName: appstream-test-stack
    DependsOn:
      - AppStreamFleet
      - AppStreamStack
```

使用 Firelens 日志路由器为 Amazon ECS 创建自定义日志解析器

由 Varun Sharma (AWS) 创建

摘要

Firelens 是适用于 Amazon Elastic Container Service （Amazon ECS）和 AWS Fargate 的日志路由器。您可以使用 Firelens 将容器日志从 Amazon ECS 路由到亚马逊 CloudWatch 和其他目的地（例如 Splunk 或 S um o Logic）。Firelens 使用 Fluentd 或 Fluent Bit 作为日志记录代理，这意味着您可以使用 Amazon ECS 任务定义参数来路由日志。

通过选择在源级别解析日志，您可以分析您的日志记录数据并执行查询，从而更加高效和有效地响应操作问题。由于不同的应用程序具有不同的日志模式，因此您需要使用自定义解析器来构建日志，以便在最终目的地更轻松地进行搜索。

此模式使用带有自定义解析器的 Firelens 日志路由器，将日志 CloudWatch 从在 Amazon ECS 上运行的示例 Spring Boot 应用程序推送到。然后，您可以使用 Amazon CloudWatch Logs Insights 根据自定义解析器生成的自定义字段筛选日志。

先决条件和限制

先决条件

- 活跃 Amazon Web Services (AWS) account
- AWS 命令行界面（AWS CLI）已在本地计算机上安装和配置。
- 已在本地计算机上安装并配置的 Docker。
- Amazon Elastic Container Registry (Amazon ECR) 上现有的基于 Spring Boot 的容器化应用程序。

架构

技术堆栈

- CloudWatch
- Amazon ECR
- Amazon ECS
- Fargate
- Docker

- Fluent Bit

工具

- Amazon ECR - Amazon Elastic Container Registry (Amazon ECR) 是一项由 AWS 托管的容器映像注册表服务，该服务安全可靠，且可扩展。
- Amazon ECS - Amazon Elastic Container Service (Amazon ECS) 是一项高度可扩展的快速容器管理服务，可轻松在集群上运行、停止和管理容器。
- AWS 身份识别和访问管理(IAM) - IAM 是一项 Web 服务，用于安全控制 Amazon Web Services 的访问。
- AWS CLI - AWS 命令行界面（AWS CLI）是一种开源工具，可让您使用命令行 Shell 中的命令与 Amazon Web Services 交互。
- Docker - Docker 是用于开发、发布和运行应用程序的开放平台。

代码

此模式附加了以下文件：

- `customFluentBit.zip` - 包含用于添加自定义解析和配置的文件。
- `firelens_policy.json` - 包含用于创建 IAM policy 的策略文档。
- `Task.json` - 包含 Amazon ECS 的示例任务定义。

操作说明

创建自定义 Fluent Bit 映像

Task	描述	所需技能
创建 Amazon ECR 存储库。	登录 AWS 管理控制台，打开 Amazon ECR 控制台，然后创建一个名为 `fluentbit_custom` 的存储库。 有关此内容的更多信息，请参阅 Amazon ECR 文档中的创建存储库。	系统管理员、开发人员

Task	描述	所需技能
解压 customFluentBit .zip 压缩包。	1. 将 `customFluentBit.zip` 软件包（附件）下载至本地计算机。 2. 运行以下命令解压缩到 `customFluentBit` 目录中：`unzip -d customFluentBit.zip` 3. 该目录包含添加自定义解析和配置所需以下文件： • `parsers/springboot_parser.conf` - 包含解析器指令并定义自定义解析器的正则表达式(regex)模式。您可以为您的特定解析器添加正则表达式模式。 • `conf/parse_springboot.conf` - 包含筛选器和服务指令。 • Dockerfile	

Task	描述	所需技能
创建自定义 Docker 映像。	1. 将目录更改为 `customFluentBit` 。 2. 打开 Amazon ECR 控制台，选择 `fluentbit_custom` 存储库，然后选择查看推送命令。 3. 上传您的项目。 4. 上传完成后，复制生成的 URL。在 Amazon ECS 中创建容器时，此 URL 是必需的 有关更多信息，请参阅 Amazon ECR 文档中的推送 Docker 映像。	系统管理员、开发人员

设置 Amazon ECS 集群

Task	描述	所需技能
创建 Amazon ECS 集群。	按照 Amazon ECS 文档中创建集群的仅联网模板部分中的说明创建 Amazon ECS 集群。 **Note** 请务必选择创建 VPC 来为您的 Amazon ECS 集群创建新的虚拟私有云 (VPC)。	系统管理员、开发人员

设置 Amazon ECS 任务

Task	描述	所需技能
设置 Amazon ECS 任务执行 IAM 角色。	使用 `AmazonECSTaskExecutionRolePolicy` 托管策略创建 Amazon ECS 任务执行 IAM 角色。有关此内容的更多信息，请参阅 Amazon ECS 文档中的 Amazon ECS 任务执行 IAM 角色。 **Note** 请务必记录该 IAM 角色的亚马逊资源名称 (ARN)。	系统管理员、开发人员
将 IAM policy 附加到 Amazon ECS 任务执行 IAM 角色。	1. 使用 `firelens_policy.json` （附加）策略文档创建 IAM policy。有关更多信息，请参阅 IAM 文档中的在 JSON 选项卡上创建策略。 2. 将此策略附加到您之前创建的 Amazon ECS 任务执行 IAM 角色。有关此内容的更多信息，请参阅 IAM 文档中的添加 IAM policy (AWS CLI)。	系统管理员、开发人员
设置 Amazon ECS 任务定义。	1. 更新 `Task.json` 示例任务定义(附加)中的以下部分： • 使用任务执行 IAM 角色的 ARN 更新	系统管理员、开发人员

Task	描述	所需技能
	`executionRoleArn` 和 `taskRoleArn` • 使用您之前创建的自定义 Fluent Bit Docker 映像更新 `containerDefinitions` 中的映像 • 使用您的应用程序映像名称更新 `containerDefinitions` 中的映像 2. 打开 Amazon ECS 控制台，选择任务定义，选择创建新任务定义，然后在选择兼容性页面上选择 Fargate。 3. 选择通过 Json 配置，将更新的 `Task.json` 文件粘贴到文本区域，然后选择保存。 4. 创建任务定义 有关此内容的更多信息，请参阅 Amazon ECS 文档中的创建任务定义。	

运行 Amazon ECS 任务

Task	描述	所需技能
运行 Amazon ECS 任务。	在 Amazon ECS 控制台上，选择集群，选择您之前创建的集群，然后运行独立任务。	系统管理员、开发人员

Task	描述	所需技能
	有关此内容的更多信息，请参阅 Amazon ECS 文档中的运行独立任务。	

验证日 CloudWatch 志

Task	描述	所需技能
验证日志。	1. 打开 CloudWatch 控制台，选择“日志组”，然后选择/aws/ecs/containerinsights/{{cluster_ARN}}/firelens/application 。 2. 验证日志，特别是自定义解析器添加的自定义字段。 3. 用于 CloudWatch 根据自定义字段筛选日志。	系统管理员、开发人员

相关资源

- Amazon ECS 的 Docker 基本信息
- AWS Fargate 上的 Amazon ECS
- 配置基本服务参数

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用和 P HashiCorp acker 创建管道 CodePipeline 和 AMI

由 Akash Kumar (AWS) 创建

摘要

注意： AWS CodeCommit 已不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

此模式提供了使用 AWS 在亚马逊网络服务 (AWS) 云中创建管道 CodePipeline 和使用 HashiCorp Packer 创建亚马逊系统映像 (AMI) 的代码示例和步骤。该模式基于持续集成实践，该实践使用基于 Git 的版本控制系统自动构建和测试代码。在此模式中，您可以使用 AWS 创建和克隆代码存储库 CodeCommit。然后，使用 AWS 创建项目并配置您的源代码 CodeBuild。最后，创建一个提交到您的存储库的 AMI。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 用于启动亚马逊弹性计算云 (亚马逊 EC2) 实例的亚马逊 Linux AMI
- HashiCorp Packer 0.12.3 或更高版本
- 亚马逊 CloudWatch 活动（可选）
- Amazon CloudWatch 日志（可选）

架构

下图显示了使用此模式的架构自动创建 AMI 的应用程序代码示例。

图表显示了以下工作流：

1. 开发者将代码更改提交到私有 CodeCommit Git 存储库。然后， CodePipeline 使用启动构建，并将准备部署 CodeBuild 到亚马逊简单存储服务 (Amazon S3) 存储桶的新项目添加到亚马逊简单存储服务 (Amazon S3) 存储桶。
2. CodeBuild 使用 Packer 根据 JSON 模板捆绑和打包 AMI。如果启用，Ev CloudWatch ents 可以在源代码发生更改时自动启动管道。

技术堆栈

- CodeBuild
- CodeCommit
- CodePipeline
- CloudWatch 活动（可选）

工具

- AWS CodeBuild — AWS CodeBuild 是一项完全托管的云端构建服务。 CodeBuild 编译您的源代码，运行单元测试，并生成随时可以部署的工件。
- AWS CodeCommit — AWS CodeCommit 是一项版本控制服务，可让您在 AWS 云中私下存储和管理 Git 存储库。 CodeCommit 您无需管理自己的源代码控制系统或担心扩展其基础架构。
- AWS CodePipeline — AWS CodePipeline 是一项持续交付服务，您可以使用它来建模、可视化和自动执行发布软件所需的步骤。
- HashiCorp Packer — HashiCorp Packer 是一款开源工具，用于自动从单一来源配置创建相同的机器映像。Packer 为轻量级，可在所有主要操作系统上运行，并可并行为多个平台创建机器映像。

代码

此模式包括以下附件：

- `buildspec.yml`— 此文件 CodeBuild 用于构建和创建用于部署的对象。
- `amazon-linux_packer-template.json` — 此文件使用 Packer 创建 Amazon Linux AMI。

操作说明

设置代码存储库

Task	描述	所需技能
创建存储库。	创建 CodeCommit 存储库。	AWS 系统管理员
克隆存储库。	通过克隆 CodeCommit 存储库来连接存储库。	应用程序开发人员

Task	描述	所需技能
将源代码推送至远程存储库。	1. 创建提交，将 `buildspec.yml` 和 `amazon-linux_packer-template.json` 文件添加至本地存储库。 2. 将提交从本地存储库推送到远程 CodeCommit 存储库。	应用程序开发人员

为应用程序创建 CodeBuild 项目

Task	描述	所需技能
创建构建项目。	1. 登录 AWS 管理控制台，打开 AWS CodeBuild 控制台，然后选择创建构建项目。 2. 对于项目名称，请输入项目名称。 3. 对于源提供商，请选择 AWS CodeCommit。 4. 对于存储库，请选择要在其中构建代码管道的存储库。 5. 对于环境映像，请选择托管映像或自定义映像。 6. 对于操作系统，请选择 Ubuntu。 7. 对于 RunTime(s)，选择 “标准”。 8. 对于 “图像”，选择:4.0 aws/codebuild/standard。	应用程序开发人员、AWS 系统管理员

Task	描述	所需技能
	9. 对于映像版本，请选择始终对此运行时版本使用最新映像。 10对于环境类型，请选择 Linux。 11选择特权复选框。 12对于服务角色，请选择新服务角色或现有服务角色。 13对于构建规范，请选择使用构建规范文件或插入构建命令。 14.（可选）对于构件部分中的类型，请选择无构件。 15.（推荐）要将生成输出日志上传到日 CloudWatch 志，请选择CloudWatch 日志。 16.（可选）若要将构建输出日志上传到 Amazon S3，请选择 S3 日志复选框。 17选择创建构建项目。	

设置管道

Task	描述	所需技能
管道名称	1. 登录 AWS 管理控制台，打开 AWS CodePipeline 控制台，然后选择创建管道。 2. 对于管道名称，请输入管道名称。 3. 对于服务角色，请选择新服务角色或现有服务角色。	应用程序开发人员、AWS 系统管理员

Task	描述	所需技能
	4. 对于角色名称，请为您的角色输入一个名称。 5. 在高级设置部分中，对于构件存储，如果您希望 Amazon S3 创建桶并将构件存储在桶中，请选择默认位置。对于存储位置，请选择使用现有 S3 存储桶。选择下一步。 6. 对于源提供商，请选择 AWS CodeCommit。 7. 对于存储库名称，请选择您之前克隆的存储库。对于分支名称，请选择您的源代码分支。 8. 对于变更检测选项，请选择 Amazon Ev CloudWatch ents（推荐）启动管道 CodePipeline，或选择 AWS 定期检查更改。选择下一步。 9. 对于构建提供商，请选择 AWS CodeBuild。 10.在 “项目名称” 中，选择您在为应用程序创建 CodeBuild 项目长篇故事中创建的构建项目。 11选择您的构建选项，然后选择下一步。 12选择跳过部署阶段。 13选择创建管道。	

相关资源

- 在 AWS 中使用存储库 CodeCommit
- 使用构建项目
- 使用中的管道 CodePipeline

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用创建管道并将项目更新部署到本地 EC2 实例 CodePipeline

由 Akash Kumar (AWS) 和 Sandeep Reddy Jogammagari (AWS) 创作

摘要

此模式提供了代码示例和步骤，用于在亚马逊网络服务 (AWS) 云中创建管道并将更新的项目部署到 AWS 中的本地亚马逊弹性计算云 (Amazon EC2) 实例 CodePipeline。该模式基于持续整合实践。这种做法使用基于 Git 的版本控制系统自动生成与测试代码。在此模式中，您可以使用 AWS 创建和克隆代码存储库 CodeCommit。然后，您可以使用 AWS 创建项目并配置源代码 CodeBuild。最后，您可以使用 AWS 创建应用程序并为本地 EC2 实例配置其目标环境 CodeDeploy。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 用户定义的标签，用于在部署期间识别 EC2 实例
- CodeDeploy 代理，安装在 EC2 实例上
- 您需要的运行时软件，安装在 EC2 实例上
- 适用于 Java Development Kit 的 Amazon Corretto 8
- Apache Tomcat Web 服务器，已安装
- 亚马逊 CloudWatch 活动（可选）
- 用于登录 Web Server 的密钥对（可选）
- 用于 Web 应用程序的 Apache Maven 应用程序项目

架构

下图显示了一个使用此模式的架构部署到本地 EC2 实例的 Java Web 应用程序示例。

图表显示了以下工作流：

1. 开发者将代码更改提交到私有 CodeCommit Git 存储库。
2. CodePipeline 用于启动构建并添加准备部署 CodeBuild 到亚马逊简单存储服务 (Amazon S3) Simple Service 存储桶中的新项目。

3. CodePipeline 使用 CodeDeploy 代理预安装部署对象更改所需的所有依赖项。
4. CodePipeline 使用 CodeDeploy 代理将 S3 存储桶中的项目部署到目标 EC2 实例。如果启用，Ev CloudWatch ents 可以在源代码发生更改时自动启动管道。

技术堆栈

- CodeBuild
- CodeCommit
- CodeDeploy
- CodePipeline
- CloudWatch 活动（可选）

工具

- AWS CodeBuild 是一项完全托管的构建服务，可帮助您编译源代码、运行单元测试和生成可随时部署的项目。 CodeBuild 编译您的源代码、运行单元测试并生成可随时部署的工件。
- AWS CodeCommit 是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。
- AWS CodeDeploy 可自动部署到亚马逊弹性计算云 (Amazon EC2) 或本地实例、AWS Lambda 函数或亚马逊弹性容器服务 (Amazon ECS) 服务。
- AWS CodePipeline 可帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件变更所需的步骤。

代码

此模式包括以下附件：

- `buildspec.yml`— 此文件指定了构建和创建用于部署的对象 CodeBuild 所需的操作。
- `appspec.yml`— 此文件指定了为本地 EC2 实例创建应用程序和配置目标环境 CodeDeploy 所需的操作。
- `install_dependencies.sh` — 此文件为 Apache Tomcat Web 服务器安装依赖项。
- `start_server.sh` — 此文件启动 Apache Tomcat 网络服务器。
- `stop_server.sh` — 此文件会停止 Apache Tomcat Web 服务器。

操作说明

设置代码存储库

Task	描述	所需技能
创建存储库。	创建 CodeCommit 存储库。	AWS 系统管理员
克隆存储库。	通过克隆 CodeCommit 存储库来连接存储库。	应用程序开发人员
将源代码推送至远程存储库。	1. 创建提交，将 `buildspec.yml` 和 `appspec.yml` 文件添加至本地存储库。 2. 将提交从本地存储库推送到远程 CodeCommit 存储库。	应用程序开发人员

为应用程序创建 CodeBuild 项目

Task	描述	所需技能
创建构建项目。	1. 登录 AWS 管理控制台，打开 AWS CodeBuild 控制台，然后选择创建构建项目。 2. 对于项目名称，请输入项目名称。 3. 对于源提供商，请选择 AWS CodeCommit。 4. 对于存储库，请选择要在其中构建代码管道的存储库。 5. 在环境映像中，选择托管映像或自定义映像。 6. 在操作系统中，选择 Amazon Linux 2。	AWS 管理员、应用程序开发人员

Task	描述	所需技能
	Note 亚马逊 Linux 2 的支持已接近终止。欲了解更多信息，请参阅亚马逊 Linux 2 FAQs。 7. 对于 RunTime(s)，选择 “标准”。 8. 对于图像，请选择 aws/codebuild/amazonlinux2-aarch64 standards: 2.0。 9. 对于映像版本，请选择始终对此运行时版本使用最新映像。 10. 在服务角色中，选择新的服务角色或现有的服务角色。 11. 在构建规范中，选择使用构建规范文件或插入构建命令。 12. （可选）选择添加构件以配置构件。 13. （可选）要将构建输出日志上传到 Amazon CloudWatch，请选择CloudWatch 日志。 14 选择 Create build project（创建构建项目）。	

为本地 EC2 实例配置构件部署

Task	描述	所需技能
创建应用程序。	1. 登录 AWS 管理控制台，打开 AWS CodeDeploy 控制台，然后选择创建应用程序。 2. 在应用程序名称中，输入您的应用程序名称。 3. 对于计算平台，请选择 EC2/本地。 4. 选择创建应用程序，然后选择创建部署组。 5. 在部署组名称，输入一个名称。 6. **Note** 为创建服务角色 CodeDeploy。：服务角色必须具有权限才能授予对目标环境的 CodeDeploy 访问权限。 7. 在服务角色中，选择您在第 6 步中选择服务角色。 8. 对于部署类型，请根据您的业务要求选择就地或蓝色/绿色 9. 在环境配置，选择满足您业务需求的选项。 10. （可选）在 Amazon EC2 控制台中单独为负载均衡器创建目标组，然后返回 AWS	AWS 系统管理员、应用程序开发人员

Task	描述	所需技能
	CodeDeploy 控制台的创建部署组页面选择您的负载均衡器和目标组。 11选择 Create deployment group（创建部署组）。	

设置管道

Task	描述	所需技能
创建管道。	1. 登录 AWS 管理控制台，打开 AWS CodePipeline 控制台，然后选择创建管道。 2. 对于管道名称，请输入管道名称。 3. 对于服务角色，请选择新服务角色或现有服务角色。 4. 对于角色名称，请为您的角色输入一个名称。 5. 在高级设置部分中，对于构件存储，如果您希望 Amazon S3 创建桶并将构件存储在桶中，请选择默认位置。对于存储位置，请选择使用现有 S3 存储桶。选择下一步。 6. 对于源提供商，请选择 AWS CodeCommit。 7. 对于存储库名称，请选择您之前克隆的存储库。对于分支名称，请选择您的源代码分支。	AWS 系统管理员、应用程序开发人员

Task	描述	所需技能
	8. 对于变更检测选项，请选择 Amazon E CloudWatch vents（推荐）或 AWS CodePipeline。选择下一步。 9. 对于构建提供商，请选择 AWS CodeBuild。 10.在 “项目名称” 中，选择您在此模式的 “为应用程序创建 CodeBuild 项目” 部分中创建的生成项目。 11选择您的构建选项，然后选择下一步。 12对于部署提供商，请选择 AWS CodeDeploy。 13选择应用程序名称和部署组，然后选择下一步。 14选择 Create pipeline（创建管道）。	

相关资源

- 在 AWS 中使用存储库 CodeCommit
- 使用构建项目
- 在中使用应用程序 CodeDeploy
- 使用中的管道 CodePipeline

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

自动为 Java 和 Python 项目创建动态 CI 管道

由 Aromal Raj Jayarajan (AWS)、Amarnath Reddy (AWS)、MAHESH RAGHUNANDANAN (AWS) 和 Vijesh Vijayakumaran Nair (AWS) 创建

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

此模式说明如何使用 AWS 开发人员工具自动为 Java 和 Python 项目创建动态持续集成（CI）管道。

随着技术堆栈的多样化和开发活动的增加，创建和维护在整个组织中保持一致的 CI 管道可能会变得困难。通过在 AWS Step Functions 中自动执行该流程，您可以确保 CI 管道的使用和方法保持一致。

为了自动创建动态 CI 管道，此模式使用以下变量输入：

- 编程语言（仅限 Java 或 Python）
- 管道名称
- 所需管道阶段

> **Note**
>
> Step Functions 使用多个 AWS 服务来协调管道的创建。有关此解决方案中使用的 Amazon Web Services 的更多信息，请参阅此模式的工具部分。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 部署此解决方案的同一 AWS 区域 中的 Amazon S3 存储桶
- 一个 AWS Identity and Access Management (IAM) 委托人，该委托人拥有创建此解决方案所需资源所需的 AWS CloudFormation 权限

限制

- 此模式仅支持 Java 和 Python 项目。

- 在此模式中预置的 IAM 角色遵循最低权限原则。必须根据您的 CI 管道需要创建的特定资源更新 IAM 角色的权限。

架构

目标技术堆栈

- AWS CloudFormation
- AWS CodeBuild
- AWS CodeCommit
- AWS CodePipeline
- IAM
- Amazon Simple Storage Service（Amazon S3）
- AWS Systems Manager
- AWS Step Functions
- AWS Lambda
- Amazon DynamoDB

目标架构

下图显示了使用 AWS 开发人员工具自动为 Java 和 Python 项目创建动态 CI 管道的示例工作流程。

图表显示了以下工作流：

1. AWS 用户以 JSON 格式提供用于创建 CI 管道的输入参数。此输入将启动一个 Step Functions 工作流（状态机），该工作流使用 AWS 开发人员工具创建 CI 管道。
2. Lambda 函数读取名为 input-reference 的文件夹，该文件夹存储在 Amazon S3 存储桶中，然后生成 buildspec.yml 文件。此生成的文件定义了 CI 管道阶段，并存储回存储参数引用的同一 Amazon S3 存储桶中。
3. Step Functions 会检查 CI 管道创建工作流程的依赖关系是否存在任何更改，并根据需要更新依赖关系堆栈。
4. Step Functions 在 CloudFormation 堆栈中创建 CI 管道资源，包括 CodeCommit 存储库、CodeBuild 项目和 CodePipeline 管道。

5. CloudFormation 堆栈将所选技术堆栈（Java 或 Python）的示例源代码和 buildspec.yml 文件复制到存储库中。 CodeCommit
6. CI 管道运行时详细信息存储在 DynamoDB 表中。

自动化和扩缩

- 此模式仅供在单个开发环境中使用。需要更改配置才能跨多个开发环境使用。
- 要添加对多个 CloudFormation 堆栈的支持，您可以创建其他 CloudFormation 模板。有关更多信息，请参阅 CloudFormation 文档 CloudFormation中的 AWS 入门。

工具

工具

- AWS Step Functions 是一项无服务器编排服务，可让您搭配使用 AWS Lambda 函数和其他 Amazon Web Services 来构建业务关键型应用程序。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS CodeBuild 是一项完全托管的构建服务，可帮助您编译源代码、运行单元测试和生成可随时部署的项目。
- AWS CodeCommit 是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。
- AWS CodePipeline 可帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件变更所需的步骤。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥，以帮助保护您的数据。
- Amazon Simple Storage Service(Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。
- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- AWS Systems Manager Parameter Store 提供安全的分层存储，用于配置数据管理和密钥管理。

代码

此模式的代码可在 GitHub automated-ci-pipeline-creation存储库中找到。存储库包含创建此模式中概述的目标架构所需的 CloudFormation 模板。

最佳实践

- 请勿在 CloudFormation 模板或 Step Functions 操作配置中直接输入令牌或密码等凭据（机密）。如果这样做，该信息将显示在 DynamoDB 日志中。相反，请使用 AWS Secrets Manager 来设置和存储密钥。然后，根据需要在 CloudFormation 模板和 Step Functions 操作配置中引用 Secrets Manager 中存储的密钥。有关更多信息，请参阅 Secrets Manager 文档中的什么是 AWS Secrets Manager。
- 为存储在 Amazon S3 中的 CodePipeline 项目配置服务器端加密。有关更多信息，请参阅 CodePipeline 文档中的为存储在 Amazon S3 中的项目配置服务器端加密。 CodePipeline
- 在配置 IAM 角色时应用最低权限。有关更多信息，请参阅 IAM 文档中的应用最低权限许可。
- 确保您的 Amazon S3 存储桶不可公开访问。有关更多信息，请参阅 Amazon S3 文档中的为您的 S3 存储桶配置阻止公有访问设置。
- 确保您为 Amazon S3 存储桶激活版本控制。有关更多信息，请参阅 Amazon S3 文档中的在 S3 存储桶中使用版本控制。
- 配置 IAM policy 时使用 IAM Access Analyzer。该工具提供可操作的建议，以帮助您编写安全且实用的 IAM policy。有关更多信息，请参阅 IAM 文档中的使用 AWS Identity and Access Management Access Analyzer。
- 如果可能，请在配置 IAM policy 时定义特定的访问条件。
- 激活 Amazon CloudWatch 日志以进行监控和审计。有关更多信息，请参阅什么是 Amazon CloudWatch 日志？ 在 CloudWatch 文档中。

操作说明

配置先决条件

Task	描述	所需技能
创建 Amazon S3 存储桶。	创建 Amazon S3 存储桶（或使用现有存储桶）来存储解决方案所需的 CloudFormation 模板、源代码和输入文件。	AWS DevOps

Task	描述	所需技能
	有关更多信息，请参阅 Amazon S3 文档中的步骤 1：创建您的第一个 S3 存储桶。 **Note** Amazon S3 存储桶必须位于您要将解决方案部署到的同一 AWS 区域中。	
克隆 GitHub 存储库。	在终端窗口中运行以下命令来克隆 GitHub automated-ci-pipeline-creation存储库： `git clone https://github.com/aws-samples/automated-ci-pipeline-creation.git` 有关更多信息，请参阅 GitHub 文档中的克隆存储库。	AWS DevOps

Task	描述	所需技能
将解决方案模板文件夹从克隆的 GitHub 存储库上传到您的 Amazon S3 存储桶。	复制克隆的 Solution-Templates 文件夹中的内容，并将其上传到您创建的 Amazon S3 存储桶中。 有关更多信息，请参阅 Amazon S3 文档中的上传数据。 **Note** 确保仅上传解决方案模板文件夹的内容。您只能在 Amazon S3 存储桶的根级别上传文件。	AWS DevOps

部署解决方案

Task	描述	所需技能
使用克隆存储库中的 template.yml 文件创建 CloudFormation 堆栈来部署解决方案。 GitHub	1. 登录 AWS 管理控制台，然后打开 AWS CloudFormation 控制台。 2. 选择创建堆栈。此时将显示一个下拉列表。 3. 在下拉列表中，选择使用新资源（标准）。创建堆栈页面将打开。 4. 在指定模板部分，选择上传模板文件旁边的复选框。 5. 选择 Choose file (选择文件)。然后，导航到克隆 GitHub 存储库的根文件夹，	AWS 管理员，AWS DevOps

Task	描述	所需技能
	然后选择 t emplate .yml 文件。然后选择 Open (打开)。 6. 选择下一步。指定堆栈详细信息页面将打开。 7. 在参数部分中，指定以下参数： • 对于 S3 TemplateBucketName，请输入您之前创建的 Amazon S3 存储桶的名称，其中包含此解决方案的源代码和参考资料。确保存储桶名称参数为小写。 • 对于 Dynamo DBTable，请输入堆栈创建的 DynamoDB 表的名称。CloudFormation • 对于 StateMachineName，输入 CloudFormation 堆栈创建的 Step Functions 状态机的名称。 8. 选择下一步。配置堆栈选项页面将打开。 9. 在配置堆栈选项页面上，请选择下一步。请勿更改任何默认值。此时将打开查看页面。 10查看堆栈创建设置。然后，选择创建堆栈以启动您的堆栈。	

Task	描述	所需技能
	Note 创建堆栈时，堆栈会列在堆栈页面上，状态为 CREAT E_IN_PROG RESS。请确保等待堆栈的状态更改为 CREATE_CO MPLETE，然后再完成此模式中的其余步骤。	

测试设置

Task	描述	所需技能
运行您创建的步骤函数。	1. 登录 AWS 管理控制台，然后打开 Step Functions 控制台。 2. 打开您创建的步骤函数。 3. 选择启动执行。然后，以 JSON 格式输入工作流的输入值（请参阅以下示例输入）。 4. 选择启动执行。 JSON 格式 `{ "details": { "tech_stack": "Name of the Tech Stack (python/java)", "project_name": "Name of the Project`	AWS 管理员，AWS DevOps

Task	描述	所需技能

```
 that you want to create
 with",
    "pre_build":
 "Choose the step if
 it required in the
 buildspec.yml file
 i.e., yes/no",
    "build": "Choose the
 step if it required in
 the buildspec.yml file
 i.e., yes/no",
    "post_build":
 "Choose the step if
 it required in the
 buildspec.yml file
 i.e., yes/no",
    "reports": "Choose
 the step if it required
 in the buildspec.yml
 file i.e., yes/no",
  }
}
```

Java JSON 输入示例

```
{
  "details": {
    "tech_stack":
 "java",
    "project_name":
 "pipeline-java-pjt",
    "pre_build": "yes",
    "build": "yes",
    "post_build":
 "yes",
    "reports": "yes"
  }
}
```

Python JSON 输入示例

Task	描述	所需技能

	<pre>{
	  "details": {
	    "tech_stack":
	 "python",
	    "project_name":
	 "pipeline-python-p
	jt",
	    "pre_build": "yes",
	    "build": "yes",
	    "post_build":
	 "yes",
	    "reports": "yes"
	  }
	}</pre>	

确认 CI 管道的 CodeCommit 存储库已创建。	1. 登录 AWS 管理控制台，然后打开CodeCommit 控制台。 2. 在存储库页面上，验证您创建的 CodeCommit 存储库的名称是否出现在存储库列表中。存储库的名称后面附有以下内容：pipeline-java-pjt-Repo 3. 打开 CodeCommit 存储库并验证示例源代码以及 buildspec.yml 文件是否已推送到主分支。	AWS DevOps

Task	描述	所需技能
检查 CodeBuild 项目资源。	1. 登录 AWS 管理控制台，然后打开CodeBuild 控制台。 2. 在“生成项目”页面上，验证您创建的 CodeBuild 项目的名称是否出现在项目列表中。项目名称后面附有以下内容：pipeline-java-pjt-Build 3. 选择您的 CodeBuild 项目名称以打开该项目。然后，查看并验证以下配置： • 项目配置 • 源 • 环境 • BuildSpec • 批量配置 • 构件	AWS DevOps
验证各个 CodePipeline 阶段。	1. 登录 AWS 管理控制台，然后打开CodePipeline 控制台。 2. 在管道页上，验证您创建的管道的名称是否显示在管道列表中。管道名称后面附有以下内容：pipeline-java-pjt-Pipeline 3. 选择管道的名称以打开管道。然后，查看并验证管道的每个阶段，包括提交和部署。	AWS DevOps

Task	描述	所需技能
确认 CI 管道已成功运行。	1. 在CodePipeline 控制台的“管道” 页面上，选择您的管道名称以查看管道的状态。 2. 验证管道的每个阶段是否具有成功状态。	AWS DevOps

清除资源

Task	描述	所需技能
删除中的资源堆栈 CloudFormation。	在中删除 CI 管道的资源堆栈 CloudFormation。 有关更多信息，请参阅 CloudFormation 文档中的在 AWS CloudFormation 控制台上删除堆栈。 **Note** 请务必删除名为 -stack 的堆栈<project_name>。	AWS DevOps
在 Amazon S3 中删除 CI 管道的依赖关系，然后 CloudFormation。	1. 清空名为的 Amazon S3 存储桶DeploymentArtifactBucket。有关更多信息，请参阅 Amazon S3 文档中的清空存储桶。 2. 在中删除 CI 管道的依赖堆栈 CloudFormation。有关更多信息，请参阅 CloudFormation 文档中的在 AWS	AWS DevOps

Task	描述	所需技能
	CloudFormation 控制台上删除堆栈。 **Note** 请务必删除名为的堆栈pipeline-creation-dependencies-stack。	
删除 Amazon S3 模版存储桶。	删除您在此模式的配置先决条件部分中创建的 Amazon s3 存储桶，该存储桶存储此解决方案的模板。 有关更多信息，请参阅 Amazon S3 文档中的删除存储桶。	AWS DevOps

相关资源

- 创建使用 Lambda 的 Step Functions 状态机（A WS Step Functions 文档）
- AWS Step Funct WorkFlow ions Studio（AWS Step Functics 文档）
- DevOps 还有 AWS
- AWS 是如何 CloudFormation 运作的？（AWS CloudFormation 文档）
- 使用 AWS CodeDeploy、AWS、AW CodeCommit S 和 AW CodeBuild S 完成 CI/CD（AW CodePipeline S 博客文章）
- IAM 和 AWS STS 配额、名称要求和字符限制（IAM 文档）

使用 Terraf CloudWatch orm 部署 Synthetics 加那利群岛

创建者：Dhrubajyoti Mukherjee（AWS）和 Jean-Francois Landreau（AWS）

摘要

重要的是要从客户的角度验证系统的运行状况，并确认客户能够连接。当客户不经常调用端点时，这就更加困难了。Amaz CloudWatch on Synthetics 支持创建加那利群岛，它可以测试公共和私有终端节点。通过使用金丝雀，即使系统未在使用中，您也可以知道该系统的状态。这些金丝雀要么是 Node.js Puppeteer 脚本，要么是 Python Selenium 脚本。

此模式描述了如何使用 HashiCorp Terraform 部署用于测试私有端点的金丝雀。它嵌入了一个用于测试 URL 是否返回 `200-OK` 的 Puppeteer 脚本。然后，可以将 Terraform 脚本与部署私有端点的脚本集成。您也可以修改解决方案以监控公有端点。

先决条件和限制

先决条件

- 具有虚拟私有云（VPC）和私有子网的有效 Amazon Web Services（AWS）账户
- 可从私有子网访问的端点的 URL
- 部署环境中安装了 Terraform

限制

当前的解决方案适用于以下 S CloudWatch ynthetics 运行时版本：

- syn-nodejs-puppeteer-3.4
- syn-nodejs-puppeteer-3.5
- syn-nodejs-puppeteer-3.6
- syn-nodejs-puppeteer-3.7

随着新的运行时系统版本的发布，您可能需要更新当前解决方案。您还需要修改解决方案以跟上安全更新的步伐。

产品版本

- Terraform 1.3.0

架构

Amaz CloudWatch on Synthetics 以 CloudWatch Lambda 和亚马逊简单存储服务 (Amazon S3) 为基础。亚马逊 CloudWatch 提供了一个用于创建加那利群岛的向导和一个显示金丝雀运行状态的控制面板。Lambda 函数运行脚本。Amazon S3 存储了金丝雀运行的日志和屏幕截图。

此模式通过部署在目标子网中的亚马逊弹性计算云 (Amazon EC2) 实例模拟私有终端节点。Lambda 函数需要在部署私有端点的 VPC 中使用弹性网络接口。

此图显示以下内容：

1. Synthetics 金丝雀启动金丝雀 Lambda 函数。
2. 金丝雀 Lambda 函数连接到弹性网络接口。
3. 金丝雀 Lambda 函数监控端点的状态。
4. Synthetics 金丝雀将运行数据推送到 S3 存储桶和 CloudWatch 指标。
5. 根据指标启动 CloudWatch 警报。
6. CloudWatch 警报会启动亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 主题。

工具

Amazon Web Services

- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和在 AWS 上运行的应用程序的指标。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。此模式使用 VPC 端点和弹性网络接口。

其他服务

- HashiCorp Terraform 是一种开源基础设施即代码 (IaC) 工具，可帮助您使用代码来配置和管理云基础架构和资源。这种模式使用 Terraform 来部署基础设施。
- Puppeteer 是一个 Node.js 库。Synt CloudWatch hetics 运行时使用 Puppeteer 框架。

代码

该解决方案可在 GitHub 云 watch-synthetics-canary-terraform存储库中找到。有关更多信息，请参阅其他信息部分。

操作说明

实施监控私有 URL 的解决方案

Task	描述	所需技能
收集监控私有 URL 的要求。	收集完整的 URL 定义：域名、参数和标头。要与 Amazon S3 和亚马逊进行私密通信 CloudWatch，请使用 VPC 终端节点。请注意端点如何访问 VPC 和子网。考虑一下金丝雀运行的频率。	云架构师、网络管理员
修改现有解决方案以监控私有 URL。	修改 `terraform.tfvars` 文件： • `name` – 金丝雀的名称。 • `runtime_version` – 金丝雀的运行时系统版本。我们建议使用 syn-nodejs-puppeteer -3.7。 • `take_screenshot` – 是否应该截屏。 • `api_hostname` – 受监控的端点的主机名。 • `api_path` – 受监控的端点的路径。	云架构师

Task	描述	所需技能
	• `vpc_id` – 金丝雀 Lambda 函数使用的 VPC ID。 • `subnet_ids` — 金丝雀 Lambda 函数使用的子网 IDs 。 • `frequency` – 金丝雀的运行频率（以分钟为单位）。 • `alert_sns_topic` — CloudWatch 警报通知发送到的 SNS 主题。	

部署和操作解决方案。	要部署该解决方案，请执行以下操作：
	1. 在开发环境的 `cloudwatch-synthetics-canary-terraform` 目录中，初始化 Terraform。
	`terraform init`
	2. 计划并查看更改。
	`terraform plan`
	3. 部署解决方案。
	`terraform apply`
		云架构师、 DevOps 工程师

故障排除

事务	解决方案
删除预调配的资源会导致卡顿。	按顺序手动删除金丝雀 Lambda 函数、相应的弹性网络接口和安全组。

相关资源

- 使用 Synthetics 监控
- 使用 Amazon S CloudWatch ynthetics 监控 API Gateway 终端节点（博客文章）

其他信息

存储库构件

存储库构件采用如下结构。

```
.
### README.md
### main.tf
### modules
#   ### canary
#   ### canary-infra
### terraform.tfvars
### tf.plan
### variable.tf
```

该 `main.tf` 文件包含核心模块，它部署了两个子模块：

- `canary-infra` 部署了金丝雀所需基础设施。
- `canary` 部署了金丝雀。

解决方案的输入参数位于 `terraform.tfvars` 文件中。您可以使用以下代码示例创建一个金丝雀。

```
module "canary" {
    source = "./modules/canary"
    name   = var.name
```

```
    runtime_version = var.runtime_version
    take_screenshot = var.take_screenshot
    api_hostname = var.api_hostname
    api_path = var.api_path
    reports-bucket = module.canary_infra.reports-bucket
    role = module.canary_infra.role
    security_group_id = module.canary_infra.security_group_id
    subnet_ids = var.subnet_ids
    frequency = var.frequency
    alert_sns_topic = var.alert_sns_topic
}
```

相应的 .var 文件如下。

```
name   = "my-canary"
runtime_version = "syn-nodejs-puppeteer-3.7"
take_screenshot = false
api_hostname = "mydomain.internal"
api_path = "/path?param=value"
vpc_id = "vpc_id"
subnet_ids = ["subnet_id1"]
frequency = 5
alert_sns_topic = "arn:aws:sns:eu-central-1:111111111111:yyyyy"
```

清理解决方案

如果您在开发环境中对此进行测试，则可以清理解决方案以避免累积成本。

1. 在 AWS 管理控制台 上，导航到 Amazon S3 控制台。清空解决方案创建的 Amazon S3 存储桶。如果需要，请务必备份数据。
2. 在开发环境中，从 `cloudwatch-synthetics-canary-terraform` 目录中运行该 `destroy` 命令。

```
terraform destroy
```

在 Amazon ECS 上部署 Java 微服务 CI/CD 管道

由 Vijay Thompson (AWS) 和 Sankar Sangubotla (AWS) 创建

摘要

此模式将指导您完成部署持续集成和持续交付的步骤（CI/CD) pipeline for Java microservices on an existing Amazon Elastic Container Service (Amazon ECS) cluster by using AWS CodeBuild. When the developer commits the changes, the CI/CD管道已启动并开始构建过程） CodeBuild。 构建完成后，项目会被推送到亚马逊弹性容器注册表 (Amazon ECR) Container Registry，然后从亚马逊 ECR 中提取最新版本并推送到亚马逊 ECS 服务。

先决条件和限制

先决条件

- 在 Amazon ECS 上运行的现有 Java 微服务应用程序
- 熟悉 AWS CodeBuild 和 AWS CodePipeline

架构

源技术堆栈

- 在 Amazon ECS 上运行的 Java 微服务
- Amazon ECR 中的代码存储库
- AWS Fargate

源架构

目标技术堆栈

- Amazon ECR
- Amazon ECS
- AWS Fargate
- AWS CodePipeline

- AWS CodeBuild

目标架构

自动化和扩缩

CodeBuild `buildspec.yml` 文件：

```
version: 0.2

phases:
  pre_build:
    commands:
      - echo Logging in to Amazon ECR...
      - aws --version
      - $(aws ecr get-login --region $AWS_DEFAULT_REGION --no-include-email)
      - REPOSITORY_URI=$AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/
$IMAGE_REPO
      - COMMIT_HASH=$(echo $CODEBUILD_RESOLVED_SOURCE_VERSION | cut -c 1-7)
      - IMAGE_TAG=build-$(echo $CODEBUILD_BUILD_ID | awk -F":" '{print $2}')
  build:
    commands:
      - echo Build started on `date`
      - echo building the Jar file
      - mvn clean install
      - echo Building the Docker image...
      - docker build -t $REPOSITORY_URI:$BUILD_TAG .
      - docker tag $REPOSITORY_URI:$BUILD_TAG $REPOSITORY_URI:$IMAGE_TAG
  post_build:
    commands:
      - echo Build completed on `date`
      - echo Pushing the Docker images...
      - docker push $REPOSITORY_URI:$BUILD_TAG
      - docker push $REPOSITORY_URI:$IMAGE_TAG
      - echo Writing image definitions file...
      - printf '[{"name":"%s","imageUri":"%s"}]' $DOCKER_CONTAINER_NAME
 $REPOSITORY_URI:$IMAGE_TAG > imagedefinitions.json
      - cat imagedefinitions.json
artifacts:
    files:
     - imagedefinitions.json
```

```
    - target/DockerDemo.jar
```

工具

Amazon Web Services

- AWS CodeBuild 是一项完全托管的构建服务，可帮助您编译源代码、运行单元测试和生成可随时部署的项目。AWS 会持续 CodeBuild 扩展并同时处理多个构建，因此您的构建不会留在队列中。
- AWS CodePipeline 可帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件变更所需的步骤。您可以将 AWS CodePipeline 与第三方服务（例如）集成 GitHub，也可以使用 AWS 服务（例如 Amazon ECR）。
- Amazon Elastic Container Registry (Amazon ECR) 是一个完全托管的 Docker 容器注册表，可让开发人员轻松地存储、管理和部署 Docker 容器映像。Amazon ECR 已与 Amazon ECS 集成，可简化您的 development-to-production工作流程。Amazon ECR 在一个可用性和可扩展性都非常高的架构中托管容器映像，从而安全可靠地为应用程序部署容器。与 AWS Identity and Access Management (IAM) 集成，可实现对每个存储库的资源级控制。
- Amazon Elastic Container Service (Amazon ECS) 是一项可扩展性高的高性能容器编排服务，支持 Docker 容器，允许您在 AWS 上轻松地运行和扩展容器化应用程序。Amazon ECS 使您无需安装和操作自己的容器编排软件、管理和扩展虚拟机集群，也不需要在这些虚拟机上调度容器。
- AWS Fargate 是一款适用于 Amazon ECS 的计算引擎，可允许您运行容器，无需管理服务器或集群。使用 AWS Fargate，您不必再预调配、配置和扩展虚拟机集群即可运行容器。这样一来，您就无需再选择服务器类型、确定扩展集群的时间和优化集群打包。

其他工具

- Docker 平台允许您在名为容器的软件包中构建、测试和交付应用程序。
- Git 是分布式版本控制系统，用于追踪软件开发期间源代码的更改。其专为协调程序员之间的工作而设计，但亦可用于追踪任何一组文件中的更改。其目标包括速度、数据完整性，以及对分布式非线性工作流程的支持。

操作说明

在 AWS 中设置构建项目 CodeBuild

Task	描述	所需技能
创建 CodeBuild 构建项目。	在 AWS CodeBuild 控制台中，创建一个构建项目并指定其名称。	应用程序开发人员、AWS 系统管理员
选择源。	此模式使用 Git 作为代码存储库，因此请GitHub 从可用选项列表中进行选择。选择公共存储库或从您的 GitHub 账户中选择。	应用程序开发人员、AWS 系统管理员
选择存储库。	选择构建代码的存储库。	应用程序开发人员、AWS 系统管理员
选择环境。	您可以从托管映像列表中进行选择，还可以使用 Docker 选择自定义映像。此模式使用了以下托管映像： • **Note** 亚马逊 Linux 2 (: 亚马逊 Linux 2) 的支持已接近终止。 欲了解更多信息，请参阅亚马逊 Linux 2 FAQs.) • Runtime（运行时）：Standard（标准） • Image 版本 1.0	应用程序开发人员、AWS 系统管理员

Task	描述	所需技能
选择服务角色。	您可以创建服务角色，或从现有服务角色列表中选择服务角色。	应用程序开发人员、AWS 系统管理员
设置环境变量。	在其他配置部分中，配置以下环境变量： • AWS_DEFAULT_REGION 表示默认 AWS 区域 • AWS_ACCOUNT用户账 @@ 号的 _ID • Amazon ECR 私有存储库的 IMAGE_REPO • 构建版本的 BUILD_TAG （最新构建版本是此变量的值） • DOCKER_CONTAINER_NAME 表示任务中容器的名称 这些变量是 `buildspec.yml` 文件中的占位符，将替换为各自的值。	应用程序开发人员、AWS 系统管理员
创建 buildspec 文件。	您可以在与 `pom.xml` 相同的位置创建 `buildspec.yml` 文件，并添加此模式所提供的配置，还可以使用在线 buildspec 编辑器添加配置。按提供的步骤为环境变量配置适当的值。	应用程序开发人员、AWS 系统管理员
为项目配置构件。	（可选）如果需要，则为构件配置构建项目。	应用程序开发人员、AWS 系统管理员

Task	描述	所需技能
配置 Amazon CloudWatch 日志。	（可选）如果需要，为构建项目配置 Amazon CloudWatch 日志。虽然此步骤是可选的，但我们建议您这样做。	应用程序开发人员、AWS 系统管理员
配置 Amazon S3 日志。	（可选）如果您希望存储构建项目的 Amazon Simple Storage Service (Amazon S3) 日志，则为构建项目配置此类日志。	应用程序开发人员、AWS 系统管理员

在 AWS 中配置管道 CodePipeline

Task	描述	所需技能
创建管道。	在 AWS CodePipeline 控制台上，创建管道并指定其名称。有关创建管道的更多信息，请参阅 AWS CodePipeline 文档。	应用程序开发人员、AWS 系统管理员
选择服务角色。	创建服务角色，或从现有服务角色列表中选择服务角色。如果要创建服务角色，请提供该角色的名称并选择用于 CodePipeline 创建该角色的选项。	应用程序开发人员、AWS 系统管理员
选择构件商店。	在高级设置中，如果希望 Amazon S3 创建存储桶并在其中存储构件，请使用构件存储的默认位置。或者，选择自定义位置并指定现有存储桶。您	应用程序开发人员、AWS 系统管理员

Task	描述	所需技能
	还可以选择使用加密密钥对构件加密。	
指定源提供程序。	对于源提供商，请选择 GitHub（版本 2）。	应用程序开发人员、AWS 系统管理员
选择代码的存储库与分支。	如果您尚未登录，请提供要连接的连接详细信息 GitHub，然后选择存储库名称和分支名称。	应用程序开发人员、AWS 系统管理员
更改检测选项。	选择源代码更改时启动管道，然后移至下一页。	应用程序开发人员、AWS 系统管理员
选择构建提供程序。	对于构建提供商，请选择 AWS CodeBuild，然后提供构建项目的 AWS 区域和项目名称详细信息。 对于构建类型，请选择单一版本。	应用程序开发人员、AWS 系统管理员
选择部署提供程序。	对于部署提供程序，请选择 Amazon ECS。选择集群名称、服务名称、映像定义文件（如有）和部署超时值（如需要）。选择创建管道。	应用程序开发人员、AWS 系统管理员

相关资源

- AWS ECS 文档
- AWS ECR 文档
- AWS CodeBuild 文档
- AWS CodePipeline 文档
- 以 Amazon ECR 为来源，为您的容器映像构建持续交付管道（博客文章）

部署 ChatOps 解决方案来管理 SAST 扫描结果，方法是在聊天应用程序中使用 Amazon Q Developer 自定义操作和 AWS CloudFormation

由 Anand Bukkapatnam Tirumala (AWS) 创作

摘要

这种模式提供了一个全面的解决方案，它在聊天应用程序中使用 Amazon Q Developer 来简化对报告的静态应用程序安全测试 (SAST) 扫描失败的 SonarQube管理。这种创新的方法将自定义操作和通知集成到对话界面中，从而在开发团队中实现高效的协作和决策流程。

在当今快节奏的软件开发环境中，高效管理 SAST 扫描结果对于维护代码质量和安全性至关重要。但是，许多组织面临着以下重大挑战：

- 由于通知系统效率低下，对关键漏洞的意识延迟
- 审批工作流程脱节导致决策过程缓慢
- 对 SAST 扫描失败缺乏即时、可行的响应
- 围绕安全调查结果的沟通和协作支离破碎
- 为安全工具手动设置基础架构既耗时又容易出错

这些问题通常会导致安全风险增加、发布延迟和团队工作效率降低。为了有效地应对这些挑战，需要一种能够简化 SAST 结果管理、增强团队协作和自动配置基础架构的解决方案。

该解决方案的主要功能包括：

- 自定义通知 — 实时警报和通知直接发送到团队聊天频道，确保及时意识到 SAST 扫描漏洞或失败并采取行动。
- 对话式批准 — 利益相关者可以在聊天界面中无缝启动和完成 SAST 扫描结果的批准工作流程，从而加快决策流程。
- 自定义操作 — 团队可以根据 SAST 扫描结果定义和执行自定义操作，例如在质量门失败时自动触发电子邮件，增强对安全问题的响应能力。
- 集中协作 — 所有与 SAST 扫描相关的讨论、决策和操作都保存在统一的聊天环境中，从而促进了团队成员之间的协作和知识共享。
- 基础设施即代码 (IaC) — 整个解决方案采用 AWS CloudFormation 模板封装，可实现更快、更可靠的基础架构配置，同时减少手动设置错误。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- 一个 AWS Identity and Access Management (IAM) 角色，有权创建和管理与工具中 AWS 服务 列出的资源相关联的资源。
- 一个 Slack 工作空间。
- 聊天应用程序中的 Amazon Q Developer 作为插件添加到所需的 Slack 工作区。有关更多信息，请参阅 Slack 文档中的向你的 Slack 工作区添加应用程序。请记下成功注册 AWS Management Console 后的 Slack 工作区 ID，如图所示。
- 聊天应用程序客户端中配置的 Amazon Q Developer，工作区 ID 随时可供在 AWS CloudFormation 控制台中输入。有关说明，请参阅 Amazon Q 开发者聊天应用程序管理员指南中的配置 Slack 客户端。
- 在 Amazon Simple Email Service (Amazon SES) 中创建和验证的源电子邮件账户，用于发送批准电子邮件。有关设置说明，请参阅《Amazon 简单电子邮件服务开发者指南》中的创建和验证电子邮件身份。
- 用于接收批准通知的目标电子邮件地址。此地址可以是共享收件箱或特定的团队通讯组列表。
- 可从您的访问的操作 SonarQube 实例 AWS 账户。有关更多信息，请参阅SonarQube 安装说明。
- 一种 SonarQube用户令牌，具有通过管道触发和创建项目的权限。

限制

- 在此解决方案中，创建自定义操作按钮是一个手动过程。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

架构

下图显示了此模式的工作流和体系结构组件。

下图显示了自动代码质量保证工作流程：

1. 代码准备和上传：

- 开发人员将代码库压缩成.zip 文件。
- 开发者手动将.zip 文件上传到指定的亚马逊简单存储服务 (Amazon S3) 存储桶。

2. Amazon S3 事件触发器和 AWS Step Functions 编排：
   - Amazon S3 上传事件会触发 Step Functions 工作流程。
   - Step Functions 使用编排 SAST 扫描。 SonarQube
   - 该工作流会监控 AWS CodeBuild 作业状态以确定后续操作。如果 CodeBuild 成功（质量门通过），则工作流程终止。如果 CodeBuild 失败，则调用 AWS Lambda 函数进行诊断。有关更多详细信息，请参阅本节后面的AWS Step Functions 逻辑。
3. AWS CodeBuild 执行：
   - 该 CodeBuild 作业对上传的代码库执行 SonarQube 扫描。
   - 扫描项目存储在单独的 Amazon S3 存储桶中，用于审计和分析。
4. 故障分析（Lambda 函数）：
   - CodeBuild 失败时，会触发 `CheckBuildStatus` Lambda 函数。
   - CodeBuild 成功后，该过程即告终止，无需采取进一步行动。
5. Lambda 函数分析故障原因（质量门失效或其他问题）
   - 该`CheckBuildStatus`函数创建包含详细故障信息的自定义负载。
   - 该`CheckBuildStatus`函数向亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 主题发布自定义有效负载。
6. 通知系统：
   - 亚马逊 SNS 在聊天应用程序中将有效负载转发给 Amazon Q Developer，用于集成 Slack。
7. Slack 集成：
   - 聊天应用程序中的 Amazon Q 开发者在指定的 Slack 频道中发布通知。
8. 批准流程：
   - 批准者在 Slack 通知中查看失败详情。
   - 批准者可以使用 Slack 中的 “批准” 按钮启动审批。
9. 批准处理人：
   - 批准 Lambda 函数处理来自 Slack 的批准操作。
   - 批准功能会将自定义消息发布到 Amazon SES。
10. 已生成消息：
    - 批准功能会生成一条用于通知开发者的自定义消息。
11. 开发者通知：

- Amazon SES 会向开发者发送一封包含后续步骤或所需操作的电子邮件。

该工作流程将手动代码上传与自动质量检查相结合，通过 Slack 提供即时反馈，并在必要时允许人工干预，从而确保稳健而灵活的代码审查流程。

AWS Step Functions 逻辑

如前面的架构图所示，如果质量门 SonarQube 失败，则工作流程将转到 `CheckBuildStatus` Lambda 函数。该`CheckBuildStatus`函数在 Slack 频道上触发通知。每份通知都包含建议的后续步骤的信息。以下是通知的类型：

- 应用程序在代码安全扫描中失败 — 当上传的代码未通过 SonarQube 安全扫描时，用户会收到此通知。用户可以选择 “批准” 来接受构建。但是，该通知建议用户注意潜在的代码质量差和安全风险。该通知包括以下详细信息：
  - 后续步骤：错误：质量门禁状态：失败 — 通过提供的 URL 查看详细信息。
  - 按照文档中提供的网址对漏洞进行分类。
  - CodeBuild 有关详细信息，请访问所提供的 URL。
- 应用程序扫描管道由于其他原因而失败 — 当管道由于代码安全扫描失败以外的其他原因而失败时，用户会收到此通知。该通知包括以下详细信息：
  - 有关后续步骤，请访问提供的链接以进行进一步的故障排除。

要查看通知在 Slack 频道中显示的屏幕截图，请前往 GitHub chatops-slack 存储库中的资产文件夹。

下图显示了质量门禁失败后 Step Functions 步骤状态的示例。

工具

AWS 服务

- 聊天应用程序中的 Amazon Q Developer 使您能够使用 Amazon Chime、Microsoft Teams 和 Slack 聊天频道来监控和响应应用程序中的操作事件。 AWS 终止支持通知： AWS 将于 2026 年 2 月 20 日终止对 Amazon Chime 服务的支持。2026 年 2 月 20 日之后，您将无法再访问 Amazon Chime 控制台或 Amazon Chime 应用程序资源。欲了解更多信息，请访问博客文章。这不会影响 Amazon Chime 软件开发工具包服务的可用性。
- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。

- AWS CodeBuild是一项完全托管的生成服务，可帮助您编译源代码、运行单元测试和生成可随时部署的工件。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥以帮助保护您的数据。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Secrets Manager 有助于您通过对 Secrets Manager 的 API 调用来替换代码中的硬编码凭证（包括密码），以编程方式检索密钥。
- 亚马逊简单电子邮件服务 (Amazon SES) Service 可帮助您使用自己的电子邮件地址和域名发送和接收电子邮件。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Step Functions是一项无服务器编排服务，可帮助您组合 AWS Lambda 功能和其他功能 AWS 服务 来构建关键业务应用程序。

其他工具

- Salesforce 推出的 Slack 是一款由 AI 驱动的对话平台，可提供聊天和视频协作，无需代码即可自动执行流程，并支持信息共享。
- SonarQube是一款本地分析工具，旨在检测 30 多种语言、框架和 IaC 平台中的编码问题。

代码存储库

此模式的代码可在 GitHub chatops-s lack 存储库中找到。

最佳实践

- CloudFormation 堆栈管理-如果您在 CloudFormation 堆栈执行过程中遇到任何故障，我们建议您删除失败的堆栈。然后，使用正确的参数值重新创建它。这种方法支持干净的部署，有助于避免潜在的冲突或部分实现。
- 共享收件箱电子邮件配置-配置`SharedInboxEmail`参数时，请使用所有相关开发者都可以访问的通用通讯组列表。这种方法提高了透明度，并有助于将重要的通知传达给相关的团队成员。

- 生产批准工作流程 — 对于生产环境，限制对用于生成批准的 Slack 频道的访问权限。只有指定的批准者才能成为该渠道的成员。这种做法保持了明确的责任链，并通过限制谁可以批准关键变更来增强安全性。
- IAM 权限-遵循最低权限原则，授予执行任务所需的最低权限。有关更多信息，请参阅 IAM 文档中的授予最低权限和安全最佳实践。

操作说明

执行初始设置

Task	描述	所需技能
克隆存储库。	要克隆此模式的 chatops-slack 存储库，请使用以下命令。 `git clone “git@github.com:aws-samples/chatops-slack.git“`	AWS DevOps、构建主管、DevOps 工程师、云管理员

创建包含 Lambda 代码的.zip 文件。	为和`ApprovalEmail` 功能的 AWS Lambda 函数代码创建.zip 文件。`CheckBuildStatus` 要创建`notification.zip` 和`approval.zip` ，请使用以下命令。
	`cd chatops-slack/src`
	`chmod -R 775 *`
	`zip -r approval.zip approval`
	`zip -r notification.zip notification`
		AWS DevOps、构建主管、DevOps 工程师、云管理员

部署先决条件.yml 堆栈文件

Task	描述	所需技能
执行pre-requisite.yml 堆栈文件。	pre-requisite.yml CloudFormation 堆栈文件会部署在您执行堆app-security.yml 栈文件之前所需的初始资源。要执行该pre-requisite.yml 文件，请执行以下操作： 1. 登录并 AWS Management Console打开AWS CloudFormation控制台。选择 “创建堆栈”，然后从下拉列表中选择 “使用新资源（标准）”。 2. 在创建堆栈页面上，选择选择现有模板并上传模板文件。然后，选择 “选择文件”，然后选择 p re-requisite.yml。选择下一步。 3. 在 “指定堆栈详细信息” 页面上，输入参数值，如其他信息中所述。然后选择下一步。 4. 在配置堆栈选项页面上，选择用于创建资源的 IAM 角色，如先决条件中所述。然后选择下一步。 5. 在 “查看并创建” 页面上，选择 “提交”。 6. 在堆栈的详细信息页面上，选择资源和输出选项卡。记下以下步骤中使用	AWS 管理员、AWS DevOps、构建主管、 DevOps 工程师

Task	描述	所需技能
	的S3LambdaCKMSKeyArn 、和CKMSKeyId 参数的值。	
将.zip 文件上传到亚马逊 S3 存储桶。	将您之前创建的notification.zip 和approval.zip 文件上传到名为 的 Amazon S3 存储桶S3LambdaBucket 。app-security.yml CloudFormation 堆栈文件用于S3LambdaBucket 配置 Lambda 函数。	AWS DevOps，构建主管，DevOps 工程师，AWS 系统管理员

执行 app-security.yml 堆栈文件

Task	描述	所需技能
执行app-security.yml 堆栈文件。	app-security.yml 堆栈文件为通知和批准系统部署剩余的基础架构。要执行该app-security.yml 文件，请执行以下操作： 1. 登录并 AWS Management Console打开AWS CloudFormation控制台。选择 “创建堆栈”，然后从下拉列表中选择 “使用新资源（标准）”。 2. 在创建堆栈页面上，选择选择现有模板并上传模板文件。然后，选择 “选择文件”，然后选择 app-security.yml。选择下一步。	AWS DevOps、AWS 系统管理员、 DevOps 工程师、构建主管

Task	描述	所需技能
	3. 在 “指定堆栈详细信息” 页面上，输入参数值，如其他信息中所述。然后选择下一步。 4. 在配置堆栈选项页面上，选择用于创建资源的 IAM 角色，如先决条件中所述。然后选择下一步。 5. 在 “查看并创建” 页面上，选择 “提交”。	

Task	描述	所需技能
测试通知设置。	要测试通知设置，请执行以下操作： 1. 打开 Amazon SNS 控制台。在左侧导航窗格中，选择主题。 2. 选择以 LambdaToAWSSlackChat bot 结尾的主题名称。 3. 在主题的详细信息页面上，选择发布消息。 4. 在将消息发布到主题页面上，要将消息正文发送到终端节点，请输入以下内容： `{ "version": "1.0", "source": "custom", "content": { "description": ":warning: This is a test notification" } }` 5. 选择发布消息。 成功发送测试消息后，您应该会在 Slack 频道上看到一条通知。有关更多信息，请参阅聊天应用程序中的 Amazon Q 开发者管理员指南中的测试从 AWS 服务 Slack 发送的通知。	AWS DevOps、AWS 系统管理员、 DevOps 工程师、构建主管

设置批准流程

Task	描述	所需技能
配置自定义 Lambda 操作。	要设置自定义 AWS Lambda 操作，请执行以下操作： 1. 在 Slack 频道中，选择配送通知底部的垂直省略号按钮。 2. 在管理操作中，选择创建。 3. 输入自定义操作名称，例如“批准”。此名称是您的自定义操作的唯一标识符。 4. 输入自定义操作按钮的名称，例如 “批准”。此名称显示在通知上的按钮上。此名称应不超过 20 个字符，并且可以包含表情符号。 5. 对于自定义操作类型，请选择 Lambda 操作。 6. 选择下一步。 7. 选择要部署此操作的 AWS 账户 和 AWS 区域 位置。 8. 选择加载 Lambda。 9. 在定义 Lambda 函数中，选择一个结尾为的 Lambda 函数。ApprovalEmailLambda然后选择下一步。 10要创建 “批准” 按钮，请在 “显示标准” 页面上选择 “保存”。	AWS 管理员、AWS DevOps、构建主管、 DevOps 工程师、Slack 管理员

Task	描述	所需技能
验证批准流程。	要验证批准流程是否按预期运行，请在 Slack 中选择 “批准” 按钮。 Slackbot 应在消息会话中发送一条通知，其中包含成功发送的确认字符串 “批准电子邮件”。	AWS 管理员、AWS DevOps、 DevOps 工程师、Slack 管理员

故障排除

事务	解决方案
Slack 配置错误	有关疑难解答与 Slack 配置错误相关的问题的信息，请参阅 Amazon Q 开发者聊天应用程序管理员指南中的 Amazon Q 开发者疑难解答。
由于其他原因，扫描失败	此错误表示代码生成任务已失败。要解决问题，请转到消息中的链接。代码生成任务失败可能有以下原因： • 应用程序打包不正确。该`sonar-scanner` 命令找不到`sonar.project.env.properties` 文件。 • `SonarFileName` `SonarFileDirectory` 、或`SonarToken` 参数的值不正确。检查这些值，然后再次执行堆栈文件。 • 无法访问声纳主机。 • 您可以使用日志来解决的其他问题。

相关资源

AWS 文档

- 配置 Slack 客户端
- 创建自定义操作
- 创建电子邮件地址身份程序
- 教程：开始使用 Slack

其他资源

- 将应用程序添加到你的 Slack 工作区（Slack 文档）
- 生成和使用令牌（SonarQube 文档）
- 服务器安装简介（SonarQube 文档）

其他信息

此解决方案强调聊天应用程序中的 Amazon Q Developer 自定义操作以用于发布管理。但是，您可以通过修改特定用例的 Lambda 代码来重复使用该解决方案，并在此基础上进行构建。

CloudFormation 堆栈文件的参数

下表显示了 CloudFormation 堆栈文件的参数及其描述`pre-requisite.yml`。

键	描述
`StackName`	CloudFormation 堆栈的名称。
`S3LambdaBucket`	您在其中上传 Lambda 代码的 Amazon S3 存储桶的名称。此名称必须全局唯一。
`SonarToken`	先决条件中所述的 SonarQube 用户令牌。

下表显示了 CloudFormation 堆栈文件的参数及其描述`app-security.yml`。

键	描述
`CKMSKeyArn`	在此堆栈中创建的 IAM 角色和 Lambda 函数中使用的 AWS KMS key 亚马逊资源名称 (ARN)。

CKMSKeyId	在此堆栈中创建的 Amazon SNS 主题中使用的 AWS KMS key ID。
EnvironmentType	用于部署应用程序扫描管道的客户端环境的名称。从允许值的下拉列表中选择环境名称。
S3LambdaBucket	包含和notification.zip 文件的 Amazon S3 存储桶approval.zip 的名称。
SESEmail	在 Amazon SES 中注册的电子邮件身份的名称，如先决条件中所述。此身份是源电子邮件地址。
SharedInboxMail	发送扫描通知的目标电子邮件地址。
SlackChannelId	你想要向其发送通知的 Slack 频道的频道 ID。要查找频道 ID，请在 Slack 应用程序的 “频道详情” 中右键单击频道名称。频道 ID 位于底部。
SlackWorkspaceId	Slack 工作空间 ID，如先决条件中所述。要查找 Slack 工作空间 ID，请登录，在聊天应用程序控制台中打开 Amazon Q Developer，然后选择已配置的客户端、Slack、Wor kspace ID。 AWS Management Console
StackName	CloudFormation 堆栈的名称。
SonarFileDirectory	包含该sonar.project.<env>.properties 文件的目录。
SonarFileName	sonar.project.<env>properties 文件名。
SourceCodeZip	包含该文件和源代码的.zip sonar.project.<env>properties 文件的名称。

使用 AWS Network Firewall 和 AWS Transit Gateway 部署防火墙

由 Shrikant Patil (AWS) 编写

摘要

此模式向您介绍如何使用 AWS Network Firewall 和 AWS Transit Gateway 部署防火墙。Network Firewall 资源是使用 AWS CloudFormation 模板部署的。Network Firewall 会根据您的网络流量自动扩缩，并且可以支持数十万个连接，因此您不必担心构建和维护自己的网络安全基础设施。传输网关是一个网络传输中心，可用于互连虚拟私有云 (VPCs) 和本地网络。

在此模式中，您还将学习在网络架构中加入检查 VPC。最后，此模式说明了如何使用 Amazon CloudWatch 为您的防火墙提供实时活动监控。

> **Tip**
>
> 最佳做法是避免使用 Network Firewall 子网部署其他 AWS 服务。原因是 Network Firewall 无法检查来自防火墙子网内源或目标的流量。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- AWS Identity and Access Management (IAM) 角色和策略权限
- CloudFormation 模板权限

限制

您可能在域筛选方面遇到问题，需要另一种配置。有关更多信息，请参阅 Network Firewall 文档中的 AWS Network Firewall 中有状态域列表规则组。

架构

技术堆栈

- Amazon CloudWatch 日志

- Amazon VPC
- AWS Network Firewall
- AWS Transit Gateway

目标架构

下图显示了如何使用 Network Firewall 和 Transit Gateway 检查您的流量：

例架包括以下组件：

- 您的应用程序托管在两个分支中 VPCs。 VPCs 它们由 Network Firewall 监控。
- 出口 VPC 可直接访问互联网网关，但不受 Network Firewall 保护。
- 检查 VPC 是部署 Network Firewall 的位置。

自动化和扩缩

您可以使用基础架构即代码CloudFormation来创建此模式。

工具

Amazon Web Services

- Amazon CloudWatch Lo gs 可帮助您集中管理来自所有系统、应用程序和 AWS 服务的日志，以便您可以监控它们并安全地将其存档。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。
- AWS Network Fire wall 是一项有状态的托管式网络防火墙以及入侵检测和防御服务，适用于 AWS 云 VPCs 中。
- AWS Transit Gateway 是一个连接 VPCs 本地网络的中心枢纽。

代码

此模式的代码可在带有 Tr ansit Gateway 存储库 GitHub 的 AWS Network Firewall 部署中找到。您可以使用此存储库中的 CloudFormation 模板部署使用 Network Firewall 的单个检查 VPC。

操作说明

创建分支 VPC 和检查 VPC

Task	描述	所需技能
准备并部署 CloudFormation 模板。	1. 从 GitHub 存储库下载`cloudformation/aws_nw_fw.yml` 模板。 2. 使用您的值更新模板。 3. 部署模板。	AWS DevOps

创建中转网关和路由

Task	描述	所需技能
创建中转网关。	1. 登录 AWS 管理控制台，打开 Amazon VPC 控制台。 2. 在导航窗格中，选择 Transit Gateways（中转网关）。 3. 选择 Create Transit Gateway(创建中转网关)。 4. 对于 Name tag（名称标签），输入中转网关的名称。 5. 对于 Description（描述），输入中转网关的描述。 6. 对于 Amazon side 自治系统号 (ASN)，保留默认 ASN。 7. 选择 DNS 支持选项。 8. 选择 VPN ECMP 支持选项。 9. 选择默认路由表关联选项。此选项自动将中转网关连接	AWS DevOps

Task	描述	所需技能
	与中转网关的默认路由表关联。 10选择默认路由表传播选项。此选项自动将中转网关连接传播至与中转网关的默认路由表。 11选择 Create Transit Gateway(创建中转网关)。	
创建中转网关连接。	为以下内容创建中转网关连接： • 检查 VPC 和 Transit Gateway 子网的检查附件 • 分支 VPCA 和私有子网中的 SpokeVPCA 附件 • 分支 VPCB 和私有子网中的 SpokevPCB 附件 • 出口 VPC 和私有子网中的 EgressVPC 附件	AWS DevOps

Task	描述	所需技能
创建中转网关路由表。	1. 为分支 VPC 创建中转网关路由表。此路由表必须与除检查 VPC 之 VPCs 外的所有路由表相关联。 2. 为防火墙创建中转网关路由表。此路由表只能关联到检查 VPC。 3. 将路由添加到防火墙的中转网关路由表： • 对于 `0.0.0/0`，请使用 egressVPC 附件。 • 对于 SpokeVPCA C IDR 块，请使用 Spoke 附件。VPC1 • 对于 SpokevPCB CIDR 块，请使用 Spoke 附件。VPC2 4. 将路由添加到分支 VPC 的中转网关路由表。对于`0.0.0/0`，请使用检查 VPC 附件。	AWS DevOps

创建防火墙与路由

Task	描述	所需技能
在检查 VPC 中创建防火墙。	1. 登录 AWS 管理控制台，打开 Amazon VPC 控制台。 2. 在导航窗格中的 Network Firewall 下，选择防火墙。 3. 选择创建防火墙。	AWS DevOps

Task	描述	所需技能
	4. 对于 Name (名称)，输入要用于标识此防火墙的名称。在创建防火墙后，您无法更改其名称。 5. 对于 VPC，请选择您的检查 VPC。 6. 在可用区和子网中，选择您确定的区域和防火墙子网。 7. 在关联的防火墙策略部分，选择关联现有防火墙策略，然后选择您之前创建的防火墙策略。 8. 选择创建防火墙。	

Task	描述	所需技能
创建防火墙策略。	1. 登录 AWS 管理控制台，打开 Amazon VPC 控制台。 2. 在导航窗格中的 Network Firewall 下，选择防火墙策略。 3. 在描述防火墙策略页面，选择创建防火墙策略。 4. 对于名称，输入要用于防火墙策略的名称。稍后在此模式中将策略与防火墙关联，您将使用该名称来标识该策略。在创建防火墙策略后，您无法更改其名称。 5. 选择 Next(下一步)。 6. 在添加规则组页面的 无状态规则组部分，选择添加无状态规则组。 7. 在从现有规则组添加对话框中，选中之前创建的无状态规则组对应的复选框。选择添加规则组。注意：在页面底部，防火墙策略的容量计数器显示在防火墙策略允许的最大容量旁边添加此规则组所消耗的容量。 8. 将无状态默认操作设置为转发到有状态规则。 9. 在有状态的规则组部分，选择添加有状态的规则组，然后选中前面创建的有状态规则组的复选框。选择添加规则组。	AWS DevOps

Task	描述	所需技能
	10选择下一步，逐步完成安装向导的其余部分，然后选择创建防火墙策略。	
更新VPC路由表。	检查 VPC 路由表 1. 在ANF子网路由表 (`Inspection-ANFRT`) 中，添加 `0.0.0/0` 到 Transit Gateway ID。 2. 在 Transit Gateway 子网路由表 (`Inspection-TGWRT`) 中，添加 `0.0.0/0` 到 egress VPC。 SpokeVPCA 路由表 在私有路由表中，添加 `0.0.0.0/0` 到 Transit Gateway ID。 分支 VPCB 路由表 在私有路由表中，添加 `0.0.0.0/0` 到 Transit Gateway ID。 VPC 路由表 在出口公共路由表中，将 SpokeVPCA 和 Spoke VPCB CIDR 块添加至 Transit Gateway ID 中。对私有子网重复相同步骤。	AWS DevOps

设置 CloudWatch 为执行实时网络检查

Task	描述	所需技能
更新防火墙的日志配置。	1. 登录 AWS 管理控制台，打开 Amazon VPC 控制台。 2. 在导航窗格中的 Network Firewall 下，选择防火墙。 3. 在 Firewalls 页面，选择要修改的防火墙名称。 4. 选择防火墙详细信息选项卡。在 日志记录 部分中，选择 Edit (编辑)。 5. 根据需要调整日志类型选择。您可为警报和流日志配置日志记录。 • 警报-发送与操作设置为警报或丢弃的任何状态规则相匹配的流量日志。有关有状态的规则和规则组的更多信息，请参阅 AWS Network Firewall 中的规则组。 • 流 – 发送无状态引擎转发到有状态规则引擎的所有网络流量的日志。 6. 对于每个选定的日志类型，选择目标类型，然后提供日志记录目标的信息。有关更多信息，请参阅 AWS Network Firewall 文档中的在 AWS Network Firewall 日志记录目的地 7. 选择保存。	AWS DevOps

验证设置

Task	描述	所需技能
启动 EC2 实例以测试设置。	在分支 VPC 中@@ 启动两个亚马逊弹性计算云 (Amazon EC2) 实例：一个用于 Jumpbox，一个用于测试连接。	AWS DevOps
检查指标。	指标的分组首先依据服务命名空间，然后依据每个命名空间内的各种维度组合。Network Firewall 的 CloudWatch 命名空间是`AWS/NetworkFirewall` 。 1. 登录 AWS 管理控制台并打开CloudWatch 控制台。 2. 在导航窗格中，选择指标。 3. 在所有指标选项卡上，选择区域，然后选择 AWS/NetworkFirewall。	AWS DevOps

相关资源

- 带互联网网关的简单单区域架构
- 带互联网网关的多区域架构
- 带互联网网关和 NAT 网关的架构

使用 AWS CodePipeline CI/CD 管道部署 AWS Glue 作业

由 Bruno Klein (AWS) and Luis Henrique Massao Yamada (AWS) 编写

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

此模式演示了如何将 AWS CodeCommit 和 AWS CodePipeline 与 AWS Glue 集成，并在开发人员将其更改推送到远程 AWS 存储库后立即使用 AWS CodeCommit Lambda 启动作业。

当开发者向提取、转换和加载 (ETL) 存储库提交更改并将更改推送到 AWS 时 CodeCommit，将调用新的管道。管道启动 Lambda 函数，该函数会启动包含这些更改的 AWS Glue 作业。AWS Glue 作业执行 ETL 任务。

当企业、开发人员和数据工程师希望在提交更改并将其推送到目标存储库后立即启动作业时，此解决方案非常有用。它有助于实现更高水平的自动化和可重复性，从而避免作业启动和生命周期期间出现错误。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- Git 已安装在本地计算机上
- 安装在本地计算机上的 Amazon Cloud Development Kit (Amazon CDK)
- 安装在本地机器中的 Python
- 附件部分中的代码

限制

- AWS Glue 作业成功启动后，管道即完成。它不会等待作业结束。
- 附件中提供的代码仅用于演示目的。

架构

目标技术堆栈

- AWS Glue
- AWS Lambda
- AWS CodePipeline
- AWS CodeCommit

目标架构

该过程包括这些步骤：

1. 开发人员或数据工程师对 ETL 代码进行修改、提交并将更改推送到 AWS CodeCommit。
2. 推送启动管道。
3. 管道启动 Lambda 函数，它会调用存储库上的 `codecommit:GetFile` 并将文件上传到Amazon Simple Storage Service (Amazon S3).。
4. Lambda 函数使用 ETL 代码启动新的 AWS Glue 作业。
5. Lambda 函数完成管道。

自动化和扩缩

示例附件演示了如何将 AWS Glue 与 AWS 集成 CodePipeline。它提供了一个基准示例，您可对其进行自定义或扩展以供自己使用。有关详细信息，请参阅操作说明部分。

工具

- AWS CodePipeline — AWS CodePipeline 是一项完全托管的持续交付服务，可帮助您自动执行发布渠道，实现快速可靠的应用程序和基础设施更新。
- AWS CodeCommit — AWS CodeCommit 是一项完全托管的源代码控制服务，可托管基于 Git 的安全存储库。
- AWS Lambda – AWS Lambda 是一项无服务器计算服务，可帮助您运行代码，无需预置或管理服务器。
- AWS Glue – AWS Glue 是一项无服务器数据集成服务，可轻松发现、准备和组合数据，以用于分析、机器学习和应用程序开发。
- Git 客户端 — Git 提供 GUI 工具，或者你可以使用命令行或桌面工具从中查看所需的工件 GitHub。
- AWS CDK – The AWS CDK 是开源软件开发框架，帮助您使用熟悉的编程语言定义云应用程序资源。

操作说明

部署示例代码

Task	描述	所需技能
配置 AWS CLI。	将 AWS 命令行界面（AWS CLI）配置为目标并使用您当前的 Amazon Web Services account 进行身份验证。有关说明，请参阅 AWS CLI 文档。	开发人员、 DevOps 工程师
提取项目文件示例。	从附件中提取文件，以创建包含示例项目文件的文件夹。	开发人员、 DevOps 工程师

部署示例代码。	解压缩文件后，从提取位置运行以下命令，以创建基准示例：
	`cdk bootstrap`
	`cdk deploy`
	`git init`
	`git remote add origin <code-commit-repository-url>`
	`git stage .`
	`git commit -m "adds sample code"`
	`git push --set-upstream origin main`
	执行最后一个命令后，您可以监控管道和 AWS Glue 作业的状态。
		开发人员、 DevOps 工程师

自定义代码。	根据您的业务需求自定义 etl.py 文件代码。您可修改 ETL 代码、修改管道阶段或者扩展解决方案。	数据工程师

相关资源

- AWS CDK 入门
- 在 AWS Glue 中添加作业
- 源操作集成 CodePipeline
- 在中的管道中调用 AWS Lambda 函数 CodePipeline
- AWS Glue 编程
- AWS CodeCommit GetFile API

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用实例配置文件从 AWS Cloud9 部署 Amazon EKS EC2 集群

由 Sagar Panigrahi (AWS) 编写

摘要

注意： AWS Cloud9 不再向新客户开放。的现有客户 AWS Cloud9 可以继续照常使用该服务。了解更多

此模式描述了如何使用 AWS Cloud9 和 AWS CloudFormation 创建亚马逊弹性 Kubernetes Service (Amazon EKS) 集群，该集群无需为亚马逊网络服务 (AWS) 账户中的用户启用编程访问即可运行。

AWS Cloud9 是一种基于云的集成式开发环境（IDE），它可以帮助您使用浏览器编写、运行和调试代码。AWS Cloud9 用作控制中心，使用亚马逊弹性计算云 (Amazon EC2) 实例配置文件和 AWS 模板来配置 Amazon EKS CloudFormation 集群。

如果您不想创建 AWS Idtity and Acccess Management(IAM) 用户，而是想改用 IAM 角色，则可以使用此模式。基于角色的访问控制 (RBAC) 根据个人用户的角色来控制对资源的访问。此模式演示了如何在 Amazon EKS 集群中更新 RBAC，以允许访问特定的 IAM 角色。

该模式的设置还可以帮助您的 DevOps 团队使用 AWS Cloud9 功能来维护和开发用于创建 Amazon EKS 基础设施的基础设施即代码 (IaC) 资源。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 为账户创建 IAM 角色与策略的权限。用户的 IAM 角色必须包含 `AWSCloud9Administrator` 策略。还必须创建 `AWSServiceRoleForAmazonEKS` 和 `eksNodeRoles` 角色，因为它们是创建 Amazon EKS 集群所需。
- 了解 Kubernetes 的概念。

限制

- 此模式介绍如何创建基本 Amazon EKS 集群。对于生产集群，您必须更新 AWS CloudFormation 模板。
- 该模式不会部署额外的 Kubernetes 组件(例如 Fluentd、入口控制器或存储控制器)。

架构

技术堆栈

- Amazon Cloud9
- AWS CloudFormation
- Amazon EKS
- IAM

自动化和扩缩

您可扩展此模式并将其整合到持续集成和持续部署 (CI/CD) 管道中，以自动完成 Amazon EKS 的完整配置。

工具

- AWS CloudFormation — AWS CloudFormation 可帮助您建模和设置 AWS 资源，这样您就可以减少管理这些资源的时间，将更多时间集中在应用程序上。
- AWS Cloud9 – AWS Cloud9 提供丰富的代码编辑体验，对多种编程语言和运行时系统调试程序的支持以及内置终端。
- AWS CLI - AWS 命令行界面（AWS CLI）是一种开源工具，它使您能够使用命令行 shell 中的命令与 Amazon Web Services 交互。
- Kubectl – `kubectl` 是命令行实用程序，用于与 Amazon EKS 集群交互。

操作说明

为 EC2 实例配置文件创建 IAM 角色

Task	描述	所需技能
创建 IAM policy。	登录 AWS 管理控制台，打开 IAM 控制台，选择策略，然后选择创建策略。选择 JSON 选	云管理员

Task	描述	所需技能
	项卡，然后粘贴 policy-role-eks-instance-profile-for-cloud9.json 文件（附后）中的内容。 解决策略验证过程中生成的任何安全警告、错误或常规警告，然后选择 Review policy（查看策略）。输入策略的名称。策略名称，建议使用 `eks-instance-profile-for-cloud9` 。 查看策略摘要以查看您的策略授予的权限。然后选择创建策略。	
使用策略创建 IAM 角色。	在 IAM 控制台中，选择角色，然后选择创建角色。选择 AWS 服务，然后EC2从列表中选择。 选择下一步：权限并搜索您之前创建的 IAM policy。根据要求选择合适的标签。 在Review（查看）部分，输入角色的名称。角色名称，建议使用 `role-eks-instance-profile-for-cloud9` 。然后选择 Create role(创建角色)。	云管理员

为 Amazon EKS RBAC 创建 IAM policy 和角色。

Task	描述	所需技能
创建 IAM policy。	在 IAM 控制台中，选择 Policies（策略），然后选择 Create policy（创建策略）。选择 JSON 选项卡，然后粘贴 policy-for-eks-rbac .json 文件（附后）中的内容。 解决策略验证过程中生成的任何安全警告、错误或常规警告，然后选择 Review policy（查看策略）。输入策略的名称。策略名称，建议使用 `policy-for-eks-rbac` 。查看策略摘要以查看您的策略授予的权限。然后选择创建策略。	云管理员
使用策略创建 IAM 角色。	在 IAM 控制台中，选择角色，然后选择创建角色。选择 AWS 服务，然后EC2从列表中选择。选择下一步：权限并搜索您之前创建的 IAM policy。根据要求选择合适的标签。 在Review（查看）部分，输入角色的名称。角色名称，建议使用 `role-eks-admin-for-rbac` 。然后选择 Create role(创建角色)。	云管理员

创建 AWS Cloud9 环境

Task	描述	所需技能
创建 AWS Cloud9 环境。	打开 AWS Cloud9 控制台并选择创建环境。在 Name environment（命名环境）页面，输入环境的名称。环境名称，建议使用 `eks-management-env` 。根据您的要求配置其余设置，然后选择下一步。 在 Review(查看)页面中，选择 Create environment(创建环境)。等待 AWS Cloud9 创建环境。这个过程可能需要几分钟。 有关可用配置选项的更多信息，请参阅 AWS Cloud9 文档中的创建 EC2 环境。	云管理员
移除 AWS Cloud9 的临时 IAM 凭证。	配置 AWS Cloud9 环境后，选择齿轮图标中的设置。在首选项，选择 AWS 设置，然后选择凭证。 关闭 AWS 托管的临时凭证并关闭选项卡。	云管理员
将 EC2 实例配置文件附加到底层 EC2 实例。	打开亚马逊 EC2 控制台，在 AWS Cloud9 中选择与您的环境相匹配的 EC2 实例。如果您使用我们推荐的名称，则会调用该 EC2 实例`aws-cloud9-eks-management-env` 。	云管理员

Task	描述	所需技能
	选择实 EC2 例，选择操作，然后选择实例设置。选择附加/替换 IAM 角色。搜索 `role-eks-instance-profile-for-cloud9` 或您之前创建的 IAM 角色的名称，然后选择应用。	

创建 Amazon EKS 集群

Task	描述	所需技能
创建 Amazon EKS 集群。	下载并打开适用于 AWS 的 eks-cfn.yaml（附后）模板。CloudFormation根据您的要求编辑模板。 打开 AWS Cloud9 环境并选择新文件。将您之前创建的 AWS CloudFormation 模板粘贴到字段中。建议您使用 eks-cfn.yaml 为模板名称。 在 AWS Cloud9 终端中，运行以下命令，以创建 Amazon EKS 集群： `aws cloudformation create-stack --stack-name eks-cluster --template-body file://eks-cfn.yaml --region <your_AWS_Region>`	云管理员

Task	描述	所需技能
	如果 AWS CloudFormation 调用成功，您将在输出中收到 AWS CloudFormation 堆栈的亚马逊资源名称 (ARN)。堆栈创建可能需要 10 到 20 分钟。	
验证 Amazon EKS 集群的状态。	在 AWS CloudFormation 控制台上，打开堆栈页面，然后选择堆栈名称。 堆栈状态代码显示 `CREATE_COMPLETE` 时即会创建堆栈。有关更多信息，请参阅 AWS CloudFormation 文档中的查看 AWS CloudFormation 堆栈数据和资源。	云管理员

访问 Amazon EKS 集群中的 Kubernetes 资源

Task	描述	所需技能
在 AWS Cloud9 环境安装 kubectl。	按 Amazon EKS 文档中的安装 kubectl 在您的 AWS Cloud9 环境中安装`kubectl`。	云管理员
在 AWS Cloud9 中更新新的 Amazon EKS 配置。	在 AWS Cloud9 终端中运行以下命令，将 `kubeconfig` 从 Amazon EKS 集群更新到 AWS Cloud9 环境： `aws eks update-kubeconfig --name EKS-DEV2 --region <your_AWS_Region>`	云管理员

Task	描述	所需技能
	Important EKS-DEV2是您用来创建集群的 AWS CloudFormation 模板中的 Amazon EKS 集群的名称。 运行 `kubectl get all -A` 命令查看所有 Kubernetes 资源。	

Task	描述	所需技能

将管理员 IAM 角色添加至 Kubernetes RBAC。	在您的 AWS Cloud9 终端中运行以下命令，以编辑模式打开 Amazon EKS 的 RBAC 配置图：
	
	`kubectl edit cm/aws-auth -n kube-system`
	
	在 `mapRoles` 部分下方添加以下几行：
	
	`- groups:`
	`- system:masters`
	`rolearn: <ARN_of_IAM_role _from_second_epic>`
	`username: eksadmin`
	
	检查 YAML 格式的文件，以避免语法错误。使用 `vi` 命令保存文件，然后退出该文件。
	
	Note
	通过添加此部分，您可以通知 Kubernetes RBAC，后者将获得对 Amazon `<ARN_of_IAM_role _from_second_epic>` EKS 集群的完全管理员访问权限。这意味着所识别的 IAM 角色可以对 Kubernetes 集群执行管理操作。AWS 在配
		云管理员

Task	描述	所需技能
	置 Amazon EKS 集群时在 `mapRoles` 下方添加了现有部分。	

相关资源

参考

- 模块化和可扩展 Amazon EKS 架构（快速入门）。
- 管理 Amazon EKS 集群的用户或 IAM 角色
- 用于创建新的 Amazon EKS 控制平面的 AWS CloudFormation 模板

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS CodePipeline、AWS 和 AWS 在多个 AWS CodeCommit 区域部署代码 CodeBuild

由 Anand Krishna Varanasi (AWS) 编写

摘要

此模式演示了如何使用 AWS 跨多个 Amazon Web Services (AWS) 区域构建基础设施或架构 CloudFormation。它包括跨多个 AWS 区域的持续集成（CI）/持续部署（CD），以实现更快的部署。例如，此模式中的步骤已经过测试，用于创建部署到三个 AWS 区域的 AWS CodePipeline 任务。您可以基于用例来更改区域数量。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- • 在 AmazonS3 FullAccess 和CloudWatchFullAccess政策中 CodeBuild 扮演的角色。这些政策 CodeBuild 允许 CodeCommit 通过亚马逊观看 AWS 的事件， CloudWatch 以及使用亚马逊简单存储服务 (Amazon S3) Simple Storage Service 作为工件存储。
  - 具有以下策略的 AWS CloudFormation 角色，这些策略使 AWS CloudFormation 能够在最后的构建阶段创建或更新 AWS Lambda 函数、推送或监视 Amazon CloudWatch 日志，以及创建和更新更改集。
    - AWSLambdaFullAccess
    - AWSCodeDeployFullAccess
    - CloudWatchFullAccess
    - AWSCloudFormationFullAccess
    - AWSCodePipelineFullAccess

> **Note**
>
> 两个 AWS 和 AWS 的 AWS CodeBuild Identity and Access Management (IAM) 角色 CloudFormation 具有适当的策略，用于 CodeBuild 执行 CI 任务，即测试、捆绑、打包工件以及并行部署到多个 AWS 区域。 交叉检查由创建的策略 CodePipeline 以验证是否 CodeBuild 和 AWS 在 CI 和 CD 阶段 CloudFormation 具有适当的权限。

架构

此模式的多区域架构和工作流程包括以下步骤。

1. 您将代码发送到存储 CodeCommit 库。
2. 在收到任何代码更新或提交后， CodeCommit 调用一个 CloudWatch 事件，该事件反过来会启动作 CodePipeline 业。
3. CodePipeline 使用由 CodeBuild处理的 CI。将执行以下任务。
   - 测试 AWS CloudFormation 模板（可选）
   - 打包部署中包含的每个区域的 AWS CloudFormation 模板。例如，此模式与三个 AWS 区域并行部署，因此将 AWS CloudFormation 模板 CodeBuild 打包到三个 S3 存储桶中，每个指定区域一个。S3 存储桶仅供 CodeBuild 用作项目存储库。
4. CodeBuild 将工件打包为下一个部署阶段的输入，该阶段将在三个 AWS 区域并行运行。如果您指定不同数量的区域，则 CodePipeline 会部署到这些区域。

工具

工具

- AWS CodePipeline — CodePipeline 是一项持续交付服务，可用于对持续发布软件变更所需的步骤进行建模、可视化和自动化。
- AWS CodeBuild — CodeBuild 是一项完全托管的构建服务，可编译您的源代码、运行单元测试并生成可随时部署的项目。
- AWS CodeCommit — CodeCommit 是一项由 Amazon Web Services 托管的版本控制服务，您可以使用它来私下存储和管理云中的资产（例如源代码和二进制文件）。
- AWS CloudFormation — AWS CloudFormation 是一项服务，可帮助您建模和设置 Amazon Web Services 资源，这样您就可以花更少的时间管理这些资源，而将更多的时间集中在在 AWS 中运行的应用程序上。
- AWS Identity and Access Management– AWS Identity and Access Management (IAM) 是一项 Web 服务，可帮助您安全地控制对 AWS 资源的访问。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项面向互联网的存储服务。该服务旨在降低开发人员进行网络规模级计算的难度。

代码

以下示例代码适用于该 BuildSpec.yaml 文件（构建阶段）。

```
---
artifacts:
discard-paths: true
files:
- packaged-first-region.yaml
- packaged-second-region.yaml
- packaged-third-region.yaml
phases:
build:
commands:
- echo "********BUILD PHASE - CF PACKAGING**********"
- "aws cloudformation package --template-file sam-template.yaml --s3-bucket
 $S3_FIRST_REGION --output-template-file packaged-first-region.yaml --region
 $FIRST_REGION"
- "aws cloudformation package --template-file sam-template.yaml --s3-bucket
 $S3_SECOND_REGION --output-template-file packaged-second-region.yaml --region
 $SECOND_REGION"
- "aws cloudformation package --template-file sam-template-anand.yaml --s3-bucket
 $S3_THIRD_REGION --output-template-file packaged-third-region.yaml --region
 $THIRD_REGION"
install:
commands:
- echo "********BUILD PHASE - PYTHON SETUP**********"
runtime-versions:
python: 3.8
post_build:
commands:
- echo "********BUILD PHASE - PACKAGING COMPLETION**********"
pre_build:
commands:
- echo "********BUILD PHASE - DEPENDENCY SETUP**********"
- "npm install --silent --no-progress"
- echo "********BUILD PHASE - DEPENDENCY SETUP DONE**********"
version: 0.2
```

操作说明

准备代码和 CodeCommit 存储库

Task	描述	所需技能
选择用于部署的主要 AWS 区域。	登录 Amazon Web Services account，选择要部署的主区域。 CodeCommit 存储库将位于主区域。	DevOps
创建 CodeCommit 存储库。	创建 CodeCommit 存储库，并将所需的代码推送到其中。该代码通常包括 AWS CloudFormation 或 AWS SAM 模板、Lambda 代码（如果有）以及作为 AWS 输入的 CodeBuild `buildspec.yaml` 文件。 CodePipeline	DevOps
将代码推送到 CodeCommit 存储库中。	在附件部分中，下载此示例的代码，然后将所需代码推送到其中。通常，代码可以包含 AWS CloudFormation 或 AWS SAM 模板、Lambda 代码和作为管道输入的 CodeBuild `buildspec.yaml` 文件。	DevOps

源代码阶段：创建管线

Task	描述	所需技能
创建作 CodePipeline 业。	在 CodePipeline 控制台上，选择创建管道。	DevOps
为 CodePipeline 作业命名并选择服务角色设置。	输入作业的名称，并保留默认的服务角色设置，以便	DevOps

Task	描述	所需技能
	CodePipeline 创建附加必要策略的角色。	
指定构件存储的位置。	在“高级设置”下，保留默认选项，以便 CodePipeline 创建用于存储代码项目的 S3 存储桶。如果您改用现有 S3 存储桶，则该存储桶必须位于您在第一个操作说明中指定的主区域。	DevOps
指定加密密钥。	保留默认选项：默认 AWS 托管式密钥，或者选择使用您自己的 AWS Key Management Service（AWS KMS）客户托管密钥。	DevOps
指定源提供程序。	在来源提供商下，选择 AWS CodeCommit。	DevOps
指定存储库。	选择您在第一部长篇故事中创建的 CodeCommit 存储库。如果您将代码放在分支中，请选择该分支。	DevOps
指定如何检测代码更改。	保留默认的 Amazon Ev CloudWatch en ts 作为启动 CodePipeline 任务的 CodeCommit 更改触发器。	DevOps

构建阶段：配置管线

Task	描述	所需技能
指定构建提供程序。	对于构建提供商，请选择 AWS CodeBuild。	DevOps
指定 AWS 区域。	选择您在第一个操作说明中指定的主区域。	DevOps

构建阶段：创建和配置项目

Task	描述	所需技能
创建项目	选择创建项目，然后输入项目的名称。	DevOps
指定环境映像。	对于此模式演示，请使用默认的 CodeBuild 托管映像。如果您有自定义的 Docker 映像，您还可以选择使用该 Docker 映像。	DevOps
指定操作系统。	选择 Amazon Linux 2 或 Ubuntu。 **Note** 亚马逊 Linux 2 的支持已接近终止。欲了解更多信息，请参阅亚马逊 Linux 2 FAQs。	DevOps
指定服务角色。	选择您在开始创建 CodePipeline 作业 CodeBuild 之前为其创建的角色。（请参阅先决条件部分。）	DevOps

Task	描述	所需技能
设置其他选项。	对于超时和队列超时，请保留默认值。对于证书，除非您有要使用的自定义证书，否则请保留默认设置。	DevOps
创建环境变量。	对于您要部署到的每个 AWS 区域，请通过提供 S3 存储桶名称和区域名称（例如 us-east-1）来创建环境变量。	DevOps
如果不是 buildspec.yml，请提供 buildspec 文件名。	如果文件名为默认名称 `buildspec.yaml` ，则将此字段留空。如果您重命名了 buildspec 文件，请在此处输入名称。确保它与 CodeCommit 存储库中文件的名称相匹配。	DevOps
指定日志记录。	要查看 Amazon CloudWatch 事件的日志，请保留默认设置。或者，您可以定义任何特定的组或记录器名称。	DevOps

跳过“部署”阶段

Task	描述	所需技能
跳过部署阶段并完成管线的创建。	设置管道时，只 CodePipeline 允许您在 “部署” 阶段创建一个阶段。要部署到多个 AWS 区域，请跳过此阶段。创建管线后，您可以添加“部署阶段”的多个阶段。	DevOps

部署阶段：配置管线以部署到第一个区域

Task	描述	所需技能
向部署阶段添加阶段。	编辑管线，然后在“部署”阶段选择添加阶段。第一个阶段适用于主区域。	DevOps
提供阶段的操作名称。	输入反映第一个（主）阶段和区域的唯一名称。例如，输入 primary_<region>_deploy。	DevOps
指定操作提供程序。	对于操作提供者，请选择 AWS CloudFormation。	DevOps
为第一个阶段配置区域。	选择第一个（主要）区域，即设置 CodePipeline 和 CodeBuild 的相同区域。这是您要在其中部署堆栈的主区域。	DevOps
指定输入构件。	选择 BuildArtifact。这是构建阶段的输出。	DevOps
指定要采取的操作。	对于操作模式，选择创建或更新堆栈。	DevOps
输入堆 CloudFormation 栈的名称。		DevOps
为第一个区域指定模板。	选择由第一个（主）区域打包 CodeBuild 并转储到第一个（主）区域的 S3 存储桶中的特定于区域的软件包名称。	DevOps
指定功能。	如果堆栈模板包含 IAM 资源，或者您直接使用包含宏的模板创建堆栈，则需要功能。对于这种模式，请使用 CAPABILIT	DevOps

Task	描述	所需技能
	Y_IAM、CAPABILITY_NAMED_IAM、CAPABILITY_AUTO_EXPAND。	

部署阶段：配置管线以部署到第二个区域

Task	描述	所需技能
将第二个阶段添加到“部署”阶段。	要为第二个区域添加阶段，请编辑管线并在“部署”阶段选择添加阶段。重要：创建第二个区域的过程与第一个区域的创建过程相同，但以下值除外。	DevOps
提供第二个阶段的操作名称。	输入反映第二个阶段和第二个区域的唯一名称。	DevOps
为第二个阶段配置区域。	选择您希望在其上部署堆栈的第二个区域。	DevOps
为第二个区域指定模板。	选择由第二个区域打包 CodeBuild 并转储到 S3 存储桶中的特定于区域的软件包名称。	DevOps

部署阶段：配置管线以部署到第三个区域

Task	描述	所需技能
将第三个阶段添加到“部署”阶段。	要为第三个区域添加阶段，请编辑管线并在“部署”阶段选择添加阶段。重要：创建第三个区域的过程与前两个区域的创建过程相同，但以下值除外。	DevOps

Task	描述	所需技能
提供第三个阶段的操作名称。	输入反映第三个阶段和第三个区域的唯一名称。	DevOps
为第三个阶段配置区域。	选择您希望在其上部署堆栈的第三个区域。	DevOps
为第三个区域指定模板。	选择由第三个区域打包 CodeBuild 并转储到 S3 存储桶中的特定于区域的软件包名称。	DevOps

清理部署

Task	描述	所需技能
删除 AWS 资源。	要清理部署，请删除每个区域中的 CloudFormation 堆栈。然后从主区域中删除 CodeCommit CodeBuild、和 CodePipeline 资源。	DevOps

相关资源

- 什么是 AWS CodePipeline？
- AWS 无服务器应用程序模型
- AWS CloudFormation
- 适用于 AWS 的 AWS CloudFormation 架构结构参考 CodePipeline

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 Terraform 执行 Amazon Redshift SQL 查询

由 Sylvia Qi (AWS) 和 Aditya Ambati (AWS) 创作

摘要

使用基础设施即代码 (IaC) 来部署和管理 Amazon Redshift 是一种普遍的做法。 DevOpsIaC 有助于部署和配置各种 Amazon Redshift 资源，例如集群、快照和参数组。但是，IaC 并未扩展到数据库资源的管理，例如表、架构、视图和存储过程。这些数据库元素通过 SQL 查询进行管理，IaC 工具不直接支持。尽管有用于管理这些资源的解决方案和工具，但您可能不希望在技术堆栈中引入其他工具。

此模式概述了一种使用 Terraform 部署 Amazon Redshift 数据库资源（包括表、架构、视图和存储过程）的方法。该模式区分了两种类型的 SQL 查询：

- 不可重复查询 — 这些查询在最初的 Amazon Redshift 部署期间执行一次，以建立基本的数据库组件。
- 可重复查询 — 这些查询是不可变的，可以在不影响数据库的情况下重新运行。该解决方案使用 Terraform 来监视可重复查询中的变化并相应地应用它们。

有关更多详细信息，请参阅 “其他信息” 中的 “解决方案演练”。

先决条件和限制

先决条件

您必须激活 AWS 账户 并在部署计算机上安装以下内容：

- AWS Command Line Interface (AWS CLI)
- AWS CLI 配置有 Amazon Redshift 读/写权限的个人资料
- Terraform 版本 1.6.2 或更高版本
- Python3
- Boto3

限制

- 此解决方案支持单个 Amazon Redshift 数据库，因为 Terraform 只允许在创建集群期间创建一个数据库。

- 此模式不包括在应用可重复查询之前对其进行验证的测试。我们建议您采用此类测试以增强可靠性。
- 为了说明解决方案，此模式提供了一个使用本地 Terraform 状态`redshift.tf`文件的示例文件。但是，对于生产环境，我们强烈建议您使用带有锁定机制的远程状态文件，以增强稳定性和协作性。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

产品版本

该解决方案是在亚马逊 Redshift 补丁 179 上开发和测试的。

代码存储库

此模式的代码可在 GitHub amazon-redshift-sql-deploy-terraform 存储库中找到。

架构

下图说明了 Terraform 如何通过处理不可重复和可重复的 SQL 查询来管理 Amazon Redshift 数据库资源。

图中显示以下步骤：

1. 在最初的 Amazon Redshift 集群部署期间，Terraform 会应用不可重复的 SQL 查询。
2. 开发人员提交对可重复的 SQL 查询的更改。
3. Terraform 监视可重复的 SQL 查询中的变化。
4. Terraform 将可重复的 SQL 查询应用于亚马逊 Redshift 数据库。

这种模式提供的解决方案是基于适用于 Amazon Redshift 的 Terraform 模块构建的。Terraform 模块预配置 Amazon Redshift 集群和数据库。为了增强该模块，我们使用了`terraform_data`资源，这些资源调用自定义 Python 脚本来使用 Amazon ExecuteStatementRedshift API 操作执行 SQL 查询。因此，该模块可以执行以下操作：

- 置备数据库后，使用 SQL 查询部署任意数量的数据库资源。
- 持续监控可重复的 SQL 查询中的更改，并使用 Terraform 应用这些更改。

有关更多详细信息，请参阅 “其他信息” 中的 “解决方案演练”。

工具

AWS 服务

- Amazon Redshift 是一项完全托管的 PB 级数据仓库服务。 AWS Cloud

其他工具

- Terraform 是一款基础设施即代码 (IaC) 工具 HashiCorp ，可帮助您创建和管理云和本地资源。
- Python 是一种通用的编程语言，在这种模式中用于执行 SQL 查询。

最佳实践

- 亚马逊 Redshift 最佳实践
- 使用亚马逊 Redshift 数据 API 与亚马逊 Redshift 集群进行交互

操作说明

使用 Terraform 部署解决方案

Task	描述	所需技能
克隆存储库。	要克隆包含用于配置 Amazon Redshift 集群的 Terraform 代码的 Git 存储库，请使用以下命令。 `git clone https://github.com/aws-samples/amazon-redshift-sql-deploy-terraform.git`	DevOps 工程师
更新 Terraform 变量。	要根据您的特定要求自定义 Amazon Redshift 集群部署，请更新文件中的以	DevOps 工程师

Task	描述	所需技能
	下terraform.tfvars 参数。 region = "<AWS_REGION>" cluster_identifier = "<REDSHIFT_CLUSTER_IDENTIFIER>" node_type = "<REDSHIFT_NODE_TYPE>" number_of_nodes = "<REDSHIFT_NODE_COUNT>" database_name = "<REDSHIFT_DB_NAME>" subnet_ids = "<REDSHIFT_SUBNET_IDS>" vpc_security_group_ids = "<REDSHIFT_SECURITY_GROUP_IDS>" run_nonrepeatable_queries = true run_repeatable_queries = true sql_path_bootstrap = "<BOOTSTRAP_SQLS_PATH>" sql_path_nonrepeatable = "<NON-REPEATABLE_SQLS_PATH>" sql_path_repeatable = "<REPEATABLE_SQLS_PATH>" sql_path_finalize = "<FINALIZE_SQLS_PATH>"	

Task	描述	所需技能
	`create_random_password = false` `master_username = "<REDSHIFT_MASTER_USERNAME>"`	

使用 Terraform 部署资源。	1. 要为部署过程做准备，请使用以下命令在克隆的存储库中初始化 Terraform。
	`terraform init`
	2. 要预览 Terraform 将应用于基础架构的更改，请使用以下命令创建执行计划。
	`terraform plan -var-file terraform.tfvars`
	3. 要配置 Amazon Redshift 集群和相关资源，请使用以下命令应用 Terraform 执行计划。
	`terraform apply -var-file terraform.tfvars`
		DevOps 工程师

Task	描述	所需技能

（可选）执行其他 SQL 查询。	示例存储库提供了几个 SQL 查询用于演示。要执行您自己的 SQL 查询，请将其添加到以下文件夹：
	`/bootstrap`
	`/nonrepeatable`
	`/repeatable`
	`/finalize`	

监控 SQL 语句的执行情况

Task	描述	所需技能
监控 SQL 语句的部署。	您可以监控 Amazon Redshift 集群的 SQL 执行结果。有关显示 SQL 执行失败和成功执行的输出示例，请参阅 “其他信息” 中的 SQL 语句示例。	数据库管理员、工程师 DevOps
清理资源。	要删除 Terraform 部署的所有资源，请运行以下命令。 `terraform destroy`	DevOps 工程师

验证结果

Task	描述	所需技能
验证 Amazon Redshift 集群中的数据。	1. 登录并打开亚马逊 Redshift 控制台。 AWS Management Console	DBA、AWS DevOps

Task	描述	所需技能
	2. 在导航菜单上，选择集群。在列表中选择相关的集群名称。 3. 按照亚马逊 Redshift 文档中使用亚马逊 Redshift 查询编辑器 v2 查询数据库中的说明进行操作。	

相关资源

AWS 文档

- 亚马逊 Redshift 预配置的集群
- 亚马逊 Redshift 数据 API 问题疑难解答

其他资源

- 命令：应用（Terraform 文档）

其他信息

解决方案演练

要使用该解决方案，您必须以特定的方式组织您的 Amazon Redshift SQL 查询。所有 SQL 查询都必须存储在`.sql`扩展名为的文件中。

在此模式提供的代码示例中，SQL 查询按以下文件夹结构进行组织。您可以修改代码（`sql-queries.tf`和`sql-queries.py`），使其适用于适合您独特用例的任何结构。

```
/bootstrap
     |- Any # of files
     |- Any # of sub-folders
/nonrepeatable
     |- Any # of files
     |- Any # of sub-folders
/repeatable
```

```
     /udf
          |- Any # of files
          |- Any # of sub-folders
     /table
          |- Any # of files
          |- Any # of sub-folders
     /view
          |- Any # of files
          |- Any # of sub-folders
     /stored-procedure
          |- Any # of files
          |- Any # of sub-folders
/finalize
     |- Any # of files
     |- Any # of sub-folders
```

鉴于上述文件夹结构，在 Amazon Redshift 集群部署期间，Terraform 按以下顺序执行查询：

1. `/bootstrap`
2. `/nonrepeatable`
3. `/repeatable`
4. `/finalize`

该`/repeatable`文件夹包含四个子文件夹：`/udf`、`/table`/`view`、和。`/stored-procedure`这些子文件夹指示 Terraform 执行 SQL 查询的顺序。

执行 SQL 查询的 Python 脚本是`sql-queries.py`。首先，该脚本读取特定源目录的所有文件和子文件夹，例如`sql_path_bootstrap`参数。然后，该脚本通过调用 Amazon ExecuteStatementRedshift API 操作来执行查询。一个文件中可能有一个或多个 SQL 查询。以下代码片段显示了 Python 函数，该函数针对亚马逊 Redshift 集群执行存储在文件中的 SQL 语句。

```
def execute_sql_statement(filename, cluster_id, db_name, secret_arn, aws_region):
    """Execute SQL statements in a file"""
    redshift_client = boto3.client(
        'redshift-data', region_name=aws_region)
    contents = get_contents_from_file(filename),
    response = redshift_client.execute_statement(
        Sql=contents[0],
        ClusterIdentifier=cluster_id,
        Database=db_name,
```

```
        WithEvent=True,
        StatementName=filename,
        SecretArn=secret_arn
    )
    ...
```

Terraform 脚本sql-queries.tf创建用于调用该脚本的 terraform_data 资源。sql-queries.py四个文件夹中的每一个都有一个terraform_data资源：/bootstrap、/nonrepeatable/repeatable、和/finalize。以下代码片段显示了在/bootstrap文件夹中执行 SQL 查询的terraform_data资源。

```
locals {
  program               = "${path.module}/sql-queries.py"
  redshift_cluster_name = try(aws_redshift_cluster.this[0].id, null)
}

resource "terraform_data" "run_bootstrap_queries" {
  count      = var.create && var.run_nonrepeatable_queries && (var.sql_path_bootstrap !
= "") && (var.snapshot_identifier == null) ? 1 : 0
  depends_on = [aws_redshift_cluster.this[0]]

  provisioner "local-exec" {
    command = "python3 ${local.program} ${var.sql_path_bootstrap}
 ${local.redshift_cluster_name} ${var.database_name} ${var.redshift_secret_arn}
 ${local.aws_region}"
  }
}
```

您可以使用以下变量控制是否运行这些查询。如果您不想在sql_path_bootstrap、、或sql_path_finalize中运行查询 sql_path_nonrepeatablesql_path_repeatable，请将其值设置为""。

```
  run_nonrepeatable_queries = true
  run_repeatable_queries    = true
  sql_path_bootstrap        = "src/redshift/bootstrap"
  sql_path_nonrepeatable    = "src/redshift/nonrepeatable"
  sql_path_repeatable       = "src/redshift/repeatable"
  sql_path_finalize         = "src/redshift/finalize"
```

运行时terraform apply，无论脚本的结果如何，Terraform 都会考虑脚本完成后添加的terraform_data资源。如果某些 SQL 查询失败，而您想重新运行它们，则可以手动

将该资源从 Terraform 状态中移除，然后重新运行。`terraform apply`例如，以下命令将该`run_bootstrap_queries`资源从 Terraform 状态中移除。

```
terraform state rm module.redshift.terraform_data.run_bootstrap_queries[0]
```

以下代码示例显示了`run_repeatable_queries`资源如何使用 s ha256 哈希值监视`repeatable`文件夹中的变化。如果更新了文件夹中的任何文件，Terraform 会将整个目录标记为更新。然后，Terraform 在下一次运行目录中的查询。`terraform apply`

```
resource "terraform_data" "run_repeatable_queries" {
  count      = var.create_redshift && var.run_repeatable_queries &&
 (var.sql_path_repeatable != "") ? 1 : 0
  depends_on = [terraform_data.run_nonrepeatable_queries]

  # Continuously monitor and apply changes in the repeatable folder
  triggers_replace = {
    dir_sha256 = sha256(join("", [for f in fileset("${var.sql_path_repeatable}",
 "**") : filesha256("${var.sql_path_repeatable}/${f}")]))
  }

  provisioner "local-exec" {
    command = "python3 ${local.sql_queries} ${var.sql_path_repeatable}
 ${local.redshift_cluster_name} ${var.database_name} ${var.redshift_secret_arn}"
  }
}
```

要完善代码，您可以实现一种机制，以检测并仅对`repeatable`文件夹中已更新的文件应用更改，而不是不分青红皂白地将更改应用于所有文件。

SQL 语句示例

以下输出显示了 SQL 执行失败以及错误消息。

```
module.redshift.terraform_data.run_nonrepeatable_queries[0] (local-exec): Executing:
 ["/bin/sh" "-c" "python3 modules/redshift/sql-queries.py src/redshift/nonrepeatable
 testcluster-1 db1 arn:aws:secretsmanager:us-east-1:XXXXXXXXXXXX:secret:/redshift/
master_user/password-8RapGH us-east-1"]
module.redshift.terraform_data.run_nonrepeatable_queries[0] (local-exec):
 ----------------------------------------------------------------------
module.redshift.terraform_data.run_nonrepeatable_queries[0] (local-exec): src/redshift/
nonrepeatable/table/admin/admin.application_family.sql
module.redshift.terraform_data.run_nonrepeatable_queries[0] (local-exec):
 ----------------------------------------------------------------------
```

```
module.redshift.terraform_data.run_nonrepeatable_queries[0] (local-exec): Status:
 FAILED
module.redshift.terraform_data.run_nonrepeatable_queries[0] (local-exec): SQL execution
 failed.
module.redshift.terraform_data.run_nonrepeatable_queries[0] (local-exec): Error
 message: ERROR: syntax error at or near ")"
module.redshift.terraform_data.run_nonrepeatable_queries[0] (local-exec):   Position:
 244
module.redshift.terraform_data.run_nonrepeatable_queries[0]: Creation complete after 3s
 [id=ee50ba6c-11ae-5b64-7e2f-86fd8caa8b76]
```

以下输出显示成功执行 SQL。

```
module.redshift.terraform_data.run_bootstrap_queries[0]: Provisioning with 'local-
exec'...
module.redshift.terraform_data.run_bootstrap_queries[0] (local-exec): Executing:
 ["/bin/sh" "-c" "python3 modules/redshift/sql-queries.py src/redshift/bootstrap
 testcluster-1 db1 arn:aws:secretsmanager:us-east-1:XXXXXXXXXXXX:secret:/redshift/
master_user/password-8RapGH us-east-1"]
module.redshift.terraform_data.run_bootstrap_queries[0] (local-exec):
 ----------------------------------------------------------------------
module.redshift.terraform_data.run_bootstrap_queries[0] (local-exec): src/redshift/
bootstrap/db.sql
module.redshift.terraform_data.run_bootstrap_queries[0] (local-exec):
 ----------------------------------------------------------------------
module.redshift.terraform_data.run_bootstrap_queries[0] (local-exec): Status: FINISHED
module.redshift.terraform_data.run_bootstrap_queries[0] (local-exec): SQL execution
 successful.
module.redshift.terraform_data.run_bootstrap_queries[0]: Creation complete after 2s
 [id=d565ef6d-be86-8afd-8e90-111e5ea4a1be]
```

将 AWS Organizations 中整个组织的 AWS Backup 报告导出为 CSV 文件

由 Aromal Raj Jayarajan (AWS) 和 Purushotham G K (AWS) 编写

摘要

此模式演示如何将 AWS Organizations 中整个组织的 AWS Backup 作业报告导出为 CSV 文件。该解决方案使用 AWS Lambda 和 Amazon EventBridge 根据状态对 AWS Backup 任务报告进行分类，这有助于配置基于状态的自动化。

AWS Backup 帮助组织集中管理和自动化跨 Amazon Web Services、云中和本地的数据保护。但是，对于在 AWS Organizations 中配置的 AWS Backup 作业，合并报告仅在每个组织管理账户的 AWS 管理控制台 中可用。将此报告置于管理账户之外可以减少审核所需工作量并扩大自动化、通知和警报的范围。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- AWS Organizations 中、至少包含一个管理账户和一个成员账户的活跃 组织
- AWS Organizations 组织级的 AWS Backup 配置（若要获取更多信息，请参见 AWS Blog 中的使用 AWS Backup 跨 Amazon Web Services 大规模自动化集中备份）
- Git，已在本地计算机上安装并配置

限制

此模式中提供的解决方案可识别仅为 AWS Backup 作业配置的 AWS 资源。该报告无法识别未配置为通过 AWS Backup 备份的 AWS 资源。

架构

目标技术堆栈

- AWS Backup
- AWS CloudFormation
- Amazon EventBridge
- AWS Lambda

- AWS Security Token Service (AWS STS)
- Amazon Simple Storage Service（Amazon S3）
- AWS Identity and Access Management (IAM)

目标架构

下图显示了将 AWS Organizations 中整个组织的 AWS Backup 作业报告导出为 CSV 文件的示例工作流。

图表显示了以下工作流：

1. 计划 EventBridge 事件规则调用成员（报告）AWS 账户中的 Lambda 函数。
2. 然后，Lambda 函数使用 AWS STS 承担拥有连接管理账户所需权限的 IAM 角色。
3. Lambda 函数执行以下操作：
   - 从 AWS Backup 服务请求合并的 AWS Backup 作业报告
   - 根据 AWS Backup 作业状态对结果进行分类
   - 将响应转换到 CSV 格式文件
   - 将结果上传到报告账户中的 Amazon S3 存储桶，该存储桶位于根据创建日期标记的文件夹内

工具

工具

- AWS Backup 是一项完全托管式服务，帮助您在云中以及在本地集中管理和自动执行各种 Amazon Web Services 中的数据保护。
- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如，AWS Lambda 函数、使用 API 目标的 HTTP 调用端点或其他 Amazon Web Services account 中的事件总线。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。

- Amazon Simple Storage Service (Amazon S3)是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

代码

此模式的代码可在 GitHub aws-backup-report-generator存储库中找到。

最佳实践

- Amazon S3 安全最佳实践（Amazon S3 用户指南）
- AWS Lambda 函数使用最佳实践（AWS Lambda 开发人员指南）
- 管理账户的最佳实践（AWS Organizations 用户指南）

操作说明

部署解决方案组件

Task	描述	所需技能
克隆 GitHub 存储库。	在终端窗口中运行以下命令来克隆 GitHub aws-backup-report-generator存储库： `git clone https://github.com/aws-samples/aws-backup-report-generator.git` 有关更多信息，请参阅 GitHub 文档中的克隆存储库。	AWS DevOps， DevOps 工程师
在成员（报告）Amazon Web Services account 中部署解决方案组件。	1. 在成员（报告）账户中，登录 AWS 管理控制台，然后打开CloudFormation 控制台。 2. 选择创建堆栈，然后选择使用新资源（标准）。	DevOps 工程师，AWS DevOps

Task	描述	所需技能
	3. 在创建堆栈页面的指定模板部分，选择上传模板文件。 4. 选择 Choose file (选择文件)。然后，导航到本地工作站上克隆 GitHub 存储库的根文件夹，然后选择 template-reporting.yaml。 5. 选择打开，然后选择下一步。 6. 在指定堆栈详细信息页面上，在堆栈名称中输入 CloudFormation 堆栈的名称。 7. 对于 ManagementAccountID，请在 AWS Organizations 中输入贵组织管理账户的 AWS 账户 ID。 8. 请选择 Next（下一步）。 9. 在配置堆栈选项页面上，请选择下一步。 10. 在审核页面，选中复选框以确认您已查看配置。 11. 选择创建堆栈。在成员（报告）账户中部署解决方案组件时，堆栈会显示CREATE_COMPLETE状态。	

测试解决方案

Task	描述	所需技能
在测试之前，请确保该 EventBridge 规则已运行。	确保 EventBridge 规则通过等待至少 24 小时或在 CloudFormation 模板的 template- reporting.yml 文件中增加报告频率来运行。 增加报告频率 1. 在克隆的存储库中打开 template-reporting.yml 文件。 2. 在逻辑 ID 为 “LambdaSchedule” 的事件规则中，找到 “ScheduleExpression”。 3. 编辑 “ScheduleExpression” 键，使其包含有效的 cron 表达式。例如，以下 cron 表达式会将事件规则安排为每五分钟运行一次：“`cron(* /5 * * * *)`”	AWS DevOps， DevOps 工程师
检查 Amazon S3 存储桶，以获取生成的报告。	1. 在成员（报告）账户中，登录 AWS 管理控制台，然后打开CloudFormation 控制台。 2. 在堆栈列表中，选择您在创建的堆栈名称。选择资源选项卡。 3. 在资源窗格的逻辑 ID 列中，找到 BackupReportS3Bucket。然后，通过选择该逻辑 ID 旁边的物理 ID 列链	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	接，在新选项卡中打开关联的 Amazon S3 存储桶。 4. 确保存储桶包含按以下格式生成的报告：BackupReports///BackupReport---.csv<yyyy><mm><dd><BACKUP JOB STATUS><dd><Mon><yyyy>	

清除资源

Task	描述	所需技能
从成员（报告）账户中删除解决方案组件。	1. 在成员（报告）账户中，打开解决方案的 Amazon S3 存储桶。有关说明，请参阅此模式的测试解决方案部分的检查 S3 存储桶以获取生成的报告说明中的第 2 至 4 步。 2. 删除存储桶中的内容并清空桶。有关说明，请参阅 Amazon S3 User 用户指南中的清空存储桶。 3. 在成员（报告）账户中，登录 AWS 管理控制台，然后打开CloudFormation 控制台。 4. 在堆栈窗格，选中您创建的堆栈名称旁边的复选框。然后选择 Delete(删除)。	AWS DevOps， DevOps 工程师

Task	描述	所需技能
从管理账户中删除解决方案组件。	1. 在管理账户中，登录 AWS 管理控制台，然后打开CloudFormation 控制台。 2. 在堆栈窗格，选中您创建的堆栈名称旁边的复选框。然后选择 Delete(删除)。	AWS DevOps， DevOps 工程师

相关资源

- 教程：将 AWS Lambda 用于计划事件（A WS Lambda 文档）
- 创建计划事件以运行 AWS Lambda 函数（用于 JavaScript 文档的 AWS 开发工具包）
- IAM 教程：使用 IAM 角色在 AWS 账户之间委派访问权限（IAM 文档）
- AWS Organizations 术语和概念（AWS Organizations 文档）
- 使用 AWS Backup 控制台创建报告计划（AWS Backup 文档）
- 创建审计报告（AWS Backup 文档）
- 创建按需报告（AWS Backup 文档）
- 什么是 AWS Backup？（AWS Backup 文档）
- 使用 AWS Backup 在 AWS 服务中大规模自动进行集中备份（AWS 博客文章）

将亚马逊 EC2 实例列表的标签导出到 CSV 文件

创建者：Sida Ju (AWS) 和 Pac Joonhyun (AWS)

摘要

此模式显示如何以编程方式将亚马逊弹性计算云 (Amazon EC2) 实例列表的标签导出到 CSV 文件。

通过使用提供的示例 Python 脚本，您可以缩短按特定标签对您的 Amazon EC2 实例进行审查和分类所需的时间。例如，您可使用该脚本来快速识别您的安全团队已标记为需要进行软件更新的实例列表并对其进行分类。

先决条件和限制

先决条件

- 已安装并配置 Python 3
- AWS 命令行界面（AWS CLI）已安装并配置

限制

此模式中提供的示例 Python 脚本只能根据以下属性搜索 Amazon EC2 实例：

- 实例 IDs
- 私有 IPv4 地址
- 公共 IPv4 地址

工具

- Python 是通用的计算机编程语言。
- virtualenv 可以帮助您创建隔离的 Python 环境。
- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。

代码存储库

此模式的 Python 脚本示例，可在 GitHub 搜索 ec2 存储库中找到。instances-export-tags

操作说明

安装并配置先决条件

Task	描述	所需技能
克隆 GitHub 存储库。	**Note** 如果您在运行 AWS CLI 命令时收到错误，请确保您使用的是最新的 AWS CLI 版本。 在终端 GitHub 窗口中运行以下 Git 命令来克隆 search-ec2 instances-export-tags 存储库： `git clone https://github.com/aws-samples/search-ec2-instances-export-tags.git`	DevOps 工程师

安装和激活 virtualenv。	1. 通过运行以下命令安装 virtualenv：
	
	`python3 -m pip install virtualenv`
	
	2. 通过运行以下命令创建新的虚拟环境：
	
	`python3 -m venv env`
	
	3. 通过运行以下命令激活新的虚拟环境：
		DevOps 工程师

Task	描述	所需技能
	`source env/bin/activate` 有关更多信息，请参阅 virtualenv 用户指南。	

安装依赖项。	1. 通过在终端中运行以下命令打开代码目录：
	`cd search-ec2-instances-export-tags`
	2. 通过运行以下 pip 命令安装 `requirements.txt` 文件：
	`pip3 install -r requirements.txt`
		DevOps 工程师

配置 AWS 命名配置文件。	如果您尚未配置一个 AWS 命名配置文件，其中包含运行脚本所需凭证。要创建命名配置文件，请运行 aws configure 命令。 有关更多信息，请参阅 AWS CLI 文档中的使用命名配置文件。	DevOps 工程师

配置并运行 Python 脚本

Task	描述	所需技能

创建输入文件。	创建一个输入文件，其中包含您希望脚本搜索和导出标签的 Amazon EC2 实例列表。您可以列出实例 IDs、私有 IPv4 地址或公有 IPv4 地址。
	
	Important
	确保每个 Amazon EC2 实例在输入文件中单独列出。
	
	输入文件示例
	
	`i-0547c351bdfe85b9f`
	`54.157.194.156`
	`172.31.85.33`
	`54.165.198.144`
	`i-0b6223b5914111a4b`
	`172.31.85.44`
	`54.165.198.145`
	`172.31.80.219`
	`172.31.94.199`
		DevOps 工程师

运行 Python 脚本。	通过在终端中运行以下命令运行脚本： `python search_instances.py -i INPUTFILE -o OUTPUTFILE -r REGION [-p PROFILE]`	DevOps 工程师

Task	描述	所需技能
	Note INPUTFILE 替换为输入文件的名称。将 OUTPUTFILE 替换为您要为 CSV 输出文件提供的名称。REGION替换为您的 Amazon EC2 资源所在的 AWS 区域。如果您使用的是 AWS 命名的配置文件，请将 PROFILE 替换为您正在使用的命名配置文件。 要获取支持的参数及其描述的列表，请运行以下命令： `python search_instances.py -h` 要了解更多信息并查看输出文件示例，请参阅 s GitHub earch-instances-export-tags ec2 存储库中的README.md 文件。	

相关资源

- 配置 AWS CLI（AWS CLI 用户指南）

使用 Troposphere 生成包含 AWS Config 托管规则的 AWS CloudFormation 模板

创建者：Lucas Nation (AWS) 和 Freddie Wilson (AWS)

摘要

许多组织使用 AWS Config 托管规则来评估其 Amazon Web Services (AWS) 资源是否符合常见的最佳实践。但是，维护这些规则可能很耗时，这种模式可以帮助您利用 Python 库 Troposphere 生成和管理 AWS Config 托管规则。

该模式使用 Python 脚本将包含 AWS 托管规则的 Microsoft Excel 电子表格转换为 AWS CloudFormation 模板，从而帮助您管理 AWS Config 托管规则。Troposphere 充当基础设施即代码（IaC），这意味着您可以使用托管规则更新 Excel 电子表格，而不是使用 JSON 或 YAML 格式的文件。然后，您可以使用该模板启动一个 AWS CloudFormation 堆栈，该堆栈在您的 AWS 账户中创建和更新托管规则。

AWS CloudFormation 模板使用 Excel 电子表格定义每个 AWS Config 托管规则，并帮助您避免在 AWS 管理控制台中手动创建单个规则。该脚本将每个托管规则的参数默认为一个空字典，作用域的`ComplianceResourceTypes`默认值为`THE_RULE_IDENTIFIER.template file`。 有关规则标识符的更多信息，请参阅 AWS Config 文档中的使用 AWS CloudFormation 模板创建 AWS Config 托管规则。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 熟悉使用 AWS CloudFormation 模板创建 AWS Config 托管规则。有关这方面的更多信息，请参阅 AWS Config 文档中的使用 AWS CloudFormation 模板创建 AWS Config 托管规则。
- Python 3，已安装并配置。有关这方面的更多信息，请参阅 Python 文档。
- 现有的集成开发环境 (IDE)。
- 在示例 `excel_config_rules.xlsx` Excel 电子表格（附后OUs）的列中标识您的组织单位 ()。

操作说明

自定义和配置 AWS Config 托管规则

Task	描述	所需技能
更新 Excel 电子表格示例。	下载示例 `excel_config_rules.xlsx` Excel 电子表格（附件），并将其标记为要使用的 AWS Config 托管规则 `Implemented` 。 标记为的规则`Implemented` 将添加到 AWS CloudFormation 模板中。	开发人员
（可选）使用 AWS Config 规则参数更新 config_rules_params.json 文件。	某些 AWS Config 托管规则需要参数，应使用 `--param-file` 选项将其作为 JSON 文件传递到 Python 脚本。例如，`access-keys-rotated` 托管规则使用以下 `maxAccessKeyAge` 参数： `{ "access-keys-rotated": { "InputParameters": { "maxAccessKeyAge": 90 } } }` 在此示例参数中，`maxAccessKeyAge` 设置为 90 天。该脚本读取参数文件并添加	开发人员

Task	描述	所需技能
	它找到的任何 `InputParameters` 。	
（可选）使用 AWS Config 更新 config_rules_params.json 文件。 ComplianceResourceTypes	默认情况下，Python 脚本会从 AWS 定义的模板中检索 `ComplianceResourceTypes` 。如果您想覆盖特定 AWS Config 托管规则范围，则需要使用 `--param-file` 选项将其作为 JSON 文件传递给 Python 脚本。 例如，以下示例代码显示了如何将 `ec2-volume-inuse-check` 的 `ComplianceResourceTypes` 设置为 `["AWS::EC2::Volume"]` 列表： `{ "ec2-volume-inuse-check": { "Scope": { "ComplianceResourceTypes": [ "AWS::EC2::Volume" ] } } }`	开发人员

运行 Python 脚本

Task	描述	所需技能
从 requirements.txt 文件处安装 pip 程序包。	下载 `requirements.txt` 文件（附件），然后在 IDE 中运行以下命令来安装 Python 程序包： `pip3 install -r requirements.txt`	开发人员
运行 Python 脚本。	1. 将 `aws_config_rules.py` 文件（附件）下载到本地计算机上。 2. **Note** 运行- `python3 aws_config_rules.py --ou <OU_NAME>` 命令。:`--ou`定义要在 Excel 电子表格中选择哪个 OU 列。 您还可以添加以下可选参数： • `--config-rule-option` – 定义要从 Excel 电子表格中选择的规则。默认值为 `Implemented` 参数。 • `--excel-file` – Excel 电子表格的路径。默认为 `aws_config_rules.xlsx` 。	开发人员

Task	描述	所需技能
	• `--param-file` – 参数 JSON 文件的路径。默认为 `config_rules_params.json` 。 • `--max-execution-frequency` – 定义评估 AWS Config 托管规则的频率。选项是 `One_Hour`、`Three_Hours` 、`Six_Hours` 、`Twelve_Hours` 或 `TwentyFour_Hours` 。默认为 `TwentyFour_Hours` 。	

部署 AWS Config 托管规则

Task	描述	所需技能
启动 AWS CloudFormation 堆栈。	1. 登录 AWS 管理控制台，打开 AWS CloudFormation 控制台，然后选择创建堆栈。 2. 在指定模板页面上，选择上传模板文件，然后上传您的 AWS CloudFormation 模板。 3. 指定堆栈名称，然后选择下一步。 4. 指定标签，然后选择下一步。 5. 选择创建堆栈。	开发人员

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

为 SageMaker 笔记本实例提供对另一个 AWS 账户中 CodeCommit 存储库的临时访问权限

创建者：Helge Aufderheide (AWS)

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

此模式演示如何授予 Amazon SageMaker 笔记本实例和用户临时访问其他 AWS 账户中的 AWS CodeCommit 存储库的权限。此模式还显示了如何为每个实体可以在每个存储库上执行的特定操作授予精细权限。

组织通常将存储 CodeCommit 库存储在与托管其开发环境的账户不同的 AWS 账户中。这种多账户设置有助于控制对存储库的访问权限并降低意外删除存储库的风险。要授予这些跨账户权限，最好做法是使用 AWS Identity and Acccess Management (IAM) 角色。然后，每个 Amazon Web Services account 中预定义的 IAM 身份可以临时代入这些角色，从而在各账户之间创建受控的信任链。

> **Note**
>
> 您可以应用类似的程序向其他 IAM 身份授予对 CodeCommit 存储库的跨账户访问权限。有关更多信息，请参阅 AWS CodeCommit 用户指南中的使用角色配置对 AWS CodeCommit 存储库的跨账户访问权限。

先决条件和限制

先决条件

- 具有 CodeCommit 存储库的活跃 AWS 账户（账户 A）
- 带有 SageMaker 笔记本实例的第二个活跃 AWS 账户（账户 B）
- 一个具有在账户 A 中创建和修改 IAM 角色的足够权限的 AWS 用户
- 第二个具有在账户 B 中创建和修改 IAM 角色的足够权限的 AWS 用户

架构

下图显示了向 SageMaker 笔记本实例和一个 AWS 账户中的用户授予跨账户访问 CodeCommit 存储库权限的示例工作流程：

图表显示了以下工作流：

1. 账户 B 中的 AWS 用户角色和 SageMaker 笔记本实例角色采用已命名的个人资料。
2. 指定配置文件的权限策略在账户 A 中指定 CodeCommit 访问角色，然后该配置文件将担任该角色。
3. 账户 A 中的 CodeCommit 访问角色的信任策略允许账户 B 中的指定配置文件担任 CodeCommit 访问角色。
4. 账户 A 中 CodeCommit 存储库的 IAM 权限策略允许 CodeCommit 访问角色访问 CodeCommit 存储库。

技术堆栈

- CodeCommit
- Git
- IAM
- pip
- SageMaker

工具

- AWS CodeCommit 是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- Git 是分布式版本控制系统，用于追踪软件开发期间源代码的更改。
- git-remote-codecommit是一个通过扩展 Git 来帮助您从 CodeCommit 存储库中推送和提取代码的实用工具。
- pip 是 Python 的软件包安装程序。您可以使用 pip 来安装来自 Python 软件包索引和其他索引中的软件包。

最佳实践

在使用 IAM policy设置权限时，请仅授予执行任务所需许可。有关更多信息，请参阅 IAM 文档中的应用最低权限许可。

在实施此模式时，请务必执行以下操作：

- 确认 IAM 原则仅具有在每个存储库中执行特定必要操作所需权限。例如，建议允许经批准的 IAM 原则将更改推送和合并到特定的存储库分支，但只能请求合并到受保护的分支。
- 确认根据每个项目各自的角色和职责，为 IAM 角色分配不同的 IAM 原则。例如，开发人员将拥有与发布管理员或 AWS 管理员不同的访问权限。

操作说明

配置 IAM 角色

Task	描述	所需技能
配置 CodeCommit 访问角色和权限策略。	**Note** 要自动执行本长篇故事中记录的手动设置过程，您可以使用 AWS CloudFormation 模板。 在包含 CodeCommit 存储库的账户（账户 A）中，执行以下操作： 1. 创建可由账户 B 中的 SageMaker 笔记本实例角色代入的 IAM 角色。 2. 创建 IAM policy，该策略授予存储库访问权限；然后将该策略附加到该角色。仅出于测试目的，请选择 AWSCodeCommitPowerUserAWS 托管策略。此策略授予除删除资源之外的所有CodeCommit 权限。	常规 AWS、AWS DevOps

Task	描述	所需技能
	3. 修改角色的信任策略，将账户 B 列为受信任实体。 **Important** 在将此设置移至生产环境之前，最佳做法是自己编写应用最低权限的 IAM 策略。有关更多信息，请参阅此模式的其他信息部分。	

Task	描述	所需技能
向账户 B 中的 SageMaker 笔记本实例角色授予在账户 A 中担任 CodeCommit 访问角色的权限。	在包含 SageMaker 笔记本实例的 IAM 角色的账户（账户 B）中，执行以下操作： 1. 创建 IAM 策略，允许 IAM 角色或用户代入账户 A 中的 CodeCommit 访问角色。 允许 IAM 角色或用户代入跨账户角色的 IAM 权限策略示例 `{ "Version": "2012-10-17", "Statement": [ { "Sid": "VisualEditor0", "Effect": "Allow", "Action": "sts:AssumeRole", "Resource": "arn:aws:iam:::accountA_ID:role/accountArole_ID" } }` 2. 将该策略附加到您的 SageMaker 笔记本实例在账户 B 中的角色。 3. 让账户 B 中 SageMaker 笔记本实例的角色代入账户 A 中的 CodeCommit 访问角色。	常规 AWS、AWS DevOps

Task	描述	所需技能
	Note 要查看存储库的 Amazon 资源名称 (ARN)，请参阅 AW S CodeCommit 用户指南中的查看 CodeCommit 存储库详情。	

在账户 B 中设置您的 SageMaker 笔记本实例

Task	描述	所需技能
在 AWS SageMaker 笔记本实例上设置用户配置文件以代入账户 A 中的角色。	**Important** 确保您安装了最新版本的 AWS 命令行界面 (AWS CLI)。 在包含 SageMaker 笔记本实例的账户（账户 B）中，执行以下操作： 1. 登录 AWS 管理控制台并打开SageMaker 控制台。 2. 访问您的 SageMaker 笔记本实例。Jupyter 界面打开。 3. 选择新建，然后选择终端。在 Jupyter 环境中会打开一个新的终端窗口。	常规 AWS、AWS DevOps

Task	描述	所需技能

	4. 导航到 SageMaker 笔记本实例的 ~/.aws/config 文件。然后，通过输入以下语句将用户配置文件添加到文件中：
	
	`------.aws/config-------------[profile remoterepouser]role_arn = arn:aws:iam::<ID of Account A>:role/<rolename>role_session_name = remoteaccesssessionregion = eu-west-1credential_source = Ec2InstanceMetadata----------`
	`----------`	

安装该 git-remote-codecommit 实用程序。	按照 AWS CodeCommit 用户指南中的步骤 2：安装 git-remote-codecommit中的说明进行操作。	数据科学家

访问存储库

Task	描述	所需技能
使用 Git 命令访问 CodeCommit 存储库或 SageMaker。	要使用 Git，请执行以下操作 在账户 B 中扮演 SageMaker 笔记本实例角色的 IAM 委托人现在可以运行 Git 命令来访问账户 A 中的 CodeCommi	Git，bash 控制台

Task	描述	所需技能
	t 存储库。例如，用户可以运行`git clone`、`git pull`和`git push`之类的命令。 有关说明，请参阅 AWS CodeCommit 用户指南中的 Connect 到 AWS CodeCommit 存储库。 有关如何使用 Git 的信息 CodeCommit，请参阅 AWS CodeCommit 用户指南 CodeCommit中的 AWS 入门。 要使用 SageMaker 要从 SageMaker 控制台使用 Git，必须允许 Git 从 CodeCommit 仓库中检索证书。有关说明，请参阅 SageMaker 文档中的将不同 AWS 账户中的 CodeCommit 存储库与笔记本实例关联。	

相关资源

- 使用角色配置对 AWS CodeCommit 存储库的跨账户访问权限（AWS CodeCommit 文档）
- IAM 教程：使用 IAM 角色在 AWS 账户之间委派访问权限（IAM 文档）

其他信息

将 CodeCommit 权限限制为特定操作

要限制 IAM 委托人可以在 CodeCommit 存储库中执行的操作，请修改 CodeCommit 访问策略中允许的操作。

有关 CodeCommit API 操作的更多信息，请参阅 AWS CodeCommit 用户指南中的CodeCommit 权限参考。

> **Note**
>
> 您还可以编辑 AWSCodeCommitPowerUserAWS 托管策略以适应您的使用案例。

限制对特定仓库的 CodeCommit 权限

要创建只有特定用户才能访问多个代码存储库的多租户环境，请执行以下操作：

1. 在账户 A 中创建多个 CodeCommit 访问角色。然后，将每个访问角色的信任策略配置为允许账户 B 中的特定用户代入该角色。
2. 通过在每个 CodeCommit 访问角色的策略中添加 “资源” 条件，限制每个角色可以担任的代码存储库。

限制 IAM 委托人访问特定 CodeCommit 存储库的 “资源” 条件示例

```
"Resource" : [<REPOSITORY_ARN>,<REPOSITORY_ARN> ]
```

> **Note**
>
> 为了帮助识别和区分同一 AWS 账户中的多个代码存储库，您可以为存储库的名称分配不同的前缀。例如，您可以使用与不同开发人员组对应的前缀来命名代码存储库，例如 myproject-subproject1-repo1 和 myproject-subproject2-repo1。然后，您可以根据为每个开发人员组分配的前缀为其创建一个 IAM 角色。例如，您可以创建一个名为 myproject-subproject1-repoaccess 的角色，并授予其访问包含前缀 myproject-subproject1 的所有代码存储库的权限。

引用包含特定前缀的代码存储库 ARN 的“资源”条件示例

```
"Resource" : arn:aws:codecommit:<region>:<account-id>:myproject-subproject1-*
```

为多 DevOps 账户环境实施 GitHub Flow 分支策略

由 Mike Stephens (AWS) 和 Abhilash Vinod (AWS) 创作

摘要

在管理源代码存储库时，不同的分支策略会影响开发团队使用的软件开发和发布流程。常见分支策略的示例包括 Trunk、 GitHub Flow 和 Gitflow。这些策略使用不同的分支，在每种环境中执行的活动也不同。正在实施 DevOps 流程的组织将受益于可视化指南，以帮助他们了解这些分支策略之间的区别。在组织中使用此视觉效果可以帮助开发团队协调工作并遵循组织标准。此模式提供了这种视觉效果，并描述了在您的组织中实施 GitHub Flow 分支策略的过程。

这种模式是关于为具有多个 AWS 账户分支机构的组织选择和实施 DevOps 分支策略的文档系列的一部分。本系列旨在帮助您从一开始就应用正确的策略和最佳实践，以简化您在云中的体验。 GitHub Flow 只是您的组织可以使用的一种可能的分支策略。本文档系列还涵盖了 T runk 和 Gitflow 分支模型。如果你还没有这样做，我们建议你先查看为多账户 DevOps 环境选择 Git 分支策略，然后再按此模式实施指南。请尽职调查为您的组织选择正确的分支策略。

本指南提供了一个示意图，显示了组织如何实施 GitHub Flow策略。建议您查看《Well-Ar DevOps chitec AWS ted 指南》，以查看最佳实践。此模式包括 DevOps 流程中每个步骤的推荐任务、步骤和限制。

先决条件和限制

先决条件

- Git，已安装。它用作源代码存储库工具。
- draw.io，已安装。此应用程序用于查看和编辑图表。

架构

目标架构

下图可以像 Punnett 方块一样使用（维基百科）。您可以将垂直轴上的分支与水平轴上的 AWS 环境对齐，以确定在每个场景中要执行的操作。这些数字表示工作流程中操作的顺序。此示例将带您从`feature`分支机构到生产环境中的部署。

有关 GitHub Flow 方法中的 AWS 账户、环境和分支的更多信息，请参阅为多账户 DevOps 环境选择 Git 分支策略。

自动化和扩缩

持续集成和持续交付（CI/CD) is the process of automating the software release lifecycle. It automates much or all of the manual processes traditionally required to get new code from an initial commit into production. A CI/CD pipeline encompasses the sandbox, development, testing, staging, and production environments. In each environment, the CI/CD pipeline provisions any infrastructure that is needed to deploy or test the code. By using CI/CD, development teams can make changes to code that are then automatically tested and deployed. CI/CD管道）还通过强制执行一致性、标准、最佳实践以及功能接受和部署的最低接受程度，为开发团队提供管理和护栏。 有关更多信息，请参阅上的 “练习持续集成和持续交付” AWS。

AWS 提供了一套开发人员服务，旨在帮助您构建 CI/CD 管道。例如，AWS CodePipeline是一项完全托管的持续交付服务，可帮助您实现发布管道的自动化，从而实现快速可靠的应用程序和基础设施更新。 AWS CodeBuild编译源代码、运行测试和生成 ready-to-deploy软件包。有关更多信息，请参阅上的开发者工具 AWS。

工具

AWS 服务和工具

AWS 提供了一套开发者服务，你可以用它们来实现这种模式：

- AWS CodeArtifact是一项高度可扩展的托管工件存储库服务，可帮助您存储和共享用于应用程序开发的软件包。
- AWS CodeBuild是一项完全托管的生成服务，可帮助您编译源代码、运行单元测试和生成可随时部署的工件。
- AWS CodeDeploy自动部署到亚马逊弹性计算云 (Amazon EC2)、本地实例、 AWS Lambda 函数或亚马逊弹性容器服务 (Amazon ECS) 服务。
- AWS CodePipeline帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件更改所需的步骤。

其他工具

- Draw.io 桌面是一款用于制作流程图和图表的应用程序。代码存储库包含 drawio 格式的 drawio 格式的 drawio 模板。
- Figma 是一款专为协作而设计的在线设计工具。代码存储库包含 Figma 的.fig 格式的模板。

代码存储库

此模式下图表的源文件可在 GitHub Git GitHub Flow 分支策略存储库中找到。它包括 PNG、draw.io 和 Figma 格式的文件。您可以修改这些图表以支持贵组织的流程。

最佳实践

遵循 Well-Architect AWS e DevOps d 指南和为多账户环境选择 Git 分支策略中的最佳实践和建议。DevOps 它们可以帮助您有效地实施 GitHub 基于 Flow 的开发、促进协作、提高代码质量和简化开发流程。

操作说明

查看 GitHub Flow 工作流程

Task	描述	所需技能
查看标准 GitHub 流程流程。	1. 在沙盒环境中，开发人员从`feature`分支创建分`main`支并使用命名模式`feature/<ticket>_<initials>_<short description>` 。 2. 开发者向`feature`分支添加一个或多个提交，每个提交都代表一个离散的更改或改进。 3. 开发者打开合并请求 (MR)，将更改合并到`main`分支中。这将启动审核流程。 4. 在审查过程中，开发人员会讨论代码变更并提供反馈。目标是确保变更的高质量并符合项目的标准。 5. 开发者创建合并请求后，将启动自动构建流程，并将`feature`分支中的更改部署到开发环境中。	DevOps 工程师

Task	描述	所需技能
	6. 自动测试可验证合并请求中封装的更改的完整性和质量。完成合并请求需要成功构建、成功部署和成功测试。 7. 审阅过程完成后，更改将合并到分`main`支中。 8. 批准者手动批准将发布构件部署到测试环境。 9. 批准者手动批准将发布构件部署到暂存环境。 10批准者手动批准将发布构件部署到生产环境。	

Task	描述	所需技能
查看错误修复 GitHub 流程流程。	1. 开发人员从该bugfix分支创建一个main分支并使用命名模式bugfix/<ticket number>_<developer initials>_<descriptor> 。 2. 开发者修复了问题，提交了修复并构建了bugfix分支。 3. 开发者打开合并请求，要求将bugfix分支合并到分main支中。这将启动审核流程。 4. 在审查过程中，开发人员会讨论代码变更并提供反馈。 5. 审核完成并获得批准后，开发者将完成分支合并到bugfix分main支的请求。 6. 批准者手动批准将发布构件部署到更高的环境。	DevOps 工程师

Task	描述	所需技能
查看修补程序 GitHub 流程流程。	GitHub Flow 旨在实现持续交付，在这种交付中，可以频繁可靠地将代码更改部署到更高的环境中。关键是每个`feature`分支都可以随时部署。 `Hotfix`类似于`feature`或`bugfix`分支的分支可以遵循与其他分支相同的过程。但是，考虑到其紧迫性，修补程序通常具有更高的优先级。根据团队的政策和情况的即时性，可以加快流程中的某些步骤。例如，修补程序的代码审查可能会被快速跟踪。因此，尽管修补程序过程与功能或错误修复过程相似，但由于围绕修补程序的紧迫性，可能需要修改程序遵守情况。必须制定有关管理修补程序的指导方针，以确保高效、安全地处理修补程序。	DevOps 工程师

故障排除

事务	解决方案
分支冲突	GitHub Flow 模型可能出现的一个常见问题是，需要在生产环境中进行修补程序，但需要在修改相同资源的`feature`、`bugfix`、或`hotfix`分支中进行相应的更改。我们建议您经常将来自较低分支的更改合并`main`到较低分支中，以避免在合并到时发生重大冲突`main`。

事务	解决方案
团队成熟度	GitHub Flow 鼓励每天部署到更高的环境，采用真正的持续集成和持续交付 (CI/CD)。团队必须具备工程成熟度，才能构建功能并为其创建自动化测试。在批准变更之前，团队必须对合并请求进行详尽的审查。这培养了一种强大的工程文化，促进了开发过程的质量、问责制和效率。

相关资源

本指南不包括针对 Git 的培训；但是，如果你需要这种培训，互联网上有许多高质量的资源可供选择。我们建议您从 Git 文档网站开始。

以下资源可以帮助您完成 GitHub Flow 分支之 AWS Cloud旅。

AWS DevOps 指导

- AWS DevOps 指南
- AWS 部署管道参考架构
- 什么是 DevOps？
- DevOps resources

GitHub 流量指导

- GitHub Flow 快速入门教程 () GitHub
- 为什么 GitHub Flow？

其他资源

- 十二因子应用程序方法论 (12factor.net)

为多账户环境实施 Gitflow 分支策略 DevOps

由 Mike Stephens (AWS)、Stephen DiCato (AWS)、Tim Wondergem (AWS) 和 Abhilash Vinod (AWS) 创作

摘要

在管理源代码存储库时，不同的分支策略会影响开发团队使用的软件开发和发布流程。常见分支策略的示例包括 Trunk、Gitflow 和 Flow。 GitHub 这些策略使用不同的分支，并且在每种环境中执行的活动也不同。正在实施 DevOps 流程的组织将受益于可视化指南，以帮助他们了解这些分支策略之间的区别。在组织中使用此视觉效果可以帮助开发团队协调工作并遵循组织标准。此模式提供了这种视觉效果，并描述了在您的组织中实施 Gitflow 分支策略的过程。

这种模式是关于为具有多个 AWS 账户分支机构的组织选择和实施 DevOps 分支策略的文档系列的一部分。本系列旨在帮助您从一开始就应用正确的策略和最佳实践，以简化您在云中的体验。Gitflow 只是您的组织可以使用的一种可能的分支策略。本文档系列还涵盖了 Tr unk 和 GitHub Flow 分支模型。如果你还没有这样做，我们建议你先查看为多账户 DevOps 环境选择 Git 分支策略，然后再按此模式实施指南。请尽职调查为您的组织选择正确的分支策略。

本指南提供的图表显示了组织如何实施 Gitflow 策略。建议您查看《Well-Ar DevOps chitec AWS ted 指南》，以查看最佳实践。此模式包括 DevOps 流程中每个步骤的推荐任务、步骤和限制。

先决条件和限制

先决条件

- Git，已安装。它用作源代码存储库工具。
- draw.io，已安装。此应用程序用于查看和编辑图表。
- （可选）Gitflow 插件，已安装。

架构

目标架构

下图可以像 Punnett 方块一样使用（维基百科）。您可以将垂直轴上的分支与水平轴上的 AWS 环境对齐，以确定在每个场景中要执行的操作。这些数字表示工作流程中操作的顺序。此示例将带您从功能分支到生产环境中的部署。

有关 Gitflow 方法中的 AWS 账户、环境和分支的更多信息，请参阅为多账户环境选择 Git 分支策略。
DevOps

自动化和扩缩

持续集成和持续交付（CI/CD) is the process of automating the software release lifecycle. It automates much or all of the manual processes traditionally required to get new code from an initial commit into production. A CI/CD pipeline encompasses the sandbox, development, testing, staging, and production environments. In each environment, the CI/CD pipeline provisions any infrastructure that is needed to deploy or test the code. By using CI/CD, development teams can make changes to code that are then automatically tested and deployed. CI/CD管道）还通过强制执行一致性、标准、最佳实践以及功能接受和部署的最低接受程度，为开发团队提供管理和护栏。 有关更多信息，请参阅上的 “练习持续集成和持续交付” AWS。

AWS 提供了一套开发人员服务，旨在帮助您构建 CI/CD 管道。例如，AWS CodePipeline是一项完全托管的持续交付服务，可帮助您实现发布管道的自动化，从而实现快速可靠的应用程序和基础设施更新。 AWS CodeBuild编译源代码、运行测试和生成 ready-to-deploy软件包。有关更多信息，请参阅上的开发者工具 AWS。

工具

AWS 服务和工具

AWS 提供了一套开发者服务，你可以用它们来实现这种模式：

- AWS CodeArtifact是一项高度可扩展的托管工件存储库服务，可帮助您存储和共享用于应用程序开发的软件包。
- AWS CodeBuild是一项完全托管的生成服务，可帮助您编译源代码、运行单元测试和生成可随时部署的工件。
- AWS CodeDeploy自动部署到亚马逊弹性计算云 (Amazon EC2)、本地实例、 AWS Lambda 函数或亚马逊弹性容器服务 (Amazon ECS) 服务。
- AWS CodePipeline帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件更改所需的步骤。

其他工具

- Draw.io 桌面是一款用于制作流程图和图表的应用程序。代码存储库包含 drawio 格式的 drawio 格式的 drawio 模板。

- Figma 是一款专为协作而设计的在线设计工具。代码存储库包含 Figma 的.fig 格式的模板。
- （可选）Gitflow 插件是 Git 扩展的集合，它们为 Gitflow 分支模型提供高级存储库操作。

代码存储库

此模式下图表的源文件可在 GitFlow存储库的 GitHub Git 分支策略中找到。它包括 PNG、draw.io 和 Figma 格式的文件。您可以修改这些图表以支持贵组织的流程。

最佳实践

遵循 Well-Architect AWS e DevOps d 指南和为多账户环境选择 Git 分支策略中的最佳实践和建议。DevOps 它们可以帮助您有效地实施基于 GitFlow 的开发、促进协作、提高代码质量和简化开发流程。

操作说明

查看 Gitflow 工作流程

Task	描述	所需技能
查看标准的 Gitflow 流程。	1. 在沙盒环境中，开发人员从`feature`分支创建分`develop`支并使用命名模式`feature/<ticket>_<initials>_<short description>` 。 2. 开发人员开发代码并以迭代方式将代码部署到沙盒环境中，以完成票证。 **Note** 开发人员可以选择创建一个`sandbox`分支来运行自动构建或在沙盒环境中部署管道。	DevOps 工程师

Task	描述	所需技能
	3. 开发者使用 squas feature h 合并创建从develop分支到分支的合并请求。 4. 持续集成和持续交付 (CI/CD) 管道会自动构建develop分支并将其部署到开发环境中。 5. （可选）开发人员在继续发布活动之前，将其他feature分支集成到开发分支中。 6. 当你准备好在develop分支中发布功能时，开发者会创建一个以该release分支命名的release/v <number> 分develop支。 7. 开发人员构建发布分支，该分支发布工件以便在其他环境中重复使用。 8. 批准者手动批准将发布构件部署到测试环境。 9. 批准者手动批准将发布构件部署到暂存环境。 10批准者手动批准将发布构件部署到生产环境。 11开发者将分支合并到release分main支中。理想情况下，开发人员使用自动脚本来执行快进合并。不要使用南瓜合并。 12开发者将分支合并到release分develop支中。理想情况下，开发人员使用	

Task	描述	所需技能
	自动脚本来执行快进合并。不要使用南瓜合并。	

Task	描述	所需技能
查看修补程序 Gitflow 流程。	1. 开发人员从该hotfix分支创建一个main分支并使用命名模式hotfix/<ticket>_<initials>_<short description> 。 2. 开发者从该release分支中创建一个main分支并将其命名release/v<number> 。 3. 开发者修复了问题，提交了修复并构建了hotfix分支。 4. 开发者使用 squas hotfix h 合并创建从release/v<number> 分支到分支的合并请求。 5. 开发人员构建release分支，该分支发布工件以便在其他环境中重复使用。 6. 批准者手动批准将发布构件部署到测试环境。 7. 批准者手动批准将发布构件部署到暂存环境。 8. 批准者手动批准将发布构件部署到生产环境。 9. 开发者将分支合并到release分main支中。理想情况下，开发人员使用自动脚本来执行快进合并。不要使用南瓜合并。	DevOps 工程师

Task	描述	所需技能
	10开发者将分支合并到 `release`分`develop`支中。理想情况下，开发人员使用自动脚本来执行快进合并。不要使用南瓜合并。 11如果检测到冲突，开发者会收到警报，并通过合并请求解决冲突。	

Task	描述	所需技能
查看错误修复 Gitflow 流程。	1. 开发者从当前bugfix分支创建一个release/v <number> 分支并使用命名模式bugfix/<ticket number>_<developer initials>_<descriptor> 。 2. 开发者修复了问题，提交了修复并构建了bugfix分支。 3. 开发者使用 squas bugfix h 合并创建从release/v <number> 分支到分支的合并请求。 4. 开发人员构建release分支，该分支发布工件以便在其他环境中重复使用。 5. 批准者手动批准将发布构件部署到测试环境。 6. 批准者手动批准将发布构件部署到舞台环境。 7. 批准者手动批准将发布构件部署到生产环境。 8. 开发者将分支合并到release分main支中。理想情况下，开发人员使用自动脚本来执行快进合并。不要使用南瓜合并。 9. 开发者将分支合并到release分develop支中。理想情况下，开发人员使用自动脚本来执行快进合并。不要使用南瓜合并。	DevOps 工程师

Task	描述	所需技能
	10如果检测到冲突，开发者会收到警报，并通过合并请求解决冲突。	

故障排除

事务	解决方案
分支冲突	Gitflow 模型可能出现的一个常见问题是，需要在生产环境中进行修补程序，但在较低的环境中需要进行相应的更改，即另一个分支正在修改相同的资源。我们建议您一次只能激活一个发布分支。如果您一次有多个分支处于活动状态，则环境中的更改可能会发生冲突，并且您可能无法将分支向前移动到生产中。
合并	应尽快将版本合并回主分支并进行开发，以便将工作整合回主分支。
南瓜合并	只有在从分支合并到`feature`分支时才使用 squas `develop` h 合并。在较高的分支中使用 squash 合并会导致将更改向下合并到较低的分支时会遇到困难。

相关资源

本指南不包括针对 Git 的培训；但是，如果你需要这种培训，互联网上有许多高质量的资源可供选择。我们建议您从 Git 文档网站开始。

以下资源可以帮助你在 Gitflow 中完成分支之旅。 AWS Cloud

AWS DevOps 指导

- AWS DevOps 指南
- AWS 部署管道参考架构

- 什么是 DevOps？
- DevOps resources

Gitflow 指南

- 最初的 Gitflow 博客（Vincent Driessen 博客文章）
- Gitflow 工作流程 (Atlassian)
- Gitflow 开启 GitHub：如何将 Git Flow 工作流程与 GitHub 基于 Repos 的存储库一起使用（视频）YouTube
- Git Flow 初始化示例（YouTube 视频）
- Gitflow 发布分支从头到尾（视频）YouTube

其他资源

十二因子应用程序方法论 (12factor.net)

为多 DevOps 账户环境实施中继分支策略

由迈克·斯蒂芬斯 (AWS) 和雷扬·威尔逊 (AWS) 创作

摘要

在管理源代码存储库时，不同的分支策略会影响开发团队使用的软件开发和发布流程。常见分支策略的示例包括 Trunk、 GitHub Flow 和 Gitflow。这些策略使用不同的分支，在每种环境中执行的活动也不同。正在实施 DevOps 流程的组织将受益于可视化指南，以帮助他们了解这些分支策略之间的区别。在组织中使用此视觉效果可以帮助开发团队协调工作并遵循组织标准。此模式提供了这种视觉效果，并描述了在组织中实施主干分支策略的过程。

这种模式是关于为具有多个 AWS 账户分支机构的组织选择和实施 DevOps 分支策略的文档系列的一部分。本系列旨在帮助您从一开始就应用正确的策略和最佳实践，以简化您在云中的体验。Trunk 只是您的组织可以使用的一种可能的分支策略。本文档系列还涵盖了 GitHub Flow 和 Gitflow 分支模型。如果你还没有这样做，我们建议你先查看为多账户 DevOps 环境选择 Git 分支策略，然后再按此模式实施指南。请尽职调查为您的组织选择正确的分支策略。

本指南提供的图表显示了组织如何实施主干策略。建议您查看官方的《Well-Ar DevOps chitec AWS ted 指南》，以查看最佳实践。此模式包括 DevOps 流程中每个步骤的推荐任务、步骤和限制。

先决条件和限制

先决条件

- Git，已安装。它用作源代码存储库工具。
- draw.io，已安装。此应用程序用于查看和编辑图表。

架构

目标架构

下图可以像 Punnett 方块一样使用（维基百科）。您可以将垂直轴上的分支与水平轴上的 AWS 环境对齐，以确定在每个场景中要执行的操作。这些数字表示工作流程中操作的顺序。此示例将带您从`feature`分支机构到生产环境中的部署。

有关 Trunk 方法中的 AWS 账户、环境和分支的更多信息，请参阅为多账户 DevOps 环境选择 Git 分支策略。

自动化和扩缩

持续集成和持续交付（CI/CD) is the process of automating the software release lifecycle. It automates much or all of the manual processes traditionally required to get new code from an initial commit into production. A CI/CD pipeline encompasses the sandbox, development, testing, staging, and production environments. In each environment, the CI/CD pipeline provisions any infrastructure that is needed to deploy or test the code. By using CI/CD, development teams can make changes to code that are then automatically tested and deployed. CI/CD管道）还通过强制执行一致性、标准、最佳实践以及功能接受和部署的最低接受程度，为开发团队提供管理和护栏。 有关更多信息，请参阅上的 “练习持续集成和持续交付” AWS。

AWS 提供了一套开发人员服务，旨在帮助您构建 CI/CD 管道。例如，AWS CodePipeline是一项完全托管的持续交付服务，可帮助您实现发布管道的自动化，从而实现快速可靠的应用程序和基础设施更新。 AWS CodeBuild编译源代码、运行测试和生成 ready-to-deploy软件包。有关更多信息，请参阅上的开发者工具 AWS。

工具

AWS 服务和工具

AWS 提供了一套开发者服务，你可以用它们来实现这种模式：

- AWS CodeArtifact是一项高度可扩展的托管工件存储库服务，可帮助您存储和共享用于应用程序开发的软件包。
- AWS CodeBuild是一项完全托管的生成服务，可帮助您编译源代码、运行单元测试和生成可随时部署的工件。
- AWS CodeDeploy自动部署到亚马逊弹性计算云 (Amazon EC2)、本地实例、 AWS Lambda 函数或亚马逊弹性容器服务 (Amazon ECS) 服务。
- AWS CodePipeline帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件更改所需的步骤。

其他工具

- Draw.io 桌面 — 用于制作流程图和图表的应用程序。
- Figma 是一款专为协作而设计的在线设计工具。代码存储库包含 Figma 的.fig 格式的模板。

代码存储库

此模式中图表的源文件可在中继的 GitHub Git 分支策略存储库中找到。它包括 PNG、draw.io 和 Figma 格式的文件。您可以修改这些图表以支持贵组织的流程。

最佳实践

遵循 Well-Architect AWS e DevOps d 指南和为多账户环境选择 Git 分支策略中的最佳实践和建议。DevOps 它们可以帮助您有效地实施基于 Trunk 的开发、促进协作、提高代码质量和简化开发流程。

操作说明

查看中继工作流程

Task	描述	所需技能
查看标准中继流程。	1. 在沙盒环境中，开发人员从`feature`分支创建分`main`支并使用命名模式`feature/<ticket>_<initials>_<short description>` 。 2. 开发人员开发代码并以迭代方式将代码部署到沙盒环境中，以完成票证。 **Note** 开发人员可以选择创建一个`sandbox`分支来运行自动构建或在沙盒环境中部署管道。 3. 开发者使用 squas `feature` h 合并创建从`main`分支到分支的合并请求。 4. 持续集成和持续交付 (CI/CD) 管道会自动生成工件并	DevOps 工程师

Task	描述	所需技能
	将其从main分支发布到开发环境。 5. 批准者手动批准将发布构件部署到开发环境。 6. 批准者手动批准将发布构件部署到测试环境。 7. 批准者手动批准将发布构件部署到暂存环境。 8. 批准者手动批准将发布构件部署到生产环境。	

故障排除

事务	解决方案
分支冲突	Trunk 模型可能出现的一个常见问题是，需要在生产环境中进行修补程序，但需要在修改相同资源的feature分支中进行相应的更改。我们建议您经常将来自下级分支的更改合并main到较低的分支中，以避免在合并到时发生重大冲突main。

相关资源

本指南不包括针对 Git 的培训；但是，如果你需要这种培训，互联网上有许多高质量的资源可供选择。我们建议您从 Git 文档网站开始。

以下资源可以帮助你完成中继分支之旅。 AWS Cloud

AWS DevOps 指导

- AWS DevOps 指南
- AWS Deployment Pipeline Reference Architecture
- 什么是 DevOps？

- DevOps resources

后备箱引导

- 基于主干的开发

其他资源

- 十二因子应用程序方法论 (12factor.net)

在部署 AWS 基础设施之前，实施集中式自定义 Checkov 扫描以强制执行策略

由本杰明·莫里斯 (AWS) 创作

摘要

这种模式提供了一个 GitHub 操作框架，用于在一个存储库中编写可在整个组织中重复使用的自定义 Checkov 策略。 GitHub 通过遵循这种模式，信息安全团队可以根据公司要求编写、添加和维护自定义策略。自定义策略可以自动拉入 GitHub 组织中的所有管道。这种方法可用于在资源部署之前强制执行公司的资源标准。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 使用 GitHub 操作的 GitHub 组织
- AWS 使用 HashiCorp Terraform 部署的基础架构或 AWS CloudFormation

限制

- 此模式是为 GitHub 操作编写的。但是，它可以适应类似的持续集成和持续交付 (CI/CD) 框架，例如。 GitLab不需要特定的付费版本。 GitHub
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅 AWS 文档中的服务终端节点和配额，然后选择该服务的链接。

架构

此模式旨在部署为包含 GitHub 可重用工作流程和自定义 Checkov 策略的 GitHub 存储库。可重复使用的工作流程可以扫描 Terraform 和 CloudFormation 基础设施即代码 (IaC) 存储库。

下图将可重用 GitHub 工作流程存储库和自定义 Checkov 策略存储库作为单独的图标显示。但是，您可以将这些存储库作为单独的存储库或单个存储库来实现。示例代码使用单个存储库，工作流程文件 (`.github/workflows`) 和自定义策略文件（`custom_policies`文件夹和`.checkov.yml`配置文件）位于同一个存储库中。

图表显示了以下工作流：

1. 用户在 GitHub 仓库中创建拉取请求。
2. 管道工作流程从 Acti GitHub ons 开始，包括对 Checkov 可重复使用工作流程的引用。
3. 管道工作流程从外部存储库下载引用的 Checkov 可重用工作流程，并使用 GitHub 操作运行该 Checkov 工作流程。
4. Checkov 可重用工作流程从外部存储库下载自定义策略。
5. Checkov 可重用工作流程根据内置和自定义 Checkov 策略评估 GitHub 存储库中的 IaC。Checkov 可重复使用的工作流程通过或失败取决于是否发现了安全问题。

自动化和扩缩

这种模式允许集中管理 Checkov 配置，因此可以在一个位置应用策略更新。但是，这种模式确实要求每个存储库都使用包含对中央可重复使用工作流程的引用的工作流程。您可以手动添加此引用，也可以使用脚本将文件推送到每个存储库`.github/workflows`的文件夹。

工具

AWS 服务

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在资源的整个生命周期中跨地区对其 AWS 账户 进行管理。Checkov 可以扫描 CloudFormation。

其他工具

- Checkov 是一款静态代码分析工具，用于检查 IaC 是否存在安全性和合规性错误配置。
- GitHub 操作已集成到 GitHub 平台中，可帮助您在 GitHub 仓库中创建、共享和运行工作流程。您可以使用 GitHub Actions 来自动执行诸如构建、测试和部署代码之类的任务。
- Terraform 是一款 IaC 工具 HashiCorp ，可帮助您创建和管理云和本地资源。Checkov 可以扫描 Terraform。

代码存储库

此模式的代码可在 GitHub centralized-custom-checkov-sast存储库中找到。

最佳实践

- 要保持一致的安全态势，请将公司的安全策略与Checkov政策保持一致。

- 在实施 Checkov 自定义策略的早期阶段，您可以在 Checkov 扫描中使用 soft-fail 选项来允许合并存在安全问题的 IaC。随着流程的成熟，从软失败选项切换到硬失败选项。

操作说明

为自定义策略创建中央Checkov存储库

Task	描述	所需技能
创建一个中央Checkov存储库。	创建一个存储库来存储将在组织内使用的自定义 Checkov 策略。 为了快速入门，您可以将此模式 GitHub centralized-custom-checkov-sast 存储库的内容复制到中央的 Checkov 存储库中。	DevOps 工程师
为可重复使用的工作流程创建存储库。	如果已存在可重复使用工作流程的存储库，或者您计划在与自定义 Checkov 策略相同的存储库中包含可重复使用的工作流程文件，则可以跳过此步骤。 创建 GitHub 存储库以保存可重复使用的工作流程。其他仓库的管道将引用此存储库。	DevOps 工程师

创建可重复使用的 Checkov 工作流程和示例

Task	描述	所需技能
添加可重复使用的 Checkov 工作流程。	在可重复使用的工作流程存储库中创建可重复 GitHub 使用的 Checkov Actions 工作流程	DevOps 工程师

Task	描述	所需技能
	（YAML 文件）。您可以从此模式中提供的工作流程文件中调整此可重复使用的工作流程。 您可能想要进行的更改的一个示例是将可重复使用的工作流程更改为使用 soft-fail 选项。设置`soft-fail` 为`true`允许任务成功完成，即使 Checkov 扫描失败。有关说明，请参阅 Checkov 文档中的硬失败和软失败。	
添加示例工作流程。	添加引用该工作流程的 Checkov 工作`reusable`流程示例。这将为如何重用`reusable`工作流程提供一个模板。在示例存储库中，`checkov-source.yaml` 是可重复使用的工作流程，`checkov-scan.yaml` 也是使用的示例。`checkov-source` 有关编写示例 Checkov 工作流程的更多详细信息，请参阅其他信息。	DevOps 工程师

将公司政策与 Checkov 自定义策略相关联

Task	描述	所需技能
确定可以使用 Checkov 强制执行的政策。	1. 查看与基础架构安全相关的公司政策以及应满足哪些要求。 2. 确定哪些要求可以使用 Checkov 自定义策略来实现。 3. 创建将策略控制映射到 Checkov 自定义策略的命名约定。通常，Checkov 自定义策略具有标识符，其中包含 Checkov 名称、策略来源（自定义）和策略编号（例如）。`CKV2_CUSTOM_123` 有关创建 Checkov 自定义策略的更多详细信息，请参阅 Checkov 文档中的自定义策略概述。	安全性与合规性
添加 Checkov 自定义策略。	将已确定的公司政策转换为中央存储库中的自定义 Checkov 策略。你可以用 Python 或 YAML 编写简单的 Checkov 策略。	安全性

实施集中式 Checkov 自定义策略

Task	描述	所需技能
将 Checkov 可重复使用的工作流程添加到所有存储库。	此时，您应该有一个引用可重用工作流程的 Checkov 工作流	DevOps 工程师

Task	描述	所需技能
	程示例。将引用可重用工作流程的示例 Checkov 工作流程复制到每个需要它的存储库。	
创建一种机制来确保 Checkov 在合并之前运行。	为确保每个拉取请求都运行 Checkov 工作流程，请创建一个状态检查，要求在合并拉取请求之前成功执行 Checkov 工作流程。 GitHub 允许您要求在合并拉取请求之前运行特定的工作流程。	DevOps 工程师
创建组织范围的 PAT，并将其作为密钥共享。	如果您的 GitHub 组织公开可见，则可以跳过此步骤。 这种模式要求 Checkov 工作流程能够从 GitHub 组织中的自定义策略存储库下载自定义策略。您必须提供权限，这样 Checkov 工作流程才能访问这些存储库。 为此，请创建具有读取组织仓库权限的个人访问令牌 (PAT)。与存储库共享此 PAT，可以是组织范围的密钥（如果是付费套餐），也可以是每个存储库中的密钥（免费版本）。在示例代码中，密钥的默认名称为`ORG_PAT`。	DevOps 工程师

Task	描述	所需技能
（可选）保护 Checkov 工作流程文件不被修改。	要保护 Checkov 工作流程文件免受不必要的更改，您可以使用CODEOWNERS 文件。该CODEOWNERS 文件通常部署在目录的根目录中。 例如，要要求在修改 checkov-scan.yaml 文件时获得 GitHub 组织secEng群组的批准，请将以下内容附加到存储库CODEOWNERS 的文件中： `[Checkov] .github/workflows /checkov-scan.yaml @myOrg/secEng` CODEOWNERS 文件特定于它所在的存储库。要保护存储库使用的 Checkov 工作流程，您必须在每个存储库中添加（或更新）一个CODEOWNERS 文件。 有关保护 Checkov 工作流程文件的更多信息，请参阅其他信息。有关CODEOWNERS 文件的更多信息，请参阅 CI/CD 提供商的官方文档（例如 GitHub）。	DevOps 工程师

相关资源

- Checkov 自定义策略概述
- CloudFormation 配置扫描

- GitHub 操作可重复使用的工作流程

其他信息

编写 Checkov 工作流程文件

写作时checkov-scan.yaml，请考虑你希望它何时运行。顶级on密钥决定工作流程何时运行。在示例存储库中，当有针对该main分支的拉取请求时（以及该拉取请求的源分支被修改时），工作流程就会运行。由于workflow_dispatch密钥的原因，也可以根据需要运行工作流程。

您可以根据希望工作流程的运行频率来更改工作流程触发条件。例如，您可以将工作流程更改为每次将代码推送到任何分支时运行，方法是将密钥pull_request替换为push并移除branches密钥。

您可以修改在单个存储库中创建的示例工作流程文件。例如，production如果仓库是围绕分支构造的，则可以main将目标production分支的名称从调整为。

保护 Checkov 工作流程文件

Checkov 扫描可提供有关潜在安全配置错误的有用信息。但是，一些开发人员可能会认为这是他们工作效率的障碍，因此试图删除或禁用扫描工作流程。

有几种方法可以解决这个问题，包括更好地宣传安全扫描的长期价值，以及关于如何部署安全基础设施的更清晰的文档。这些是重要的 “软” DevSecOps 协作方法，可以看作是解决这个问题根本原因的办法。但是，您也可以使用诸如CODEOWNERS文件之类的技术控件作为护栏，以帮助开发人员走上正确的道路。

在沙箱中测试模式

要在沙盒环境中测试此模式，请执行以下步骤：

1. 创建新 GitHub 组织。创建对组织中所有仓库具有只读访问权限的令牌。由于此令牌适用于沙盒环境，而不是付费环境，因此您将无法将此令牌存储在组织范围内的密钥中。
2. 创建一个checkov存储库来存放 Checkov 配置，并创建一个github-workflows存储库来保存可重复使用的工作流程配置。使用示例存储库的内容填充存储库。
3. 创建应用程序存储库，然后将checkov-scan.yaml工作流程复制并粘贴到其.github/workflows文件夹。向存储库中添加一个密钥，其中包含您为组织只读访问而创建的 PAT。默认密钥是ORG_PAT。
4. 创建一个拉取请求，向应用程序存储库中添加一些 Terraform 或 CloudFormation 代码。Checkov 应该扫描并返回结果。

自动检测变化并为 monorepo 启动不同的 CodePipeline 管道 CodeCommit

由 Helton Ribeiro (AWS)、Petrus Batalha (AWS) 和 Ricardo Morais (AWS) 创作

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

注意： AWS Cloud9 不再向新客户开放。的现有客户 AWS Cloud9 可以继续照常使用该服务。了解更多

这种模式可以帮助您自动检测基于 monorepo 的应用程序源代码的更改，然后在其中启动运行持续集成 AWS CodeCommit 和持续交付（管道）的CI/CD) automation for each microservice. This approach means that each microservice in your monorepo-based application can have a dedicated CI/CD管道，从而确保更好的可见性、更轻松的代码共享以及改进的协作、标准化和可发现性。 AWS CodePipeline

此模式中描述的解决方案不会在 monorepo 内部的微服务之间执行任何依赖关系分析。它仅检测源代码中的更改并启动匹配的 CI/CD 管道。

该模式 AWS Cloud9 用作集成开发环境 (IDE)，并使用两个 AWS CloudFormation 堆栈 AWS Cloud Development Kit (AWS CDK) 来定义基础架构：`MonoRepoStack`和`PipelinesStack`。`MonoRepoStack`堆栈在中创建 monorepo AWS CodeCommit 和启动 CI/CD 管道的 AWS Lambda 函数。`PipelinesStack` 堆栈定义了管道基础设施。

> **Important**
>
> 此模式的工作流程是概念验证 (PoC)。我们建议您仅在测试环境中使用它。如果您想在生产环境中使用此模式的方法，请参阅 AWS Identity and Access Management (IAM) 文档中的 IAM 中的安全最佳实践，并对您的 IAM 角色和进行必要的更改 AWS 服务。

先决条件和限制

先决条件

- 一个活跃的 AWS 账户。

- AWS Command Line Interface (AWS CLI)，已安装并配置。有关更多信息，请参阅 AWS CLI 文档 AWS CLI中的安装、更新和卸载。
- Python 3 和 `pip`，已安装于本地计算机。有关更多信息，请参阅 Python 文档。
- AWS CDK，已安装并配置。有关更多信息，请参阅 AWS CDK 文档AWS CDK中的入门。
- 已安装和配置的 AWS Cloud9 IDE。有关更多信息，请参阅 AWS Cloud9 文档 AWS Cloud9中的设置。
- GitHub AWS CodeCommit monorepo 多管道触发器存储库，克隆到您的本地计算机上。
- 包含要用来生成和部署的应用程序代码的现有目录 CodePipeline。
- 熟悉 DevOps 最佳实践并有经验. AWS Cloud为了增加对它的熟悉程度 DevOps，你可以使用 “使用 DevOps 实践和 AWS 规范指南” AWS Cloud9网站上的 “构建与微服务松散耦合的架构” 模式。

架构

下图显示了如何使用 AWS CDK 来定义具有两个 AWS CloudFormation 堆栈的基础架构：`MonoRepoStack`和。`PipelinesStack`

图表显示了以下工作流：

1. 引导过程使用创建 AWS CloudFormation 堆栈和`MonoRepoStack`。 AWS CDK `PipelinesStack`
2. `MonoRepoStack`堆栈会为您的应用程序创建 CodeCommit 存储库，并在每次提交后启动的 `monorepo-event-handler` Lambda 函数。
3. `PipelinesStack`堆栈在 CodePipeline 其中创建由 Lambda 函数启动的管道。每项微服务都必须有定义的基础设施管道。
4. 的管道`microservice-n`由 Lambda 函数启动，并根据中的源代码启动其隔离的 CI/CD 阶段。CodeCommit
5. 的管道`microservice-1`由 Lambda 函数启动，并根据中的源代码启动其隔离的 CI/CD 阶段。CodeCommit

下图显示了 AWS CloudFormation 堆栈`MonoRepoStack`和账户`PipelinesStack`中的部署情况。

1. 用户在其中一个应用程序微服务中更改代码。

2. 用户将更改从本地存储库推送到 CodeCommit 存储库。
3. 推送活动启动 Lambda 函数，该函数接收所有到存储库的推送。 CodeCommit
4. Lambda 函数读取参数存储中的参数（一种功能）以检索最新的提交 ID。 AWS Systems Manager 该参数的命名格式为：`/MonoRepoTrigger/{repository}/{branch_name}/LastCommit`。如果未找到该参数，Lambda 函数会从 CodeCommit 存储库中读取最后一次提交 ID，并将返回的值保存在参数存储中。
5. 识别提交 ID 和更改的文件后，Lambda 函数会识别每个微服务目录的管道并启动所需的管道。 CodePipeline

工具

- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，用于在代码中定义云基础架构并通过它进行配置 AWS CloudFormation。
- Python 是一种编程语言，可让您快速工作并更有效地集成系统。

代码

此模式的源代码和模板可在 GitHub AWS CodeCommit monorepo 多管道触发器存储库中找到。

最佳实践

- 此示例架构不包括针对已部署基础设施的监控解决方案。如果您想在生产环境中部署此解决方案，我们建议您启用监控。有关更多信息，请参阅 AWS Serverless Application Model (AWS SAM) 文档中的使用 Application Insights 监控您的无服务器 CloudWatch 应用程序。
- 编辑此模式提供的示例代码时，请遵循 AWS CDK 文档中开发和部署云基础设施的最佳实践。
- 定义微服务管道时，请查看 AWS CodePipeline 文档中的安全最佳实践。
- 您也可以使用 cdk-na g 实用程序检查 AWS CDK 代码以了解最佳实践。此工具使用一组按包分组的规则来评估您的代码。可用的包有：
  - AWS 解决方案库
  - Health Insurance 流通与责任法案 (HIPAA) 安全
  - 美国国家标准与技术研究所 (NIST) 800-53 修订版 4
  - NIST 800-53 第 5 版
  - 支付卡行业数据安全标准 (PCI DSS) 3.2.1

操作说明

设置环境

Task	描述	所需技能
创建虚拟 Python 环境。	在 AWS Cloud9 IDE 中，通过运行以下命令创建虚拟 Python 环境并安装所需的依赖项： `make install`	开发人员
Bootstrap AWS 账户 和 f AWS 区域 or. AWS CDK	通过运行以下命令引导所需的 AWS 账户 区域和区域： `make bootstrap account-id=<your-AWS-account-ID> region=<required-region>`	开发人员

为微服务添加新管道

Task	描述	所需技能
将示例代码添加至应用程序目录。	将包含您的示例应用程序代码的目录添加到克隆的 GitHub AWS CodeCommit monorepo 多管道触发器存储库中的`monorepo-sample` 目录中。	开发人员
编辑 `monorepo-main.json` 文件。	将应用程序代码的目录名和管道名称添加到克隆存储库中的`monorepo-main.json` 文件中。	开发人员

Task	描述	所需技能
创建管道。	在存储库的Pipelines 目录中，class为您的应用程序添加管道。该目录包含两个示例文件，pipeline_hotsite.py 和pipeline_demo.py 。每个文件都有三个阶段：源代码、生成和部署。 您可以复制其中一个文件，并根据应用程序要求对其进行更改。	开发人员

Task	描述	所需技能

编辑 monorepo_config.py 文件。	在 service_map 中，添加应用程序的目录名称和为管道创建的分类。
	
	例如，以下代码显示了 Pipelines 目录中的管道定义，该定义使用名为 pipeline_mysample.py 的 MySamplePipeline 类文件：
	
	`...`
	`# Pipeline definition imports`
	`from pipelines.pipeline_demo import DemoPipeline`
	`from pipelines.pipeline_hotsite import HotsitePipeline`
	`from pipelines.pipeline_mysample import MySamplePipeline`
	
	`### Add your pipeline configuration here`
	`service_map: Dict[str, ServicePipeline] = {`
	`# folder-name -> pipeline-class`
	`'demo': DemoPipeline(),`
	`'hotsite': HotsitePipeline(),`
	`'mysample': MySamplePipeline()`
	`}`
		开发人员

部署堆 MonoRepoStack 栈

Task	描述	所需技能
部署 AWS CloudFormation 堆栈。	通过运行`make deploy-core` 命令将带有默认参数值的 AWS CloudFormation `MonoRepoStack` 堆栈部署到克隆存储库的根目录中。 您可以通过运行 `make deploy-core monorepo-name=<repo_name>` 命令更改存储库名称。 **Note** 您可以使用`make deploy monorepo-name=<repo_name>` 命令同时部署两个管道。	开发人员
验证 CodeCommit 存储库。	通过运行 `aws codecommit get-repository --repository-name <repo_name>` 命令验证资源是否已创建。 **Important** 由于 AWS CloudFormation 堆栈会创建存储 monorepo 的存储 CodeCommit 库，因此如果您已开始向其中推送修改，请不要	开发人员

Task	描述	所需技能
	运行该`cdk destroy MonoRepoStack` 命令。	
验证 AWS CloudFormation 堆栈结果。	通过运行以下命令验证 AWS CloudFormation `MonoRepoStack` 堆栈的创建和配置是否正确： `aws cloudformation list-stacks --stack-status-filter CREATE_COMPLETE --query 'StackSummaries[?StackName == 'MonoRepoStack']'`	开发人员

部署堆 PipelinesStack 栈

Task	描述	所需技能
部署 AWS CloudFormation 堆栈。	AWS CloudFormation `PipelinesStack` 堆栈必须在部署堆`MonoRepoStack` 栈后部署。当将新微服务添加至 monorepo 的代码库中时，堆栈的大小会增加，并在加入新微服务时重新部署。 通过运行`make deploy-pipelines` 命令部署 PipelinesStack 堆栈。	开发人员

Task	描述	所需技能

	Note
	您也可以通过运行`make deploy monorepo-name=<repo_name>` 命令同时部署两个管道。
	
	以下示例输出显示了 `PipelinesStacks` 部署在 URLs 实现结束时如何打印微服务的内容：
	
	`Outputs:`
	`PipelinesStack.demourl = .cloudfront.net`
	`PipelinesStack.hotsiteurl = .cloudfront.net`	

验证 AWS CloudFormation 堆栈结果。	通过运行以下命令验证 AWS CloudFormation `Pipelines Stacks` 堆栈的创建和配置是否正确： `aws cloudformation list-stacks --stack-status-filter CREATE_COMPLETE UPDATE_COMPLETE --query 'StackSummaries[?StackName == 'PipelinesStack']'`	开发人员

清理资源

Task	描述	所需技能
删除您的 AWS CloudFormation 堆栈。	运行 `make destroy`命令。	开发人员
删除管道的 S3 存储桶。	1. 登录AWS Management Console并打开亚马逊简单存储服务 (Amazon S3) 控制台。 2. 删除与您的管道关联的 S3 存储桶并使用以下名称：`pipelinesstack-codepipeline*`	开发人员

故障排除

事务	解决方案
我遇到了 AWS CDK 问题。	请参阅 AWS CDK 文档中的常见 AWS CDK 问题疑难解答。
我推送了我的微服务代码，但微服务管道没有运行。	设置验证 验证分支配置： • 确保将代码推送到正确的分支。此管道配置为仅在对分`main`支进行更改时运行。除非经过专门配置，否则推送到其他分支不会启动管道。 • 推送代码后，请检查提交是否在中可见，AWS CodeCommit 以确保推送成功且本地环境与存储库之间的连接完好无损。如果推送代码时出现问题，请刷新您的凭证。

事务	解决方案
	验证配置文件： • 确认中的`service_map` 变量`monorepo_config.py` 准确反映了微服务的当前目录结构。此变量在将您的代码推送映射到相应的管道方面起着至关重要的作用。 • 请确保更新`monorepo-main.json` 以包含您的微服务的新映射。该文件对于管道识别和正确处理微服务更改至关重要。 在控制台上进行故障排除 AWS CodePipeline 支票： • 在上 AWS Management Console，确认您 AWS 区域 所在的管道所在地。打开CodePipeline 控制台，检查与您的微服务对应的管道是否已启动。 错误分析：如果管道已启动但失败，请查看提供的任何错误消息或日志 CodePipeline ，以了解出了什么问题。 AWS Lambda 疑难解答： • 在AWS Lambda 控制台上，打开 `monorepo-event-handler` Lambda 函数。确认该函数是为了响应代码推送而启动的。 日志分析：检查 Lambda 函数的日志中是否存在任何问题。这些日志可以提供函数运行时发生的事情的详细见解，并帮助确定函数是否按预期处理了事件。

事务	解决方案
我需要重新部署我所有的微服务。	有两种方法可以强制重新部署所有微服务。选择符合您要求的选项。 方法 1：删除参数存储中的参数 此方法涉及删除 Systems Manager 参数存储区中用于跟踪上次用于部署的提交 ID 的特定参数。删除此参数时，系统将被迫在下次触发时重新部署所有微服务，因为它会将其视为全新状态。 步骤： 1. 找到包含您的 monorepo 的提交 ID 或相关部署标记的特定参数存储条目。参数名称遵循以下格式：`"/MonoRepoTrigger/{repository}/{branch_name}/LastCommit"` 2. 如果参数值很重要，或者您希望在重置之前保留部署状态的记录，请考虑对其进行备份。 3. 使用 AWS Management Console AWS CLI、或 SDKs 删除已识别的参数。此操作会重置部署标记。 4. 删除后，下一次推送到存储库应该会导致系统部署所有微服务，因为它会寻找要考虑部署的最新提交。 优点： • 只需最少的步骤即可简单快捷地实施。 • 不需要进行任意代码更改即可启动部署。 缺点： • 对部署过程的控制不那么精细。

事务	解决方案
	• 如果使用参数存储来管理其他关键配置，则可能存在风险。 方法 2：在每个 monorepo 子文件夹中推送一个提交 此方法包括进行较小的更改，然后将其推送到 monorepo 中的每个微服务子文件夹中，以启动其各自的管道。 步骤： 1. 列出 monorepo 中所有需要重新部署的微服务。 2. 对于每项微服务，在其子文件夹中进行最小的、无影响的更改。这可能是更新README文件、在配置文件中添加注释或任何不影响服务功能的更改。 3. 使用明确的消息（例如 “启动微服务的重新部署”）提交这些更改，然后将其推送到存储库。确保将更改推送到启动部署的分支。 4. 监控每项微服务的管道，以确认它们已成功启动并完成。 优点： • 提供对重新部署哪些微服务的精细控制。 • 更安全，因为它不涉及删除可能用于其他目的的配置参数。 缺点： • 更耗时，尤其是在使用大量微服务时。

事务	解决方案
	• 需要进行不必要的代码更改，这可能会使提交历史记录混乱。

相关资源

- 使用 CDK Pipelines 进行持续集成和交付 (CI/CD)（文档）AWS CDK
- aws-cdk/管道模块（API 参考）AWS CDK

使用 AWS 将 Bitbucket 存储库与 AWS Amplify 集成 CloudFormation

由 Alwin Abraham (AWS) 创建

摘要

AWS Amplify 可帮助您快速部署和测试静态网站，无需设置通常所需基础设施。如果您的组织想要使用 Bitbucket 进行源代码控制，无论是迁移现有应用程序代码还是构建新应用程序，都可部署这种模式的方法。通过使用 AWS CloudFormation 自动设置 Amplify，您可以查看自己使用的配置。

此模式描述了如何使用 AWS 将 Bitbucket 存储库与 AWS Amplify 集成， CloudFormation 从而创建前端持续集成和持续部署 (CI/CD) 管道和部署环境。此模式方法意味着你可以为可重复的部署构建 Amplify 前端管道。

先决条件和限制

先决条件

- 活跃 Amazon Web Services (AWS) 账户
- 含管理员访问权限的活跃 Bitbucket 账户
- 访问使用 cURL 或 Postman 应用程序的终端
- 熟悉 Amplify
- 熟悉 AWS CloudFormation
- 熟悉 YAML 格式的文件

架构

技术堆栈

- Amplify
- AWS CloudFormation
- Bitbucket

工具

- AWS Amplify — Amplify 帮助开发人员开发和部署基于云的移动与网络应用程序。

- AWS CloudFormation — AWS CloudFormation 是一项服务，可帮助您建模和设置 AWS 资源，这样您就可以减少管理这些资源的时间，将更多时间集中在在 AWS 中运行的应用程序上。
- Bitbucket – Bitbucket 是一款专为专业团队设计的 Git 存储库管理解决方案。它为你提供了集中位置来管理 Git 存储库、协作处理源代码以及指导你完成开发流程。

代码

该`bitbucket-amplify.yml`文件（附后）包含此模式的 AWS CloudFormation 模板。

操作说明

配置 Bitbucket 存储库

Task	描述	所需技能
（可选）创建 Bitbucket 存储库。	1. 登录您的 Bitbucket 账户，并创建新的存储库。有关这方面的更多信息，请参阅 Bitbucket 文档中的创建 Git 存储库。 2. 记录工作空间名称。 **Note** 您也可以使用现有的 Bitbucket 存储库。	DevOps 工程师
打开工作区设置。	1. 打开工作区并选择存储库选项卡。 2. 选择要与 Amplify 集成的存储库。 3. 选择位于存储库名称上方位置的工作区名称。 4. 在侧栏上，选择设置。	DevOps 工程师

Task	描述	所需技能
创建 OAuth 消费者。	1. 在 “应用程序和功能” 部分中，选择OAuth 消费者，然后选择添加消费者。 2. 输入您的消费者名称，例如，`Amplify Integration` 。 3. 输入回调 URL。尽管此字段为必填项，但它不用于完成集成，因此该值可能是 `http://localhost:3000` 4. 选中这是私人使用者复选框。 5. 选择以下权限： • 项目 – `Read` • 存储库 – `Admin` • 拉取请求 – `Read` • 网络钩子 – `Read` 和 `Write` 6. 对所有其他字段保持默认选择，然后选择提交。 7. 记录生成的密钥与机密。	DevOps 工程师

Task	描述	所需技能
获取 OAuth 访问令牌。	1. 打开终端窗口，并运行以下命令： `curl -X POST -u "KEY:SECRET" https://bitbucket.org/site/oauth2/access_token -d grant_type=client_credentials` **Important** SECRET用之前记录的密钥和密钥替换KEY和。 2. 在不使用引号的情况下，记录访问令牌。该令牌仅在有限时间内有效，默认时间为两个小时。您必须在此时间范围内运行 AWS CloudFormation 模板。	DevOps 工程师

创建和部署 AWS CloudFormation 堆栈

Task	描述	所需技能
下载 AWS CloudFormation 模板。	下载 A `bitbucket-amplify.yml` WS CloudFormation 模板（附后）。除了 Amplify 项目与分	

Task	描述	所需技能
	支外，此模板还在 Amplify 中创建 CI/CD 管道。	

Task	描述	所需技能
创建并部署 AWS CloudFormation 堆栈。	1. 登录您要部署的 AWS 区域的 AWS 管理控制台，然后打开 AWS CloudFormation 控制台。 2. 选择创建堆栈（使用新资源），然后选择上传模板文件。 3. 上传 `bitbucket-amplify.yml` 文件。 4. 选择下一步，输入堆栈名称，然后输入以下参数： • 访问令牌：粘贴您之前创建的 OAuth 访问令牌。 • 存储库 URL：添加 Bitbucket 项目存储库的网址。URL 通常采用以下格式：`https://bitbucket.org/<WORKSPACE_NAME>/<REPO_NAME>` • 分支名称：该名称必须与您的 Bitbucket 存储库分支的名称相匹配。当您运行 AWS CloudFormation 堆栈时，此分支不需要存在，但它是向环境部署代码所必需的。 • 项目名称：与 Amplify 项目关联的名称。 5. 选择下一步，然后选择创建堆栈。	DevOps 工程师

测试 CI/CD 管道

Task	描述	所需技能
将代码部署到存储库中的分支。	1. 通过运行以下命令，克隆您的 Bitbucket 存储库：`git clone https://bitbucket.org/<WORKSPACE_NAME>/<REPO_NAME>` 2. 查看运行 AWS CloudFormation 脚本时使用的分支名称。要创建并签出新分支，请运行 `git checkout -b <BRANCH_NAME>` 命令。要检出现有分支，请运行 `git checkout <BRANCH_NAME>` 命令 3. 通过运行 `git commit` 和 `git push` 命令，将代码提交至分支，并推送至远程分支。 4. 然后，Amplify 会构建并部署此应用程序。 有关这方面的更多信息，请参阅 Bitbucket 文档中的 Basic Git 命令。	应用程序开发人员

相关资源

身份验证方法（Atlassian 文档）

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 Step Functions 和 Lambda 代理函数在 AWS 账户上启动 CodeBuild 项目

由 Richard Milner-Watts (AWS) 和 Amit Anjarlekar (AWS) 创作

摘要

此模式演示了如何使用 AWS Step Functions 和 AWS Lambda 代理函数跨多个 AWS 账户异步启动 AWS CodeBuild 项目。你可以使用模式的示例 Step Functions 状态机来测试你的 CodeBuild 项目是否成功。

CodeBuild 帮助您在完全托管的运行时环境中使用 AWS 命令行界面 (AWS CLI) Line Interface 启动操作任务。您可以通过覆盖环境变量来更改 CodeBuild 项目在运行时的行为。此外，您还可以使用 CodeBuild 来管理工作流程。有关更多信息，请参阅 AWS Workshop 网站上的服务目录工具和在 AWS 数据库博客上使用 AWS 在 Amazon RDS for PostgreSQL 中安排作业，在 CodeBuild AWS 数据库博客上使用 EventBridge亚马逊安排作业。

先决条件和限制

先决条件

- 两个活跃的 AWS 账户：一个用于通过 Step Functions 调用 Lambda 代理函数的源账户和一个用于构建远程示例项目的目标账户 CodeBuild

限制

- 此模式不能用于在账户之间复制构件。

架构

下图显示此模式构建的架构。

图表显示了以下工作流：

1. Step Functions 状态机解析提供的输入映射，并为您定义的每个账户、区域和项目调用 Lambda 代理函数 (`codebuild-proxy-lambda`)。
2. Lambda 代理函数使用 AWS Security Token Service (AWS STS`codebuild-proxy-role`) 来担任 IAM 代理角色 ()，该角色与目标账户中的 IAM 策略 (`codebuild-proxy-policy`) 关联。

3. 使用代入的角色，Lambda 函数启动 CodeBuild 项目并返回 CodeBuild 任务 ID。Step Functions 状态机循环并轮询 CodeBuild 作业，直到收到成功或失败状态。

状态机逻辑如下图所示。

技术堆栈

- AWS CloudFormation
- CodeBuild
- IAM
- Lambda
- Step Functions
- X-Ray

工具

- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。
- AWS CloudFormation Designer 提供了一个集成的 JSON 和 YAML 编辑器，可帮助您查看和编辑 CloudFormation 模板。
- AWS CodeBuild 是一项完全托管的构建服务，可帮助您编译源代码、运行单元测试和生成可随时部署的项目。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Step Functions 是一项无服务器编排服务，可让您搭配使用 AWS Lambda 函数和其他 Amazon Web Services 来构建业务关键型应用程序。
- AWS X-Ray – 帮助您收集您的应用程序所服务的请求的相关数据，并提供用于查看、筛选和获取数据洞察力的工具，以确定问题和发现优化机会。

代码

此模式的示例代码可在 GitHub跨账户 CodeBuild 代理存储库中找到。此模式使用 AWS Lambda Powertools for Python 库来提供日志和跟踪功能。有关此库及其实用程序的更多信息，请参阅 Powertools for AWS Lambda (Python)。

最佳实践

1. 调整 Step Function 状态机中的等待时间值，以最大限度地减少对作业状态的轮询请求。使用 CodeBuild 项目的预期执行时间。
2. 在 Step Functions 中调整地图的`MaxConcurrency`属性以控制可以并行运行的 CodeBuild 项目数量。
3. 如有必要，请查看示例代码是否已准备就绪。考虑解决方案可能记录哪些数据，以及默认的 Amazon CloudWatch 加密是否足够。

操作说明

在源账户中创建 Lambda 代理函数和关联 IAM 角色

Task	描述	所需技能
记录 AWS 账户 IDs。	需要 AWS 账户 IDs 才能设置跨账户的访问权限。 记录您的源账户和目标账户的 Amazon Web Services account ID。有关更多信息，请参阅 IAM 文档中的查找您 Amazon Web Services account ID。	AWS DevOps
下载 AWS CloudFormation 模板。	1. 从GitHub 存储库中下载此模式的 `sample_target_codebuild_template.yaml` AWS CloudFormation 模板。 2. 从GitHub 存储库中下载此模式的 `codebuild_lambda_proxy_temp`	AWS DevOps

Task	描述	所需技能
	`late.yaml` AWS CloudFormation 模板。 **Note** 在 AWS CloudFormation 模板中，`<SourceAccountId>` 是源账户的 AWS 账户 ID，`<TargetAccountId>` 也是目标账户的 AWS 账户 ID。	

Task	描述	所需技能
创建并部署 AWS CloudFormation 堆栈。	1. 登录您的源账户的 AWS 管理控制台，打开 AWS CloudFormation 控制台，然后选择 Stacks。 2. 选择创建堆栈，然后选择使用新资源（标准）。 3. 对于 Template source(模板来源)，选择 Upload a template file(上载模板文件)。 4. 对于上传模板文件，选择文件，然后选择您下载的 `codebuild_lambda_proxy_template.yaml` 文件。选择下一步。 5. 对于堆栈名称，输入堆栈名称 (例如 `codebuild-lambda-proxy`)。 6. **Note** 将 `crossAccountTargetRoleArn` 参数替换成您的 `<TargetAccountId>` （例如，`<arn:aws:iam::123456789012:role/proxy-lambda-codebuild-role>` ）。：您无	AWS DevOps

Task	描述	所需技能
	需更新**targetCodeBuildProject** 参数的默认值。 7. 选择下一步，接受默认堆栈创建选项，然后选择下一步。 8. 选中 “我确认 AWS CloudFormation 可能会使用自定义名称创建 IAM 资源” 复选框，然后选择 “创建堆栈”。 **Note** 在目标账户中创建任何资源之前，您必须为代理 Lambda 函数创建 AWS CloudFormation 堆栈。当您在目标账户中创建信任策略时，IAM 角色会从角色名称转换至内部标识符。所以 IAM 角色必须已经存在。	

Task	描述	所需技能
确认代理函数和状态机已创建。	1. 等待 AWS CloudFormation 堆栈达到 CREATE_COMPLETE 状态。这应该需要不到 1 分钟的时间。 2. 打开 AWS Lambda 控制台，选择函数，然后查找 `lambda-proxy-ProxyLambda-<GUID>` 函数。 3. 打开 AWS Step Functions 控制台，选择状态机，然后查找 `sample-crossaccount-codebuild-state-machine` 状态机。	AWS DevOps

在目标账户中创建 IAM 角色并启动示例 CodeBuild 项目

Task	描述	所需技能
创建并部署 AWS CloudFormation 堆栈。	1. 登录目标账户的 AWS 管理控制台，打开 AWS CloudFormation 控制台，然后选择 Stacks。 2. 选择创建堆栈，然后选择使用新资源（标准）。 3. 对于 Template source(模板来源)，选择 Upload a template file(上载模板文件)。 4. 对于上传模板文件，选择选择文件，然后选择`sample_target_code`	AWS DevOps

Task	描述	所需技能
	build_template.yaml 文件。选择下一步。 5. 对于堆栈名称，请为堆栈输入名称（例如，sample-codebuild-stack ）。 6. 将 crossAccountSourceRoleArn 参数替换成您的 <SourceAccountId> （例如，<arn:aws:iam::123456789012:role/codebuild-proxy-lambda-role> ）。 7. 选择下一步，接受默认堆栈创建选项，然后选择下一步。 8. 选中 “我确认 AWS CloudFormation 可能会使用自定义名称创建 IAM 资源” 复选框，然后选择 “创建堆栈”。	
验证示例 CodeBuild 项目的创建。	1. 等待 AWS CloudFormation 堆栈达到 CREATE_COMPLETE 状态。这应该需要不到 1 分钟的时间。 2. 打开 AWS CodeBuild 控制台，然后找到该sample-codebuild-project 项目。	AWS DevOps

测试跨账户 Lambda 代理函数

Task	描述	所需技能
启动状态机。	1. 登录您的源账户的 AWS 管理控制台，打开 AWS Step Functions 控制台，选择状态机。 2. 选择 `sample-crossaccount-codebuild-state-machine` 状态机，然后选择开始执行。 3. 在输入编辑器中，输入以下 JSON，然后<TargetAccountID> 替换为包含该 CodeBuild 项目的账户的 AWS 账户 ID。 `{ "crossAccountTargetRoleArns": [ { "arn": "arn:aws:iam::<TargetAccountID>:role/proxy-lambda-codebuild-role", "region": "eu-west-1", "codeBuildProject": "sample-codebuild-project", "SampleValue1": "Value1", "SampleValue2": "Value2" } ] }`	AWS DevOps

Task	描述	所需技能
	Note 键值对作为环境变量从源账户中的函数传递到目标账户中的 CodeBuild 项目。 4. 选择启动执行。 5. 在状态机页面的详细信息选项卡，检查执行状态是否设置为成功。这可以确认您的状态机正运行。注意：状态机可能需要大约 30 秒才能达到成功状态。 6. 要查看状态机中某个 步骤的输出和输入，请在执行事件历史记录部分中展开该步骤。例如，展开 Lambda-CodeBuild 代理-启动步骤。输出包括有关被覆盖的环境变量、原始负载和 CodeBuild 作业 ID 的详细信息。	

Task	描述	所需技能
验证环境变量。	1. 使用您的目标账户登录 AWS 管理控制台。 2. 打开 AWS CodeBuild 控制台，展开 “构建”，然后选择 “构建项目”。 3. 选择`sample-codebuild-project` 项目，然后选择 “查看详细信息”。 4. 在 “生成历史记录” 选项卡上，选择项目的最新版本，然后选择 “查看日志”。 5. 在日志输出中，验证打印至 STDOUT 的环境变量是否与 Step Functions 示例状态机中的环境变量相匹配。	AWS DevOps

故障排除

事务	解决方案
Step Functions 的执行时间比预期的要长。	在 Step Function 状态机中调整地图的 `MaxConcurrency` 属性以控制可以并行运行多少 CodeBuild 项目。
CodeBuild 任务的执行时间比预期的要长。	1. 调整 Step Functions 状态机中的等待时间值，以最大限度地减少对作业状态的轮询请求。使用 CodeBuild 项目的预期执行时间。 2. 考虑使用的工具 CodeBuild 是否合适。例如，初始化 CodeBuild 任务所需的时间可能比 AWS Lambda 长得多。如果需要高吞吐量和快速完成时间，可以考虑将业务逻辑迁移到 AWS Lambda 并使用扇出架构。

使用应用程序恢复控制器管理 EMR 集群的多可用区故障转移

由 Aarti Rajput (AWS)、Ashish Bhatt (AWS)、Neeti Mishra (AWS) 和 Nidhi Sharma (AWS) 创作

摘要

这种模式为 Amazon EMR 工作负载提供了有效的灾难恢复策略，有助于确保单个可用区内的多个可用区之间的高可用性和数据一致性。 AWS 区域该设计使用 Amazon 应用程序恢复控制器和应用程序负载均衡器来管理基于 Apache Spark 的 EMR 集群的故障转移操作和流量分配。

在标准条件下，主可用区托管活动的 EMR 集群和具有完整读/写功能的应用程序。如果可用区意外出现故障，流量将自动重定向到辅助可用区，在那里启动新的 EMR 集群。两个可用区都通过专用网关终端节点访问共享的亚马逊简单存储服务 (Amazon S3) 存储桶，从而确保一致的数据管理。这种方法可最大限度地减少停机时间，并在可用区故障期间实现关键大数据工作负载的快速恢复。该解决方案在金融或零售等行业非常有用，在这些行业中，实时分析至关重要。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 亚马逊弹性计算云（亚马逊）上的亚马逊 EMR EC2
- 从 EMR 集群的主节点访问 Amazon S3。
- AWS 多可用区基础架构

限制

- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额页面，然后选择服务的链接。

产品版本

- 亚马逊 EMR 6.x 及更高版本

架构

目标技术堆栈

- 亚马逊 EMR 集群
- Amazon Application Recovery Controller
- 应用程序负载均衡器
- Amazon S3 存储桶
- 适用于 Amazon S3 的网关端点

目标架构

该架构通过使用多个可用区并通过应用程序恢复控制器实施自动恢复机制来提供应用程序弹性。

1. Application Load Balancer 将流量路由到活动的 Amazon EMR 环境，该环境通常是主可用区中的主 EMR 集群。
2. 活动的 EMR 集群处理应用程序请求，并通过其专用 Amazon S3 网关终端节点连接到 Amazon S3 以进行读取和写入操作。
3. Amazon S3 充当中央数据存储库，可能用作检查点或 EMR 集群之间的共享存储。

   EMR 集群通过`s3://`协议和 EM R 文件系统 (EMRFS) 直接写入 Amazon S3 时可以保持数据一致性。为了确保数据完整性，此模式下的解决方案对 Amazon S3 实施了预写日志 (WAL)，并使用 Amazon S3 版本控制功能来跟踪数据版本并在需要时启用回滚。对于读取操作，集群使用 Amazon S3 S elect 访问共享的 Amazon S3 存储层以优化性能，再加上 Spark 缓存机制以最大限度地减少重复访问 Amazon S3。Amazon S3 旨在实现跨多个可用区域的 99.999999999% 的持久性，提供原生 Amazon EMR 集成，并提供高度可靠的跨集群数据一致性解决方案。
4. Application Recovery Controller 持续监控主可用区的运行状况，并在必要时自动管理故障转移操作。
5. 如果应用程序恢复控制器在主 EMR 集群中检测到故障，它将采取以下措施：
   - 启动到可用区 2 中辅助 EMR 群集的故障转移过程。
   - 更新路由配置以将流量引导到辅助群集。

工具

Amazon Web Services

- Amazon 应用程序恢复控制器可帮助您管理和协调跨 AWS 区域 可用区的应用程序恢复。该服务通过减少传统工具和流程所需的手动步骤，简化了流程并提高了应用程序恢复的可靠性。

- Application Load Balancer 在应用层运行，这是开放系统互联 (OSI) 模型的第七层。它在多个可用区中的多个目标（例如 EC2 实例）之间分配传入的应用程序流量。这将提高应用程序的可用性。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行 shell 中的命令进行交互。
- Amazon EMR 是一个大数据平台，可为 Apache Spark、Apache Hive 和 Presto 等开源框架提供数据处理、交互式分析和机器学习。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- Amazon S3 提供了一个简单的 Web 服务接口，您可以使用该接口随时随地存储和检索任意数量的数据。使用此服务，您可以轻松构建利用云原生存储的应用程序。
- Amazon S3 的网关终端节点是您在路由表中指定的网关，用于通过 AWS 网络从您的虚拟私有云 (VPC) 访问 Amazon S3。

最佳实践

- 遵循安全、身份和合规性方面AWS 的最佳实践，以确保架构的强大和安全。
- 使架构与 Well-Architect AWS ed 框架保持一致。
- 使用 Amazon S3 访问授权来管理从基于 Spark 的 EMR 集群到 Amazon S3 的访问权限。有关详细信息，请参阅博客文章 “使用带有 S3 访问权限的 Amazon EMR 来扩展 Spark 对 Amazon S3 的访问权限。
- 使用亚马逊 S3 提高 Spark 的性能。

操作说明

设置您的环境

Task	描述	所需技能
登录到 AWS Management Console。	以 IAM 用户身份登录 AWS Management Console。有关说明，请参阅AWS 文档。	AWS DevOps
配置 AWS CLI。	安装 AWS CLI 或将其更新到最新版本，这样您就可以在 AWS 服务 中与之交互 AWS	AWS DevOps

Task	描述	所需技能
	Management Console。有关说明，请参阅AWS CLI 文档。	

在你的 EMR 集群上部署 Spark 应用程序

Task	描述	所需技能
创建 S3 存储桶。	1. 创建 S3 存储桶来存储输入数据集、日志、应用程序和输出数据。有关说明，请参阅 Amazon S3 文档。 2. 将存储桶整理到不同的文件夹中，用于存放输入数据 (`datasetlogs`)、日志 ()、Spark 应用程序 (`spark-app`) 和输出数据 (`output`)。	AWS DevOps
创建 EMR 集群。	1. 使用以下 AWS CLI 命令创建一个 EMR 集群（例如，版本 6.12 或更高版本），其实例跨越两个可用区（例如`us-east-1a` 和`us-east-1b` ），以实现高可用性。该命令以指定`m4.large`实例类型为例。 `aws emr create-cluster \ --ec2-attributes AvailabilityZone=<AZ-name-1> \ --release-label emr-6.12.0 \`	AWS DevOps

Task	描述	所需技能

	<pre>--instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large</pre>
	<pre>aws emr create-cluster \
	--ec2-attributes AvailabilityZone=<AZ-name-2> \
	--release-label emr-6.12.0 \
	--instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large</pre>
	有关更多信息，请参阅 create-cluster 命令和 Amazon EMR 文档。
	2. 必要时为密钥对、服务角色和实例配置文件提供所需的权限。	

Task	描述	所需技能
为 EMR 集群配置安全设置。	1. 使用 d AWS CLI es cribe-cluster 命令识别与 EMR 集群的主节点关联的安全组： `aws emr describe-cluster --cluster-id j-XXXXXXXX` 2. 要增强安全性，请修改安全组设置以允许安全外壳 (SSH) 访问主节点（TCP 端口 22），但仅限于您的特定 IP 地址。 有关更多信息，请参阅 Amazon EMR 文档。	AWS DevOps

连接到 EMR 集群。	使用提供的密钥对，通过 SSH 连接到 EMR 集群的主节点。
	确保 key pair 文件与您的应用程序位于同一目录中。
	运行以下命令为 key pair 设置正确的权限并建立 SSH 连接：
	`chmod 400 <key-pair-name>`
	`ssh -i ./<key-pair-name> hadoop@<master-node-public-dns>`
		AWS DevOps

Task	描述	所需技能

部署 Spark 应用程序。	建立 SSH 连接后，您将进入 Hadoop 控制台。
	
	1. 使用诸如 vim 之类的文本编辑器创建或编辑 Spark 应用程序文件 (`main.py`)：
	
	`vim main.py`
	
	有关创建和修改 Spark 应用程序的更多信息，请参阅 Amazon EMR 文档。
	2. 将 Spark 应用程序提交到 EMR 集群，在 S3 存储桶中指定输入数据和输出数据的位置：
	
	`spark-submit main.py –data_source <input-data-folder-in-s3> –output_uri <output-folder-in-s3>`
	
	例如（基于您之前设置的文件夹）：
	
	`spark-submit main.py –data_source dataset –output_uri output`
	3. 通过检查应用程序日志来监控应用程序的进度：
	
	`yarn logs -applicationId <application-id>`
		AWS DevOps

Task	描述	所需技能
监控 Spark 应用程序。	1. 打开另一个终端窗口，建立通往 EMR 集群资源管理器 Web UI 的 SSH 隧道： `ssh -i <key-pair-name> -N -L 8157:<resource-manager-public-dns>:8088 hadoop@<resource-manager-public-dns>` 2. 要监控应用程序，请在 Web 浏览器`http://localhost:8157` 中导航到资源管理器 Web UI。	AWS DevOps

将流量转移到另一个可用区

Task	描述	所需技能
创建应用程序负载均衡器。	设置目标组，在部署在两个可用区的 Amazon EMR 主节点之间路由流量。 AWS 区域 有关说明，请参阅 El astic Load Balancing 文档中的为应用程序负载均衡器创建目标组。	AWS DevOps
在应用程序恢复控制器中配置区域偏移。	在此步骤中，您将使用应用程序恢复控制器中的区域转移功能将流量转移到另一个可用区。 1. 打开应用程序恢复控制器控制台。	AWS DevOps

Task	描述	所需技能
	2. 在 “入门” 下，选择区域移动、开始区域移动。 3. 选择您要转移流量的可用区。 4. 从 “资源” 表中为区域转移选择支持的资源（例如 Application Load Balancer）。 5. 在设置可用区转移到期时间中，选择或输入可用区转移的到期时间。您可以将持续时间设置在 1 分钟到三天（72 小时）之间。 所有可用区转移都是暂时的。您必须设置到期时间，但稍后可以更新活动的可用区转移，以设置新的到期时间，最长是三天。 6. 输入有关此区域偏移的评论。 7. 选中该复选框以确认启动可用区转移，这会将流量移离该可用区，从而减少应用程序的可用容量。 8. 选择启动。 要使用 AWS CLI，请参阅应用程序恢复控制器文档中的 AWS CLI 带区域偏移的使用示例。	

Task	描述	所需技能

验证区域移位配置和进度。	1. 验证使用区域移位注册的资源：
	
	`aws arc-zonal-shift list-managed-resources --region <AWS-region-name>`
	
	例如，以下输出确认两个可用区中的资源均已启动并运行。
	
	`"appliedWeights": { "use1-az1": 1.0, "use1-az2": 1.0 },`
	
	2. 要可视化区域偏移，请使用以下 AWS CLI 命令开始区域偏移：
	
	`aws arc-zonal-shift start-zonal-shift \ --resource-identifier <application-load-balancer-arn> \ --away-from <source-AZ> \ --expires-in 10m --comment "testing" \ --region <AWS-region-name>`
	
	其中<source-AZ> 是您要转移流量的可用区域的标识符，<applicat
		AWS DevOps

Task	描述	所需技能

	ion-load-balancer-arn> 也是应用程序负载均衡器的亚马逊资源名称 (ARN)。
	3. 确认流量已转移到另一个可用区。
	`aws arc-zonal-shift get-managed-resource \ --resource-identifier <application-load-balancer-arn> \ --region <AWS-region-name>`
	你可以看到这些权重所证实的区域偏移：
	`"appliedWeights": { "use1-az1": 0.0, "use1-az2": 1.0 },`	

相关资源

- AWS CLI 命令：
  - create-cluster
  - describe-cluster
  - arc-zonal-shift
- 为@@ 竞价型实例配置 Amazon EMR 集群实例类型和最佳实践（亚马逊 EMR 文档）
- IAM 中的安全最佳实践（IAM 文档）
- 使用实例配置文件（IAM 文档）

- 使用区域移位和区域自动切换来恢复 ARC 中的应用程序（应用程序恢复控制器文档）

使用 AWS 代码服务和 AWS KMS 多区域密钥，管理微服务到多个账户和区域的蓝/绿部署

由 Balaji Vedagiri (AWS)、Ashish Kumar (AWS)、Faisal Shahdad (AWS)、Anand Krishna Varanasi (AWS)、Vanitha Dontireddy (AWS) 和 Vivek Thangamuthu (AWS) 创建

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

此模式描述了如何根据蓝/绿部署策略将全局微服务应用程序从中央 Amazon Web Services account 部署到多个工作负载账户和区域。该模式支持以下内容：

- 软件是在一个中心账户中开发的，而工作负载和应用程序则分布在多个账户和 AWS 区域中。
- 单个 AWS 密钥管理系统（AWS KMS）多区域密钥用于加密和解密，以涵盖灾难恢复。
- KMS 密钥是特定于区域的，必须在三个不同的区域中维护或创建管道构件。KMS 多区域密钥有助于跨区域保留相同的密钥 ID。
- Git 工作流分支模型由两个分支（开发分支和主分支）实现，使用拉取请求 (PRs) 合并代码。从此堆栈部署的 AWS Lambda 函数创建了一个从开发分支到主分支的 PR。PR 合并到主分支会启动 AWS CodePipeline 管道，该管道协调持续集成和持续交付 (CI/CD) 流程，并将堆栈部署到各个账户。

此模式提供了通过 AWS CloudFormation 堆栈设置的基础设施即代码 (IaC) 示例，以演示此用例。微服务的蓝/绿部署是使用 AWS 实现的。 CodeDeploy

先决条件和限制

先决条件

- 四个活跃的 Amazon Web Services account：
  - 用于管理代码管道和维护 AWS CodeCommit 存储库的工具账户。
  - 用于部署微服务工作负载的三个工作负载（测试）账户。
- 此模式使用以下区域。如果您想使用其他区域，则必须对 AWS CodeDeploy 和 AWS KMS 多区域堆栈进行适当的修改。
  - 工具 (AWS CodeCommit) 账户：`ap-south-1`
  - 工作负载（测试）账户 1：`ap-south-1`

  - 工作负载(测试)账户 2： `eu-central-1`
  - 工作负载(测试)账户 3： `us-east-1`
- 每个工作负载账户中用于部署区域的三个 Amazon Simple Storage Service (Amazon S3) 桶。（在此模式中，它们稍后被称为 `S3BUCKETNAMETESTACCOUNT1`、`S3BUCKETNAMETESTACCOUNT2` 和 `S3BUCKETNAMETESTACCOUNT3` 。）

  例如，您可以在特定账户和区域中使用唯一的桶名称创建这些桶，如下所示（将 xxxx 替换为随机数）：

```
##In Test Account 1
aws s3 mb s3://ecs-codepipeline-xxxx-ap-south-1 --region ap-south-1
##In Test Account 2
aws s3 mb s3://ecs-codepipeline-xxxx-eu-central-1 --region eu-central-1
##In Test Account 3
aws s3 mb s3://ecs-codepipeline-xxxx-us-east-1 --region us-east-1

#Example
##In Test Account 1
aws s3 mb s3://ecs-codepipeline-18903-ap-south-1 --region ap-south-1
##In Test Account 2
aws s3 mb s3://ecs-codepipeline-18903-eu-central-1 --region eu-central-1
##In Test Account 3
aws s3 mb s3://ecs-codepipeline-18903-us-east-1 --region us-east-1
```

限制

该模式使用 AWS CodeBuild 和其他配置文件来部署示例微服务。如果您有不同的工作负载类型（例如无服务器），则必须更新所有相关配置。

架构

目标技术堆栈

- AWS CloudFormation
- AWS CodeCommit
- AWS CodeBuild
- AWS CodeDeploy
- AWS CodePipeline

目标架构

自动化和扩缩

使用 AWS CloudFormation 堆栈模板 (IaC) 自动完成设置。它可以轻松扩展到多个环境和账户。

工具

Amazon Web Services

- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。
- AWS CodeBuild 是一项完全托管的构建服务，可帮助您编译源代码、运行单元测试和生成可随时部署的项目。
- AWS CodeCommit 是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。
- AWS CodeDeploy 可自动部署到亚马逊弹性计算云 (Amazon EC2) 或本地实例、AWS Lambda 函数或亚马逊弹性容器服务 (Amazon ECS) 服务。
- AWS CodePipeline 可帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件变更所需的步骤。
- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- Amazon Elastic Container Service (Amazon ECS)是一项快速且可扩展的容器管理服务，可帮助运行、停止和管理集群上的容器。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥，以帮助保护您的数据。
- Amazon Simple Storage Service(Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

其他工具

- Git 是一个开源的分布式版本控制系统，可与 AWS CodeCommit 存储库配合使用。
- Docker 是一组平台即服务（PaaS）产品，它们使用操作系统级别的虚拟化技术在容器中交付软件。该模式使用 Docker 在本地构建和测试容器映像。
- cfn-lint 和 cfn-nag 是开源工具，可帮助您检查 CloudFormation 堆栈中是否存在任何错误和安全问题。

代码存储库

此模式的代码可在多个地区的 GitHub 全球蓝/绿部署和账户存储库中找到。

操作说明

设置环境变量

Task	描述	所需技能

导出用于 CloudFormation 堆栈部署的环境变量。	定义环境变量，这些变量将在此模式的后面用作 CloudFormation 堆栈的输入。
	1. 按照前面先决条件一节中的说明，更新您在三个账户和区域中创建的桶名称：
	`export S3BUCKETNAMETESTACCOUNT1=<S3BUCKETACCOUNT1>`
	`export S3BUCKETNAMETESTACCOUNT2=<S3BUCKETACCOUNT2>`
	`export S3BUCKETNAMETESTACCOUNT3=<S3BUCKETACCOUNT3>`
	2. 定义一个随机字符串来创建构件桶，因为桶名称必须是全局唯一的：
	`export BUCKETSTARTNAME=ecs-codepipeline-artifacts-19992`
	3. 定义并导出账户 IDs 和区域：
		AWS DevOps

Task	描述	所需技能

	`export TOOLSACCOUNT=<TOOLSACCOUNT>`
	`export CODECOMMITACCOUNT=<CODECOMMITACCOUNT>`
	`export CODECOMMITREGION=ap-south-1`
	`export CODECOMMITREPONAME=Poc`
	`export TESTACCOUNT1=<TESTACCOUNT1>`
	`export TESTACCOUNT2=<TESTACCOUNT2>`
	`export TESTACCOUNT3=<TESTACCOUNT3>`
	`export TESTACCOUNT1REGION=ap-south-1`
	`export TESTACCOUNT2REGION=eu-central-1`
	`export TESTACCOUNT3REGION=us-east-1`
	`export TOOLSACCOUNTREGION=ap-south-1`
	`export ECRREPOSITORYNAME=web`	

Package 并部署 CloudFormation 基础架构的堆栈

Task	描述	所需技能
克隆存储库。	将示例存储库克隆到您工作位置的新存储库中： `##In work location`	AWS DevOps

Task	描述	所需技能
	`git clone https://github.com/aws-samples/ecs-blue-green-global-deployment-with-multiregion-cmk-codepipeline.git`	

Task	描述	所需技能

打包 Cloudformation 资源。	在此步骤中，您将打包 CloudFormation 模板引用的本地项目，以创建亚马逊虚拟私有云 (Amazon VPC) 和 Application Load Balancer等服务所需的基础设施资源。
	
	这些模板位于代码存储库的 `Infra` 文件夹中。
	
	`##In TestAccount1## aws cloudformation package \ --template-file mainInfraStack.yaml \ --s3-bucket $S3BUCKETNAMETESTACCOUNT1 \ --s3-prefix infraStack \ --region $TESTACCOUNT1REGION \ --output-template-file infrastructure_${TESTACCOUNT1}.template`
	
	`##In TestAccount2## aws cloudformation package \ --template-file mainInfraStack.yaml \ --s3-bucket $S3BUCKETNAMETESTACCOUNT2 \ --s3-prefix infraStack \ --region $TESTACCOUNT2REGION \`
		AWS DevOps

Task	描述	所需技能
	--output-template-file infrastructure_${TESTACCOUNT2}.template ##In TestAccount3## aws cloudformation package \ --template-file mainInfraStack.yaml \ --s3-bucket $S3BUCKETNAMETESTACCOUNT3 \ --s3-prefix infraStack \ --region $TESTACCOUNT3REGION \ --output-template-file infrastructure_${TESTACCOUNT3}.template	

Task	描述	所需技能

验证程序包模板。	验证程序包模板：
	
	`aws cloudformation validate-template \ --template-body file://infrastructure_${TESTACCOUNT1}.template`
	
	`aws cloudformation validate-template \ --template-body file://infrastructure_${TESTACCOUNT2}.template`
	
	`aws cloudformation validate-template \ --template-body file://infrastructure_${TESTACCOUNT3}.template`
		AWS DevOps

Task	描述	所需技能

将包文件部署到工作负载账户中，	1. 根据您的设置更新 infraParameters.json 脚本中的占位符值和账户名称。
	2. 将包模板部署到您的三个工作负载账户中。
	
	`##In TestAccount1##`
	`aws cloudformation deploy \`
	`--template-file infrastructure_${TESTACCOUNT1}.template \`
	`--stack-name mainInfrastack \`
	`--parameter-overrides file://infraParameters.json \`
	`--region $TESTACCOUNT1REGION \`
	`--capabilities CAPABILITY_IAM CAPABILITY_NAMED_IAM`
	
	`##In TestAccount2##`
	`aws cloudformation deploy \`
	`--template-file infrastructure_${TESTACCOUNT2}.template \`
	`--stack-name mainInfrastack \`
	`--parameter-overrides file://infraParameters.json \`
	`--region $TESTACCOUNT2REGION \`
		AWS DevOps

Task	描述	所需技能

	``
	    --capabilities
	 CAPABILITY_IAM
	 CAPABILITY_NAMED_I
	AM
	
	##In TestAccount3##
	aws cloudformation
	 deploy \
	    --template-file
	 infrastructure_${T
	ESTACCOUNT3}.templ
	ate \
	    --stack-name
	 mainInfrastack \
	    --parameter-
	overrides file://in
	fraParameters.json \
	    --region $TESTACCO
	UNT3REGION \
	    --capabilities
	 CAPABILITY_IAM
	 CAPABILITY_NAMED_I
	AM
	``	

推送示例图像并扩展 Amazon ECS

Task	描述	所需技能
将示例图像推送到 Amazon ECR 存储库。	将一个示例 (NGINX) 图像推送到名为 web 的 Amazon Elastic Container Registry (Amazon ECR) 存储库中（如参数中所设置）。您可以根据需要自定义图像。 要登录并设置用于将图像推送到 Amazon ECR 的凭证，请按	AWS DevOps

Task	描述	所需技能

	照 Amazon ECR 文档中的说明进行操作。
	
	命令包括：
	
	`docker pull nginx`
	`docker images`
	`docker tag <imageid> aws_account_id.dkr.ecr.region.amazonaws.com/<web>:latest`
	`docker push <aws_account_id>.dkr.ecr.<region>.amazonaws.com/<web>:tag`	

扩展 Amazon ECS 并验证访问权限。	1. 扩展 Amazon ECS 以创建两个副本： `aws ecs update-service --cluster QA-Cluster --service Poc-Service --desired-count 2` 其中，`Poc-Service` 指的是您的示例应用程序。 2. 通过使用浏览器中的完全限定域名 (FQDN) 或 DNS 或使用 curl 命令，验证是否可以从应用程序负载均衡器访问服务。	AWS DevOps

设置代码服务和资源

Task	描述	所需技能
在工具账户中创建 CodeCommit 存储库。	使用模板在工具帐户中创建 CodeCommit 存储库，该`codecommit.yaml` 模板位于 GitHub 存储库的`code`文件夹中。您只能在计划开发代码的单个区域中创建此存储库。 `aws cloudformation deploy --stack-name codecommitrepoStack --parameter-overrides CodeCommitReponame= $CODECOMMITREPONAME \ ToolsAccount=$TOOLSACCOUNT --template-file codecommit.yaml --region $TOOLSACCOUNTREGION \ --capabilities CAPABILITY_NAMED_IAM`	AWS DevOps
创建 S3 存储桶，用于管理由生成的项目 CodePipeline。	创建一个 S3 存储桶，用于管理 CodePipeline 通过使用`pre-reqs-bucket.yaml` 模板生成的项目，该模板位于 GitHub 存储库的`code`文件夹中。堆栈必须部署在所有三个工作负载（测试）和工具账户和区域中。 `aws cloudformation deploy --stack-name pre-reqs-artifacts-bucket --parameter-overrides BucketSta`	AWS DevOps

Task	描述	所需技能
	rtName=$BUCKETSTAR TNAME \ TestAccount1=$TE STACCOUNT1 TestAccou nt2=$TESTACCOUNT2 \ TestAccount3=$TE STACCOUNT3 CodeCommi tAccount=$CODECOMM ITACCOUNT ToolsAcco unt=$TOOLSACCOUNT \ --template-file pre- reqs_bucket.yaml --region $TESTACCO UNT1REGION --capabil ities CAPABILIT Y_NAMED_IAM aws cloudformation deploy --stack-name pre-reqs-artifacts -bucket --parameter- overrides BucketSta rtName=$BUCKETSTAR TNAME \ TestAccount1=$TE STACCOUNT1 TestAccou nt2=$TESTACCOUNT2 \ TestAccount3=$TE STACCOUNT3 CodeCommi tAccount=$CODECOMM ITACCOUNT ToolsAcco unt=$TOOLSACCOUNT \ --template-file pre- reqs_bucket.yaml --region $TESTACCO UNT2REGION --capabil ities CAPABILIT Y_NAMED_IAM aws cloudformation deploy --stack-name pre-reqs-artifacts	

Task	描述	所需技能
	-bucket --parameter-overrides BucketStartName=$BUCKETSTARTNAME \ TestAccount1=$TESTACCOUNT1 TestAccount2=$TESTACCOUNT2 \ TestAccount3=$TESTACCOUNT3 CodeCommitAccount=$CODECOMMITACCOUNT ToolsAccount=$TOOLSACCOUNT \ --template-file pre-reqs_bucket.yaml --region $TESTACCOUNT3REGION --capabilities CAPABILITY_NAMED_IAM aws cloudformation deploy --stack-name pre-reqs-artifacts-bucket --parameter-overrides BucketStartName=$BUCKETSTARTNAME \ TestAccount1=$TESTACCOUNT1 TestAccount2=$TESTACCOUNT2 \ TestAccount3=$TESTACCOUNT3 CodeCommitAccount=$CODECOMMITACCOUNT ToolsAccount=$TOOLSACCOUNT \ --template-file pre-reqs_bucket.yaml --region $TOOLSACCOUNTREGION --capabilities CAPABILITY_NAMED_IAM	

Task	描述	所需技能
设置多区域 KMS 密钥。	1. 使用要使用的主密钥和副本密钥创建多区域 KMS 密钥。 CodePipeline 在我们的示例中，`ToolsAccount1region - ap-south-1` 将是主要区域。 `aws cloudformation deploy --stack-name ecs-codepipeline-pre-reqs-KMS \ --template-file pre-reqs_KMS.yaml --parameter-overrides \ TestAccount1=$TESTACCOUNT1 TestAccount2=$TESTACCOUNT2 \ TestAccount3=$TESTACCOUNT3 CodeCommitAccount=$CODECOMMITACCOUNT ToolsAccount=$TOOLSACCOUNT --region $TOOLSACCOUNTREGION` 2. 设置要传递给项目的 CMKARN 变量。 CodeBuild 这些值可在 ecs-codepipeline-pre-reqs-KMS 模板堆栈的输出中找到（密钥 ID 在所有区域中都相同，开头为`mrk-`）。或者，您也可以从工具账户中获取 CMKARN 值。在所有账户会话中将其导出：	AWS DevOps

Task	描述	所需技能

	`export CMKARN1=arn:aws:kms:ap-south-1:<TOOLSACCOUNTID>:key/mrk-xxx`
	`export CMKARN2=arn:aws:kms:eu-central-1:<TOOLSACCOUNTID>:key/mrk-xxx`
	`export CMKARN3=arn:aws:kms:us-east-1:<TOOLSACCOUNTID>:key/mrk-xxx`
	`export CMARNTOOLS=arn:aws:kms:ap-south-1:<TOOLSACCOUNTID>:key/mrk-xxx`	

Task	描述	所需技能

在工具账户中设置 CodeBuild 项目。	1. 使用 GitHub 存储库code文件夹中的codebuild_IAM.yaml 模板在工具账户的单个区域 CodeBuild 中为 AWS 设置适用于 AWS 的 AWS 身份与访问管理 (IAM)：
	`#In ToolsAccount aws cloudformation deploy --stack-name ecs-codebuild-iam \ --template-file codebuild_IAM.yaml --region $TOOLSACCOUNTREGION \ --capabilities CAPABILITY_NAMED_IAM`
	2. 使用codebuild.yaml 模板为您的构建项目 CodeBuild 进行设置。在所有三个区域部署此模板，如下所示：
	`aws cloudformation deploy --stack-name ecscodebuildstack --parameter-overrides ToolsAccount=$TOOLSACCOUNT \ CodeCommitRepoName=$CODECOMMITREPONAME ECRRepositoryName=$ECRREPOSITORYNAME APPACCOUNTID=$TESTACCOUNT1 \`
		AWS DevOps

Task	描述	所需技能
	TestAccount3=$TE STACCOUNT3 CodeCommi tRegion=$CODECOMMI TREGION CMKARN=$C MKARN1 \ --template-file codebuild.yaml --region $TESTACCO UNT1REGION --capabil ities CAPABILIT Y_NAMED_IAM aws cloudformation deploy --stack-name ecscodebuildstack -- parameter-overrides ToolsAccount=$TOOL SACCOUNT \ CodeCommitRepoName= $CODECOMMITREPONAME ECRRepositoryName= $ECRREPOSITORYNAME APPACCOUNTID=$TEST ACCOUNT2 \ TestAccount3=$TE STACCOUNT3 CodeCommi tRegion=$CODECOMMI TREGION CMKARN=$C MKARN2 \ --template-file codebuild.yaml --region $TESTACCO UNT2REGION --capabil ities CAPABILIT Y_NAMED_IAM aws cloudformation deploy --stack-name ecscodebuildstack -- parameter-overrides ToolsAccount=$TOOL SACCOUNT \	

Task	描述	所需技能
	CodeCommitRepoName=$CODECOMMITREPONAME ECRRepositoryName=$ECRREPOSITORYNAME APPACCOUNTID=$TESTACCOUNT3 \ CodeCommitRegion=$CODECOMMITREGION CMKARN=$CMKARN3 \ --template-file codebuild.yaml --region $TESTACCOUNT3REGION --capabilities CAPABILITY_NAMED_IAM	

Task	描述	所需技能

CodeDeploy 在工作负载帐户中进行设置。	使用 GitHub 存储库`code`文件夹中的`codedeploy.yaml` 模板在所有三个工作负载帐户 CodeDeploy 中进行设置。的输出`mainInfraStack` 包括 Amazon ECS 集群的亚马逊资源名称 (ARNs) 和 Application Load Balancer 侦听器。
	
	Note
	基础设施堆栈中的值已经导出，因此它们由 CodeDeploy 堆栈模板导入。
	
	`##WorkloadAccount1##`
	`aws cloudformation deploy --stack-name ecscodedeploystack \`
	`--parameter-overrides ToolsAccount=$TOOLSACCOUNT mainInfrastackname=mainInfrastack \`
	`--template-file codedeploy.yaml --region $TESTACCOUNT1REGION --capabilities CAPABILITY_NAMED_IAM`
	
	`##WorkloadAccount2##`
	`aws cloudformation deploy --stack-name ecscodedeploystack \`
		AWS DevOps

Task	描述	所需技能

	`--parameter-overrides ToolsAccount=$TOOLSACCOUNT mainInfrastackname=mainInfrastack \ --template-file codedeploy.yaml --region $TESTACCOUNT2REGION --capabilities CAPABILITY_NAMED_IAM`
	
	`##WorkloadAccount3## aws cloudformation deploy --stack-name ecscodedeploystack \ --parameter-overrides ToolsAccount=$TOOLSACCOUNT mainInfrastackname=mainInfrastack \ --template-file codedeploy.yaml --region $TESTACCOUNT3REGION --capabilities CAPABILITY_NAMED_IAM`	

CodePipeline 在工具账户中设置

Task	描述	所需技能
在工具账户中创建代码管道。	在工具账户中，运行以下命令： `aws cloudformation deploy --stack-name ecscodepipelinestack`	AWS DevOps

Task	描述	所需技能
	--parameter-overrides \ TestAccount1=$TESTACCOUNT1 TestAccount1Region=$TESTACCOUNT1REGION \ TestAccount2=$TESTACCOUNT2 TestAccount2Region=$TESTACCOUNT2REGION \ TestAccount3=$TESTACCOUNT3 TestAccount3Region=$TESTACCOUNT3REGION \ CMKARNTools=$CMKTROOLSARN CMKARN1=$CMKARN1 CMKARN2=$CMKARN2 CMKARN3=$CMKARN3 \ CodeCommitRepoName=$CODECOMMITREPONAME BucketStartName=$BUCKETSTARTNAME \ --template-file codepipeline.yaml --capabilities CAPABILITY_NAMED_IAM	

Task	描述	所需技能
在 AWS KMS 密钥策略 CodePipeline 和 S3 存储桶策略中提供访问权限和 CodeBuild 角色。	1. 在 AWS KMS 密钥策略中为 CodePipeline CodeBuild 角色提供访问权限： `aws cloudformation deploy --stack-name ecs-codepipeline-pre-reqs-KMS \ --template-file pre-reqs_KMS.yaml --parameter-overrides \ CodeBuildCondition=true TestAccount1=$TESTACCOUNT1 TestAccount2=$TESTACCOUNT2 \ TestAccount3=$TESTACCOUNT3 CodeCommitAccount=$CODECOMMITACCOUNT ToolsAccount=$TOOLSACCOUNT --region $TOOLSACCOUNTREGION` 2. 更新 S3 存储桶策略以允许访问 CodePipeline 和 CodeDeploy 角色： `aws cloudformation deploy --stack-name pre-reqs-artifacts-bucket --parameter-overrides BucketStartName=$BUCKETSTARTNAME \ PutS3BucketPolicy=true TestAccount1=$TESTACCOUNT1`	AWS DevOps

Task	描述	所需技能
	TestAccount2=$TEST ACCOUNT2 \ TestAccount3=$TE STACCOUNT3 CodeCommi tAccount=$CODECOMM ITACCOUNT ToolsAcco unt=$TOOLSACCOUNT \ --template-file pre- reqs_bucket.yaml --region $TESTACCO UNT1REGION --capabil ities CAPABILIT Y_NAMED_IAM aws cloudformation deploy --stack-name pre-reqs-artifacts -bucket --parameter- overrides BucketSta rtName=$BUCKETSTAR TNAME \ PutS3BucketPolic y=true TestAccou nt1=$TESTACCOUNT1 TestAccount2=$TEST ACCOUNT2 \ TestAccount3=$TE STACCOUNT3 CodeCommi tAccount=$CODECOMM ITACCOUNT ToolsAcco unt=$TOOLSACCOUNT \ --template-file pre- reqs_bucket.yaml --region $TESTACCO UNT2REGION --capabil ities CAPABILIT Y_NAMED_IAM aws cloudformation deploy --stack-name pre-reqs-artifacts -bucket --parameter-	

Task	描述	所需技能
	overrides BucketStartName=$BUCKETSTARTNAME \ PutS3BucketPolicy=true TestAccount1=$TESTACCOUNT1 TestAccount2=$TESTACCOUNT2 \ TestAccount3=$TESTACCOUNT3 CodeCommitAccount=$CODECOMMITACCOUNT ToolsAccount=$TOOLSACCOUNT \ --template-file pre-reqs_bucket.yaml --region $TESTACCOUNT3REGION --capabilities CAPABILITY_NAMED_IAM aws cloudformation deploy --stack-name pre-reqs-artifacts-bucket --parameter-overrides BucketStartName=$BUCKETSTARTNAME \ PutS3BucketPolicy=true TestAccount1=$TESTACCOUNT1 TestAccount2=$TESTACCOUNT2 \ TestAccount3=$TESTACCOUNT3 CodeCommitAccount=$CODECOMMITACCOUNT ToolsAccount=$TOOLSACCOUNT \ --template-file pre-reqs_bucket.yaml --region $TOOLSACCOUNTREGION --capabil	

Task	描述	所需技能
	`ities CAPABILIT Y_NAMED_IAM`	

调用并测试管道

Task	描述	所需技能

将更改推送到 CodeCommit 存储库。	1. `codecommitrepoStac k` 按照 AWS CodeCommit 文档中所述，使用`git clone`命令克隆在中创建的 CodeCommit 存储库。
	2. 使用所需详细信息更新输入构件：
	• JSON 文件：在此文件的三个位置更新文件中的 `AccountID` 。重命名这三个文件以包含该帐户 IDs。
	• YAML 文件：更新任务定义 ARN 和版本。重命名这三个文件以包含该帐户 IDs。
	3. 修改 `index.html` 文件以对主页进行一些细微的更改。
	4. 将以下文件复制到存储库并提交：
	`index.html`
	`Dockerfile`
	`buildspec.yaml`	

Task	描述	所需技能
	appspec_<accountid>.yaml (3 files - one per account) taskdef<accountid>.json (3 files - one per account) 5. 启动或重启管道并验证结果。 6. 使用 FQDN 或 DNS 从应用程序负载均衡器访问服务，并验证是否已部署更新。	

清理

Task	描述	所需技能

清理所有已部署的资源。	1. 将 Amazon ECS 缩减到零个实例：
	`aws ecs update-service --cluster QA-Cluster --service Poc-Service --desired-count 0`
	2. 删除每个账户和区域中的 CloudFormation 堆栈：
	`##In Tools Account## aws cloudformation delete-stack --stack-name ecscodepipelinestack --region $TOOLSACCOUNTREGION aws cloudformation delete-stack --stack-name ecscodebu`	

Task	描述	所需技能
	ildstack --region $TESTACCOUNT1REGION aws cloudformation delete-stack -- stack-name ecscodebu ildstack --region $TESTACCOUNT2REGION aws cloudformation delete-stack -- stack-name ecscodebu ildstack --region $TESTACCOUNT3REGION aws cloudformation delete-stack -- stack-name ecs-codep ipeline-pre-reqs-K MS --region $TOOLSACC OUNTREGION aws cloudformation delete-stack -- stack-name codecommi trepoStack --region $TOOLSACCOUNTREGION aws cloudformation delete-stack -- stack-name pre-reqs- artifacts-bucket --region $TESTACCO UNT1REGION aws cloudformation delete-stack -- stack-name pre-reqs- artifacts-bucket --region $TESTACCO UNT2REGION aws cloudformation delete-stack -- stack-name pre-reqs- artifacts-bucket --region $TESTACCO UNT3REGION	

Task	描述	所需技能
	aws cloudformation delete-stack --stack-name pre-reqs-artifacts-bucket --region $TOOLSACCOUNTREGION aws cloudformation delete-stack --stack-name ecs-codebuild-iam --region $TOOLSACCOUNTREGION ##NOTE: Artifact buckets will not get deleted if there are artifacts so it has to be emptied manually before deleting.##	
	##In Workload / Test Accounts## ##Account:1## aws cloudformation delete-stack --stack-name ecscodedeploystack --region $TESTACCOUNT1REGION aws cloudformation delete-stack --stack-name mainInfrastack --region $TESTACCOUNT1REGION ##Account:2## aws cloudformation delete-stack --stack-name ecscodedeploystack --region $TESTACCOUNT2REGION aws cloudformation delete-stack --	

Task	描述	所需技能
	`stack-name mainInfra stack --region $TESTACCOUNT2REGION ##Account:3## aws cloudformation delete-stack --stack-name ecscodedeploystack --region $TESTACCOUNT3REGION aws cloudformation delete-stack --stack-name mainInfra stack --region $TESTACCOUNT3REGION ##NOTE: Amazon ECR (web) will not get deleted if the registry still includes images. It can be manually cleaned up if not required.`	

故障排除

事务	解决方案
您提交到存储库的更改不会得到部署。	• 检查 CodeBuild 日志，查看 Docker 构建操作中是否存在错误。有关更多信息，请参阅 CodeBuild 文档。 • 检查 CodeDeploy 部署中是否存在任何 Amazon ECS 部署问题。

相关资源

- 推送 Docker 映像（Amazon ECR 文档）

- 连接到 AWS CodeCommit 存储库（AWS CodeCommit 文档）
- AWS 疑难解答 CodeBuild（AWS CodeBuild 文档）

使用 AWS 和 AWS CloudFormation Config 监控亚马逊 ECR 存储库的通配符权限

由 Vikrant Telkar (AWS)、Sajid Momin (AWS) 和 Wassim Benhallam (AWS) 创建

摘要

在 Amazon Web Services (AWS) Cloud 上，Amazon Elastic Container Registry (Amazon ECR) 是一项托管容器映像注册表服务，支持使用 AWS Identity and Access Management（AWS IAM）且具有基于资源权限的私有存储库。

IAM 在资源和操作属性中均支持“*”通配符，这使得自动选择多个匹配项变得更加容易。在您的测试环境中，您可以通过在存储库策略语句的主体元素中使用 `ecr:*` 通配符权限来允许所有经过身份验证的 AWS 用户访问 Amazon ECR 存储库。在无法访问生产数据的开发账户中进行开发和测试时，`ecr:*` 通配符权限非常有用。

但是，您必须确保在生产环境中不使用 `ecr:*` 通配符权限，因为它可能会导致严重的安全漏洞。此模式的方法可帮助您识别在存储库策略语句中包含 `ecr:*` 通配符权限的 Amazon ECR 存储库。 该模式提供了在 AWS Config 中创建自定义规则的步骤和 AWS CloudFormation 模板。然后，AWS Lambda 函数会监控您的 Amazon ECR 存储库策略语句中是否有 `ecr:*` 个通配符权限。如果发现不合规的存储库策略声明，Lambda 会通知 AWS Config 向亚马逊发送事件，然后启动 EventBridge EventBridge 亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 主题。SNS 主题通过电子邮件通知您有关不合规的存储库策略语句。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 已安装和配置 AWS 命令行界面（AWS CLI）。有关更新 Amazon CLI 的信息，请参阅 AWS CLI 文档中的安装、更新和卸载 Amazon CLI。
- 在测试环境中安装和配置的带有附加策略语句的现有 Amazon ECR 存储库。有关更多信息，请参阅 Amazon ECR 文档中的创建私有存储库和设置存储库策略语句。
- AWS Config，已在您首选的 AWS 区域中配置。有关此内容的更多信息，请参阅 AWS Config 文档中的 AWS Config 入门。
- `aws-config-cloudformation.template` 文件（附加）已下载到本地计算机。

限制

- 此模式的解决方案是区域性的，您的资源必须在同一区域中创建。

架构

下图显示了 AWS Config 如何评估 Amazon ECR 存储库策略语句。

图表显示了以下工作流：

1. AWS Config 启动自定义规则。
2. 自定义规则调用 Lambda 函数来评估 Amazon ECR 存储库策略语句的合规性。然后，Lambda 函数会识别不合规的存储库策略语句。
3. Lambda 函数将不合规状态发送到 AWS Config。
4. AWS Config 向发送事件 EventBridge。
5. EventBridge 向 SNS 主题发布不合规通知。
6. Amazon SNS 会向您或授权用户发送电子邮件提醒。

自动化和扩缩

此模式的解决方案可以监控任意数量的 Amazon ECR 存储库策略语句，但您要评估的所有资源必须在同一区域中创建。

工具

- AWS CloudFormation — AWS CloudFormation 可帮助您建模和设置 AWS 资源，快速一致地配置这些资源，并在资源的整个生命周期中对其进行管理。您可以使用模板来描述资源及其依赖关系，然后将它们作为堆栈一起启动和配置，而不必单独管理资源。您可以跨多个 Amazon Web Services account 和 AWS 区域管理和预置堆栈。
- AWS Config - AWS Config 提供 Amazon Web Services account 中 AWS 资源配置的详细视图。这些信息包括资源之间的关联方式以及资源以前的配置方式，让您了解资源的配置和关系如何随着的时间的推移而更改。
- Amazon ECR - Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的 AWS 托管容器映像注册表服务。Amazon ECR 支持私有存储库，其具有使用 IAM 的基于资源的权限。

- Ama zon EventBridge — Amazon EventBridge 是一项无服务器事件总线服务，可用于将应用程序与来自各种来源的数据连接起来。 EventBridge 将来自您的应用程序、软件即服务 (SaaS) 应用程序和 AWS 服务的实时数据流传输到目标，例如 AWS Lambda 函数、使用 API 目标的 HTTP 调用终端节点或其他账户中的事件总线。
- AWS Lambda – AWS Lambda 是一项计算服务，支持无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。您只需为消耗的计算时间付费 - 代码未运行时不产生费用。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 可协调和管理发布者和客户端之间消息的传送或发送，包括 Web 服务器和电子邮件地址。订阅用户接收所有发布至他们所订阅主题的消息，并且一个主题的所有订阅用户收到的消息都相同。

代码

此模式的代码可在 `aws-config-cloudformation.template` 文件（附件）中获取。

操作说明

创建 AWS CloudFormation 堆栈

Task	描述	所需技能
创建 AWS CloudFormation 堆栈。	在 AWS CLI 中运行以下命令来创建 AWS CloudFormation 堆栈： `$ aws cloudformation create-stack --stack-name=AWSConfigECR \ --template-body file://aws-config-cloudformation.template \ --parameters ParameterKey=<email>,ParameterValue=<myemail@example.com> \ --capabilities CAPABILITY_NAMED_IAM`	AWS DevOps

测试 AWS Config 自定义规则

Task	描述	所需技能
测试 AWS Config 自定义规则。	1. 登录 AWS 管理控制台，打开 AWS Config 控制台，然后选择资源。 2. 在资源清单页上，可以按资源类别、资源类型和合规性状态进行筛选。 3. 包含 `ecr:*` 的 Amazon ECR 存储库是 `NON-COMPLIANT?` ，不包含 `ecr:*` 的 Amazon ECR 存储库是 `COMPLIANT` 。 4. 如果 Amazon ECR 存储库包含不合规的策略语句，则订阅 SNS 主题的电子邮件地址将收到通知。	AWS DevOps

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS CDK 和 GitHub 操作工作流程优化多账户无服务器部署

由 Sarat Chandra Pothula (AWS) 和 VAMSI KRISHNA SUNKAVALLI (AWS) 创作

摘要

跨多个 AWS 账户 环境部署无服务器基础架构的组织经常会遇到诸如代码重复、手动流程和做法不一致之类的挑战。此模式的解决方案展示了如何使用 Go and Actions AWS Cloud Development Kit (AWS CDK) 中的可重复使用工作流程来简化多账户无服务器基础架构管理。 GitHub 该解决方案演示了如何将云资源定义为代码、实现标准化integration/continuous deployment (CI/CD（持续）流程以及如何创建模块化、可重复使用的组件。

通过使用这些工具，组织可以高效地管理跨账户资源，实施一致的部署管道，并简化复杂的无服务器架构。该方法还通过强制使用标准化实践来增强安全性和合规性 AWS 账户，从而最终提高工作效率并减少无服务器应用程序开发和部署中的错误。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- AWS Identity and Access Management (IAM) 角色和权限适用于部署过程。这包括访问亚马逊弹性容器注册表 (Amazon ECR) 存储库、 AWS Lambda 创建函数以及跨目标的任何其他所需资源的权限。 AWS 账户
- AWS Command Line Interface (AWS CLI) 版本 2.9.11 或更高版本，已安装并配置。
- AWS Cloud Development Kit (AWS CDK) 版本 2.114.1 或更高版本，已安装并已引导。
- 已安装 1.22 或更高版本。
- 已安装 Docker 24.0.6 或更高版本。

限制

- 语言兼容性 — Go 是无服务器应用程序的常用语言。但是，除了 Go 之外，还 AWS CDK 支持其他编程语言，包括 C#、Java、Python 和 TypeScript。如果您的组织拥有其他语言的现有代码库或专业知识，则可能需要调整或学习 Go 才能充分使用模式中描述的解决方案。
- 学习曲线 — 采用 AWS CDK、Go（如果对组织来说是新手）和 GitHub 可重复使用的工作流程可能会涉及开发人员和 DevOps 团队的学习曲线。为确保顺利采用和有效使用这些技术，可能需要培训和文档。

架构

下图显示了此模式的工作流和体系结构组件。

此解决方案执行以下步骤：

1. 开发人员克隆存储库，创建新分支，并在本地环境中更改应用程序代码。
2. 开发者提交这些更改并将新分支推送到 GitHub 存储库。
3. 开发者在 GitHub 存储库中创建拉取请求，提议将其功能或新功能分支合并到主分支中。
4. 此拉取请求会触发持续集成 (CI) GitHub 操作工作流程。这种模式中的 CI 和持续部署 (CD) 工作流程使用可重复使用的工作流程，这些工作流程是预定义的模块化模板，可以在不同的项目或存储库中共享和执行。可重复使用的工作流程促进了 CI/CD 流程的标准化和效率。
5. CI 工作流程设置必要的环境，为镜像生成 Docker 标签，并使用应用程序代码构建 Docker 镜像。
6. CI 工作流程使用中央 AWS 账户 GitHub OIDC 角色进行身份验证。 AWS 对于 CI 工作流程， AWS 账户 GitHub OIDC 的中心角色使用 AWS Security Token Service (AWS STS) 来获取临时证书。这些证书允许该角色构建 Docker 镜像并将其推送到中央的 Amazon ECR 存储库。 AWS 账户
7. CI 工作流程将构建的 Docker 镜像推送到 Amazon ECR。
8. CI 工作流程将图像标签存储到 Systems Manager 参数存储区。
9. CI 工作流程成功完成后，将输出 Docker 镜像标签。
10. 触发 CD 工作流程时，开发人员手动输入他们要部署的 Docker 镜像的镜像标签。此图片标签对应于 CI 工作流程中生成并推送到 Amazon ECR 的标签。
11. 开发人员手动触发 CD 工作流程，该工作流程使用 CD 可重复使用工作流程。
12. CD 工作流程 AWS 使用中央 AWS 账户 GitHub OIDC 角色进行身份验证。对于 CD 工作流程， AWS STS 首先用于担任 AWS 账户 GitHub OIDC 的中心角色。然后，此角色将担任目标账户部署的 CDK 引导角色。
13. CD 工作流程使用 AWS CDK 来合成 AWS CloudFormation 模板。
14. CD 工作流程使用 CDK 部署将应用程序部署 AWS 账户 到目标，使用手动为 Lambda 函数指定的图像标签。

工具

AWS 服务

- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，可帮助您在代码中定义和配置 AWS Cloud 基础架构。
- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。 CloudFormation 是 AWS CDK 部署过程不可或缺的一部分。CDK 合成 CloudFormation 模板，然后 CloudFormation 用于创建或更新环境中的 AWS 资源。
- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Systems Manager Parameter Store 为配置数据管理和密钥管理提供安全的分层存储。

其他工具

- Docker 是一组平台即服务（PaaS）产品，它们使用操作系统级别的虚拟化技术在容器中交付软件。
- GitHub A@@ c tions 是一个持续集成和持续交付 (CI/CD) 平台，与 GitHub 存储库紧密集成。您可以使用 GitHub Actions 来自动执行构建、测试和部署管道。
- Go 是谷歌支持的开源编程语言。

代码存储库

此模式的代码可在cicd-github-actions存储 GitHub aws-cdk-golang-serverless库中找到。

最佳实践

- 模块化设计 — 将您的 AWS CDK 代码组织成模块化和可重复使用的构造或堆栈，从而促进跨多个账户和项目的代码重用和可维护性。
- 问题分离 — 将基础架构代码与应用程序代码分开，允许独立部署和管理每个组件。
- 版本控制和不可变性 — 将您的基础设施视为代码 (IaC)，并使用 Git 进行版本控制。通过创建新资源而不是修改现有资源来接受不可变的基础架构原则。
- 测试和验证-实施全面的测试策略，包括单元测试、集成 end-to-end测试和测试，以帮助支持 AWS CDK 代码和部署的正确性和可靠性。

- 安全性与合规性-遵循 AWS 安全最佳实践，例如最低权限访问、安全通信和数据加密。实施合规检查和审计机制，确保遵守组织政策和监管要求。为容器映像实施安全最佳实践，例如扫描漏洞、强制执行映像签名以及遵守组织的合规性要求。
- 监控和记录-设置监控和日志记录机制，以跟踪无服务器应用程序和基础设施的运行状况和性能。AWS 服务 像 Amazon CloudWatch、 AWS CloudTrail、一样使用， AWS X-Ray 用于监控和审计。
- 自动化和 CI/CD — 使用 GitHub 可重复使用的工作流程和其他 CI/CD 工具来自动执行构建、测试和部署流程，这有助于支持跨多个账户进行一致且可重复的部署。
- 环境管理-维护单独的环境（例如，开发、暂存和生产）。实施促进环境间变更的策略，确保在生产部署之前进行适当的测试和验证。
- 文档和协作-记录您的基础架构代码、部署流程和最佳实践，以促进团队内部的知识共享和协作。
- 成本优化-实施成本监控和优化策略，例如合理调整资源规模、使用自动缩放以及利用 AWS 成本优化服务（如 AWS Budgets 和）。 AWS Cost Explorer
- 灾难恢复和备份 — 通过为无服务器应用程序和基础设施资源实施备份和还原机制，规划灾难恢复方案。
- 持续改进 — 定期审查并更新您的实践、工具和流程，以适应无服务器生态系统中的最新最佳实践、安全建议和技术进步。
- 改善安全状况 — 通过AWS PrivateLink为 Amazon ECR 和 P AWS Systems Manager arameter Store 配置接口 VPC 终端节点， AWS Lambda用于改善虚拟私有云 (VPC) 的安全状况。

操作说明

设置环境

Task	描述	所需技能
在中央 AWS 账户创建一个 Amazon ECR 存储库。	要跨多个容器镜像共享 AWS 账户，您必须为 Amazon ECR 配置跨账户访问权限。首先，在中央 AWS 账户创建一个 Amazon ECR 存储库。 要创建 Amazon ECR 存储库，请运行以下命令：	AWS DevOps

Task	描述	所需技能
	`aws ecr create-repository --repository-name sample-repo` 在以后的任务中，向需要使用容器镜像的其他 AWS 账户 人授予拉取访问权限。	

Task	描述	所需技能

向 Amazon ECR 存储库添加跨账户权限。	要向中心的 Amazon ECR 存储库添加跨账户权限 AWS 账户，请运行以下代码：
	
	{
	  "Version": "2008-10-17",
	  "Statement": [
	  {
	  "Sid": "LambdaECRImageRetrievalPolicy",
	  "Effect": "Allow",
	  "Principal": {
	  "Service": "lambda.amazonaws.com"
	  },
	  "Action": [
	  "ecr:BatchGetImage",
	  "ecr:GetDownloadUrlForLayer",
	  ],
	  "Condition": {
	  "StringLike": {
	  "aws:sourceArn": "arn:aws:lambda:<Target_Region>:<Target_Account_ID>:function:*"
	  }
	  }
	  },
	  {
	  "Sid": "new statement",
	  "Effect": "Allow",
	  "Principal": {
		AWS DevOps

Task	描述	所需技能

	<pre>"AWS": "arn:aws:iam::<Target_Account_ID>:root"
	},
	"Action": [
	"ecr:BatchGetImage",
	"ecr:GetDownloadUrlForLayer",
	],
	}
	]
	}</pre>	

在中心为 GitHub OIDC 角色配置角色。 AWS 账户	1. AWS 将信任 GitHub的 OIDC 配置为联合身份，包括将 GitHub OIDC 提供商添加到 IAM AWS 以及在 IAM 中配置角色和信任策略。为此，请按照文档中在亚马逊 Web Services 中配置 OpenID Connect 中的 GitHub 说明进行操作。 2. 创建角色后，向该角色添加必要的权限。例如，为 Amazon ECR 和 P AWS Systems Manager arameter Store 添加权限。有关更多信息，请参阅 IAM 文档中的为 GitHub OIDC 身份提供商配置角色。	AWS DevOps

Task	描述	所需技能
引导目标 AWS 账户中的 AWS 环境。	在特定 AWS 账户 环境中设置 CDK 环境 AWS 区域 ，该环境允许从中央账户进行跨账户部署，并将最低权限原则应用于执行角色。 CloudFormation 要引导 AWS 环境，请运行以下命令： `cdk bootstrap aws://<Target_Account_ID>/<Target_Region> --trust <Central_Account_ID> --cloudformation-execution-policies arn:aws:iam::aws:policy/<Least_Privilege_Policy>`	AWS DevOps

Task	描述	所需技能

授予中央 AWS 账户 OIDC 角色对目标 AWS 账户 引导角色的访问权限。	CDK bootstrap 创建了以下 IAM 角色，这些角色旨在由中央在 CDK AWS 账户 部署过程的各个阶段担任：
	• 文件发布角色
	• 映像发布角色
	• 查找角色
	• 部署角色
	
	每个角色都有根据其目的量身定制的特定权限，遵循最低权限原则。每个角色名称`Target_Region` 中的`Target_Account_ID` 和有助于表明这些角色在不同的 AWS 账户 地区中是唯一的。这种方法支持在多账户、多区域设置中进行清晰的识别和管理。
	
	`Target Account CDK Bootstrap Roles`
	`arn:aws:iam::<Target_Account_ID>:role/cdk-deploy-role-<Target_Account_ID>-<Target_Region>`
	`arn:aws:iam::<Target_Account_ID>:role/cdk-file-publishing-role-<Target_Account_ID>-<Target_Region>`
	`arn:aws:iam::<Target_Account_ID>:role/cdk-image-publishing-role-`
		AWS DevOps

Task	描述	所需技能
	<Target_Account_ID>- <Target_Region> arn:aws:iam::<Target_Account_ID>:role/cdk-lookup-role-<Target_Account_ID>-<Target_Region> • 更新中央账户中 OIDC 角色的权限策略，使其能够在目标账户中扮演角色。此配置允许跨不同堆栈部署 CDK 堆栈。 AWS 账户通过允许中央账户的 OIDC 角色采用目标账户的必要权限，您可以为跨账户 CDK 部署创建安全桥梁。这种方法可以保持适当的访问控制，同时促进无缝的多账户基础设施管理。 要在中心更新 OIDC 角色的权限策略 AWS 账户，请使用以下代码： { "Version": "2012-10-17", "Statement": [{ "Effect": "Allow", "Action": "sts:AssumeRole", "Resource": ["arn:aws:iam::<Target_Account_ID>:ro	

Task	描述	所需技能

	<pre>le/cdk-deploy-role-
	<Target_Account_ID>-
	<Target_Region>”,
	
	“arn:aws:iam::<Tar
	get_Account_ID>:role/
	cdk-file-publishing-
	role-<Target_Account_I
	D>-<Target_Region>”,
	
	“arn:aws:iam::<Tar
	get_Account_ID>:role/
	cdk-image-publishing-
	role-<Target_Account_
	ID>-<Target_Region>”,
	
	“arn:aws:iam::<Tar
	get_Account_ID>:ro
	le/cdk-lookup-role-
	<Target_Account_ID>-
	<Target_Region>”
	]
	}
	]
	}</pre>	

构建 Docker 镜像

Task	描述	所需技能

克隆项目存储库。	要克隆此模式的GitHub 存储库，请运行以下命令：
	
	<pre>git clone https://g
	ithub.com/aws-samp
	les/aws-cdk-golang
	-serverless-cicd-g
	ithub-actions.git</pre>
		AWS DevOps

Task	描述	所需技能
前往 Dockerfile 路径。	要导航到 Dockerfile 路径，请运行以下命令： `cd lambda`	AWS DevOps
使用亚马逊 ECR 对 Docker 进行身份验证。	Amazon ECR 需要安全访问您的私有容器存储库。通过这种方式登录，您就允许本地计算机或 CI/CD 环境上的 Docker 安全地与 Amazon ECR 进行交互。 要使用 Amazon ECR 对 Docker 进行身份验证，请运行以下命令： `aws ecr get-login -password --region <AWS_REGION> | docker login --username AWS --password-stdin <AWS_ACCOUNT_ID>.dkr.ecr.<AWS_REGION>.amazonaws.com` 修改占位`AWS_Account_ID` 符`AWS_REGION` 和您的信息。	AWS DevOps
构建 Docker 映像。	要构建 Docker 镜像，请运行以下命令： `docker build --platform linux/arm64 -t sample-app .`	AWS DevOps

Task	描述	所需技能

标记并推送 Docker 镜像。	要标记 Docker 镜像并将其推送到 Amazon ECR 存储库，请运行以下命令：
	`docker tag sample-app:latest <AWS_ACCOUNT_ID>.dkr.ecr.<AWS_REGION>.amazonaws.com/<ECR_REPOSITORY>:<DOCKER_TAG>`
	`docker push <AWS_ACCOUNT_ID>.dkr.ecr.<AWS_REGION>.amazonaws.com/<ECR_REPOSITORY>:<DOCKER_TAG>`
	使用您的信息修改占位符`AWS_Account_ID` `AWS_REGION` `ECR_REPOSITORY` 、、和`DOCKER_TAG` 。
		AWS DevOps

部署 AWS CDK 应用程序

Task	描述	所需技能
使用特定于环境的变量合成 CDK 堆栈。	要按照 CDK 代码中的定义为基础架构生成 CloudFormation 模板，请运行以下命令： `ENV=<environment> IMAGETAG=<image_tag> ECR_ARN=<ecr_repo_arn> cdk synth`	AWS DevOps

Task	描述	所需技能
	使用您的信息修改以下占位符： • `environment` — 替换为特定的环境名称`dev`，例如`staging`、或`prod`。 • `image_tag` — 替换为 Docker 镜像的特定标签，例如`v1.0.0`或`latest`。 • `ecr_repo_arn` — 替换为亚马逊 ECR 存储库的亚马逊资源名称 (ARN)。	
部署 CDK 堆栈。	要将 CDK 堆栈部署到您的 AWS 账户，请运行以下命令。该`--require-approval never`标志表示 CDK 将自动批准并执行所有更改。这包括 CDK 通常会标记为需要手动审查的更改（例如 IAM 政策变更或资源移除）。在生产环境中使用该`--require-approval never`标志之前，请确保您的 CDK 代码和 CI/CD 管道经过充分测试且安全。 `ENV=<environment> IMAGETAG=<image_tag> ECR_ARN=<ecr_repo_arn> cdk deploy --require-approval never`	AWS DevOps

使用 GitHub 操作工作流程自动化 CI/CD

Task	描述	所需技能

创建功能分支，然后添加您的更改。	使用您之前创建的克隆存储库，创建功能分支，然后将更改添加到应用程序代码中。使用以下命令：
	`git checkout -b <feature_branch>`
	`git add .`
	`git commit -m “add your changes”`
	`git push origin <feature_branch>`
	以下是更改的示例：
	• 对 Lambda 函数逻辑的更改
	• 向 Lambda 代码添加新特性或功能
	• 修复错误或优化 Lambda 函数中的现有代码
	GitHub 操作将使用可重复使用的工作流程并触发 CI/CD 管道。
		AWS DevOps

合并您的更改。	创建拉取请求，并将您的更改合并到 main 中。	AWS DevOps

故障排除

事务	解决方案
`AccessDenied` 例如 AWS 账户，跨部署资源时会出错`AccessDenied: User not authorized to perform: “sts:AssumeRole"` 。	要帮助解决此问题，请执行以下操作来验证跨账户权限： • 确保为跨账户部署准备了必要的 IAM 角色和策略。 • 检查`assume`角色权限配置是否正确。
由于版本不匹配而导致的兼容性问题，例如，过时的 CDK 版本`undefined: awscdkStack` 出错。	要帮助解决此问题，请执行以下操作以验证您使用的是 AWS CDK 和 Go 的必需版本： • 确保你使用的是 AWS CDK 和 Go 的兼容版本。 • 检查最新版本中是否存在任何已知问题或重大更改。
例如，由`Error: No such file or directory` 于 YAML 配置不正确或受保护分支导致的 CI/CD 管道故障`Permission denied`。	为了帮助解决 GitHub 操作配置问题，请验证是否正确引用和配置了可重复使用的工作流程。

相关资源

AWS 资源

- AWS 安全性、身份和合规性最佳实践
- AWS CDK 工作坊
- AWS Cloud 开发套件库
- 使用容器映像创建 Lambda 函数
- 适用于亚马逊弹性容器注册表的身份和访问管理
- 在 Go 中使用 AWS CDK

其他资源

- 在亚马逊 Web Services 中配置 OpenID Connect（文档）GitHub
- Golang 文档
- GitHub 操作快速入门（GitHub 文档）
- 重复使用工作流程（GitHub 文档）

从 AWS CodeCommit 事件中执行自定义操作

由 Abdullahi Olaoye (AWS) 编写

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

当您使用 AWS CodeCommit 存储库存储代码时，您可能需要监控存储库并在发生特定事件时启动操作工作流程。例如，您可能希望在用户评论提交中的一行代码时发送电子邮件通知，或者在提交后启动 AWS Lambda 函数，以对存储库内容执行安全扫描。此模式概述了为自定义操作配置 CodeCommit 存储库的步骤。该模式使用 AWS CodeCommit 通知规则捕获感兴趣的事件，然后将这些事件发送到已配置的目标。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 熟悉 Git 命令。
- AWS CodeCommit，设置完毕。有关说明，请参阅为 AWS 进行设置 CodeCommit。
- （推荐）安装和配置 AWS 命令行界面（AWS CLI）。有关说明，请参阅 AWS CLI 入门。

架构

工具

Amazon Web Services

- AWS CodeCommit 是一项完全托管的源代码控制服务，可托管基于 Git 的安全存储库。它使团队可以轻松地在安全且高度可扩展的生态系统中就代码进行协作。 CodeCommit 无需操作自己的源代码控制系统或担心扩展其基础架构
- Amazon Simple Notification Service (Amazon SNS)是一项 Web 服务，通过该服务，应用程序、终端用户和设备可以发送和接收来自云的通知。Amazon SNS 为高吞吐量、基于推送的消息传递提供主题（通信渠道）。 many-to-many使用 Amazon SNS 主题，发布者可以向大量订阅用户分发消息以进行并行处理，包括 Amazon Simple Queue Service (Amazon SQS) 队列、AWS Lambda 函数以

及 HTTP/S 网络钩子。您还可以使用 Amazon SNS，通过移动推送、SMS 和电子邮件向最终用户发送通知。

操作说明

设置存储 CodeCommit 库

Task	描述	所需技能
创建 CodeCommit 存储库。	使用 CodeCommit 控制台或 AWS CLI 创建 CodeCommit 存储库。有关说明，请参阅创建 CodeCommit 存储库。	DevOps 工程师
将内容推送到 CodeCommit 存储库。	创建存储后，使用 Git 命令向其中添加内容。您可以从计算机迁移现有 Git 存储库的内容，或本地未受版本控制的内容。有关说明，请参阅向存储库添加文件或迁移到 AWS CodeCommit。	DevOps 工程师

设置 Amazon SNS

Task	描述	所需技能
创建 SNS 主题。	此 SNS 主题接收来自的事件。CodeCommit有关说明，请参阅创建 Amazon SNS 主题。	云架构师、 DevOps 工程师
为自定义操作创建资源。	若要执行自定义操作，必须创建相应的资源。例如，如果您的自定义操作是运行 Lambda 代码并将消息发送至 SQS 队列，则必须创建 Lambda 函数和 SQS 队列。电子邮件和短信通知等操作不需要资源。有关	云架构师、 DevOps 工程师

Task	描述	所需技能
	更多信息，请参阅您正在创建资源类型的 AWS 文档。	
将自定义操作资源订阅至 SNS 主题。	根据自定义操作，您可按相应的协议创建订阅。例如，您订阅电子邮件地址以接收电子邮件通知，订阅用于运行自定义代码的 Lambda 函数，或者订阅用于向 Amazon SQS 发送事件的 SQS 队列。对于电子邮件和短信等订阅协议，您需要分别通过发送电子邮件或电话号码的链接确认订阅。有关说明，请参阅 Amazon SNS 主题。	云架构师、 DevOps 工程师

配置通知规则

Task	描述	所需技能
为 CodeCommit 存储库创建通知规则。	创建通知规则时，您可以选择启动通知的 Git 事件，选择 SNS 主题作为目标类型，然后选择之前创建的 SNS 主题。您也可为存储库配置多个目标。有关说明，请参阅创建通知规则。	DevOps 工程师
测试自定义操作。	配置以下事件，以发起通知。例如，如果您选择拉取请求作为触发条件，请创建拉取请求。您应看到自定义操作总数。例如，如果您为一个电子邮件地址订阅 SNS 主题，您应接收电子邮件通知。	DevOps 工程师

相关资源

- AWS CodeCommit 文档
- Amazon SNS 文档
- Git 文档

使用 GitHub 操作基于 AWS CloudFormation 模板配置 AWS Service Catalog 产品

由 Ashish Bhatt (AWS) 和 Ruchika Modi (AWS) 创作

摘要

这种模式为组织提供了一种简化的方法，使用AWS Service Catalog产品和产品组合来 AWS 服务 跨团队进行标准化和合规的配置。 AWS CloudFormation有助于整合 Service Catalog 产品和产品组合中的基本组件，用于配置基础网络基础架构 AWS Cloud。这种模式还通过使用 Actions 将基础设施即代码 (IaC) 集成到自动化开发工作流程中，从而促进 DevOps 实践GitHub 。

AWS Service Catalog 使组织能够在上创建和管理经批准的 IT 服务 AWS，从而提供标准化、集中控制、自助服务配置和成本管理等好处。通过操作自动部署 Service Catalog 产品组合和产品，公司可以执行以下 GitHub 操作：

- 实现一致且可重复的部署。
- 对 IaC 使用版本控制。
- 将云资源管理与现有开发工作流程集成。

这种组合简化了云端运营，加强了合规性，加快了经批准的服务的交付，同时减少了人为错误并提高了整体效率。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 访问GitHub 存储库
- 对 AWS CloudFormation 和的基本理解 AWS Service Catalog
- 用于托管模板的亚马逊简单存储服务 (Amazon S3) Service 存储桶 CloudFormation
- 一个名为的 AWS Identity and Access Management (IAM) 角色`github-actions`，用于 GitHub 和之间的连接 AWS

限制

- 此模式的可重用代码仅通过 Actions 进行了 GitHub 测试。

- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

产品版本

此模式的解决方案是使用以下 GitHub Marketplace 操作及其相应的版本创建的：

- `actions/checkout@v4`
- `aws-actions/configure-aws-credentials@v2`
- `aws-actions/aws-cloudformation-github-deploy@v1.2.0`

架构

下图显示了此解决方案的架构。

1. 管理员或平台工程师将标准化 CloudFormation 模板推送到 GitHub 存储库，在那里维护这些模板。该 GitHub 存储库还包含 AWS Service Catalog 使用 GitHub 操作自动配置的工作流程。
2. GitHub 操作会触发一个工作流，该工作流 AWS Cloud 使用 OpenID Connect (OIDC) 提供程序连接到以配置 Service Catalog。
3. Service Catalog 包含开发人员可以直接用来配置标准化 AWS 资源的产品组合和产品。这种模式捆绑了诸如虚拟私有云 (VPCs)、子网、NAT 和 Internet 网关以及路由表之类的 AWS 资源。
4. 开发者创建服务目录产品后，Service Catalog 会将其转换为预先配置的标准化 AWS 资源。因此，开发人员可以节省时间，因为他们无需配置单个资源和手动配置这些资源。

工具

AWS 服务

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。它是一种基础设施即代码 (IaC) 服务，可以很容易地用作产品类型之一。 AWS Service Catalog
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Service Catalog帮助您集中管理已获批准的 IT 服务目录 AWS。最终用户可在遵循组织设定约束的情况下快速部署他们所需已获得批准的 IT 服务。

- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

其他

- GitHub A@@ c tions 是一个持续集成和持续交付 (CI/CD) 平台，与 GitHub 存储库紧密集成。您可以使用 GitHub Actions 来自动执行构建、测试和部署管道。

代码存储库

此模式的代码可在 GitHub service-catalog-with-github- actions 存储库中找到。该存储库包含以下感兴趣的文件：

- `github/workflows`:
  - `e2e-test.yaml`— 此文件调用`workflow.yaml`，这是可重复使用的工作流程。只要分支上有提交和推送，就会触发此工作流程。
  - `workflow.yaml`— 此文件包含此解决方案的可重复使用的工作流程，并配置`workflow_call`为其触发器。作为可重复使用的工作流程，`workflow.yaml`可以从任何其他工作流程中调用。
- `templates`:
  - `servicecatalog-portfolio.yaml`— 此 CloudFormation 模板包括用于配置服务目录产品组合和服务目录产品的资源。该模板包含在配置 Service Catalog 产品组合和产品时使用的一组参数。其中一个参数接受上传模板`vpc.yaml`的 Amazon S3 文件网址。尽管此模式包括 AWS 用于配置资源的`vpc.yaml`文件，但您也可以使用参数 S3 文件 URL 进行配置。
  - `vpc.yaml`— 此 CloudFormation 模板包含要添加到 Service Catalog 产品中的 AWS 资源。 AWS 资源包括子网 VPCs、互联网网关、NAT 网关和路由表。该`vpc.yaml`模板是如何将任何模板与 Service Catalog 产品和产品组合 CloudFormation 模板配合使用的示例。

最佳实践

- 请参阅 AWS Service Catalog 文档 AWS Service Catalog中的安全最佳实践。
- 有关 GitHub 操作的信息，请参阅 GitHub 文档中的安全加固。

操作说明

设置本地工作站

Task	描述	所需技能
在本地工作站上设置 Git。	要在本地工作站上安装和配置 Git，请使用 Git 文档中的入门-安装 Git 说明。	应用程序开发人员

克隆 GitHub 项目存储库。	要克隆 GitHub 项目存储库，请执行以下操作：
	1. 打开此GitHub 模式的存储库
	2. 选择 “代码” 以查看克隆选项，然后复制 HTTPS 选项卡中提供的 URL。
	3. 在工作站上为项目创建文件夹。
	4. 打开终端，然后导航到此文件夹
	5. 要克隆 GitHub 存储库，请使用您在步骤 2 中复制的 URL 运行以下命令：
	`git clone https://github.com/aws-samples/service-catalog-with-github-actions.git`
	6. 克隆完成后，要更改为项目文件夹中的克隆存储库，请运行以下命令：
	`cd <folder-name>/service-catalog-with-github-actions`
		DevOps 工程师

Task	描述	所需技能
	7. 在您选择的集成开发环境 (IDE) 中打开项目。	

设置 OIDC 提供商

Task	描述	所需技能
配置 OIDC 提供商。	创建一个 OpenID Connect (OIDC) 提供程序，允许 GitHub 操作工作流程访问其中的资源 AWS，而无需将 AWS 凭据存储为长期存在的机密。有关说明，请参阅 GitHub 文档中的在亚马逊 Web Services 中配置 OpenID Connect。GitHub 配置 OIDC 提供商后，将更新先决条件中前面提到的 IAM 角色`github-actions` 的信任策略。	AWS 管理员、AWS DevOps、常规 AWS

触发 GitHub 操作管道以部署 Service Catalog 产品组合和产品

Task	描述	所需技能
更新 `e2e-test.yaml`	该`e2e-test.yaml` 文件在上触发了可重复使用的工作流程`workflow.yaml` 。更新并验证中以下输入参数的值`e2e-test.yaml` ： • `aws_account_id` — 指定正确的 AWS 账户。	DevOps 工程师

Task	描述	所需技能

	• aws_region — 指定正确的 AWS 区域。
	• s3BucketName — 指定用于存放 CloudFormation 模板的 Amazon S3 存储桶。
	• 工作流程文件需要两个 IAM 角色作为输入：
	  • LaunchConstraintRole -最终用户启动、更新或终止产品时 AWS Service Catalog 扮演的 IAM 角色。
	  • PrincipalArn -将与 Service Catalog 产品组合关联的委托人（IAM 用户、角色或群组）的亚马逊资源名称 (ARN)。如果PrincipalType 是IAM，则支持的值为完全定义的 IAM Amazon 资源名称 (ARN)。如果PrincipalType 是IAM_PATTERN ，则支持的值为不带的 IAM ARN，AccountID 格式如下：arn:partition:iam::<account_ID>:resource-type/resource-id	

验证部署

Task	描述	所需技能
验证 Service Catalog 资源。	要验证 Service Catalog 资源，请执行以下操作： 1. 登录到 AWS Management Console AWS 账户，并验证是否正 AWS 区域 确。 2. 导航到 “管理” AWS Service Catalog、“投资组合” 下是否存在投资组合。 3. 选择产品组合，并验证 “产品”、“约束” 和 “访问权限” 选项卡上的信息。	AWS DevOps

清理资源

Task	描述	所需技能
删除 CloudFormation 堆栈。	要删除 CloudFormation 堆栈，请执行以下操作： 1. 在 https://console.aws.amazon.com/cloudformation 上打开 AWS CloudFormation 控制台。 2. 在屏幕顶部的导航栏上，选择堆栈 AWS 区域 所在的位置。 3. 在堆栈页面上，选择您要删除的堆栈。该堆栈当前必须处于运行状态。 4. 在堆栈详细信息窗格中，选择删除。	DevOps 工程师，AWS 管理员

Task	描述	所需技能
	5. 在系统提示时，选择删除堆栈。 有关更多信息，请参阅 CloudFormation 文档中的从 CloudFormation 控制台删除堆栈	

故障排除

事务	解决方案

`e2e-test`
 	
`Can't find 'action.yml', 'action.yaml' or 'Dockerfile' under '*/home/runner/work/service-catalog-with-github-actions/service-catalog-with-github-actions`
 	
`Did you forget to run actions/checkout before running your local action?`
	要确保启用了正确的存储库设置，请执行以下操作：
	
	1. 导航到 Github 存储库的 “设置” 选项卡。
	2. 从左侧菜单中选择 “操作”、“常规”
	3. 转到 “访问权限” 部分，然后选择 “可从 “XXX” 组织中的存储库访问选项。

相关资源

AWS 文档

- Service Catalog 概述

其他资源

- 关于触发工作流程的事件（GitHub 文档）

- 重复使用工作流程（GitHub 文档）

其他信息

要查看与 Epics 相关的屏幕截图，请转到此模式存储库中的 Images 文件夹。 GitHub 以下屏幕截图可用：

- AWS Service Catalog 投资组合，管理部分
- AWS Service Catalog 产品，“管理” 部分
- AWS Service Catalog 产品，“用户/配置” 部分

通过部署角色自动售货机解决方案来配置权限最低的 IAM 角色

由本杰明·莫里斯 (AWS)、Aman Kaur Gandhi (AWS)、Chad Moon (AWS) 和 Nima Fotouhi (AWS) 创作

摘要

管道的超范围 AWS Identity and Access Management (IAM) 角色权限可能会给组织带来不必要的风险。开发人员有时会在开发过程中授予广泛的权限，但在对代码进行故障排除后却忽略了缩小权限范围。这会导致一个问题，即在没有业务需求的情况下存在强大的角色，并且可能从未经过安全工程师的审查。

这种模式为这个问题提供了解决方案：角色自动售货机 (RVM)。RVM 使用安全的集中式部署模型，演示了如何以最少的开发人员努力为各个 GitHub 仓库的管道配置权限最低的 IAM 角色。由于 RVM 是一个中央解决方案，因此您可以根据需要配置安全团队以批准更改。这种方法允许安全部门拒绝权限过大的管道角色请求。

RVM 将 Terraform 代码作为输入，并生成可用于管道的 IAM 角色作为输出。所需的输入是 AWS 账户 ID、 GitHub 存储库名称和权限策略。RVM 使用这些输入来创建角色的信任策略和权限策略。由此产生的信任策略允许指定的 GitHub 存储库担任该角色并将其用于管道操作。

RVM 使用 IAM 角色（在引导期间配置）。该角色有权在组织 role-provisioning-role中的每个账户中扮演一个。该角色可通过 Account Factor AWS Control Tower y for Terraform (AFT) 或。 AWS CloudFormation StackSets role-provisioning-roles这些角色实际上是为开发人员创建管道角色。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- 一种用于通过 GitHub 操作部署基础设施即代码 (IaC) 的 GitHub 组织。 （GitHub Enterprise/Premium/Ultimate不是必填项。 ）
- 多账户 AWS 环境。这个环境不一定是其中的一部分 AWS Organizations。
- 一种用于在所有角色中部署 IAM 角色的机制 AWS 账户 （例如，AFT 或 CloudFormation StackSets ）。
- 已安装并配置 Terraform 版本 1.3 或更高版本。
- 安装并配置了 Terraform P AWS rovider 版本 4 或更高版本。

限制

- 此模式的代码特定于 Actions 和 Terraform。 GitHub 但是，该模式的一般概念可以在其他持续集成和交付 (CI/CD) 框架中重复使用。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅按地区划分的AWS 服务。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

架构

下图说明了此模式的工作流。

角色自动售货机的典型使用工作流程包括以下步骤：

1. 开发人员将包含新请求的 IAM 角色的 Terraform 代码的代码推送到 R GitHub VM 存储库。此操作会触发 RVM GitHub 操作管道。
2. 该管道使用 OpenID Connect (OIDC) 信任策略来担任 RVM 角色假设角色。
3. 当 RVM 管道运行时，它将在预配置开发人员新 IAM 角色的账户中担任 RVM 工作流程角色。（ RVM 工作流程角色是使用 AFT 或 CloudFormation StackSets配置的。 ）
4. RVM 创建具有适当权限和信任的开发人员的 IAM 角色，以便其他应用程序管道可以担任该角色。
5. 应用程序开发者可以配置其应用程序管道以承担这个由 RVM 配置的角色。

创建的角色包括开发者请求的权限和`ReadOnlyAccess`策略。只有针对开发者指定存储库`main`分支运行的管道才能担任该角色。这种方法有助于确保需要分支机构保护和审查才能使用该角色。

自动化和扩缩

最低权限权限需要注意正在配置的每个角色的细节。该模型降低了创建这些角色所需的复杂性，使开发人员无需太多的学习或努力即可创建所需的角色。

工具

AWS 服务

- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Organizations是一项账户管理服务，可帮助您将多个账户整合 AWS 账户 到一个由您创建和集中管理的组织中。

其他工具

- Git 是一个开源的分布式版本控制系统。它包括创建组织帐户的功能。
- GitHub A@@ c tions 是一个持续集成和持续交付 (CI/CD) 平台，与 GitHub 存储库紧密集成。您可以使用 GitHub Actions 来自动执行构建、测试和部署管道。
- Terraform 是一款基础设施即代码 (IaC) 工具 HashiCorp ，可帮助您创建和管理云和本地资源。

代码存储库

此模式的代码可在 GitHub role-vending-machine存储库中找到。

最佳实践

- 让正确的方法变得容易，让错误的方式变得艰难 — 让正确的事情变得容易。如果开发人员在 RVM 配置过程中苦苦挣扎，他们可能会尝试通过其他方式创建角色，这会削弱 RVM 的中心性质。请确保您的安全团队就如何安全有效地使用 RVM 提供明确的指导。

  你还应该让开发者很难做错事。使用服务控制策略 (SCPs) 或权限边界来限制哪些角色可以创建其他角色。这种方法可以帮助将角色创建限制为仅限于 RVM 和其他可信来源。
- 举个@@ 好例子 — 不可避免的是，一些开发者会将 RVM 存储库中的现有角色调整为向其新角色授予权限的非正式模板。如果您有最低权限的示例可供他们复制，则可以降低开发人员请求宽泛、通配符密集型权限的风险。如果你从具有大量通配符的高权限角色开始，那么随着时间的推移，这个问题可能会成倍增加。
- 使用命名约定和条件 — 即使开发人员不知道其应用程序将创建的所有资源名称，他们仍应使用命名约定来限制角色权限。例如，如果他们在创建 Amazon S3 存储桶，则其资源密钥的值可能看起来像，`arn:aws:s3:::myorg-myapp-dev-*`这样他们的角色除了与该名称匹配的存储桶之外没有其他权限。通过 IAM 策略强制执行命名约定还有一个额外的好处，那就是提高对命名约定的合规性。之所以出现这种改进，是因为不允许创建不匹配的资源。
- 需要拉取请求 (PR) 审查 — RVM 解决方案的价值在于它创建了一个可以审查新管道角色的中心位置。但是，只有当有护栏可以帮助确保将安全、高质量的代码提交到 RVM 时，这种设计才有用。保护用于部署代码的分支（例如`main`），使其免受直接推送，并要求批准任何针对它们的合并请求。
- 配置只读角色-默认情况下，RVM 会为每个请求的角色配置一个`readonly`版本。此角色可用于不写入数据的 CI/CD 管道，例如`terraform plan`管道工作流程。这种方法有助于防止只读工作流程行为不当时发生不必要的更改。

默认情况下，AWS 托管ReadOnlyAccess策略同时附加到只读角色和读写角色。此策略减少了在确定所需权限时进行迭代的需求，但对于某些组织来说，它可能过于宽松。如果你愿意，你可以从 Terraform 代码中删除该策略。

- 授予最低权限-遵循最低权限原则，授予执行任务所需的最低权限。有关更多信息，请参阅 IAM 文档中的授予最低权限和安全最佳实践。

操作说明

准备环境

Task	描述	所需技能
将示例存储库复制到您的 GitHub 组织。	克隆此模式的存储库或将此存储库分支到您的 GitHub 组织，以便您可以根据需要对其进行调整。 • 如果您选择克隆此存储库，则可以使用以下命令：`git clone https://github.com/aws-samples/role-vending-machine` • 如果您选择将存储库分支到您的 GitHub 组织，则可以使用以下命令： `gh repo fork aws-samples/role-vending-machine --org YOUR_ORGANIZATION_NAME`	DevOps 工程师
确定 R AWS 账户 VM 的。	确定 AWS 账户 要为 RVM 使用哪种基础架构部署。请勿使用管理账户或 root 账户。	云架构师

Task	描述	所需技能
（可选）允许组织创建管道 PRs。	**Note** 只有在要允许创建`generate_providers_and_account_vars` 工作流程时，才需要执行此步骤 PRs。 要允许贵组织的管道创建 PRs，请使用以下步骤： 1. 转到 `https://github.com/organizations/YOUR_ORG/settings/actions` 。 2. 选择“允许 GitHub 操作”以创建和批准拉取请求。 有关更多信息，请参阅 GitHub 文档中的管理存储库的 GitHub 操作设置。	DevOps 工程师

Task	描述	所需技能
向 RVM 账户授予只读权限。	在您的管理账户中创建授予您的 RVM 账户只读权限的委托策略。这允许你的 RVM GitHub 工作流程在`generate_providers_and_account_vars.py` 脚本运行时动态提取 AWS 组织账户列表。 使用以下代码并`<YOUR RVM Account ID>`替换为您在步骤 2 中选择的 AWS 账户 ID： `{ "Version": "2012-10-17", "Statement": [{ "Sid": "Statement", "Effect": "Allow", "Principal": { "AWS": "arn:aws:iam::<YOUR RVM Account ID>:root" }, "Action": ["organizations:ListAccounts", "organizations:DescribeOrganization", "organizations:DescribeOrganizationalUnit", "organizations:ListRoots", "organizations:ListAWSServi`	云管理员

Task	描述	所需技能

	<pre>ceAccessForOrganiz
	ation",
	"organiza
	tions:ListDelegate
	dAdministrators"
	],
	"Resource": "*"
	}
	]
	}</pre>	

更新示例存储库中的默认值。	要将 RVM 配置为在您的特定环境中运行 AWS 区域，请执行以下操作： 1. 使用您所在的相应区域更新`scripts/generate_providers_and_account_vars.py` 和其他文件（例如`bootstrap` 文件夹）。 2. 使用 AWS 账户 ID、Region 和您要指定的任何其他变量更新`.github/workflows/.env` 文件。	DevOps 工程师

初始化基础设施

Task	描述	所需技能
引导 RVM 存储库。	此步骤对于创建 RVM 管道本身使用的 OIDC 信任和 IAM 角色是必要的，这样它才能开始操作和出售其他角色。	DevOps 工程师

Task	描述	所需技能
	在您的 RVM 帐户的上下文中，从scripts/bootstrap 目录中手动运行terraform apply命令。根据变量文档提供任何必需的值。	

配置操作

Task	描述	所需技能
将github-workflow-rvm 和github-workflow-rvm-readonly 角色部署到所有账户。	选择符合组织实践的部署方法，例如 AFT 或 StackSets。使用该方法将scripts/assumed_role/main.tf 文件中的两个 IAM 角色（默认名称github-workflow-rvm 和github-workflow-rvm-readonly ）部署到您希望 RVM 能够在其中创建管道角色的每个账户。 这些 IAM 角色具有信任策略，允许 RVM 账户的角色承担角色（或其readonly等效角色）担任该角色。这些角色还具有 IAM 权限策略，允许他们读取和写入匹配的readonly角色（除非使用角色）github-workflow-role-* 。	AWS 管理员
运行工作generate_providers_and_account_vars 流程。	要将 RVM 配置为可以创建管道角色，请执行以下操作：	DevOps 工程师

Task	描述	所需技能
	1. 使用 workflow_dispatch 运行工作generate_providers_and_account_vars 流程。 2. 合并工作流程创建的 PR。 工作流程完成后，RVM 已准备好： • 接受新管道角色的请求。 • 根据要求创建只读或读写角色。 • 将角色部署到指定的 AWS 账户。	

故障排除

事务	解决方案
我使用 RVM 创建了一个角色，但 GitHub 无法假设。	验证 GitHub 存储库的名称是否与提供给github_workflow_roles 模块的名称相匹配。角色的作用域是为了只有一个存储库可以代替它们。 同样，请验证 GitHub 管道中使用的分支是否与提供给github_workflow_roles 模块的分支名称相匹配。通常，RVM 创建的具有写入权限的角色只能由范围限于分支的工作流程（即main源自分支）使用。main
我的只读角色无法运行其管道，因为它缺乏读取特定资源的权限。	尽管该ReadOnlyAccess 策略提供了广泛的只读权限，但该策略没有一些读取操作（例如，某些 AWS Security Hub 操作）。

事务	解决方案
	您可以使用`github-workflow-roles` 模块的`inline_policy_readonly` 参数添加特定的操作权限。

相关资源

- 使用的最佳实践 AWS CloudFormation StackSets
- 使用多个账户组织您的 AWS 环境
- 适用于 Terraform (AFT) 的 Account Factory 概述 AWS Control Tower
- 政策最佳实践

其他信息

使用 GitHub 环境

GitHub 环境是替代基于分支的角色访问限制的另一种方法。如果您更喜欢使用 GitHub 环境，以下是 IAM 信任策略中附加条件的语法示例。此语法指定只有在`Production`环境中运行 GitHub 操作时才能使用该角色。

```
"StringLike": {
    "token.actions.githubusercontent.com:sub": "repo:octo-org/octo-
repo:environment:Production"
}
```

示例语法使用以下占位符值：

- `octo-org`是 GitHub 组织名称。
- `octo-repo`是存储库名称。
- `Production`是特定的 GitHub 环境名称。

将亚马逊 CloudWatch 指标发布到 CSV 文件

由 Abdullahi Olaoye (AWS) 编写

摘要

此模式使用 Python 脚本来检索 Amazon CloudWatch 指标，并将指标信息转换为逗号分隔值 (CSV) 文件以提高可读性。该脚本将应检索其指标的 Amazon Web Services 作为必需参数。您可将 AWS 区域和 AWS 凭证配置文件指定为可选参数。如果您不指定这些参数，脚本将使用为运行脚本的工作站配置的默认区域和配置文件。脚本运行后，它会生成 CSV 文件并将其存储在同一目录中。

有关此模式提供的脚本和关联文件，请参阅附件部分。

先决条件和限制

先决条件

- Python 3.x
- AWS 命令行界面（AWS CLI）

限制

脚本当前支持以下 Amazon Web Services：

- AWS Lambda
- 亚马逊弹性计算云（亚马逊 EC2）
  - 默认情况下，脚本不收集Amazon Elastic Block Store(Amazon EBS) 卷指标。要收集 Amazon EBS 指标，您必须修改所附 `metrics.yaml` 文件。
- Amazon Relational Database Service(Amazon RDS)
  - 但是，该脚本不支持 Amazon Aurora。
- 应用程序负载均衡器
- 网络负载均衡器
- Amazon API Gateway

工具

- Amazon CloudWatch 是一项专为 DevOps 工程师、开发人员、站点可靠性工程师 (SREs) 和 IT 经理构建的监控服务。 CloudWatch 提供数据和切实可行的见解，帮助您监控应用程序、响应系统范围

的性能变化、优化资源利用率并获得统一的运营状况视图。 CloudWatch 以日志、指标和事件的形式收集监控和运营数据，并提供在 AWS 和本地服务器上运行的 AWS 资源、应用程序和服务的统一视图。

操作说明

安装并配置先决条件

Task	描述	所需技能
安装先决条件。	运行以下命令： `$ pip3 install -r requirements.txt`	开发人员
配置 AWS CLI。	运行以下命令： `$ aws configure`	开发人员

配置 Python 脚本

Task	描述	所需技能
打开脚本。	要更改脚本的默认配置，请打开 `metrics.yaml` 。	开发人员
为脚本设置周期。	即获取时间范围。默认值为 5 分钟 (300 秒)。您可更改时间段，但请注意以下限制： • 如果您指定的小时值介于 3 小时到 15 天前，请对该时段使用 60 秒（1 分钟）的倍数。 • 如果您指定的小时值介于 15 小时到 63 天前，请对该时段	开发人员

Task	描述	所需技能
	使用 300 秒（5 分钟）的倍数。 • 如果您指定的小时值大于 63 天前，请对该时段使用 3,600 秒（1 小时）的倍数。 否则，API 操作不会返回任何数据点。	
设置脚本时间。	此值指定您想要获取多少小时的指标。默认值为 1 小时。要检索多天指标，请提供以小时为单位的值。例如对于 2 天，指定 48。	开发人员
更改脚本的统计数据值。	（可选）全局统计值为 `Average`，在获取未分配特定统计值的指标时使用该值。该脚本支持统计值 `Maximum`、`SampleCount` 和 `Sum`。	开发人员

运行 Python 脚本

Task	描述	所需技能
运行 脚本。	使用以下命令： `$ python3 cwreport.py <service>` 要查看服务值列表以及可选 `profile` 参数和 `region` 参数，请运行以下命令：	开发人员

Task	描述	所需技能
	`$ python3 cwreport.py -h` 有关可选参数的更多信息，请参阅其他信息部分。	

相关资源

- 配置 AWS CLI
- 使用亚马逊 CloudWatch 指标
- 亚马逊 CloudWatch 文档
- EC2 CloudWatch Metrics
- AWS Lambda 指标
- Amazon RDS 指标
- 应用程序负载均衡器指标
- 网络负载均衡器指标
- Amazon API Gateway 指标

其他信息

脚本用法

```
$ python3 cwreport.py -h
```

语法示例

```
python3 cwreport.py <service> <--region=Optional Region> <--profile=Optional credential
 profile>
```

参数

- 服务（必填）- 您要运行脚本的服务。该脚本目前支持以下服务：AWS Lambda、Amazon、A EC2 mazon RDS、应用程序负载均衡器、网络负载均衡器和 API Gateway。

- 区域（可选）- 要从中获取指标的 AWS 区域。默认选项是 ap-southeast-1。
- 配置文件（可选）- 要使用的 AWS CLI 命名配置文件。如果未指定此参数，则使用默认配置凭证配置文件。

示例

- 要使用默认区域和ap-southeast-1默认配置的凭证来获取 Amazon EC2 指标，请执行以下操作：`$ python3 cwreport.py ec2`
- 若要指定区域并获取 API Gateway 指标：`$ python3 cwreport.py apigateway --region us-east-1`
- 要指定 AWS 配置文件并获取亚马逊 EC2 指标，请执行以下操作：`$ python3 cwreport.py ec2 --profile testprofile`
- 要指定区域和概况以获取 Amazon EC2 指标，请执行以下操作：`$ python3 cwreport.py ec2 --region us-east-1 --profile testprofile`

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS Lambda 自动化功能 AWS 账户 从 AWS Managed Microsoft AD 正面移除 Amazon EC2 条目

由 Rahul Sharad Gaikwad 博士 (AWS) 和 Tamilselvan P (AWS) 编写

摘要

Active Directory (AD) 是一款 Microsoft 脚本工具，用于管理域信息和用户与网络服务的交互。它在托管服务提供商 (MSPs) 中被广泛用于管理员工凭证和访问权限。由于 AD 攻击者可以使用不活跃的帐户来尝试入侵组织，因此找到不活跃的帐户并在例行维护计划中将其禁用非常重要。使用 AWS Directory Service for Microsoft Active Directory，你可以将 Microsoft Active Directory 作为托管服务运行。此模式可以帮助您配置 AWS Lambda 自动化，以快速查找和删除不活跃的帐户。

如果以下情况适用于您的组织，则此模式可以为您提供帮助：

- 集中式广告管理 — 如果您的组织有多个账户 AWS 账户，每个账户都有自己的广告部署，那么在所有账户中一致地管理用户帐户和访问权限可能会很困难。借助跨账户 AD 清理解决方案，您可以集中方式禁用或删除所有 AD 实例中的非活动账户。
- AD 重组或迁移 — 如果您的组织计划重组或迁移其 AD 部署，则跨账户 AD 清理解决方案可以帮助您准备环境。该解决方案可以帮助您删除不必要或不活跃的帐户，简化迁移过程并减少潜在的冲突或问题。

当你使用这种模式时，你可以获得以下好处：

- 提高数据库和服务器性能，修复来自非活动帐户的安全漏洞。
- 如果您的 AD 服务器托管在云中，则移除不活跃的帐户还可以降低存储成本，同时提高性能。您的每月账单可能会减少，因为带宽费用和计算资源都可能下降。
- 使用干净的活动目录将潜在的攻击者拒之门外。

先决条件和限制

先决条件

- 一个活跃的家长 AWS 账户 和一个或多个子女账户。在这种模式中，父账户是创建活动目录的地方。子帐户托管 Windows 服务器，并通过父帐户 Active Directory 加入。
- Git 已在本地工作站上安装和配置。

- 在本地工作站上安装和配置 Terraform。
- AWS Managed Microsoft AD 在家长帐户中配置并共享给所有子女帐户的目录。有关更多详细信息，请参阅《管理指南》中的教程：共享您的 AWS Managed Microsoft AD 目录以实现无缝 EC2 域加入。AWS Directory Service
- （父账户）的 VPC 和亚马逊弹性计算云 (Amazon) 实例 AWS Directory Service （子账户）的 VPC 之间可用的虚拟私有云 (VPC EC2) 对等 AWS Transit Gateway 连接或连接。有关更多详细信息，请参阅《AWS Directory Service 管理指南》中的在目录所有者和目录使用者账户之间配置 VPC 对等连接。
- 一台 Windows 计算机，在所有父帐户和子帐户上都配置了该`EC2WindowsUserdata`脚本。该脚本文件位于此模式的代码存储库的根目录中。
- 每个子账户上都有一个跨账户 AWS Identity and Access Management (IAM) 角色，该账户配置了信任策略，允许使用父账户中的 AWS Lambda 功能。有关更多信息，请参阅《亚马逊 EventBridge 用户指南》 AWS 账户 中的在亚马逊 EventBridge之间发送和接收事件。
- 父账户的 AWS Systems Manager 参数存储中提供以下密钥值：
  - `domainJoinUser`— 目录服务的用户名
  - `domainJoinPassword`— 目录服务的密码

  有关密钥的更多信息，请参阅《AWS Secrets Manager 用户指南》中的创建 AWS Secrets Manager 密钥。

限制

- Terraform 不会自动在儿童账户中创建资源。您必须使用手动创建以下资源 AWS Management Console：
  - 亚马逊 EventBridge 规定将亚马逊终 EC2 止事件发送到母账户
  - 使用信任政策在子 EC2 账户中创建亚马逊跨账户角色
  - VPC 对等连接或 Transit Gateway 连接
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

产品版本

- Terraform 版本 1.1.9 或更高版本
- Terraform P AWS rovider 3.0 或更高版本

架构

下图显示了该解决方案的高级架构。

架构图说明了以下过程：

1. 在儿童账户中，该 EventBridge 规则会收集所有 Amazon EC2 终止事件。该规则将父账户中存在的事件发送到 EventBridge 这些事件。
2. 从父账户 EventBridge 收集所有事件并包含触发 Lambda 函数的规则。`ADcleanup-Lambda`
3. 父账户从父账户或子账户接收任何终止事件并触发 Lambda 函数。
4. Lambda 函数使用 Python boto 模块调用 Amazon A EC2 uto Scaling 群组并获取随机实例 ID。实例 ID 用于执行 Systems Manager 命令。
5. Lambda 函数使用 boto 模块再次调 EC2 用亚马逊。Lambda 函数获取正在运行的 Windows 服务器的私有 IP 地址，并将这些地址存储在临时变量中。在步骤 5.1 和 5.2 中，正在运行的 Windows EC2 实例是从子账户中收集的。
6. Lambda 函数再次调用 Systems Manager 以获取连接到的计算机信息。 AWS Directory Service
7. AWS Systems Manager 文档有助于在 Amazon EC2 Windows 服务器上执行 PowerShell 命令，以获取连接到 AD 的计算机的私有 IP 地址。（Systems Manager 文档使用在步骤 4 中获得的实例 ID。）
8. AD 域用户名和密码存储在 AWS Systems Manager 参数存储中。 AWS Lambda 然后 Systems Manager 调用 Parameter Store 并获取用于连接 AD 的用户名和密码值。
9. 使用 Systems Manager 文档，使用先前在步骤 4 中获得的实例 ID 在亚马逊 EC2 Windows 服务器上执行 PowerShell 脚本。
10. Amazon 使用 PowerShell 命令 EC2 连接并移除未使用或处于非活动状态的计算机。 AWS Directory Service

工具

AWS 服务

- AWS Directory Service提供了多种使用微软 Active Directory (AD) 的方法， AWS 服务 例如亚马逊弹性计算云 (亚马逊 EC2)、适用于 SQL Server 的亚马逊关系数据库服务 (Amazon RDS) 和 FSx 适用于 Windows 文件服务器的亚马逊。

- AWS Directory Service for Microsoft Active Directory允许你的目录感知工作负载和 AWS 资源使用中的 Microsoft Active Directory。 AWS Cloud
- 亚马逊弹性计算云 (Amazon EC2) 在中提供可扩展的计算容量 AWS Cloud。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如， AWS Lambda 函数、使用 API 目的地的 HTTP 调用端点或其他 AWS 账户目的地的事件总线。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。借助 IAM，您可以指定谁或什么人可以访问中的服务和资源 AWS，集中管理细粒度的权限，并分析访问权限以优化权限。 AWS
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Systems Manager 可帮助您管理在 AWS Cloud中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。
- AWS Systems Manager 文档定义了 Systems Manager 对您的托管实例执行的操作。Systems Manager 包括 100 个预先配置的文档，您可以在运行时通过指定参数进行使用。
- AWS Systems Manager Parameter Store 是一项 AWS Systems Manager 功能，可为配置数据管理和密钥管理提供安全的分层存储。

其他工具

- HashiCorp Terraform 是一款基础设施即代码 (IaC) 工具，可帮助您使用代码来配置和管理云基础架构和资源。
- PowerShell是一款在 Windows、Linux 和 macOS 上运行的微软自动化和配置管理程序。
- Python 是通用的计算机编程语言。

代码存储库

此模式的代码可在 GitHub aws-lambda-ad-cleanup-terraform- samples 存储库中找到。

最佳实践

- 自动加入域名。当您启动要成为 AWS Directory Service 域一部分的 Windows 实例时，请在实例创建过程中加入该域，而不是稍后手动添加该实例。要自动加入域，请在启动新实例时从 “域加入目录”

下拉列表中选择正确的目录。有关更多详细信息，请参阅《AWS Directory Service 管理指南》中的“将 Amazon EC2 Windows 实例无缝加入您的 AWS Managed Microsoft AD 活动目录”。

- 删除未使用的帐户。在 AD 中找到从未使用过的账户是很常见的。与保留在系统中的已禁用或非活动帐户一样，被忽视的未使用帐户可能会减慢广告系统的速度或使您的组织容易受到数据泄露的影响。
- 自动清理活动目录。为了帮助降低安全风险并防止过时的账户影响广告性能，应定期进行广告清理。您可以通过编写脚本来完成大多数 AD 管理和清理任务。示例任务包括删除已禁用和非活动帐户、删除空组和非活动群组以及查找过期的用户帐户和密码。

操作说明

设置儿童账户

Task	描述	所需技能
在子账号中创建跨账户角色。	要在儿童账户中创建跨账户角色，请执行以下操作： 1. 对于每个子账号，使用名为`ec2crossaccountrole` 的托管策略创建一个名为的角色`AmazonEC2ReadOnlyAccess` 。（有关更多信息，请参阅 IAM 文档中的使用自定义信任策略创建角色。） 2. 在自定义信任策略部分，添加以下代码： `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": {`	DevOps 工程师

Task	描述	所需技能

	<pre>"Service": "ec2.amazonaws.com"
	},
	"Action": "sts:AssumeRole"
	},
	{
	"Effect": "Allow",
	"Principal": {
	"AWS": "arn:aws:iam::${Parentaccountid}:role/ADcleanuprole"
	},
	"Action": "sts:AssumeRole"
	}
	]
	}</pre>	

Task	描述	所需技能
在儿童账户中创建活动规则。	要为每个儿童账户创建 EventBridge 规则，请执行以下操作： 1. 登录孩子的账户 AWS 账户，然后打开亚马逊 EventBridge 控制台，网址为https://console.aws.amazon.com/events/。 2. 在导航窗格中，选择规则。 3. 选择创建规则。 4. 输入规则的名称和描述（可选）。 5. 对于事件总线，请选择 AWS 默认事件总线。 6. 对于规则类型，选择带有事件模式的规则。 7. 选择下一步。 8. 对于事件模式，请粘贴以下代码： `{ "source": ["aws.ec2"], "detail-type": ["EC2 Instance State-change Notification"], "detail": { "state": ["terminated"] } }` 9. 选择下一步。	DevOps 工程师

Task	描述	所需技能
	10对于目标类型，请选择其他账户或区域中的事件总线。对于事件总线作为目标，请输入父账户的事件总线 Amazon 资源名称 (ARN)。 11对于执行角色，选择为此特定资源创建新角色。 12选择 “下一步” 查看新规则的详细信息，然后选择 “创建”。 有关更多详细信息，请参阅《亚马逊 EventBridge 用户指南》中的创建对亚马逊 EventBridge事件做出反应的规则。	

Task	描述	所需技能
创建 EC2 实例并加入 AD。	要为 Windows 创建 EC2 实例，请执行以下操作： 1. 使用此模式的代码存储库中可用的`EC2WindowsUserdata` 脚本。 2. 在用户数据脚本中，修改以下代码以使用您来自父账户的`Directory service addresses` 值： `set-DnsClientServerAddress -InterfaceIndex 6 -ServerAddresses $(Directory service addresses)`	DevOps 工程师

设置本地工作站

Task	描述	所需技能
创建项目文件夹并添加文件。	要克隆存储库并创建项目文件夹，请执行以下操作： 1. 打开此模式的GitHub 存储库。 2. 选择 “代码” 按钮，在 “克隆” 下拉列表中查看要克隆的选项。 3. 在 HTTPS 选项卡上，使用网址复制克隆中提供的网址。	DevOps 工程师

Task	描述	所需技能

	4. 在您的计算机上创建一个文件夹，并使用您的项目名称命名。
	5. 在本地计算机上打开终端，然后导航到该文件夹。
	6. 要克隆 git 存储库，请使用以下命令。
	`git clone <repository-URL>.git`
	7. 克隆存储库后，使用以下命令转到克隆的目录。
	`cd <directory name>/terraform-aws-lambda-ad-cleanup/multiple-account-cleanup`
	8. 在克隆的存储库中，在您选择的集成开发环境 (IDE) 中打开此项目。	

生成`adcleanup.zip` 文件。	要压缩`lambda_function.py` 文件，请运行以下命令： `zip -r adcleanup.zip lambda_function.py`	DevOps 工程师

使用 Terraform 配置配置目标架构

Task	描述	所需技能
为 Terraform 变量提供值。	对于子账户，在`terraform.tfvars` 文件中提供以	DevOps 工程师

Task	描述	所需技能
	下arn变量的值作为字符串类型： • lambda_env_cross_role_arn • child_account_cross_role_arn	
初始化 Terraform 配置。	要初始化包含 Terraform 文件的工作目录，请运行以下命令： terraform init	DevOps 工程师
预览更改。	在部署基础架构之前，您可以预览 Terraform 将对基础架构所做的更改。要验证 Terraform 是否会根据需要进行更改，请运行以下命令： terraform plan –-var-file=examples/terraform.tfvars	DevOps 工程师
执行建议的操作。	要验证terraform plan命令的结果是否符合预期，请执行以下操作： 1. 运行以下命令：terraform apply 2. 登录并 AWS Management Console验证资源是否存在。	DevOps 工程师

验证部署

Task	描述	所需技能
执行并测试 Lambda 函数。	要验证部署是否成功完成，请执行以下操作： 1. 登录并打开 Lambda 控制台。 AWS Management Console打开函数页面，然后选择以 ADcleanup-Lambda- * 开头的函数名称。 2. 在函数概述页面上，在 “代码源” 部分的 “代码” 选项卡上选择 “测试”。 3. 要保存测试事件，请提供该事件的名称并选择保存。要测试事件，请再次选择测试。 执行结果显示函数的输出。	DevOps 工程师
查看父账户执行 EventBridge 规则的结果。	要查看基于父账户的 Amazon EC2 终止事件的 EventBridge 规则结果，请执行以下操作： 1. 终止父账户中的 EC2 实例。 2. 打开父账户的 Lambda 控制台。打开函数页面，然后选择以 ADcleanup-Lambda- * 开头的函数名称。 3. 选择监控选项卡，然后选择查看 CloudWatch 日志。	DevOps 工程师

Task	描述	所需技能
	在 CloudWatch 控制台中，日志组页面显示 Lambda 函数的结果。	
查看子账号执行 EventBridge 规则的结果。	要查看基于子女账户的 Amazon EC2 终止事件的 EventBridge 规则结果，请执行以下操作： 1. 终止子账号中的 EC2 实例。 2. 打开父账户的 Lambda 控制台。打开函数页面，然后选择以 ADcleanup-Lambda- * 开头的函数名称。 3. 选择监控选项卡，然后选择查看 CloudWatch 日志。 在 CloudWatch 控制台中，日志组页面显示 Lambda 函数的结果。	DevOps 工程师

使用后清理基础架构

Task	描述	所需技能
清理基础架构。	要清理您创建的基础架构，请使用以下命令： `terraform destroy` 要确认该`destroy`命令，请键入`yes`。	DevOps 工程师
清理后进行验证。	验证资源是否已成功删除。	DevOps 工程师

故障排除

事务	解决方案
AWS Directory Service （父账户）和 Amazon EC2 实例（子账户）之间的连接问题 — 即使 VPC 对等连接可用，您也无法将子账户的计算机加入到 AD。	在中添加路由 VPCs。有关说明，请参阅 AWS Directory Service 文档中的在目录所有者和目录使用者账户之间配置 VPC 对等连接。

相关资源

AWS 文档

- 亚马逊 EventBridge 和 AWS Identity and Access Management
- 配置 Systems Manager 所需的实例权限
- 的身份和访问管理 AWS Directory Service
- 适用于 Lambda 的基于身份的 IAM 策略
- 手动将亚马逊 EC2 Windows 实例加入您的 AWS Managed Microsoft AD 活动目录
- 使用 AWS Lambda 自动化功能 AWS 账户 从 AWS Managed Microsoft AD 中移除 Amazon EC2 条目

其他资源

- AWS 提供者（Terraform 文档）
- 后端配置（Terraform 文档）
- 安装 Terraform（Terraform 文档）
- Python boto 模块（Python Package 索引存储库）
- Terraform 二进制文件下载（Terraform 文档）

使用 AWS Lambda 自动化功能 AWS 账户 从 AWS Managed Microsoft AD 中移除 Amazon EC2 条目

由 Rahul Sharad Gaikwad 博士 (AWS) 和 Tamilselvan P (AWS) 编写

摘要

Active Directory (AD) 是一款 Microsoft 脚本工具，用于管理域信息和用户与网络服务的交互。它在托管服务提供商 (MSPs) 中被广泛用于管理员工凭证和访问权限。由于 AD 攻击者可以使用不活跃的帐户来尝试入侵组织，因此找到不活跃的帐户并在例行维护计划中将其禁用非常重要。使用 AWS Directory Service for Microsoft Active Directory，你可以将 Microsoft Active Directory 作为托管服务运行。

此模式可以帮助您配置 AWS Lambda 自动化，以快速查找和删除不活跃的帐户。当你使用这种模式时，你可以获得以下好处：

- 提高数据库和服务器性能，修复来自非活动帐户的安全漏洞。
- 如果您的 AD 服务器托管在云中，则移除不活跃的帐户还可以降低存储成本，同时提高性能。您的每月账单可能会减少，因为带宽费用和计算资源都可能下降。
- 使用干净的活动目录将潜在的攻击者拒之门外。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- Git 已在本地工作站上安装和配置。
- 在本地工作站上安装和配置 Terraform。
- 带有活动目录模块的 Windows 计算机 (`ActiveDirectory`)。
- 中的目录 AWS Managed Microsoft AD 和存储在参数存储区AWS Systems Manager 参数中的凭证。
- AWS Identity and Access Management (IAM) 角色拥有工具中 AWS 服务 列出的权限。有关 IAM 的更多信息，请参阅相关资源。

限制

- 此模式不支持跨账户设置。

- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

产品版本

- Terraform 版本 1.1.9 或更高版本
- Terraform AWS Provider 3.0 或更高版本

架构

下图显示了此模式的工作流和体系结构组件。

图表显示了以下工作流：

1. Amazon 根据 cron 表达式 EventBridge 触发该 AWS Lambda 函数。（对于这种模式，cron 表达式计划为每天一次。）
2. 所需的 IAM 角色和策略是 AWS Lambda 通过 Terraform 创建并附加到的。
3. 使用 Python boto 模块执行该 AWS Lambda 函数并调用亚马逊弹性计算云 (Amazon EC2) Auto Scaling Groups。Lambda 函数获取随机实例 ID。实例 ID 用于执行 AWS Systems Manager 命令。
4. AWS Lambda EC2 使用 boto 模块再次调用 Amazon，获取正在运行的 Windows 服务器的私有 IP 地址，并将这些地址存储在临时变量中。
5. AWS Lambda 再次调用 Systems Manager 以获取所连接的计算机信息 AWS Directory Service。
6. AWS Systems Manager 文档有助于在 Amazon EC2 Windows 服务器上执行 PowerShell 脚本，以获取与 AD 连接的计算机的私有 IP 地址。
7. AD 域用户名和密码存储在 AWS Systems Manager 参数存储中。 AWS Lambda 然后 Systems Manager 调用 Parameter Store 并获取用于连接 AD 的用户名和密码值。
8. 使用 Systems Manager 文档，使用先前在步骤 3 中获得的实例 ID 在亚马逊 EC2 Windows 服务器上执行 PowerShell 脚本。
9. Amazon 使用 PowerShell 命令 AWS Directory Service 进行 EC2 连接，并移除未使用或处于非活动状态的计算机。

工具

Amazon Web Services

- AWS Directory Service提供了多种使用微软 Active Directory (AD) 的方法， AWS 服务 例如亚马逊弹性计算云 (亚马逊 EC2)、适用于 SQL Server 的亚马逊关系数据库服务 (Amazon RDS) 和 FSx 适用于 Windows 文件服务器的亚马逊。
- AWS Directory Service for Microsoft Active Directory允许你的目录感知工作负载和 AWS 资源使用中的 Microsoft Active Directory。 AWS Cloud
- 亚马逊弹性计算云 (Amazon EC2) 在中提供可扩展的计算容量 AWS Cloud。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如， AWS Lambda 函数、使用 API 目的地的 HTTP 调用端点或其他 AWS 账户目的地的事件总线。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS资源，从而帮助您安全地管理对资源的访问权限。借助 IAM，您可以指定谁或什么人可以访问中的服务和资源 AWS，集中管理细粒度的权限，并分析访问权限以优化权限。 AWS
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Systems Manager 可帮助您管理在 AWS Cloud中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。
- AWS Systems Manager 文档定义了 Systems Manager 对您的托管实例执行的操作。Systems Manager 包括 100 个预先配置的文档，您可以在运行时通过指定参数进行使用。
- AWS Systems Manager Parameter Store 是一项 AWS Systems Manager 功能，可为配置数据管理和密钥管理提供安全的分层存储。

其他工具

- HashiCorp Terraform 是一款开源基础设施即代码 (IaC) 工具，可帮助您使用代码来配置和管理云基础架构和资源。
- PowerShell是一款在 Windows、Linux 和 macOS 上运行的微软自动化和配置管理程序。
- Python 是通用的计算机编程语言。

C 代码存储库

此模式的代码可在 GitHub 自定义 AD 清理自动化解决方案存储库中找到。

最佳实践

- 自动加入域名。当您启动要成为 AWS Directory Service 域一部分的 Windows 实例时，请在实例创建过程中加入该域，而不是稍后手动添加该实例。要自动加入域，请在启动新实例时从 “域加入目录” 下拉列表中选择正确的目录。有关更多详细信息，请参阅《AWS Directory Service 管理指南》中的 “将 Amazon EC2 Windows 实例无缝加入您的 AWS Managed Microsoft AD 活动目录”。
- 删除未使用的帐户。在 AD 中找到从未使用过的账户是很常见的。与保留在系统中的已禁用或非活动帐户一样，被忽视的未使用帐户可能会减慢广告系统的速度或使您的组织容易受到数据泄露的影响。
- 自动清理活动目录。为了帮助降低安全风险并防止过时的账户影响广告性能，应定期进行广告清理。您可以通过编写脚本来完成大多数 AD 管理和清理任务。示例任务包括删除已禁用和非活动帐户、删除空组和非活动群组以及查找过期的用户帐户和密码。

操作说明

设置您的环境

Task	描述	所需技能
创建项目文件夹，然后添加文件。	要克隆存储库并创建项目文件夹，请执行以下操作： 1. 打开此模式的GitHub 存储库。 2. 选择 “代码” 按钮，在 “克隆” 下拉列表中查看要克隆的选项。 3. 在 HTTPS 选项卡上，使用网址复制 Clone 中提供的网址。 4. 在您的计算机上创建一个文件夹，并使用您的项目名称命名。 5. 在本地计算机上打开终端，然后导航到该文件夹。 6. 要克隆 git 存储库，请使用以下命令。	DevOps 工程师

Task	描述	所需技能

	`git clone <repository-URL>.git`
	7. 克隆存储库后，使用以下命令转到克隆的目录。
	`cd <directory name>`
	8. 在克隆的存储库中，在您选择的集成开发环境 (IDE) 中打开此项目。	

使用 Terraform 配置来配置目标架构

Task	描述	所需技能
初始化 Terraform 配置。	要初始化包含 Terraform 文件的工作目录，请运行以下命令。 `terraform init`	DevOps 工程师
预览更改。	在部署基础架构之前，您可以预览 Terraform 将对基础架构所做的更改。要验证 Terraform 是否会根据需要进行更改，请运行以下命令。 `terraform plan`	DevOps 工程师
执行建议的操作。	要验证`terraform plan`命令的结果是否符合预期，请执行以下操作： 1. 运行以下命令。 `terraform apply`	DevOps 工程师

Task	描述	所需技能
	2. 登录并 AWS Management Console验证资源是否存在。	
清理基础架构。	要清理您创建的基础架构，请使用以下命令。 `terraform destroy` 要确认销毁命令，请键入`yes`。	DevOps 工程师

验证部署

Task	描述	所需技能
执行并测试 Lambda 函数。	要验证部署是否成功完成，请执行以下操作： 1. 登录 AWS Management Console 并打开控制台。打开函数页面，然后选择以 ADcleanup-Lambda- * 开头的函数名称。 2. 在函数概述页面上，在 “代码源” 部分的 “代码” 选项卡上选择 “测试”。 3. 要保存测试事件，请提供该事件的名称并选择保存。然后，要测试该事件，请再次选择测试。 执行结果显示函数的输出。	DevOps 工程师

Task	描述	所需技能
查看 Lambda 函数的结果。	在这种模式中， EventBridge 规则每天执行 Lambda 函数一次。要查看 Lambda 函数的结果，请执行以下操作： 1. 登录 AWS Management Console 并打开 AWS Lambda 控制台。打开函数页面，选择以 ADcleanup-Lambda- * 开头的函数名称。 2. 选择 “监控” 选项卡，然后选择 “查看 CloudWatch 日志”。 在 CloudWatch 控制台中，日志组页面显示 Lambda 函数的结果。	DevOps 工程师

使用后清理基础架构

Task	描述	所需技能
清理基础设施。	要清理您创建的基础架构，请使用以下命令。 `terraform destroy` 要确认销毁命令，请键入`yes`。	DevOps 工程师
清理后进行验证。	验证资源是否已成功删除。	DevOps 工程师

故障排除

事务	解决方案
如果您尝试移除 AD 计算机，则会收到 “访问被拒绝” 消息。无法删除 AD 计算机，因为默认情况下，该操作会尝试删除作为 AD 服务一部分连接的两个私有 IP 地址。	要避免出现此错误，请在列出 AD 计算机输出与运行 Windows 的计算机的输出之间的差异时，使用以下 Python 操作忽略前两台计算机。 `Difference = Difference[2:]`
当 Lambda 在 Windows 服务器上执行 PowerShell 脚本时，它预计 Active Directory 模块在默认情况下可用。如果模块不可用，Lambda 函数会生成一个错误，指出 “未在实例上安装 G AdComputer et-”。	为避免此错误，请使用 EC2 实例的用户数据安装所需的模块。使用此模式 GitHub 存储库中的EC2WindowsUserdata脚本。

相关资源

AWS 文档

- 亚马逊 EventBridge 和 AWS Identity and Access Management
- 配置 Systems Manager 所需的实例权限
- 的身份和访问管理 AWS Directory Service
- 手动将亚马逊 EC2 Windows 实例加入您的 AWS Managed Microsoft AD 活动目录
- 在中使用基于身份的 IAM 策略 AWS Lambda

其他资源

- AWS 提供者（Terraform 文档）
- 后端配置（Terraform 文档）
- 安装 Terraform（Terraform 文档）
- Python boto 模块（Python Package 索引存储库）
- Terraform 二进制文件下载（Terraform 文档）

AWS Glue 使用 pytest 框架对 Python ETL 作业运行单元测试

创建者：Praveen Kumar Jeyarajan (AWS) 和 Vaidy Sankaran (AWS)

摘要

您可以在本地开发环境 AWS Glue中为 Python 提取、转换和加载 (ETL) 作业运行单元测试，但是在 DevOps 管道中复制这些测试可能既困难又耗时。在技术堆栈上 AWS 对大型机 ETL 流程进行现代化改造时，单元测试可能特别具有挑战性。此模式向您展示了如何简化单元测试，同时保持现有功能完好无损，在发布新功能时避免中断关键应用程序功能，并维护高质量的软件。您可以使用此模式中的步骤和代码示例， AWS Glue 通过使用中的 pytest 框架，为 Python ETL 作业运行单元测试。 AWS CodePipeline您也可以使用此模式来测试和部署多个 AWS Glue 作业。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- AWS Glue 您的库的亚马逊 Elastic Container Registry (Amazon ECR) 图片 URI，从亚马逊 ECR 公共画廊下载
- 带有目标 AWS 账户 配置文件的 Bash 终端（在任何操作系统上）和 AWS 区域
- Python 3.10 或更高版本
- Pytest
- 用于测试的 Moto Python 库 AWS 服务

架构

下图描述了如何将基于 Python 的 AWS Glue ETL 流程的单元测试整合到典型的企业级管道 AWS DevOps 中。

图表显示了以下工作流：

1. 在源代码阶段， AWS CodePipeline 使用版本控制的亚马逊简单存储服务 (Amazon S3) 存储桶来存储和管理源代码资产。这些资源包括 Python ETL 作业示例 (`sample.py`)、单元测试文件 (`test_sample.py`) 和 AWS CloudFormation 模板。然后，将最新的代码从主分支 CodePipeline 传输到 AWS CodeBuild 项目以进行进一步处理。

2. 在构建和发布阶段，在 AWS Glue 公共 Amazon ECR 映像的帮助下，对上一个源代码阶段的最新代码进行了单元测试。然后，将测试报告发布到 CodeBuild 报告组。 AWS Glue 图书馆的公共 Amazon ECR 存储库中的容器镜像包括本地运行所需的所有二进制文件和PySpark基于单元测试的 ETL 任务。 AWS Glue 公共容器存储库有三个图像标签，每个版本对应一个镜像标签 AWS Glue。出于演示目的，此模式使用了 `glue_libs_4.0.0_image_01` 映像标签。要在中 CodeBuild将此容器映像用作运行时映像，请复制与您要使用的图像标签相对应的图像 URI，然后更新`TestBuild`资源 GitHub 存储库中的`pipeline.yml`文件。
3. 在部署阶段， CodeBuild 项目将启动，如果所有测试都通过，它将代码发布到 Amazon S3 存储桶。
4. 用户使用`deploy`文件夹中的 CloudFormation 模板部署 AWS Glue 任务。

工具

AWS 服务

- AWS CodeBuild是一项完全托管的生成服务，可帮助您编译源代码、运行单元测试和生成可随时部署的工件。
- AWS CodePipeline帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件更改所需的步骤。
- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- AWS Glue是一项完全托管的 ETL 服务。它可以帮助您在数据存储和数据流之间对数据进行可靠地分类、清理、扩充和移动。
- Amazon Simple Storage Service (Amazon S3) 是一项对象存储服务，可提供业界领先的可扩展性、数据可用性、安全性和性能。

其他工具

- Python 是一种高级解释型通用编程语言。
- Moto 是一个用于测试的 Python 库 AWS 服务。
- Pytest 是一个用于编写小型单元测试的框架，单元测试可扩展以支持应用程序和库的复杂功能测试。
- 的 Python ETL 库 AWS Glue 是 Python 库的存储库，这些库用于本地开发的 PySpark 批处理作业。 AWS Glue

代码存储库

此模式的代码可在 GitHub aws-glue-jobs-unit- testing 存储库中找到。存储库包含以下资源：

- 文件夹中基于 Python 的 AWS Glue 作业示例 `src`
- `tests` 文件夹中的关联单元测试用例（使用 pytest 框架构建）
- 文件夹中的 CloudFormation 模板（用 YAML 编写）`deploy`

最佳实践

CodePipeline 资源安全

最佳做法是对连接到您的管道的源存储库使用加密和身份验证 CodePipeline。有关更多信息，请参阅 CodePipeline 文档中的安全最佳实践。

监控和记录 CodePipeline 资源

最佳做法是使用 AWS 日志记录功能来确定用户在您的账户中执行了哪些操作以及他们使用了哪些资源。日志文件显示以下内容：

- 操作的时间和日期
- 操作的源 IP 地址
- 由于权限不足而失败的操作

记录功能可在 AWS CloudTrail 和 Amazon Ev CloudWatch ents 中使用。您可以使用 CloudTrail 记录由您或代表您进行的 AWS API 调用和相关事件 AWS 账户。有关更多信息，请参阅 CodePipeline 文档 AWS CloudTrail中的使用记录 CodePipeline API 调用。

您可以使用 “ CloudWatch 事件” 来监控您的 AWS Cloud 资源和正在运行的应用程序 AWS。您也可以在 “ CloudWatch 事件” 中创建警报。有关更多信息，请参阅 CodePipeline 文档中的监控 CodePipeline 事件。

操作说明

部署源代码

Task	描述	所需技能

准备代码存档以进行部署。	1. `code.zip`从 GitHub aws-glue-jobs-unit-testing 存储库下载，或者使用命令行工具自己创建.zip 文件。例如，您可以在 Linux 或 Mac 上通过在终端中运行以下命令来创建 .zip 文件：
	`git clone https://github.com/aws-samples/aws-glue-jobs-unit-testing.git`
	`cd aws-glue-jobs-unit-testing`
	`git checkout master`
	`zip -r code.zip src/ tests/ deploy/`
	2. 登录AWS Management Console并选择您选择 AWS 区域 的。
	3. 创建 Amazon S3 存储桶，然后将.zip 压缩包和`code.zip`文件（之前已下载）上传到您创建的 Amazon S3 存储桶。
		DevOps 工程师

创建 CloudFormation 堆栈。	1. 登录 AWS Management Console 并打开控制CloudFormation 台。 2. 选择创建堆栈，然后选择使用现有的资源（导入资源）。	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	3. 在创建堆栈页面的指定模板部分，选择上传模板文件，然后选择 pipeline.yml 模板（从存储库下载）。GitHub 然后选择下一步。 4. 在堆栈名称中 glue-unit-testing-pipeline，输入或选择您选择的堆栈名称。 5. 对于 ApplicationStackName，请使用预先填充的 glue-codepipeline-app名称。这是管道创建的 CloudFormation 堆栈的名称。 6. 对于 BucketName，请使用预先填充的 aws-glue-artifacts-us-east-1 存储桶名称。这是包含.zip 文件的 Amazon S3 存储桶的名称，管道使用该存储桶来存储代码项目。 7. 对于 CodeZipFile，请使用预先填充的 code.zip 值。这是示例代码 Amazon S3 对象的密钥名称。此对象应为 .zip 文件。 8. 对于 TestReportGroupName，请使用预先填充的glue-unittest-report名称。这是为存储单元 CodeBuild 测试报告而创建的测试报告组的名称。	

Task	描述	所需技能
	9. 选择下一步，然后在配置堆栈选项页面上再次选择下一步。 10.在 “查看” 页面的 “能力” 下，选择 “我确认 CloudFormation 可能会使用自定义名称创建 IAM 资源” 选项。 11选择提交。堆栈创建完成后，您可以在资源选项卡上看到已创建的资源。创建堆栈需要约 5-7 分钟的时间。 堆栈使用 Amazon S3 作为源来创建 CodePipeline 视图。在上面的步骤中，管道是 aws-glue-unit-test-pipelin e。	

运行单元测试

Task	描述	所需技能
运行管线中的单元测试。	1. 要测试已部署的管道，请登录 AWS Management Console，然后打开CodePipeline 控制台。 2. 选择 CloudFormation 堆栈创建的管道，然后选择 Release change。管道开始运行（使用 Amazon S3 存储桶中的最新代码）。 3. Test_and_Build 阶段完成后，选择详细信息选项卡，然后检查日志。	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	4. 选择报告选项卡，然后从报告历史记录中选择测试报告以查看单元测试结果。 5. 部署阶段完成后，在 AWS Glue 控制台上运行并监控已部署的 AWS Glue 作业。有关更多信息，请参阅 AWS Glue 文档 AWS Glue中的监控。	

清理所有 AWS 资源

Task	描述	所需技能
在环境中清理资源。	为避免额外的基础设施成本，请务必在尝试此模式中提供的示例后删除堆栈。 1. 打开CloudFormation 控制台，然后选择您创建的堆栈。 2. 选择删除。这将删除您的堆栈创建的所有资源，包括 AWS Identity and Access Management (IAM) 角色、IAM 策略和 CodeBuild 项目。	AWS DevOps， DevOps 工程师

故障排除

事务	解决方案
CodePipeline 服务角色无法访问 Amazon S3 存储桶。	• 对于附加到您的 CodePipeline 服务角色的策略，`s3:ListBucket` 请添加到策略中的操作列表中。有关查看服务角色策略的说明，请参阅查看管道 ARN 和服务角色 ARN（控制台）。按照向服务角色添加权限中所述编辑 CodePipeline 服务角色的策略声明。 • 对于附加到管道的 Amazon S3 工件存储桶的基于资源的策略（也称为项目存储桶策略），请添加一条允许 CodePipeline 服务角色使用该`s3:ListBucket` 权限的声明。
CodePipeline 返回一个错误，指示 Amazon S3 存储桶未进行版本控制。	CodePipeline 要求对源 Amazon S3 存储桶进行版本控制。在您的 Amazon S3 存储桶上启用版本控制。有关说明，请参阅在存储桶上启用版本控制。

相关资源

- AWS Glue
- 在本地开发和测试 AWS Glue 作业
- AWS CloudFormation 适用于 AWS Glue

其他信息

此外，您可以使用 AWS Command Line Interface (AWS CLI) 部署 AWS CloudFormation 模板。有关更多信息，请参阅 CloudFormation 文档中的使用转换快速部署模板。

在 Amazon S3 中设置 Helm v3 图表存储库

创建者：Abhishek Sharma (AWS)

摘要

注意： CodeCommit AWS 不再向新客户开放。AWS 的现有客户 CodeCommit 可以继续照常使用该服务。了解更多

此模式通过将 Helm v3 存储库集成到 Amazon Web Services (AWS) 云上的 Amazon Simple Storage Service (Amazon S3) 中，帮助您高效管理 Helm v3 图表。要使用此模式，您必须熟悉 Kubernetes 和 Helm（Kubernetes 包管理器）。使用 Helm 存储库来存储图表和控制图表版本，可以缩短停机期间的平均恢复时间 (MTTR)。

这种模式使用 AWS CodeCommit 创建 Helm 存储库，并使用 S3 存储桶作为 Helm 图表存储库，这样整个组织的开发人员就可以集中管理和访问图表。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- Python 版本 2.7.12 或更高版本
- pip
- 带有子网和亚马逊弹性计算云 (Amazon) 实例的虚拟私有云 (VPC EC2)
- Git 安装在 EC2 实例上
- 用于创建 S3 存储桶的 AWS Identity and Access Management (IAM) 访问权限
- 通过客户端计算机访问 Amazon S3 的 IAM（编程或角色）
- AWS CodeCommit 存储库
- AWS 命令行界面（AWS CLI）

产品版本

- Helm v3
- Python 版本 2.7.12 或更高版本

架构

目标技术堆栈

- Amazon S3
- AWS CodeCommit
- Helm
- Kubectl
- Python 和 pip
- Git
- helm-3 插件

目标架构

自动化和扩缩

- 您可以将 Helm 整合到现有的（连续integration/continuous delivery (CI/CD）自动化工具中，以自动执行 Helm 图表的打包和版本控制（超出此模式的范围）。
- GitVersion 或者 Jenkins 内部版本号可用于自动控制图表的版本。

工具

- Helm – Helm 是 Kubernetes 的软件包管理器，可帮助您在 Kubernetes 集群上安装和管理应用程序。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项面向互联网的存储服务。您可以通过 Amazon S3 随时在 Web 上的任何位置存储和检索的任意大小的数据。
- helm-s3 插件 – helm-s3 插件支持与 Amazon S3 的交互。它可与 Helm v2 或 Helm v3 一起使用。

操作说明

安装并验证 Helm v3

Task	描述	所需技能
安装 Helm v3 客户端。	要在本地系统上下载并安装 Helm 客户端，请运行以下命令：`sudo curl https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 | bash`	云管理员、 DevOps 工程师
验证 Helm 安装。	要验证 Helm 客户端，请运行以下命令：`helm version --short`	云管理员、 DevOps 工程师

将 S3 存储桶初始化为 Helm 存储库

Task	描述	所需技能
为 Helm 图表创建 S3 存储桶。	创建唯一 S3 存储桶。在存储桶中，创建一个名为 `stable/myapp` 的文件夹。此模式中的示例使用 `s3://my-helm-charts/stable/myapp` 作为目标图表存储库。	云管理员、 DevOps 工程师
安装适用于 Amazon S3 的 helm-s3 插件。	要在客户端计算机上安装 helm-s3 插件，请运行以下命令：`helm plugin install https://github.com/hypnoglow/helm-s3.git`	云管理员、 DevOps 工程师

Task	描述	所需技能
初始化 Amazon S3 Helm 存储库。	要将目标文件夹初始化为 Helm 存储库，请使用以下命令：`helm s3 init s3://my-helm-charts/stable/myapp` 该命令在目标系统中创建 `index.yaml` 文件，用于跟踪存储在该位置的所有图表信息。	云管理员、 DevOps 工程师
验证新创建的 Helm 存储库。	要验证 `index.yaml` 文件是否已创建，请运行以下命令：`aws s3 ls s3://my-helm-charts/stable/myapp/`	云管理员、 DevOps 工程师
将 Amazon S3 存储库添加至客户端计算机的 Helm 中。	要向 Helm 客户端计算机中添加目标存储库别名，请使用以下命令：`helm repo add stable-myapp s3://my-helm-charts/stable/myapp/`	云管理员、 DevOps 工程师

在 Amazon S3 Helm 存储库中打包和发布图表

Task	描述	所需技能
克隆您的 Helm 图表。	如果您的 CodeCommit 存储库中没有本地 Helm 图表，请运行以下命令从存储 GitHub 库中克隆它们：`git clone <url_of_your_helm_source_code>.git`	云管理员、 DevOps 工程师

Task	描述	所需技能
打包本地 Helm 图表。	要打包您创建或克隆的图表，请使用以下命令：`helm package ./my-app` 例如，此模式使用 `my-app` 图表。该命令将 `my-app` 图表文件夹的所有内容打包成存档文件，该文件使用 `Chart.yaml` 文件中提到的版本号命名。	云管理员、 DevOps 工程师
将本地数据包存储在 Amazon S3 Helm 存储库中。	要将本地数据包上传至 Amazon S3 中的 Helm 存储库，请运行以下命令：`helm s3 push ./my-app-0.1.0.tgz stable-myapp` 在该命令中，`my-app` 是您的图表文件夹名称，`0.1.0` 是 `Chart.yaml` 中提及的图表版本，`stable-myapp` 是目标存储库别名。	云管理员、 DevOps 工程师
搜索 Helm 图表。	要确认图表同时出现在本地和 Amazon S3 Helm 存储库中，请运行以下命令：`helm search repo stable-myapp`	云管理员、 DevOps 工程师

升级 Helm 存储库

Task	描述	所需技能
修改和打包图表。	在 `values.yaml` 中，将 `replicaCount` 值设置为	云管理员、 DevOps 工程师

Task	描述	所需技能
	1，然后打包图表，这次将 `Chart.yaml` 中的版本更改为 `0.1.1`。理想情况下，版本控制是通过在 CI/CD 管道中使用诸如 GitVersion 或 Jenkins 内部版本号之类的工具通过自动化来实现的。自动生成的版本号超出了这种模式范围。要打包图表，请运行以下命令：`helm package ./my-app/`	
将新版本推送至 Amazon S3 中的 Helm 存储库。	要将新软件包（0.1.1 版）推送到 Amazon S3 中的 my-helm-chartsHelm 存储库，请运行以下命令：`helm s3 push ./my-app-0.1.1.tgz stable-myapp`	云管理员、 DevOps 工程师

验证更新的 Helm 图表。	要确认更新后的图表同时出现在本地和 Amazon S3 Helm 存储库中，请运行以下命令。
	`helm repo update`
	`helm search repo stable-myapp`
		云管理员、 DevOps 工程师

从 Amazon S3 Helm 存储库中搜索并安装图表

Task	描述	所需技能
搜索 my-app 图表的所有版本。	要查看图表的所有可用版本，请使用 `--versions` 标志运行以下命令：`helm search`	DevOps 工程师

Task	描述	所需技能
	`repo my-app --versions` 如果没有标志，Helm 默认会显示图表的最新上传版本。	
从 Amazon S3 Helm 存储库安装图表。	自动安装超出了此模式的范围，但您可手动安装。上一个任务的搜索结果显示了 `my-app` 图表的多个版本。要从 Amazon S3 Helm 存储库安装新版本 (0.1.1)，请使用以下命令：`helm upgrade --install my-app-release stable-myapp/my-app --version 0.1.1 --namespace dev`	DevOps 工程师

通过使用 Helm 回滚至以前的版本

Task	描述	所需技能
查看特定修订版的详细信息。	自动回滚超出了此模式的范围，但您可手动回滚到较早的版本。在切换或回滚至工作版本前，以及要在安装修订版之前进行额外验证，请使用以下命令查看已传递至每个修订版的值：`helm get values --revision=2 my-app-release`	DevOps 工程师
回滚至以前的版本。	自动回滚超出了此模式范围。要手动回滚至以前的修订版，请使用以下命令：`helm`	DevOps 工程师

Task	描述	所需技能
	`rollback my-app-release 1` 此示例正在回滚至修订版号 1。	

相关资源

- HELM 文档
- helm-s3 插件（MIT 许可证）
- Amazon S3

使用 AWS 和 AW CodePipeline S CDK 设置 CI/CD 管道

创建者：Konstantin Zarudaev (AWS)、Cizer Pereira (AWS)、Lars Kinder (AWS) 和 Yasha Dabas (AWS)

主页

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多通过持续集成和持续交付实现软件构建和发布过程的自动化（CI/CD) supports repeatable builds and rapid delivery of new features to your users. You can quickly and easily test each code change, and you can catch and fix bugs before releasing your software. By running each change through your staging and release process, you can verify the quality of your application or infrastructure code. CI/CD体现了一种文化、一套操作原则和一系列实践，可帮助应用程序开发团队更频繁、更可靠地交付代码变更）。该实现也被称之为 CI/CD 管线。

此模式定义了亚马逊网络服务 (AWS) 上与 AWS 存储库的可重复使用的持续集成和持续交付 (CI/CD) 管道。 CodeCommit AWS CodePipeline 管道是使用 AW S Cloud Development Kit (AWS CDK) v2 编写的。

使用 CodePipeline，您可以通过 AWS 管理控制台界面、AWS 命令行界面 (AWS CLI)、AWS 或 AWS 对软件发布过程的不同阶段进行建模。 CloudFormation SDKs此模式演示了使用 AWS CDK 的实现 CodePipeline 及其组件。除了构造库外，AWS CDK 还包括一个工具包（CLI 命令 `cdk`），它是与您的 AWS CDK 应用程序交互的主要工具。除其他功能外，该工具包还提供将一个或多个堆栈转换为 CloudFormation 模板并将其部署到 AWS 账户的功能。

该管线包括验证第三方库安全性的测试，有助于确保在指定环境中快速、自动发布。您可通过对应用程序进行验证来提高应用程序的整体安全性。

这种模式的目的是加快您使用 CI/CD 管道来部署代码，同时确保您部署的资源符合 DevOps 最佳实践。实施示例代码后，您将拥有一个 AWS， CodePipeline其中包含整理、测试、安全检查、部署和部署后流程。此模式还包括 Makefile 步骤。通过 Makefile，开发人员可以在本地重现 CI/CD 步骤并提高开发过程的速度。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 基本定义如下：

- AWS CDK
- AWS CloudFormation
- AWS CodePipeline
- TypeScript

限制

此模式 TypeScript 仅将 AWS CDK 用于。它不包含 AWS CDK 支持的其他语言。

产品版本

使用以下工具的最新版本：

- AWS 命令行界面（AWS CLI）
- cfn_nag
- git-remote-codecommit
- Node.js

架构

目标技术堆栈

- AWS CDK
- AWS CloudFormation
- AWS CodeCommit
- AWS CodePipeline

目标架构

该管道由 AWS CodeCommit 存储库的更改触发 (`SampleRepository`)。一开始， CodePipeline 构建工件，更新自身，然后启动部署过程。生成的管线将解决方案部署至三个独立的环境：

- 开发 – 在活跃的开发环境进行三步代码检查
- 测试 - 集成和回归测试环境
- Prod – 生产环境

开发阶段包含的三个步骤是 linting、安全性和单元测试。这些步骤并行运行，以加快流程。为确保管线仅提供可运行的构件，每当流程中的某个步骤失败时，它就会停止运行。开发阶段部署后，管线运行验证测试以验证结果。如果成功，管线会将构件部署到测试环境，其中包含部署后验证。最后一步是将构件部署至 Prod 环境。

下图显示了从 CodeCommit 存储库到由执行的构建和更新过程的工作流程 CodePipeline、三个开发环境步骤，以及三个环境中每个环境中的后续部署和验证。

工具

Amazon Web Services

- AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，可帮助您在代码中定义和预调配 Amazon Web Services Cloud 基础设施。
- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。在这种模式中， CloudFormation 模板可用于创建 CodeCommit 存储库和 C CodePipeline I/CD 管道。
- AWS CodeCommit 是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。
- AWS CodePipeline 是一项 CI/CD 服务，可帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件更改所需的步骤。
- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。

其他工具

- cfn_nag 是一个开源工具，可在 CloudFormation 模板中查找模式以识别潜在的安全问题。
- git-remote-codecommit是一个通过扩展 Git 从 CodeCommit 存储库中推送和提取代码的实用工具。
- Node.js 是一个事件驱动的 JavaScript 运行时环境，专为构建可扩展的网络应用程序而设计。

代码

此模式的代码可在 GitHub AWS CodePipeline with CI/CD 实践存储库中找到。

最佳实践

查看资源（例如 AWS Identity and Access Management (IAM) 策略），以确认它们符合您的组织最佳实践。

操作说明

安装工具

Task	描述	所需技能

在 macOS 或 Linux 上安装工具。	如果您使用的是 macOS 或 Linux，则可以通过在首选终端中运行以下命令或使用 Homebrew for Linux 安装这些工具。
	
	`brew install`
	`brew install git-remote-codecommit`
	`brew install ruby brew-gem`
	`brew-gem install cfn-nag`
		DevOps 工程师

设置 AWS CLI。	要设置 AWS CLI，请使用适合操作系统的说明： • Windows：使用 AWS CLI 凭证帮助程序在 Windows 上设置与 AWS CodeCommit 存储库的 HTTPS 连接的步骤 • Linux、macOS、Unix：使用 AWS CLI 凭证助手在 Linux、macOS 或 Unix 上使用 HTTPS 连接到 AWS	DevOps 工程师

Task	描述	所需技能
	CodeCommit 存储库的设置步骤	

设置初始部署

Task	描述	所需技能

下载或克隆代码。	要获取此模式使用的代码，请执行以下操作之一：
	• 从 GitHub repo 的版本中下载最新的源代码，然后将下载的文件解压缩到一个文件夹中。
	• 通过运行以下命令克隆项目。
	`git clone --depth 1 https://github.com/aws-samples/aws-codepipeline-cicd.git`
	从克隆的存储库中删除 `.git` 目录。
	`cd ./aws-codepipeline-cicd`
	`rm -rf ./.git`
	稍后，您将使用新创建的 AWS CodeCommit 存储库作为远程来源。
		DevOps 工程师

连接至 Amazon Web Services account。	您可使用临时安全令牌或登录区身份验证进行连接。要确认	DevOps 工程师

Task	描述	所需技能

	您使用的是正确的账户和 AWS 区域，请运行以下命令。
	
	`AWS_REGION="eu-west-1"`
	`ACCOUNT_NUMBER=$(aws sts get-caller-identity --query Account --output text)`
	`echo "${ACCOUNT_NUMBER}"`	

引导环境。	要引导 AWS CDK 环境，请运行以下命令。
	
	`npm install`
	`npm run cdk bootstrap "aws://${ACCOUNT_NUMBER}/${AWS_REGION}"`
	
	成功引导环境后，应该显示以下输出。
	
	`#  Bootstrapping environment aws://{account}/{region}...`
	`#  Environment aws://{account}/{region} bootstrapped`
	
	有关更多信息，请参阅 AWS CDK 文档中的 AWS CDK 引导。
		DevOps 工程师

Task	描述	所需技能

合成模板。	要合成 AWS CDK 应用程序，请使用 `cdk synth` 命令。
	`npm run cdk synth`
	您应当看到如下输出。
	`Successfully synthesized to <path-to-directory>/aws-codepipeline-cicd/cdk.out`
	`Supply a stack id (CodePipeline, Dev-MainStack) to display its template.`
		DevOps 工程师

Task	描述	所需技能

部署 CodePipeline 堆栈。	现在，您已经引导并合成了 CloudFormation 模板，可以对其进行部署。部署将创建 CodePipeline 管道和 CodeCommit 存储库，这将是管道的来源和触发器。
	`npm run cdk -- deploy CodePipeline --require-approval never`
	运行命令后，您应该会看到 CodePipeline 堆栈的成功部署和输出信息。`CodePipeline.RepositoryName` 为您提供 AWS 账户中 CodeCommit 存储库的名称。
	`CodePipeline: deploying...`
	`CodePipeline: creating CloudFormation changeset...`
	`# CodePipeline Outputs:`
	`CodePipeline.RepositoryName = SampleRepository`
	`Stack ARN: arn:aws:cloudformation:REGION:ACCOUNT-ID:stack/CodePipeline/STACK-ID`
		DevOps 工程师

Task	描述	所需技能

设置远程 CodeCommit 存储库和分支。	成功部署后， CodePipeline 将启动管道的首次运行，您可以在 AWS CodePipeline 控制台中找到该管道。由于 AWS CDK 且 CodeCommit 不启动默认分支，因此此初始管道运行将失败并返回以下错误消息。
	
	`The action failed because no branch named main was found in the selected AWS CodeCommit repository SampleRepository. Make sure you are using the correct branch name, and then try again. Error: null`
	
	要修复此错误，请将远程源设置为 `SampleRepository` ，然后创建所需 `main` 分支。
	
	`RepoName=$(aws cloudformation describe-stacks --stack-name CodePipeline --query "Stacks[0].Outputs[?OutputKey=='RepositoryName'].OutputValue" --output text)`
	`echo "${RepoName}"`
	`#`
	`git init`
	`git branch -m master main`
		DevOps 工程师

Task	描述	所需技能

	`git remote add origin codecommit://${RepoName}`
	`git add .`
	`git commit -m "Initial commit"`
	`git push -u origin main`	

测试已部署的 CodePipeline 管道

Task	描述	所需技能
提交更改，以激活管线。	成功完成初始部署后，您应该拥有完整的 CI/CD 管线，其中包含一个 `main` 分支 `SampleRepository` 作为源分支。一旦您提交对 `main` 分支的更改，管线就会启动并运行以下操作序列： 1. 从 CodeCommit 存储库中获取您的代码。 2. 构建您的代码。 3. 更新管线本身 (`UpdatePipeline`)。 4. 运行三个并行作业以进行 linting、安全性和单元测试检查。 5. 如果成功，管线会将 `Main` 堆栈从 `./lib/main-stack.ts` 部署至开发环境。 6. 对已部署资源进行部署后检查。您可以在 CodePipel	DevOps 工程师

Task	描述	所需技能
	ine 控制台中按照所有 CodePipeline 步骤和结果进行操作。 7. 成功后，管线将对测试和生产环境重复部署和验证。	

使用 Makefile 在本地测试

Task	描述	所需技能
使用 Makefile 运行开发过程。	您可使用 `make` 命令在本地运行整个管线，也可以运行单个步骤（例如 `make linting`）。 要使用 `make` 进行测试，请执行以下操作： • 实现本地管线：`make` • 仅运行单元测试：`make unittest` • 部署到当前账户：`make deploy` • 清理环境：`make clean`	应用程序开发者、 DevOps 工程师

清理资源

Task	描述	所需技能
删除 AWS CDK 应用程序资源。	要清理您的 AWS CDK 应用程序，请运行以下命令。 `cdk destroy --all`	DevOps 工程师

Task	描述	所需技能
	请注意，引导启动期间创建的 Amazon Simple Storage Service (Amazon S3) 存储桶将不会自动删除。它们需要允许删除的保留策略，或者您需要在您的 Amazon Web Services account 中手动删除它们。	

故障排除

事务	解决方案
该模板未按预期工作。	如果出现问题且模板无法正常工作，请确保您具备以下条件： • 工具的正确版本。 • 访问目标 Amazon Web Services account（网络连接）。 • 有足够的权限访问 Amazon Web Services account。

相关资源

- IAM Identity Center 中的常见任务入门
- AWS CodePipeline 文档
- AWS CDK

使用证书管理器和 “让我们加 end-to-end密” 为 Amazon EKS 上的应用程序设置加密

由 Mahendra Revanasiddappa (AWS) 和 Vasanth Jeyaraj (AWS) 创作

摘要

实施 end-to-end加密可能很复杂，您需要管理微服务架构中每项资产的证书。尽管您可以使用网络负载均衡器或 Amazon API Gateway 终止亚马逊网络服务 (AWS) 网络边缘的传输层安全 (TLS) 连接，但有些组织需要 end-to-end加密。

此模式使用 Nginx Ingress Controller 作为入口。这是因为当您创建 Kubernetes 入口时，入口资源使用网络负载均衡器。网络负载均衡器不允许上传客户端证书。因此，您无法通过 Kubernetes 入口实现双向 TLS。

这种模式旨在针对需要在其应用程序中所有微服务之间相互认证的组织。双向 TLS 减轻了用户名或密码的维护负担，还可以使用交钥匙安全框架。如果您的组织具有大量连接的设备或必须遵守严格的安全指南，则此模式的方法是兼容的。

这种模式通过对在 Amazon Elastic Kubernetes Service (Amazon EKS) 上运行的应用程序实施 end-to-end加密，有助于提高组织的安全状况。此模式在 Amazon EKS 存储库中的 GitHub End-to-end 加密中提供了示例应用程序和代码，以显示微服务如何在 Amazon EKS 上通过 end-to-end加密运行。该模式的方法使用 cert-manager（Kubernetes 的附加组件），将 Let 's Encrypt 作为证书颁发机构 (CA)。Let's Encrypt 是一款经济实惠的证书管理解决方案，它提供有效期为 90 天的免费证书。当在 Amazon EKS 上部署新的微服务时，CERT-MANAGER自动化证书的按需供应和旋转。

目标受众

建议使用 Kubernetes、TLS、Amazon Route 53 和域名系统（DNS）的用户使用此模式。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 现有 Amazon EKS 集群。
- 已在 macOS、Linux 或 Windows 上安装和配置的 AWS 命令行界面（AWS CLI）1.7 或更高版本
- `kubectl` 命令行实用程序，已安装并配置以便访问 Amazon EKS 集群。有关这方面的更多信息，请参阅 Amazon EKS 文档中的安装 kubectl。

- 用于测试应用程序的现有 DNS 名称。有关更多信息，请参阅 Amazon Route 53 文档中的使用 Amazon Route 53 注册域名。
- 最新 Helm 版本，安装在您的本地计算机上。有关这方面的更多信息，请参阅 Amazon EKS 文档和 Helm 存储库中的将 Hel GitHub m 与 Amazon EKS 配合使用。
- Amazon EKS 存储库上的 GitHub End-to-end 加密已克隆到您的本地计算机。
- 替换 Amazon EKS 存储库中克隆 GitHub End-to-end 加密的`policy.json`和`trustpolicy.json`文件中的以下值：
  - `<account number>` — 替换为您要在其中部署解决方案的账户的 Amazon Web Services account ID。
  - `<zone id>` — 替换为域名的 Route 53 区域 ID。
  - `<node_group_role>` – 替换为与 Amazon EKS 节点关联的 AWS Identity and Access Management（IAM）角色的名称。
  - `<namespace>` — 替换为在其中部署 NGINX Ingress Controller和示例应用程序的 Kubernetes 命名空间。
  - `<application-domain-name>` — 替换为 Route 53 中的 DNS 域名。

限制

- 该模式无法描述如何旋转证书，而仅演示如何在 Amazon EKS 上使用微服务的证书。

架构

下图显示了此模式的工作流和体系结构组件。

图表显示了以下工作流：

1. 客户端发送请求将应用程序访问到DNS名称。
2. Route 53 记录为网络负载均衡器的别名记录。
3. 网络负载均衡器将请求转发至配置了 TLS 侦听器的 NGINX Ingress Controller。NGINX Ingress Controller与网络负载均衡器之间的通信遵循 HTTPS 协议。
4. NGINX Ingress Controller 根据客户端对应用程序服务的请求进行基于路径的路由。
5. 应用程序服务将请求转发至应用程序容器组（pod）。该应用程序旨在通过调用秘密使用相同的证书。

6. 容器组（pod）使用证书管理器证书运行示例应用程序。NGINX Ingress Controller 和容器组（pod）之间的通信使用 HTTPS。

> **Note**
>
> 证书管理器在自己的命名空间中运行。它使用 Kubernetes 集群角色来提供特定名称空间中的秘密证书。您可以将这些名称空间附加到应用程序容器组（pod）和 Nginx Ingress Controller。

工具

Amazon Web Services

- Amazon Elastic Kubernetes Service (Amazon EKS) 是一项托管服务，可让您在 AWS 上轻松运行 Kubernetes，而无需安装、操作或维护您自己的 Kubernetes 控制面板或节点。
- 弹性负载均衡会自动将您的传入流量分配到多个目标、容器和 IP 地址。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- Amazon Route 53 是一种可用性高、可扩展性强的 DNS Web 服务。

其他工具

- cert-manager 是 Kubernetes 的附加组件，用于请求证书、将证书分发到 Kubernetes 容器和自动续订证书。
- NGINX Ingress Controller 是一款适用于 Kubernetes 和容器化环境中的云原生应用程序的流量管理解决方案。

操作说明

使用 Route 53 创建与配置公共托管区域

Task	描述	所需技能
在 Route 53 中创建一个公有托管区。	登录 AWS 管理控制台，打开 Amazon Route 53 控制台，选择托管区域，然后选择创建托管区域。创建公共托管区域，并记录区域 ID。有关更多信息，请参阅 Amazon Route 53 文档中的创建公有托管区。 **Note** ACME DNS01 使用 DNS 提供商发布质疑，要求证书管理器颁发证书。这项挑战要求您通过将特定值放在该域名下的TXT记录中来证明您控制域名的 DNS。在 Let's Encrypt 向您的 ACME 客户端提供令牌后，您的客户端会创建一条从该令牌和您的账户密钥派生的 TXT 记录，并将该记录置于 `_acme-challenge.<YOURDOMAIN>` 。然后 Let's Encrypt 在 DNS 中查询此记录。如找到匹配项，则可以继续颁发证书。	AWS DevOps

配置 IAM 角色，以允许证书管理员访问公共托管区域

Task	描述	所需技能
为证书管理器创建 IAM policy。	要求 IAM policy 为证书管理器提供许可，以验证您拥有53号公路域。A mazon EKS 存储库中的克隆 GitHub End-to-end 加密`1-IAMRole` 目录中提供了`policy.json` 示例 IAM 策略。 在 AWS CLI 中输入以下命令以创建 IAM policy。 `aws iam create-policy \ --policy-name PolicyForCertManager \ --policy-document file://policy.json`	AWS DevOps
为证书管理器创建 IAM 角色。	创建 IAM policy后，必须创建 IAM 角色。`1-IAMRole` 目录中提供了 `trustpolicy.json` 示例 IAM 角色。 在 AWS CLI 中输入以下命令以创建 IAM 角色。 `aws iam create-role \ --role-name RoleForCertManager \ --assume-role-policy-document file://trustpolicy.json`	AWS DevOps
将 策略附加到该角色。	在 AWS CLI 中输入以下命令以将 IAM policy 附加到 IAM 角	AWS DevOps

Task	描述	所需技能
	色。将 AWS_ACCOUNT_ID 替换为您的 Amazon Web Services account 的 ID。 `aws iam attach-role-policy \ --policy-arn arn:aws:iam::AWS_ACCOUNT_ID:policy/PolicyForCertManager \ --role-name RoleForCertManager`	

在Amazon EKS 中设置 NGINX Ingress Controller

Task	描述	所需技能
部署 NGINX Ingress Controller。	使用 Helm 安装最新版nginx-ingress 。在部署 nginx-ingress 配置之前，您可以根据自己的要求对其进行修改。此模式使用带注释的、面向内部的网络负载均衡器，该网络负载均衡器可在 5-Nginx-Ingress-Controller 目录中找到。 通过从 5-Nginx-Ingress-Controller 目录中运行以下 Helm 命令来安装 NGINX Ingress Controller。 `helm install test-nginx nginx-stable/nginx-ingress -f 5-Nginx-Ingress-Co`	AWS DevOps

Task	描述	所需技能
	`ntroller/values_internal_nlb.yaml`	
验证 NGINX Ingress Controller 是否已安装。	输入 `helm list`命令。输出应显示 NGINX Ingress Controller 已安装。	AWS DevOps

Task	描述	所需技能
创建 Route 53 A 记录。	创建指向 NGINX Ingress Controller 的网络负载均衡器的别名记录。 1. 获取网络负载均衡器的 DNS 名称。有关说明，请参阅获取 ELB 负载均衡器的 DNS 名称。 2. 在 Amazon Route 53 控制台，选择托管区域。 3. 选择要在其中创建记录的公共托管区域，然后选择创建记录。 4. 输入记录的名称。 5. 在记录类型中，选择 A-将流量路由到 IPv4 和一些 AWS 资源。 6. 启用别名。 7. 在流量路由目标，执行以下操作： a. 选择网络负载均衡器的别名。 b. 选择部署网络负载均衡器的 AWS 区域。 c. 输入网络负载均衡器的 DNS 名称。 8. 选择创建记录。	AWS DevOps

在 VirtualServer 亚马逊 EKS 上设置 NGINX

Task	描述	所需技能
部署 NGINX VirtualServer。	NGINX VirtualServer 资源是一种负载平衡配置，可以替代入口资源。创建 NGINX VirtualServer 资源的配置可在目录中的`nginx_virtualserver.yaml` 文件中找到。`6-Nginx-Virtual-Server` 在中输入以下命令`kubectl`以创建 NGINX 资源 VirtualServer 。 `kubectl apply -f nginx_virtualserver.yaml` **Important** 确保更新`nginx_virtualserver.yaml` 文件中的应用程序域名、证书密钥和应用程序服务名称。	AWS DevOps
验证是否已创建 NGINX VirtualServer 。	在中输入以下命令`kubectl`以验证 NGINX VirtualServer 资源是否已成功创建。 `kubectl get virtualserver`	AWS DevOps

Task	描述	所需技能
	Note 验证该Host列是否与您的应用程序的域名相匹配。	
在启用 TLS 的情况下部署 NGINX Web 服务器。	此模式使用启用了 TLS 的 NGINX Web 服务器作为测试 end-to-end加密的应用程序。部署测试应用程序所需配置文件可在 `demo-webserver` 目录中找到。 在 `kubectl` 中输入以下命令以部署测试应用程序。 `kubectl apply -f nginx-tls-ap.yaml`	AWS DevOps

Task	描述	所需技能
验证测试应用程序资源是否已创建。	在 kubectl 中输入以下命令，以验证是否为测试应用程序创建了所需资源： • `kubectl get deployments` **Note** 验证Ready列和Available 列。 • `kubectl get pods | grep -i example-deploy` **Note** Pod 应处于running状态。 • `kubectl get configmap` • `kubectl get svc`	AWS DevOps
验证应用程序。	1. 将 <application-domain-name> 替换为您之前创建的 Route53 DNS 名称，输入以下命令。 `curl --verbose https://<application-domain-name>` 2. 验证您是否可访问应用程序。	AWS DevOps

相关资源

AWS 资源

- 使用 Amazon Route 53 控制台创建记录（Amazon Route 53 文档）
- 在 Amazon EKS 上使用带有 NGINX 入口控制器的网络负载均衡器（AWS Blog 文章）

其他资源

- Route 53（证书管理器文档）
- 配置 DNS01 Challenge 提供程序（证书管理器文档）
- Let’s encrypt DNS 问题 (Let’s Encrypt 文档)

使用 Flux 简化 Amazon EKS 多租户应用程序部署

由 Nadeem Rahaman (AWS)、Aditya Ambati (AWS)、Aniket Dekate (AWS) 和 Shrikant Patil (AWS) 创作

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

许多提供产品和服务的公司都是受数据监管的行业，需要在内部业务职能之间保持数据屏障。此模式描述了如何使用 Amazon Elastic Kubernetes Service (Amazon EKS) 中的多租户功能来构建一个数据平台，在共享单个 Amazon EKS 集群的租户或用户之间实现逻辑和物理隔离。该模式通过以下方法提供隔离：

- Kubernetes 命名空间隔离
- 基于角色的访问控制 (RBAC)
- 网络策略
- 资源配额
- AWS Identity and Access Management 服务账户的 (IAM) 角色 (IRSA)

此外，此解决方案使用 Flux 在部署应用程序时保持租户配置不可变。您可以通过在配置中指定包含 Flux `kustomization.yaml` 文件的租户存储库来部署租户应用程序。

此模式实现了以下内容：

- 通过手动部署 Terraform 脚本创建的 AWS CodeCommit 存储库、 AWS CodeBuild 项目和 AWS CodePipeline 管道。
- 托管租户所需的网络和计算组件。它们是 CodeBuild 通过 CodePipeline 和使用 Terraform 创建的。
- 租户命名空间、网络策略和资源配额，它们通过 Helm 图表进行配置。
- 属于不同租户的应用程序，使用 Flux 进行部署。

我们建议您根据自己的独特要求和安全考虑，仔细规划和构建自己的多租户架构。这种模式为您的实施提供了一个起点。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- AWS Command Line Interface (AWS CLI) 版本 2.11.4 或更高版本，已安装并配置
- 本地计算机上安装了 Terraform 版本 0.12 或更高版本
- Terraform AWS Provider 版本 3.0.0 或更高版本
- Kubernetes 提供程序版本 2.10 或更高版本
- Helm Provid er 版本 2.8.0 或更高版本
- Kubectl 提供者版本 1.14 或更高版本

限制

- 对 Terraform 手动部署的依赖：工作流程的初始设置（包括创建 CodeCommit 存储库、 CodeBuild 项目和 CodePipeline 管道）依赖于手动 Terraform 部署。这在自动化和可扩展性方面带来了潜在的限制，因为它需要手动干预才能进行基础架构的更改。
- CodeCommit 存储库依赖关系：工作流程依赖于 CodeCommit 存储库作为源代码管理解决方案，并与之紧密结合 AWS 服务。

架构

目标架构

这种模式部署了三个模块来为数据平台构建管道、网络和计算基础架构，如下图所示。

管道架构：

网络架构：

计算架构：

工具

AWS 服务

- AWS CodeBuild是一项完全托管的生成服务，可帮助您编译源代码、运行单元测试和生成可随时部署的工件。
- AWS CodeCommit是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。
- AWS CodePipeline帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件更改所需的步骤。
- 亚马逊 Elastic Kubernetes Service（亚马逊 EKS）可帮助你在上面运行 AWS Kubernetes，而无需安装或维护自己的 Kubernetes 控制平面或节点。
- AWS Transit Gateway是连接虚拟私有云 (VPCs) 和本地网络的中央集线器。
- Amazon Virtual Private Cloud（亚马逊 VPC）可帮助您将 AWS 资源启动到您定义的虚拟网络中。该虚拟网络类似于您在数据中心中运行的传统网络，并具有使用 AWS的可扩展基础设施的优势。

其他工具

- Cilium 网络策略支持 Kubernetes L3 和 L4 网络策略。它们可以通过 L7 策略进行扩展，为 HTTP、Kafka 和 gRPC 以及其他类似协议提供 API 级别的安全性。
- Flux 是一款基于 Git 的持续交付 (CD) 工具，可自动在 Kubernetes 上部署应用程序。
- Helm 是一款适用于 Kubernetes 的开源软件包管理器，可帮助你在 Kubernetes 集群上安装和管理应用程序。
- Terraform 是一款基础设施即代码 (IaC) 工具 HashiCorp ，可帮助您创建和管理云和本地资源。

代码存储库

此模式的代码可在 GitHub EKS 多租户 Terraform 解决方案存储库中找到。

最佳实践

有关使用此实现的指南和最佳实践，请参阅以下内容：

- Amazon EKS 多租户最佳实践
- Flux 文档

操作说明

为 Terraform 构建、测试和部署阶段创建管道

Task	描述	所需技能
克隆项目存储库。	在终端 GitHub 窗口中运行以下命令，克隆 EKS 多租户 Terraform 解决方案存储库： `git clone https://github.com/aws-samples/aws-eks-multitenancy-deployment.git`	AWS DevOps

引导 Terraform S3 存储桶和亚马逊 DynamoDB。	1. 在该`bootstrap` 文件夹中，打开`bootstrap.sh` 文件并更新 S3 存储桶名称、DynamoDB 表名称的变量值以及： AWS 区域
	`S3_BUCKET_NAME="<S3_BUCKET_NAME>"`
	`DYNAMODB_TABLE_NAME="<DYNAMODB_NAME>"`
	`REGION="<AWS_REGION>"`
	2. 运行 `bootstrap.sh` 脚本。该脚本需要 AWS CLI，您已将其作为先决条件的一部分进行安装。
	`cd bootstrap`
	`./bootstrap.sh`
		AWS DevOps

更新`run.sh`和`locals.tf` 文件。	1. 成功完成引导过程后，从`variables` 脚本部分复	AWS DevOps

Task	描述	所需技能

	制 S3 存储桶和 DynamoDB 表名称：bootstrap.sh
	
	``
	# Variables
	S3_BUCKET_NAME="
	S3_BUCKET_NAME>"
	DYNAMODB_TABLE_NAME
	="<DYNAMODB_NAME"
	``
	
	2. 将这些值粘贴到run.sh脚本中，该脚本位于项目的根目录中：
	
	``
	BACKEND_BUCKET_ID=
	"<SAME_NAME_AS_S3_
	BUCKET_NAME>"
	DYNAMODB_ID="
	<SAME_NAME_AS_DYNA
	MODB_NAME>"
	``
	
	3. 将项目代码上传到存储 CodeCommit 库。您可以通过 Terraform 自动创建此存储库，方法是在文件true中将以下变量设置为：demo/pipeline/locals.tf
	
	``
	create_new_repo =
	 true
	``
	
	4. 根据您的要求更新locals.tf 文件以创建管道资源。	

Task	描述	所需技能

部署管道模块。	要创建管道资源，请手动运行以下 Terraform 命令。没有自动运行这些命令的编排。
	
	`./run.sh -m pipeline -e demo -r <AWS_REGION> -t init`
	`./run.sh -m pipeline -e demo -r <AWS_REGION> -t plan`
	`./run.sh -m pipeline -e demo -r <AWS_REGION> -t apply`
		AWS DevOps

创建网络基础架构

Task	描述	所需技能
启动管道。	1. 在该`templates`文件夹中，确保`buildspec`文件将以下变量设置为`network`： `TF_MODULE_TO_BUILD: "network"` 2. 在CodePipeline 控制台的管道详细信息页面上，通过选择发布更改来启动管道。 第一次运行后，每当你向 CodeCommit 存储库主分支提交更改时，管道就会自动启动。 该管道包括以下阶段：	AWS DevOps

Task	描述	所需技能
	• `validate`初始化 Terraform，使用 checkov 和 tfsec 工具运行 Terraf orm 安全扫描，并将扫描报告上传到 S3 存储桶。 • `plan` 显示了 Terraform 计划并将该计划上传到 S3 存储桶。 • `apply`应用来自 S3 存储桶的 Terraform 计划输出并创建 AWS 资源。 • `destroy`移除在`apply`阶段中创建的 AWS 资源。要启用此可选阶段，请在`demo/pipeline/locals.tf` 文件`true`中将以下变量设置为： `enable_destroy_stage = true`	

Task	描述	所需技能
验证通过网络模块创建的资源。	确认以下 AWS 资源是在成功部署管道后创建的： • 具有三个公有子网和三个私有子网、互联网网关和 NAT 网关的出口 VPC。 • 具有三个私有子网的 Amazon EKS VPC。 • 租户 1 和租户 2 各 VPCs 有三个私有子网。 • 带有所有 VPC 连接和路由到每个私有子网的中转网关。 • 目标 CIDR 块为 Amazon EKS 出口 VPC 的静态传输网关路由。`0.0.0.0/0` 这是让所有 VPCs 人都能通过 Amazon EKS 出口 VPC 进行出站互联网访问所必需的。	AWS DevOps

创建计算基础架构

Task	描述	所需技能
更新`locals.tf` 以启用 CodeBuild 项目对 VPC 的访问权限。	要为 Amazon EKS 私有集群部署插件，必须将该 CodeBuild 项目连接到 Amazon EKS VPC。 1. 在`demo/pipeline` 文件夹中，打开`locals.tf` 文件，并将`vpc_enabled` 变量设置为`true`。	AWS DevOps

Task	描述	所需技能

	2. 运行run.sh脚本将更改应用于流水线模块：
	
	`demo/pipeline/locals.tf`
	`./run.sh -m pipeline -env demo -region <AWS_REGION> -tfcmd init`
	`./run.sh -m pipeline -env demo -region <AWS_REGION> -tfcmd plan`
	`./run.sh -m pipeline -env demo -region <AWS_REGION> -tfcmd apply`	

更新buildspec 文件以生成计算模块。	在该templates 文件夹中，在所有 buildspec YAML 文件中，将TF_MODULE_TO_BUILD 变量的值设置为：networkcompute `TF_MODULE_TO_BUILD: "compute"`	AWS DevOps

Task	描述	所需技能

更新租户管理 Helm 图表的values文件。	1. 在以下位置打开values.yaml 文件：
	`cd cfg-terraform/demo/compute/cfg-tenant-mgmt`
	该文件如下所示：
	`---`
	`global:`
	`  clusterRoles:`
	`  operator: platform-tenant`
	`  flux: flux-tenant-applier`
	`  flux:`
	`  tenantCloneBaseUrl: ${TEANT_BASE_URL}`
	`  repoSecret: ${TENANT_REPO_SECRET}`
	`tenants:`
	`  tenant-1:`
	`  quotas:`
	`  limits:`
	`  cpu: 1`
	`  memory: 1Gi`
	`  flux:`
	`  path: overlays/tenant-1`
	`  tenant-2:`
	`  quotas:`
	`  limits:`
	`  cpu: 1`
	`  memory: 2Gi`
	`  flux:`
		AWS DevOps

Task	描述	所需技能

	<pre>path: overlays/
	tenant-2</pre>
	2. 在global和tenants部分中，根据您的要求更新配置：
	• tenantCloneBaseUrl — 托管所有租户代码的仓库路径（我们对所有租户使用相同的 Git 存储库）
	• repoSecret — Kubernetes 密钥，其中包含用于向全局租户 Git 存储库进行身份验证的 SSH 密钥和已知主机
	• quotas— 你想为每个租户申请的 Kubernetes 资源配额
	• flux path— 全局租户存储库中租户应用程序 YAML 文件的路径	

Task	描述	所需技能
验证计算资源。	在前面的步骤中更新文件后，将自动 CodePipeline 启动。确认它已为计算基础设施创建了以下 AWS 资源： • 带有私有终端节点的 Amazon EKS 群 • 亚马逊 EKS 工作节点 • Amazon EKS 附加组件：外部机密`aws-loadbalancer-controller`、和 `metrics-server` • GitOps 模块、Flux Helm 图表、Cilium Helm 图表和租户管理 Helm 图表	AWS DevOps

查看租户管理和其他资源

Task	描述	所需技能
在 Kubernetes 中验证租户管理资源。	运行以下命令以检查租户管理资源是否在 Helm 的帮助下成功创建。 1. 租户命名空间已创建，如中所述：`values.yaml` `kubectl get ns -A` 2. 配额分配给每个租户命名空间，具体如以下所述`values.yaml` ：	AWS DevOps

Task	描述	所需技能

	`kubectl get quota --namespace=<tenant_namespace>`
	3. 每个租户命名空间的配额详细信息都是正确的：
	`kubectl describe quota cpu-memory-resource-quota-limit -n <tenant_namespace>`
	4. Cilium 网络策略已应用于每个租户命名空间：
	`kubectl get CiliumNetworkPolicy -A`	

Task	描述	所需技能

验证租户应用程序部署。	运行以下命令以验证租户应用程序是否已部署。
	1. Flux 能够连接到 GitOps 模块中指定的 CodeCommit 存储库：
	`kubectl get gitrepositories -A`
	2. Flux 自定义控制器已在存储库中部署了 YAML 文件：CodeCommit
	`kubectl get kustomizations -A`
	3. 所有应用程序资源都部署在其租户命名空间中：
	`kubectl get all -n <tenant_namespace>`
	4. 已为每个租户创建了一个入口：
	`kubectl get ingress -n <tenant_namespace>`	

故障排除

事务	解决方案
您会遇到一条类似于以下内容的错误消息：	请按照以下步骤解决问题：

事务	解决方案

`Failed to checkout and determine revision: unable to clone unknown error: You have successfully authenticated over SSH. You can use Git to interact with AWS CodeCommit.`	1. 验证租户应用程序存储库：错误可能是存储库为空或配置错误所致。确保租户应用程序存储库包含所需的代码。
	2. 重新部署`tenant_mgmt` 模块：在`tenant_mgmt` 模块配置文件中，找到模`app`块，然后将`deploy`参数设置为：`0`
	`deploy = 0`
	运行 Terraform `apply` 命令后，将`deploy`参数值改回：`1`
	`deploy = 1`
	3. 重新检查状态：运行前面的步骤后，使用以下命令检查问题是否仍然存在：
	`kubectl get gitrepositories -A`
	如果问题仍然存在，可以考虑更深入地研究 Flux 日志以获取更多详细信息，或者参阅 Flux 一般故障排除指南。

相关资源

- 适用于 Terraform 的 Amazon EKS 蓝图
- Amazon EKS 最佳实践指南，多租户部分
- Flux 网站
- Helm 网站

其他信息

以下是用于部署租户应用程序的存储库结构示例：

```
applications
sample_tenant_app
### README.md
### base
#   ### configmap.yaml
#   ### deployment.yaml
#   ### ingress.yaml
#   ### kustomization.yaml
#   ### service.yaml
### overlays
    ### tenant-1
    #   ### configmap.yaml
    #   ### deployment.yaml
    #   ### kustomization.yaml
    ### tenant-2
        ### configmap.yaml
        ### kustomization.yaml
```

使用自定义资源将多个电子邮件端点订阅 SNS 主题

创建者：Ricardo Morais (AWS)

摘要

注意，2022 年 8 月：AWS CloudFormation 现在支持通过AWS::SNS::Topic对象及其订阅属性订阅多个资源。

此模式描述如何订阅多个电子邮件地址以接收来自 Amazon Simple Notification Service (Amazon SNS)主题的通知。它使用 AWS Lambda 函数作为 AWS CloudFormation 模板中的自定义资源。Lambda 函数与一个输入参数相关联，该参数指定 SNS 主题的电子邮件端点。

目前，您可以使用 AWS CloudFormation 模板对象AWS::SNS::Topic和AWS::SNS::Subscription为单个终端节点订阅 SNS 主题。要订阅多个端点，您必须多次调用该对象。通过将 Lambda 函数用作自定义资源，您可以通过输入参数订阅多个端点。您可以将此 Lambda 函数用作任何 AWS CloudFormation 模板中的自定义资源。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 在本地环境中使用访问密钥和私有密钥配置的 AWS 配置文件。
- 以下内容的权限：
  - AWS Identity and Access Management (IAM) 角色和策略
  - AWS Lambda 函数
  - 用于上传 Lambda 函数 的 Amazon Simple Storage Service (Amazon S3)
  - Amazon SNS 主题和策略
  - AWS CloudFormation 堆栈

限制

- 此代码支持 Linux 和 macOS 工作站。

产品版本

- AWS 命令行界面（AWS CLI）版本 2 或更高版本。

架构

目标技术堆栈

- AWS CloudFormation
- Amazon SNS
- AWS Lambda

工具

工具

- AWS CLI 版本 2

代码

附件包括以下文件：

- Lambda 函数：`lambda_function.py`
- AWS CloudFormation 模板：`template.yaml`
- 用于处理多个或单个电子邮件端点订阅的两个参数文件：`parameters-multiple-values.json`（用作默认值）和 `parameters-one-value.json`

要部署堆栈，您可使用任一参数文件。要指定多个电子邮件端点，请执行以下操作：

```
./deploy.sh -p <YOUR_AWS_PROFILE_NAME> -r <YOUR_AWS_PROFILE_REGION>
```

要指定单个电子邮件端点，请执行以下操作：

```
./deploy.sh -p <YOUR_AWS_PROFILE_NAME> -r <YOUR_AWS_PROFILE_REGION> -f parameters-one-value.json
```

操作说明

选项 1 - 通过电子邮件订阅部署 SNS 主题

Task	描述	所需技能
为 SNS 主题订阅配置电子邮件端点。	编辑文件 `parameters-one-value.json`（附件），然后更改 `pSNSNotificationsEmail` 参数的值以反映您要使用的电子邮件地址，例如 `someone@example.com`。	
部署用于创建资源和订阅的 AWS CloudFormation 堆栈。	使用您的 AWS 个人资料名称、AWS 区域 和 `parameters-one-value.json` 文件运行 deploy.sh 命令。 `./deploy.sh -p <YOUR_AWS_PROFILE_NAME> -r <YOUR_AWS_PROFILE_REGION> -f parameters-one-value.json`	具有适当权限的 IAM 角色

选项 2 - 部署包含两个或更多电子邮件订阅的 SNS 主题

Task	描述	所需技能
为 SNS 主题订阅配置电子邮件端点。	编辑文件 `parameters-multiple-values.json`（附件），然后更改 `pSNSNotificationsEmail` 参数的值以反映您要使用的电子邮件地址（用逗号分隔），如下所示：`someone1@`	

Task	描述	所需技能
	example.com, someone2@example.com 。	
部署用于创建资源和订阅的 AWS CloudFormation 堆栈。	使用您的 AWS 配置文件名称和 AWS 区域运行 deploy.sh 命令。您不必指定 parameters-multiple-values.json 文件，因为默认情况下会使用该文件。 `./deploy.sh -p <YOUR_AWS_PROFILE_NAME> -r <YOUR_AWS_PROFILE_REGION>`	具有适当权限的 IAM 角色

选项 3-通过 AWS 模板部署 SNS CloudFormation 主题

Task	描述	所需技能
创建 SNS 主题。	通过 AWS 模板创建 SNS 主题，无需在 CloudFormation AWS::SNS::Topic 模板对象中指定订阅终端节点。您可使用附件中的 template.yaml 作为起点。	具有适当权限的 IAM 角色
创建 SNS 主题策略。	在 AWS CloudFormation 模板中创建 SNS 主题策略。	具有适当权限的 IAM 角色
在电子邮件端点列表中订阅 SNS 主题。	根据电子邮件端点（一个或多个）列表，为端点订阅您创建的 SNS 主题。	具有适当权限的 IAM 角色

相关资源

参考

- AWS CloudFormation 自定义资源（AWS 文档）
- 使用 Python、AWS Lambda 和 crhelper 创建 AWS CloudFormation 自定义资源（博客文章）

必要工具

- AWS CLI 版本 2

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS Fargate WaitCondition 挂钩构造协调资源依赖关系和任务执行

由 Stan Fan (AWS) 创作

摘要

此模式描述了 WaitCondition hook (`waitcondition-hook-for-aws-fargate-task`) npm 包，它是一种云原生解决方案，专为在亚马逊弹性容器服务 (Amazon ECS) 集群AWS Fargate中编排任务而设计。

WaitCondition 钩子是专门为与集成而量身定制的 AWS Cloud Development Kit (AWS CDK) 构造 AWS CloudFormation。该 WaitCondition 挂钩提供以下关键功能：

- 充当等待条件机制，在指定的 Fargate 任务完成之前暂停 CloudFormation 堆栈执行，这有助于有序部署和资源配置。
- 支持 TypeScript 和 Python，使其成为 AWS CDK 项目的理想之选。
- 允许开发人员和架构师通过协调容器化应用程序的任务完成和资源管理来协调部署。 AWS
- 允许在 CloudFormation 生命周期中嵌入一个或多个容器的情况下运行 Fargate 任务。并且可以处理任务失败并在任务失败后回滚 CloudFormation 堆栈。
- 允许灵活地在资源和 Fargate 任务执行结果之间添加依赖关系，从而启用自定义任务或调用其他端点。例如，您可以暂停 CloudFormation 堆栈并等待数据库迁移（由 Fargate 任务完成），然后配置其他可能取决于数据库迁移成功与否的资源。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- AWS Cloud Development Kit (AWS CDK) 命令行界面 (CLI) 安装在本地工作站上。有关更多信息，请参阅 AWS CDK 文档中的 AWS CDK CLI 参考。
- 节点包管理器 (npm)，安装在本地工作站上并配置为AWS CDK 中。 TypeScript有关更多信息，请参阅 npm 文档中的下载和安装 Node.js 和 npm。
- Yarn 安装在本地工作站上。有关更多信息，请参阅 Yarn 文档中的安装。

限制

- 此解决方案已部署到单个 AWS 账户。
- 容器的预期返回码是成功0的。任何其他返回代码都表示失败， CloudFormation 堆栈将回滚。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

架构

下图显示了构造架构。

该图显示了以下工作流程waitcondition-hook-for-aws-fargate-task：

1. WaitCondition并WaitConditionHandler被设置为监听来自 AWS Lambda 函数的响应。
2. 根据任务的结果，要么由 Fargate 任务的完成触发。CallbackFunction ErrorHandlerFunction
3. Lambda 函数向发送成功或失败信号。WaitConditionHandler
4. WaitConditionHandler如果 Fargate 任务的执行结果成功，则继续配置资源；如果任务失败，则回滚堆栈。

下图显示了执行数据库迁移的工作流程示例。

示例工作流使用该waitcondition-hook-for-aws-fargate-task结构执行数据库迁移，如下所示：

1. 已预配置亚马逊关系数据库服务 (Amazon RDS) 实例。
2. 该waitcondition-hook-for-aws-fargate-task构造运行数据库迁移任务，并将堆栈作为亚马逊弹性计算云 (Amazon EC2) 实例暂停。
3. 如果迁移任务成功完成，则会向发送成功信号 CloudFormation。否则，它会向堆栈发送失败信号，CloudFormation 然后回滚堆栈。

工具

Amazon Web Services

- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，可帮助您在代码中定义云基础架构并通过它进行配置 AWS CloudFormation。
- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。
- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和运行的应用程序 AWS 的指标。
- Amazon Elastic Container Service (Amazon ECS)是一项快速且可扩展的容器管理服务，可帮助运行、停止和管理集群上的容器。
- AWS Fargate无需管理服务器或 Amazon EC2 实例，即可帮助您运行容器。它与 Amazon ECS 配合使用。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Step Functions是一项无服务器编排服务，可帮助您组合 AWS Lambda 功能和其他功能 AWS 服务 来构建关键业务应用程序。
- Amazon Virtual Private Cloud（亚马逊 VPC）可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心中运行的传统网络，其优点是使用的可扩展基础架构。 AWS

其他工具

- npm 是在 Node.js 环境中运行的软件注册表，用于共享或借用软件包以及管理私有软件包的部署。
- Yarn 是一个开源软件包管理器，你可以用它来管理 JavaScript 项目中的依赖关系。Yarn 可以帮助您安装、更新、配置和删除软件包依赖关系。

代码存储库

此模式的代码可在 GitHub waitcondition-hook-for-aws-fargate- task 存储库中找到。

最佳实践

- 在构建 AWS CDK 应用程序时，请按照 AWS CDK v2 文档 AWS CDK中的开发和部署云基础架构的最佳实践进行操作。
- 对于此 AWS Fargate 任务，请按照 Amazon ECS 文档中的 Amazon ECS 容器映像最佳实践进行操作。

操作说明

设置 AWS CDK

Task	描述	所需技能
安装 AWS CDK。	要 AWS CDK 在本地计算机或其他环境中安装，请运行以下命令： `npm install -g aws-cdk@latest`	云架构师、应用程序开发人员
Bootstrap 的。 AWS CDK	引导是为部署准备环境的过程。要为目标引导 AWS CDK 工具包 AWS 账户 AWS 区域，请运行以下命令： `cdk bootstrap aws://ACCOUNT-NUMBER-1/REGION-1` 此命令创建一个名为 的 CloudFormation 堆栈`CDKToolkit` 。	云架构师

运行 AWS Fargate 任务构造 WaitCondition 挂钩

Task	描述	所需技能
创建 CDK 项目。	使用您喜欢的语言创建 CDK 项目。此模式使用 TypeScript. 要使用创建 CDK 项目 TypeScript，请运行以下命令： `cdk init app –language typescript`	云架构师

Task	描述	所需技能
安装 软件包。	`npm install`在 CDK 项目的根路径上执行。安装 CDK 库后，运行以下命令进行安装`waitcondition-hook-for-aws-fargate-task` ： `yarn add waitcondition-hook-for-aws-fargate-task`	云架构师

Task	描述	所需技能

构建您的 CDK 应用程序和 Amazon ECS 组件。	构建你的 CDK 项目。需要一个 Amazon ECS 任务定义资源。有关创建任务定义的信息，请参阅 Amazon ECS 文档中的 Amazon ECS 任务定义。
	
	以下示例使用此构造：
	
	`import * as cdk from 'aws-cdk-lib';`
	`import { Vpc } from 'aws-cdk-lib/aws-ec2';`
	`import * as ecr from 'aws-cdk-lib/aws-ecr';`
	`import * as ecs from 'aws-cdk-lib/aws-ecs';`
	`import { Construct } from 'constructs';`
	`import { FargateRunner } from 'waitcondition-hook-for-aws-fargate-task';`
	`import { Queue } from 'aws-cdk-lib/aws-sqs';`
	
	`export class FargateRunnerStack extends cdk.Stack {`
	`  constructor(scope: Construct, id: string, props?: cdk.StackProps) {`
	`  super(scope, id, props);`
	`  // Define the VPC`
		云架构师

Task	描述	所需技能

	<pre>const vpc = new Vpc(this, 'MyVpc')
	// Define the Fargate Task
	const taskDefinition = new ecs.FargateTaskDefinition(this, 'MyTask', {});
	// Import exiting ecr repo
	const repo = ecr.Repository.fromRepositoryName(this, 'MyRepo', 'RepoName');
	// Add a container to the task
	taskDefinition.addContainer('MyContainer', {
	image: ecs.ContainerImage.fromEcrRepository(repo),
	});
	// Create the Fargate runner
	const myFargateRunner = new FargateRunner(this, 'MyRunner', {
	fargateTaskDef: taskDefinition,
	timeout: `${60 * 5}`,
	vpc: vpc,
	});
	// Create the SQS queue
	const myQueue = new Queue(this, 'MyQueue', {});</pre>	

Task	描述	所需技能
	`// Add dependenc y myQueue.n ode.addDependency( myFargateRunner); } }`	

合成并启动 CDK 应用程序。	1. 要生成资源和 CloudForm ation 模板，请在 CDK 根路径中运行以下命令：
	`cdk synth`
	2. `synth`命令成功后，运行以下命令部署资源：
	`cdk deploy`
	
	该`waitcondition-hook -for-aws-fargate-t ask` 构造运行 Fargate 任务。
		云架构师

清理

Task	描述	所需技能
清理资源。	要清理上一步中配置的资源，请运行以下命令： `cdk destroy`	云架构师

故障排除

事务	解决方案
一般 CloudFormation 堆栈故障	要帮助排除一般 CloudFormation 堆栈故障，请添加`--no-rollback` 标志，如以下示例所示： `cdk deploy --no-rollback` 此命令将暂停 CloudFormation 堆栈回滚，从而为您提供故障排除资源。有关更多信息，请参阅 AWS CloudFormation 文档中的选择配置资源时如何处理故障。
AWS Step Functions 失败	AWS Step Functions 状态机可能由于不同的原因而无法执行。`–disable-rollback` 配置完成后，使用以下步骤进行故障排除： 1. 登录 AWS Management Console，在 “搜索” 字段中输入 Step Fun ctions，然后选择 Step Functions 服务。 2. 在左侧导航窗格中，选择 St at e machines，然后选择由堆 CloudFormation 栈置备的状态机。 3. 在执行中，选择意外失败的执行的名称。 4. 在事件视图中，选择失败的步骤。 有关更多信息，请参阅 AWS Step Functions 文档中的 Step F unctions 中的问题疑难解答和在 Step Functions 控制台中查看执行详情。
AWS Lambda 功能失败	此构造提供了两个 Lambda 函数：`CallbackF unction` 和。`ErrorhandlerFunction` 它们可能由于各种原因而失败，例如未处理的异常。使用以下步骤进行故障排除：

事务	解决方案
	1. 登录 AWS Management Console，在 “搜索” 字段CloudWatch中输入，然后选择 CloudWatch 服务。 2. 在左侧导航窗格中，选择日志组。 3. 在搜索字段中，输入 Lambda 函数的名称。 4. 选择与 Lambda 函数关联的日志组名称。 5. 要查看 Lambda 函数的执行结果，请选择最新的日志流。 有关更多信息，请参阅文档中的 Lambda 问题疑难解答。 AWS Lambda

相关资源

AWS 文档

- AWS CDK 构建 API 参考
- 开始使用 AWS CDK
- 学习如何创建和使用 Amazon ECS 资源
- 学习如何开始使用 Step Functions
- 什么是 AWS CDK？

其他资源

- AWS Fargate 任务的等待条件挂钩 (npm)
- waitcondition-hook-for-aws-fargate-task 1.0.6 (pypi.org)

在 AWS 中使用第三方 Git 源存储库 CodePipeline

由 Kirankumar Chandrashekar (AWS) 创建

摘要

注意： AWS CodeCommit 已不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

此模式描述了如何将 AWS CodePipeline 与第三方 Git 源存储库配合使用。

AWS CodePipeline 是一项持续交付服务，可自动执行构建、测试和部署软件的任务。该服务目前支持由 AWS CodeCommit 和 Atlassian Bit GitHub bucket 管理的 Git 存储库。但是，部分企业使用与其单点登录（SSO）服务和 Microsoft Active Directory 集成的第三方 Git 存储库进行身份验证。您可以 CodePipeline 通过创建自定义操作和 webhook 将这些第三方 Git 存储库用作来源。

Webhook 是一种 HTTP 通知，用于检测其他工具（例如 GitHub 存储库）中的事件，并将这些外部事件连接到管道。当您在中创建 webhook 时 CodePipeline，该服务会返回一个网址，您可以在 Git 存储库 webhook 中使用该网址。如果您将代码推送到 Git 存储库的特定分支，Git webhook 会通过此 URL 启动 CodePipeline webhook，并将管道的源代码阶段设置为 “进行中”。当管道处于此状态时，作业工作人员会轮 CodePipeline 询自定义作业，运行该作业，并将成功或失败状态发送到 CodePipeline。在本示例中，由于管道处于源代码阶段，因此作业工作程序使用所轮询的作业提供的对象密钥获取 Git 存储库的内容，压缩内容，然后将其上传到存储管道构件的 Amazon Simple Storage Service (Amazon S3) 存储桶。您还可以将自定义操作的过渡与 Amazon 中的事件关联起来 CloudWatch，并根据该事件启动任务工作人员。通过此设置，您可以使用该服务本身不支持的第三方 Git 存储库作为来源。CodePipeline

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 一个支持 webhook 的 Git 存储库，可以通过互联网连接到 CodePipeline webhook 网址
- AWS 命令行界面（AWS CLI）已安装并配置以使用 Amazon Web Services account

架构

此模式涉及这些步骤：

1. 用户将代码提交到至 Git 存储库。
2. 调用 Git 网络钩子。
3. 这个 CodePipeline webhook 被称为。
4. 管道设置为进行中，源阶段设置为进行中状态。
5. 源阶段操作会启动 “ CloudWatch 事件” 规则，表示该规则已启动。
6. 该 CloudWatch 事件启动一个 Lambda 函数。
7. Lambda 函数获取自定义操作作业详细信息。
8. Lambda 函数启动 AW CodeBuild S 并将所有与工作相关的信息传递给它。
9. CodeBuild 从 Secrets Manager 获取访问 HTTPS Git 的 SSH 公钥或用户证书。
10. CodeBuild 克隆特定分支的 Git 存储库。
11. CodeBuild 压缩存档并将其上传到用作 CodePipeline 项目存储的 S3 存储桶。

工具

- AWS CodePipeline — AWS CodePipeline 是一项完全托管的持续交付服务，可帮助您自动执行发布渠道，实现快速可靠的应用程序和基础设施更新。 CodePipeline 根据您定义的发布模型，针对每次代码变更自动执行发布过程的构建、测试和部署阶段。这让您可以快速而可靠地交付各种功能和更新。您可以将 AWS CodePipeline 与第三方服务（例如 GitHub 您自己的自定义插件）集成。
- AWS Lambda – AWS Lambda 让您无需预调配或管理服务器即可运行代码。借助 Lambda，您可以为几乎任何类型的应用程序或后端服务运行代码，而无需进行任何管理。您只需上传您的代码，Lambda 就会处理以高可用性运行和扩展您的代码所需一切。您可以将您的代码设置为自动从其他 Amazon Web Services 启动，或者直接从任何 Web 或移动应用程序调用。
- AWS CodeBuild — AWS CodeBuild 是一项完全托管的持续集成服务，可编译源代码、运行测试并生成可随时部署的软件包。使用 CodeBuild，您无需预置、管理和扩展自己的构建服务器。CodeBuild 持续扩展并同时处理多个构建，因此您的构建不会在队列中等待。您可以使用预先打包的构建环境快速开始，也可以创建使用您自己的构建工具的自定义构建环境。
- AWS Secrets Manager – AWS Secrets Manager 可帮助您保护访问您的应用程序、服务和 IT 资源所需密钥。该项服务让您可以在数据库凭证、API 密钥和其他密钥的整个生命周期内对其进行轮换、管理和检索。用户和应用程序通过调用 Secrets Manager 来检索机密 APIs，而不必以纯文本格式对敏感信息进行硬编码。Secrets Manager 通过与Amazon Relational Database Service (Amazon RDS)、Amazon Redshift 和Amazon DocumentDB 的内置集成，提供密钥轮换。可以扩展该服务以

支持其他类型的机密，包括 API 密钥和 OAuth 令牌。此外，Secrets Manager 使您能够使用精细权限控制对密钥的访问，并集中审计 AWS Cloud、第三方服务和本地环境资源的密钥轮换。

- Ama zon CloudWatch — Amazon CloudWatch 是一项专为 DevOps 工程师、开发人员、站点可靠性工程师 (SREs) 和 IT 经理构建的监控和观察服务。 CloudWatch 为您提供数据和切实可行的见解，以监控您的应用程序、响应系统范围的性能变化、优化资源利用率并获得统一的运营状况视图。CloudWatch 以日志、指标和事件的形式收集监控和运营数据，为您提供在 AWS 和本地服务器上运行的 AWS 资源、应用程序和服务的统一视图。您可以使用 CloudWatch 来检测环境中的异常行为、设置警报、并排可视化日志和指标、采取自动操作、解决问题以及发现见解，以保持应用程序平稳运行。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项对象存储服务，可让您存储和保护任意数量的数据，用于网站、移动应用程序、备份和还原、归档、企业应用程序、IoT 设备和大数据分析。Amazon S3 提供 easy-to-use管理功能，可帮助您整理数据并配置经过微调的访问控制，以满足您的特定业务、组织和合规要求。

操作说明

在中创建自定义操作 CodePipeline

Task	描述	所需技能
使用 AWS CLI 或 AWS 创建自定义操作 CloudFormation。	此步骤涉及创建自定义源操作，该操作可在特定区域 Amazon Web Services account 中用于管道的源代码阶段。您必须使用 AWS CLI 或 AWS CloudFormation （不是控制台）来创建自定义源操作。有关本文和其他操作说明中描述的命令和步骤的更多信息，请参阅此模式末尾的 “相关资源”部分。在 AWS CLI 中，使用 create-custom-action-type命令。使用--configuration-properties提供作业工作人员在轮询 CodePipeline 作业时需要处理的所有参数。请务必记下	常规 AWS

Task	描述	所需技能
	提供给 --provider 和 --action-version 选项的值，这样您就可以在使用此自定义源代码阶段创建管道时使用相同的值。您也可以使用资源类型在 AWS CloudFormation 中创建自定义源操作 AWS::CodePipeline::CustomActionType。	

设置身份验证

Task	描述	所需技能
创建 SSH 密钥对。	创建 Secure Shell (SSH) 密钥对。有关说明，请参阅 GitHub 文档。	系统/工程师 DevOps
在 AWS Secrets Manager 中创建密钥。	从 SSH 密钥对中复制私钥的内容，然后在 AWS Secrets Manager 创建密钥。此密钥用于在访问 Git 存储库时进行身份验证。	常规 AWS
将公有密钥添加至 Git 存储库。	将 SSH 密钥对中的公钥添加到 Git 存储库账户设置中，以便根据私钥进行身份验证。	系统/工程师 DevOps

创建管道和网络钩子

Task	描述	所需技能
创建包含自定义源操作的管道。	在中创建管道 CodePipeline。配置源阶段时，选择之前创建的自定义源操作。您可以	常规 AWS

Task	描述	所需技能
	在 AWS CodePipeline 控制台或 AWS CLI 中执行此操作。CodePipeline 提示您输入您在自定义操作上设置的配置属性。该信息是作业工作程序处理自定义操作作业的必要条件。按向导进行操作，为管道创建下一个阶段。	
创建一个 CodePipeline webhook。	为您使用自定义源操作构建的管道创建网络钩子。您必须使用 AWS CLI 或 AWS CloudFormation（不是控制台）来创建 Webhook。在 AWS CLI 中，运行 put-webhook 命令并为网络钩子选项提供相应值。记下命令返回的网络钩子 URL。如果您使用 AWS CloudFormation 创建 Webhook，请使用资源类型 AWS::CodePipeline::Webhook。请务必从创建的资源中输出网络钩子网址，并记录。	常规 AWS

Task	描述	所需技能
创建 Lambda 函数和项目。CodeBuild	在此步骤中，您将使用 Lambda 和 CodeBuild 创建一个任务工作线程，该工作人员将轮询 CodePipeline 自定义操作的任务请求，运行该作业，并将状态结果返回到。 CodePipeline创建一个 Lambda 函数，当管道的自定义源操作阶段转换为 “进行中” 时，该函数由 Amazon CloudWatch Events 规则启动。当 Lambda 函数启动时，它应通过轮询作业来获取自定义操作作业的详细信息。您可以使用 PollForJobs API 返回此信息。获得轮询的作业信息后，Lambda 函数应返回确认，然后使用从自定义操作的配置属性中所获得的数据处理该信息。当工作人员准备好与 Git 仓库对话时，你可以启动一个 CodeBuild 项目，因为使用 SSH 客户端可以方便地处理 Git 任务。	常规 AWS，代码开发人员

在中创建活动 CloudWatch

Task	描述	所需技能
创建 CloudWatch 事件规则。	创建一个 CloudWatch 事件规则，每当管道的自定义操作阶段转换为 “进行中” 时，该规则就会将 Lambda 函数作为目标启动。	常规 AWS

相关资源

在中创建自定义操作 CodePipeline

- 在中创建和添加自定义操作 CodePipeline
- AWS::CodePipeline::CustomActionType 资源

设置身份验证

- 通过 AWS Secrets Manager 创建和管理密钥

创建管道和网络钩子

- 在中创建管道 CodePipeline
- put-webhook 命令参考
- AWS::CodePipeline::Webhook 资源
- PollForJobs API 参考
- 在中创建和添加自定义操作 CodePipeline
- 在 AWS 中创建构建项目 CodeBuild

创建事件

- 通过 Amazon CloudWatch Events 检测管道状态的变化并做出反应

其他参考资料

- 使用中的管道 CodePipeline
- AWS Lambda 开发人员指南

使用 AWS 创建 CI/CD 管道以验证 Terraform 的配置 CodePipeline

由 Aromal Raj Jayarajan (AWS) 和 Vijesh Vijayakumaran Nair (AWS) 创作

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

此模式展示了如何使用 AWS 部署的持续集成和持续交付 (CI/CD) 管道来测试 HashiCorp Terraform 配置。 CodePipeline

Terraform 是一款命令行界面应用程序，可帮助您使用代码来配置和管理云基础设施和资源。此模式中提供的解决方案创建了一个 CI/CD 管道，通过运行五个阶段来帮助您验证 Terraform 配置的完整性： CodePipeline

1. “checkout”从 AWS 存储库中提取你正在测试的 Terraform 配置。 CodeCommit
2. “validate”运行基础设施即代码 (IaC) 验证工具，包括 tfsec TFLint、和 chec kov。该阶段还会运行以下 Terraform IaC 验证命令：`terraform validate` 和 `terraform fmt`。
3. “plan” 显示如果应用 Terraform 配置，将对基础架构应用哪些更改。
4. “apply” 使用生成的计划在测试环境中配置所需基础架构。
5. “destroy” 移除在 “apply” 阶段中创建的测试基础架构。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- AWS 命令行界面（ AWS CLI ），已安装并配置
- Git，已在本地计算机上安装并配置
- Terraform，已在本地计算机上安装并配置

限制

- 这种模式的方法 CodePipeline 将 AWS 部署到一个 AWS 账户中，并且仅限于 AWS 区域。多账户和多区域部署需更改配置。

- 此模式配置的 AWS Identity and Access Management (IAM) 角色 (codepiline_iam_role) 遵循最低权限的原则。必须根据您的管道需要创建的特定资源，更新此 IAM 角色的权限。

产品版本

- AWS CLI 版本 2.9.15 或更高版本
- Terraform 版本 1.3.7 或更高版本

架构

目标技术堆栈

- AWS CodePipeline
- AWS CodeBuild
- AWS CodeCommit
- Amazon IAM
- Amazon Simple Storage Service(Amazon S3)
- AWS Key Management Service（AWS KMS）
- Terraform

目标架构

下图显示了用于在中测试 Terraform 配置的 CI/CD 管道工作流程示例。 CodePipeline

图表显示了以下工作流：

1. 在中 CodePipeline，AWS 用户通过在 AWS CLI 中运行`terraform apply`命令来启动 Terraform 计划中建议的操作。
2. AW CodePipeline S 担任 IAM 服务角色，其中包括访问 CodeCommit CodeBuild、AWS KMS 和 Amazon S3 所需的策略。
3. CodePipeline 运行“`checkout`”管道阶段，从 AWS CodeCommit 存储库中提取 Terraform 配置进行测试。
4. CodePipeline 通过运行 IaC 验证工具并在项目中运行 Terraform iaC 验证命令来运行测试 Terraform 配置的“`validate`”阶段。 CodeBuild

5. CodePipeline 运行该“plan”阶段以基于 Terraform 配置在 CodeBuild 项目中创建计划。在将更改应用至测试环境前，AWS 用户可查看此计划。
6. Code Pi “apply” peline 通过使用 CodeBuild 项目在测试环境中配置所需的基础架构，来实施计划。
7. CodePipeline 运行“destroy”阶段，该阶段用于 CodeBuild 移除在“apply”阶段中创建的测试基础架构。
8. Amazon S3 存储桶存储管道项目，这些项目使用 AWS KMS 客户托管密钥进行加密和解密。

工具

工具

Amazon Web Services

- AWS CodePipeline 可帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件变更所需的步骤。
- AWS CodeBuild 是一项完全托管的构建服务，可帮助您编译源代码、运行单元测试和生成可随时部署的项目。
- AWS CodeCommit 是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥，以帮助保护您的数据。
- Amazon Simple Storage Service(Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

其他服务

- HashiCorp Terraform 是一款命令行界面应用程序，可帮助您使用代码来配置和管理云基础架构和资源。

代码

此模式的代码可在 GitHub aws-codepipeline-terraform-cicdsamples存储库中找到。存储库包含创建此模式所述目标架构所需 Terraform 配置。

操作说明

配置解决方案组件

Task	描述	所需技能
克隆 GitHub 存储库。	在终端窗口中运行以下命令来克隆 GitHubaws-codepipeline-terraform-cicdsamples存储库： `git clone https://github.com/aws-samples/aws-codepipeline-terraform-cicd-samples.git` 有关更多信息，请参阅 GitHub 文档中的克隆存储库。	DevOps 工程师
创建 Terraform 变量定义文件。	基于您的使用案例要求创建 `terraform.tfvars` 文件。您可以更新克隆存储库中 `examples/terraform.tfvars` 文件中的变量。 有关更多信息，请参阅 Terraform 文档中的为根模块变量赋值。 **Note** 存储库`Readme.md` 的文件包含有关所需变量的更多信息。	DevOps 工程师
将 AWS 配置为 Terraform 提供程序。	1. 在代码编辑器中，打开克隆存储库的 `main.tf` 文件。	DevOps 工程师

Task	描述	所需技能
	2. 添加必要的配置，以建立与目标 Amazon Web Services account 的连接。 有关更多信息，请参阅 Terraform 文档中的 AWS 提供者。	
更新用于构建 Amazon S3 复制存储桶的 Terraform 提供程序配置。	1. 通过运行以下命令打开存储库的 S3 目录： `cd ./modules/s3` 2. 通过更新 `tf` 文件中的 `region` 值，更新用于创建 Amazon S3 复制存储桶的 Terraform 提供程序配置。确保您输入您希望 Amazon S3 将对象复制到的目标区域。 3. （可选）默认情况下，Terraform 使用本地状态文件进行状态管理。若要将 Amazon S3 添加为远程后端，则必须更新 Terraform 配置。有关更多信息，请参阅 Terraform 文档中的后端配置。 **Note** 复制可以激活在 Amazon S3 存储桶之间自动异步复制对象。	DevOps 工程师

Task	描述	所需技能
初始化 Terraform 配置。	如要初始化包含 Terraform 配置文件的工作目录，在克隆存储库的根文件夹运行以下命令。 `terraform init`	DevOps 工程师
创建 Terraform 计划。	若要创建 Terraform 计划，请在克隆存储库的根文件夹中运行以下命令： `terraform plan --var-file=terraform.tfvars -out=tfplan` **Note** Terraform 会评估配置文件，以确定已声明资源的目标状态。然后，其将比较目标状态与当前状态，并创建计划。	DevOps 工程师
验证 Terraform 计划。	查看 Terraform 计划，并确认它已在您的目标 Amazon Web Services account 中配置了所需架构。	DevOps 工程师

Task	描述	所需技能
部署解决方案。	1. 若要应用 Terraform 计划，请在克隆存储库的根文件夹中运行以下命令： `terraform apply "tfplan"` 2. 输入是，以确认您要部署资源。 **Note** Terraform 创建、更新或销毁基础设施，以实现配置文件中声明的目标状态。	DevOps 工程师

通过运行管道验证 Terraform 配置

Task	描述	所需技能
设置源代码存储库。	1. 从 Terraform 输出中，获取包含要验证的 Terraform 配置的存储库的源存储库详细信息。 2. 登录 AWS 管理控制台。然后，打开CodeCommit 控制台。 3. 在名为 `main` 的源存储库中创建一个新的分支。有关说明，请参阅 CodeCommit 文档中的在 AWS CodeCommit 中创建分支。	DevOps 工程师

Task	描述	所需技能
	4. 将源存储库 的 `main` 分支克隆至您的本地工作站。有关说明，请参阅 CodeCommit 文档中的使用 AWS CLI 证书帮助程序在 Windows 上与 AWS CodeCommit 存储库进行 HTTPS 连接的设置步骤。 5. 通过运行以下命令从 GitHubaws-codepipeline-terraform-cicdsamples存储库中复制**`templates`** 文件夹： `cp -r templates $YOUR_CODECOMMIT_REPO_ROOT` **Note** 该`templates` 文件夹包含编译规范文件和源存储库根目录的验证脚本。 6. 将所需 Terraform IaC 配置添加至源存储库的根文件夹。 7. 在项目的 Terraform 配置中添加远程后端详细信息。有关更多信息，请参阅 Terraform 文档中的 S3。 8. （可选）更新 `templates` 文件夹中的变量，以激活或停用预配置的扫描、工具更	

Task	描述	所需技能
	改版本，并在自定义脚本文件中指定您的目录。有关更多信息，请参阅此模式的其他信息部分。 9. 将更改推送至源存储库的 `main` 分支。	
验证管道阶段。	1. 登录 AWS 管理控制台并打开CodePipeline 控制台。 2. 在上一节 Epic 部分的`terraform apply "tfplan"`命令生成的输出中，找到生成的名称CodePipeline。 3. 在 CodePipeline 控制台中打开管道，然后选择 “发布更改”。 4. 查看每个管道阶段，并确认其按预期运行。 有关更多信息，请参阅 AWS CodePipeline 用户指南中的查看管道详情和历史记录（控制台）。 **Important** 将更改提交到源存储库的主分支时，测试管道会自动激活。	DevOps 工程师

Task	描述	所需技能
验证报告输出。	1. 在CodePipeline 控制台的左侧导航窗格中，选择 Build。然后，选择报告历史记录。 2. 查看管道生成的 tfsec 与 checkov 扫描报告。这些报告可帮助您通过可视化和图形形式来识别问题。 **Note** 在此“validate” 阶段，该<project_name>-validate CodeBuild 项目会为您的代码生成漏洞报告。	DevOps 工程师

清除资源

Task	描述	所需技能
清理管道和关联资源。	要从您的 Amazon Web Services account 中删除测试资源，请在克隆存储库的根文件夹运行以下命令： `terraform destroy --var-file=terraform.tfvars`	DevOps 工程师

故障排除

事务	解决方案
在“apply”舞台期间您会收到AccessDenied 错误。	1. 查看与该“apply”阶段关联的 CodeBuild 项目的执行日志，找出任何缺失的 IAM 权限。有关更多信息，请参阅 AWS CodeBuild 用户指南中的在 AWS CodeBuild 中查看构建详情。 2. 在代码编辑器中，打开克隆存储库 `modules` 文件夹。然后，导航到 `iam-role` 文件夹并打开该文件夹中的 `main.tf` 文件。 3. 在 `codepipeline_policy` 语句中，添加在您的 Amazon Web Services account 中配置资源所需 IAM policy。

相关资源

- 模块数据库（Terraform 文档）
- 如何使用 CI/CD 通过 Terraform 部署和配置 AWS 安全服务（AWS 博客文章）
- 使用服务相关角色（IAM 文档）
- 创建管道（AWS CLI 文档）
- 为存储在 Amazon S3 中的项目配置服务器端加密 CodePipeline（AWS CodePipeline 文档）
- AWS 配额 CodeBuild（AWS CodeBuild 文档）
- AWS 中的数据保护 CodePipeline（AWS CodePipeline 文档）

其他信息

自定义 Terraform 模块

以下是在此模式中使用的自定义 Terraform 模块列表：

- `codebuild_terraform`创建构成管道每个阶段的 CodeBuild 项目。
- `codecommit_infrastructure_source_repo`捕获并创建源 CodeCommit 存储库。

- codepipeline_iam_role 为管道创建所需 IAM 角色。
- codepipeline_kms 为 Amazon S3 对象加密和解密创建所需 AWS KMS 密钥。
- codepipeline_terraform为源 CodeCommit 存储库创建测试管道。
- s3_artifacts_bucket 创建一个 Amazon S3 存储桶以管理管道项目。

生成规范文件

以下是此模式用于运行每个管道阶段的构建规范 (buildspec) 文件列表：

- buildspec_validate.yml 运行 “validate” 阶段。
- buildspec_plan.yml 运行 “plan” 阶段。
- buildspec_apply.yml 运行 “apply” 阶段。
- buildspec_destroy.yml 运行 “destroy”阶段。

生成规格文件变量

每个 buildspec 文件都使用以下变量激活不同的特定构建设置：

变量	默认值	描述
CODE_SRC_DIR	"."	定义源 CodeCommit 目录
TF_VERSION	"1.3.7"	为构建环境定义 Terraform 版本

每个 buildspec_validate.yml 文件都使用以下变量激活不同的特定构建设置：

变量	默认值	描述
SCRIPT_DIR	"./templates/scripts"	定义脚本目录
ENVIRONMENT	"dev"	定义环境名称
SKIPVALIDATIONFAILURE	"Y"	失败时跳过验证

ENABLE_TFVALIDATE	"Y"	激活 Terraform 验证
ENABLE_TFFORMAT	"Y"	激活 Terraform 格式
ENABLE_TFCHECKOV	"Y"	激活 checkovov 扫描
ENABLE_TFSEC	"Y"	激活 tfsec 扫描
TFSEC_VERSION	"v1.28.1"	定义 tfsec 版本

更多模式

- 使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon EKS 上私密访问容器应用程序
- 将一个 AWS CodeCommit 存储库 AWS 账户 与另一个账户中的 Amazon A SageMaker I Studio Classic 关联起来
- 使用着陆区域加速器自动创建账户 AWS
- 通过 AWS Systems Manager 自动添加或更新 Windows 注册表项
- 使用 AWS Batch 自动备份 Amazon RDS for PostgreSQL 数据库实例
- 使用 AWS SAM 自动部署嵌套应用程序
- 使用 CI/CD 管道在 Amazon EKS 中自动部署 Node Termination Handler
- 在 Amazon MQ 中自动化 RabbitMQ 配置
- 在跨区域自动复制 Amazon RDS 实例 AWS 账户
- 使用 CI/CD 管道自动构建 Java 应用程序并将其部署到 Amazon EKS
- 使用 Python 应用程序为亚马逊 DynamoDB 自动生成 PynamoDB 模型和 CRUD 函数
- 使用 CodePipeline IAM Access Analyzer 和 AWS CloudFormation 宏在 AWS 账户中自动验证和部署 IAM 策略和角色
- 在 Amazon Web Services Cloud 上的 Stromasys Charon-SSP 仿真器中备份 Sun SPARC 服务器
- 使用 AWS DataOps 开发套件构建数据管道，以提取、转换和分析 Google Analytics（分析）数据
- 使用 Amazon A EC2 uto Scaling 和 Systems Manager 构建 Micro Focus 企业服务器 PAC
- 使用 Image Builder 和 Terraform 为经过强化的容器 EC2 镜像构建管道
- 使用亚马逊 A SageMaker I 和 Azure 构建 MLOps 工作流程 DevOps
- 使用本地微软 Active Direct AWS Managed Microsoft AD ory 集中解析 DNS
- 使用无服务器方法将 Amazon Web Services 串在一起
- 使用在 Amazon Logs 中为.NET 应用程序配置 CloudWatch 日志记录 NLog
- 从 AWS 存储库持续部署现代 AWS Amplify 网络应用程序 CodeCommit
- 为 AWS Step Functions SageMaker 创建自定义 Docker 容器镜像并将其用于模型训练
- 在不支持 AWS 的 AWS 区域创建管道 CodePipeline
- 使用 Amazon CloudWatch 异常检测为自定义指标创建警报
- 使用 AWS CDK 外观和逃生舱口自定义默认角色名称
- 部署可同时检测多个代码交付项中的安全问题的管道
- 使用基础设施即代码，在 Amazon Web Services Cloud 上部署和管理无服务器数据湖

- 使用 Amazon EKS 和 Amazon S3 中的 Helm 图表存储库部署 Kubernetes 资源和软件包
- 使用 AWS CDK 部署多堆栈应用程序 TypeScript
- 使用 Terraform 为 AWS WAF 解决方案部署安全自动化
- 使用 RAG 和提示开发基于 AI 聊天的高级生成式 AI 助手 ReAct
- 使用 AWS CloudFormation 模板 GuardDuty 有条件地启用 Amazon
- 使用 Amazon EKS Pod Identity 和 KEDA 在 Amazon EKS 中设置事件驱动的自动缩放
- 使用 Amazon Personalize 生成个性化和重新排名的推荐
- 当 AWS KMS 密钥的密钥状态发生变化时获取 Amazon SNS 通知
- 迁移到 Amazon ECR 存储库时自动识别重复的容器映像
- 在 Amazon API Gateway 中使用自定义域名实现基于路径的 API 版本控制
- 通过在多个 AWS 区域、账户中启用 Amazon DevOps Guru 并 OUs 使用 AWS CDK 来提高运营绩效
- 使用 Kubernetes 在亚马逊 EKS 工作节点上安装 SSM 代理 DaemonSet
- 将 Stonebranch Universal Controller 与 AWS Mainframe Modernization 集成
- 大型机现代化： AWS 使用 Rock DevOps et 软件企业套件开启
- 使用以下 AWS IAM Identity Center 命令将权限集作为代码进行管理 AWS CodePipeline
- 通过使用 AWS CDK 设置 Amazon ECS Anywhere 来管理本地容器应用程序
- 将 DNS 记录批量迁移至 Amazon Route 53 私有托管区
- SageMaker 使用 AWS 开发人员工具将 ML 构建、训练和部署工作负载迁移到 Amazon
- 监控多个共享的 Amazon 系统映像的使用情况 AWS 账户
- 优化 AWS App2Container 生成的 Docker 映像
- 使用 AWS Step Functions 编排 ETL 管道，包含验证、转换和分区
- 使用 IaC 原则自动部署 Amazon Aurora 全球数据库
- 在非工作负载子网的多账户 VPC 设计中保留可路由的 IP 空间
- 使用代码存储库在中 AWS Service Catalog 配置 Terraform 产品
- 跨账户或区域复制已筛选的 Amazon ECR 容器映像
- 在不重启容器的情况下轮换数据库凭证
- 从 AWS Step Functions 同步运行 AWS Systems Manager Automation 任务
- 使用 AWS CDK 在 Amazon ECS Anywhere 上为混合工作负载设置 CI/CD 管道和 GitLab
- 使用 Terraform 为数据库迁移设置 CI/CD 管道
- 使用 Amazon 为 SQL Server Always On FCI 设置多可用区基础架构 FSx

- 使用 AWS 在亚马逊 EC2 上自动设置 UiPath RPA 机器人 CloudFormation
- 使用应用程序负载均衡器在 Amazon ECS 中使用双向 TLS 简化应用程序身份验证
- 使用 C# 和 AWS CDK 在 SaaS 架构中为孤岛模型进行租户登录
- 使用 Terraform 自动 GuardDuty 为组织启用亚马逊
- 使用 Amazon Bedrock 代理通过基于文本的提示在 Amazon EKS 中自动创建访问入口控制
- 在本地验证 Account Factory for Terraform (AFT) 代码
- 使用 Flask 和 AWS Elastic Beanstalk 查看人工智能/机器学习 (AI/ML) 模型结果

基础设施

主题

- 使用会话管理器和 Amazon Instance Connect 访问堡垒主 EC2 机
- 使用本地微软 Active Direct AWS Managed Microsoft AD ory 集中解析 DNS
- 使用 Amazon CloudWatch 可观测性访问管理器进行集中监控
- 在启动时检查 EC2 实例是否有必需的标签
- 使用会话管理器连接到 Amazon EC2 实例
- 在不支持 AWS 的 AWS 区域创建管道 CodePipeline
- 使用 AWS CDK 外观和逃生舱口自定义默认角色名称
- 使用私有静态在 Amazon EC2 上部署 Cassandra 集群 IPs 以避免重新平衡
- 使用 AWS Transit Gateway Connect 扩展 VRFs 到 AWS
- 当 AWS KMS 密钥的密钥状态发生变化时获取 Amazon SNS 通知
- 在非工作负载子网的多账户 VPC 设计中保留可路由的 IP 空间
- 使用代码存储库在中 AWS Service Catalog 配置 Terraform 产品
- 使用 Amazon SES 通过单个电子邮件地址注册多个 Amazon Web Services account
- 在单账户 AWS 环境中为混合网络设置 DNS 解析
- 使用 AWS 在亚马逊 EC2 上自动设置 UiPath RPA 机器人 CloudFormation
- 在 AWS 上设置高度可用的 PeopleSoft 架构
- 使用 AWS 弹性灾难恢复为 Oracle JD Edwar EnterpriseOne ds 设置灾难恢复
- 在多区域、多账户组织中设置 AWS CloudFormation 偏差检测
- 成功导入 S3 存储桶作为 AWS CloudFormation 堆栈
- 使用 AWS 在不同 AWS 区域的 Amazon EFS 文件系统之间同步数据 DataSync
- 使用 LocalStack 和 Terraform 测试来测试 AWS 基础架构
- 将 SAP Pacemaker 集群从升级到 ENSA1 ENSA2
- 在不同的 AWS 账户中 VPCs 使用一致的可用区
- IDs 在 IAM 策略中使用用户进行访问控制和自动化
- 在本地验证 Account Factory for Terraform (AFT) 代码
- 更多模式

使用会话管理器和 Amazon Instance Connect 访问堡垒主 EC2 机

由 Piotr Chotkowski (AWS) 和 Witold Kowalik (AWS) 创建

摘要

堡垒主机（有时也称为跳转盒）是一种服务器，提供从外部网络到位于专用网络中的资源的单点访问。暴露在外部公共网络（如互联网）中的服务器会给未经授权的访问带来潜在的安全风险。保护和控制对这些服务器的访问非常重要。

此模式描述了如何使用会话管理器和 Amazon EC2 Instance Connect 安全地连接到部署在您的 AWS 账户中的亚马逊弹性计算云 (Amazon EC2) 堡垒主机。会话管理器是一项功能 AWS Systems Manager。此模式的优点包括：

- 部署的堡垒主机没有任何向公共互联网公开的开放入站端口。这就减少了潜在的攻击面。
- 您无需在中存储和维护长期安全外壳 (SSH) 密钥 AWS 账户。相反，每位用户每次连接到堡垒主机时都会生成一个新的 SSH key pair。 AWS Identity and Access Management 附加到用户 AWS 证书的 (IAM) 策略控制对堡垒主机的访问权限。

目标受众

这种模式适用于对亚马逊 EC2、亚马逊虚拟私有云（亚马逊 VPC）和 Hashicorp Terraform 有基本了解的读者。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- AWS Command Line Interface (AWS CLI) 版本 2，已安装并配置
- 的会话管理器插件 AWS CLI，已安装
- Terraform CLI 已安装
- Terraform 状态的存储，例如充当远程后端来存储 Terraform 状态的 Amazon Simple Storage Service (Amazon S3) 桶和 Amazon DynamoDB 表。有关使用远程后端获得 Terraform 状态的更多信息，请参阅 Amazon S3 后端（Terraform 文档）。有关使用 Amazon S3 后端设置远程状态管理的代码示例，请参阅 remote-state-s3 后端（Terraform Registry）。请注意以下要求：
  - Amazon S3 存储桶和 DynamoDB 表必须相同。 AWS 区域

- 创建 DynamoDB 表时，分区键必须为 `LockID`（区分大小写），并且分区键类型必须为 `String`。将所有其他设置保留为默认值。有关更多信息，请参阅 DynamoDB 文档中的关于主键和创建表。
- SSH 客户端，已安装。

限制

- 该模式旨在作为概念证明(PoC)，或作为进一步开发的基础。不得将其当前形式用于生产环境。在部署之前，请调整存储库中的示例代码以满足您的要求和用例。
- 此模式假定目标堡垒主机使用 Amazon Linux 2 作为其操作系统。虽然可以使用其他 Amazon 系统映像 (AMIs)，但其他操作系统超出了这种模式的范围。

> **Note**
>
> 亚马逊 Linux 2 的支持已接近终止。欲了解更多信息，请参阅亚马逊 Linux 2 FAQs。

- 在此模式中，堡垒主机位于没有 NAT 网关和互联网网关的私有子网中。这种设计将 Amazon EC2 实例与公共互联网隔离开来。您可以添加特定的网络配置，使其能够与互联网进行通信。有关更多信息，请参阅 Amazon VPC 文档中的将虚拟私有云（VPC）连接到其他网络。同样，遵循最低权限原则，AWS 账户 除非您明确授予权限，否则堡垒主机无权访问您的任何其他资源。有关更多信息，请参阅 IAM 文档中的基于资源的策略。

产品版本

- AWS CLI 版本 2
- Terraform 版本 1.3.9

架构

目标技术堆栈

- 具有单个私有子网的 VPC
- 以下接口 VPC 端点：
  - `amazonaws.<region>.ssm`— AWS Systems Manager 服务的终端节点。
  - `amazonaws.<region>.ec2messages` - Systems Manager 使用此端点从 SSM 代理调用到 Systems Manager 服务。

  - `amazonaws.<region>.ssmmessages`— 会话管理器使用此终端节点通过安全数据通道连接到您的 Amazon EC2 实例。
- 运行`t3.nano`亚马逊 Linux 的亚马逊 EC2 实例 2
- IAM 角色与实例配置文件
- 终端节点和亚马逊 EC2 实例的 Amazon VPC 安全组和安全组规则

目标架构

该图显示了以下过程：

1. 用户假定的 IAM 角色拥有执行以下操作的权限：
   - 对 Amazon EC2 实例进行身份验证、授权和连接
   - 使用会话管理器启动会话
2. 用户通过会话管理器启动 SSH 会话。
3. 会话管理器对用户进行身份验证，验证关联的 IAM policy 中的权限，检查配置设置，并向 SSM 代理发送消息以打开双向连接。
4. 用户通过 Amazon EC2 元数据将 SSH 公钥推送到堡垒主机。这必须在每次连接之前完成。SSH 公钥在 60 秒内保持可用。
5. 堡垒主机与 Systems Manager 和 Amazon 的接口 VPC 终端节点通信。 EC2
6. 用户使用 TLS 1.2 加密的双向通信通道通过会话管理器访问堡垒主机。

自动化和扩缩

以下选项可用于自动部署或扩缩此架构：

- 您可以通过连续集成和连续交付 (CI/CD) 管道部署体系结构。
- 您可以修改代码以更改堡垒主机的实例类型。
- 您可以修改代码以部署多个堡垒主机。在 `bastion-host/main.tf` 文件的 `aws_instance` 资源块中，添加 `count` 元参数。有关更多信息，请参阅 Terraform 文档。

工具

AWS 服务

- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行 shell 中的命令进行交互。
- 亚马逊弹性计算云 (Amazon EC2) 在中提供可扩展的计算容量 AWS Cloud。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Systems Manager 可帮助您管理在 AWS Cloud中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。此模式使用 Session Manager，这是 Systems Manager 的一项功能。
- Amazon Virtual Private Cloud（亚马逊 VPC）可帮助您将 AWS 资源启动到您定义的虚拟网络中。该虚拟网络类似于您在数据中心中运行的传统网络，并具有使用 AWS的可扩展基础设施的优势。

其他工具

- HashiCorp Terraform 是一款基础设施即代码 (IaC) 工具，可帮助您使用代码来配置和管理云基础架构和资源。此模式使用 Terraform CLI。

代码存储库

此模式的代码可在使用会话管理器 GitHub 访问堡垒主机和 Amazon EC2 Instance Connect 存储库中找到。

最佳实践

- 我们建议使用自动代码扫描工具来提高代码的安全性和质量。使用 IaC 的静态代码分析工具 Checkov 对该模式进行了扫描。我们至少建议您使用 `terraform validate` 和 `terraform fmt -check -recursive` Terraform 命令执行基本验证和格式检查。
- 为 IaC 添加自动化测试是一种很好的做法。有关测试 Terraform 代码的不同方法的更多信息，请参阅测试 Terraform（ HashiCorp Terraform 博客文章）。
- 在部署过程中，每次检测到新版本的亚马逊 Linux 2 AMI 时，Terraform 都会使用替换的亚马逊 EC2 实例。这将部署新版本的操作系统，包括补丁和升级。如果未及时安排部署计划，可能会由于实例没有最新补丁而带来安全风险。经常更新已部署的 Amazon EC2 实例并对其应用安全补丁非常重要。有关更多信息，请参阅 Amazon 中的更新管理 EC2。
- 由于此模式是一种概念验证，因此它使用 AWS 托管策略，例如`AmazonSSMManagedInstanceCore`。 AWS 托管策略涵盖常见用例，但不授予最低权限权限。

根据您的使用案例需要，我们建议您创建自定义策略，以便为此架构中部署的资源授予最低权限许可。有关更多信息，请参阅开始使用 AWS 托管策略并转向最低权限权限。

- 使用密码来保护对 SSH 密钥的访问，并将密钥存储在安全位置。
- 为堡垒主机设置日志记录和监控。从操作和安全角度来看，日志和监控都是维护系统的重要组成部分。有多种方法可以监控堡垒主机中的连接和活动。有关更多信息，请参阅 Systems Manager 文档中的以下主题：
  - 监控 AWS Systems Manager
  - 登录和监控 AWS Systems Manager
  - 审核会话活动
  - 记录会话活动

操作说明

部署资源

Task	描述	所需技能
克隆代码存储库。	1. 在命令行界面中，将工作目录更改为要存储示例文件的位置。 2. 输入以下命令。 `git clone https://github.com/aws-samples/secured-bastion-host-terraform.git`	DevOps 工程师、开发人员
初始化 Terraform 工作目录。	只有首次部署时才需要这一步骤。如果您要重新部署模式，请跳至下一步。 在克隆存储库的根目录中，输入以下命令，其中：	DevOps 工程师、开发人员、Terraform

Task	描述	所需技能

	• $S3_STATE_BUCKET 是包含 Terraform 状态的 Amazon S3 存储桶的名称
	• $PATH_TO_STATE_FILE 是 Terraform 状态文件的密钥，例如 infra/bastion-host/tetfstate
	• $AWS_REGION 是部署 Amazon S3 存储桶的区域
	
	`terraform init \`
	`  -backend-config="bucket=$S3_STATE_BUCKET" \`
	`  -backend-config="key=$PATH_TO_STATE_FILE" \`
	`  -backend-config="region=$AWS_REGION`
	
	Note
	或者，您可以打开 config.tf 文件，然后在该terraform 部分中手动提供这些值。	

Task	描述	所需技能
部署资源	1. 在克隆存储库的根目录中，输入以下命令。 `terraform apply -var-file="dev.tfvars"` 2. 查看将应用于您的所有更改的列表 AWS 账户，然后确认部署。 3. 等待所有资源部署完毕。	DevOps 工程师、开发人员、Terraform

设置本地环境

Task	描述	所需技能
配置 SSH 连接。	更新 SSH 配置文件，允许通过会话管理器进行 SSH 连接。有关说明，请参阅会话管理器允许的 SSH 连接。这允许授权用户输入启动会话管理器会话并通过双向连接传输所有数据的代理命令。	DevOps 工程师
生成 SSH 密钥。	输入以下命令生成本地私密和公共 SSH 密钥对。您可使用此密钥对连接到堡垒主机。 `ssh-keygen -t rsa -f my_key`	DevOps 工程师、开发人员

使用会话管理器连接到堡垒主机

Task	描述	所需技能

获取实例 ID。	1. 要连接到已部署的堡垒主机，您需要提供 Amazon EC2 实例的 ID。执行以下任一操作以定位 ID：
	• 打开 Amazon EC2 控制台。在导航窗格中，选择 Instances (实例)。找到堡垒主机实例。
	• 在中 AWS CLI，输入以下命令。
	`aws ec2 describe-instances`
	若要筛选结果，请输入以下命令，其中 `$BASTION_HOST_TAG` 是您分配给堡垒主机的标签。该选项的默认值为 `sandbox-dev-bastion-host` 。
	`aws ec2 describe-instances \ --filters "Name=tag:Name,Values=$BASTION_HOST_TAG" \ --output text \ --query 'Reservations[*].Instances[*].InstanceId' \ --output text`
		常规 AWS

Task	描述	所需技能
	2. 复制 Amazon EC2 实例的 ID。您稍后会使用 ID。	

Task	描述	所需技能
发送 SSH 公有密钥。	**Note** 在本节中，您将公钥上传到堡垒主机的实例元数据。上传密钥后，您有 60 秒的时间启动与堡垒主机的连接。60 秒后，公有密钥将被删除。有关更多信息，请参阅本模式的疑难解答部分。快速完成后续步骤，以防止在连接到堡垒主机之前密钥被删除。 1. 使用 Amazon EC2 Instance Connect 将 SSH 密钥发送到堡垒主机。输入以下命令，其中： • `$INSTANCE_ID` 是 Amazon EC2 实例的 ID • `$PUBLIC_KEY_FILE` 是公有密钥文件的路径，例如 `my_key.pub` **Important** 请务必使用公钥而不是私钥。 `aws ec2-instance-connect send-ssh-public-key \`	常规 AWS

Task	描述	所需技能

	``
	    --instance-id
	 $INSTANCE_ID \
	    --instance-os-
	user ec2-user \
	    --ssh-public-key
	 file://$PUBLIC_KEY
	_FILE
	``
	2. 等到您收到一条消息，表明密钥已成功上传。立即继续执行下一步。	

Task	描述	所需技能
连接到堡垒主机。	1. 输入以下命令通过会话管理器连接到堡垒主机，其中： • `$PRIVATE_KEY_FILE` 是你的私钥的路径，比如 `my_key` • `$INSTANCE_ID` 是 Amazon EC2 实例的 ID `ssh -i $PRIVATE_KEY_FILE ec2-user@$INSTANCE_ID` 2. 输入 `yes` 确认连接。这将使用会话管理器打开 SSH 连接。 **Note** 还有其他选项可用于打开与堡垒主机的 SSH 连接。有关更多信息，请参阅此模式的其他信息部分中与堡垒主机建立 SSH 连接的替代方法。	常规 AWS

(可选)清除

Task	描述	所需技能
删除已部署的资源。	1. 要删除所有已部署的资源，请从克隆存储库的根目录运行以下命令。	DevOps 工程师、开发人员、Terraform

Task	描述	所需技能
	```terraform destroy -var-file="dev.tfvars"``` 2. 确认删除资源。	

故障排除

事务	解决方案
尝试连接到堡垒主机时出现 `TargetNotConnected` 错误	1. 按照 Amazon EC2 文档中重启您的实例中的说明重启堡垒主机。 2. 实例成功重新启动后，将公有密钥重新发送到堡垒主机并重新尝试连接。
尝试连接到堡垒主机时出现 `Permission denied` 错误	将公有密钥上传到堡垒主机后，您只有 60 秒的时间来启动连接。60 秒后，密钥将自动删除，您将无法使用它连接到实例。如果发生这种情况，您可以重复该步骤以将密钥重新发送到实例。

相关资源

AWS 文档

- AWS Systems Manager Session Manager（Systems Manager 文档）
- 安装用于 AWS CLI（Systems Manager 文档）的会话管理器插件
- 允许会话管理器的 SSH 连接（Systems Manager 文档）
- 关于使用 Instan EC2 ce Connect（亚马逊 EC2 文档）
- 使用 Instanc EC2 e Connect 进行连接（亚马逊 EC2 文档）
- 亚马逊@@ 的身份和访问管理 EC2（亚马逊 EC2 文档）
- 使用 IAM 角色向在 Amazon EC2 实例上运行的应用程序授予权限（IAM 文档）

- IAM 中的安全最佳实践（IAM 文档）
- 使用安全组控制流量（Amazon VPC 文档）

其他资源

- Terraform 开发人员网页
- 命令：验证（Terraform 文档）
- 命令：fmt（Terraform 文档）
- 测试 HashiCorp Terraform（HashiCorp 博客文章）
- Checkov 网页

其他信息

与堡垒主机建立 SSH 连接的替代方法

端口转发

您可以使用 `-D 8888` 选项打开具有动态端口转发的 SSH 连接。欲了解更多信息，请参阅 explainshell.com 上的说明。以下是使用端口转发打开 SSH 连接的命令示例。

```
ssh -i $PRIVATE_KEY_FILE -D 8888 ec2-user@$INSTANCE_ID
```

此类连接会打开 SOCKS 代理，该代理可通过堡垒主机转发来自本地浏览器的流量。如果您使用的是 Linux 或 MacOS，要查看所有选项，请输入 `man ssh`。这将显示 SSH 参考手册。

使用提供的脚本

您可以使用代码存储库中包含的 connect.sh 脚本，无需手动运行操作说明部分中使用会话管理器连接到堡垒主机中所描述的步骤。此脚本生成 SSH 密钥对，将公钥推送到 Amazon EC2 实例，并启动与堡垒主机的连接。运行脚本时，将标签和键名作为参数传递。以下是运行脚本的命令示例。

```
./connect.sh sandbox-dev-bastion-host my_key
```

使用本地微软 Active Direct AWS Managed Microsoft AD ory 集中解析 DNS

由 Brian Westmoreland (AWS) 编写

摘要

此模式为同时使用 AWS Directory Service for Microsoft Active Directory (AWS Managed Microsoft AD) 和 Amazon Route 53 在 AWS 多账户环境中集中 DNS 解析提供了指导。在这种模式中，AWS DNS 命名空间是本地 DNS 命名空间的子域。此模式还提供了有关如何将本地 DNS 服务器配置为在本地 DNS 解决方案使用 Microsoft Active Directory AWS 时将查询转发到的指导。

先决条件和限制

先决条件

- 使用 AWS Organizations设置的 AWS 多账户环境。
- 之间已建立网络连接 AWS 账户。
- 在 AWS 和本地环境之间建立网络连接（通过使用 AWS Direct Connect 或任何类型的 VPN 连接）。
- AWS Command Line Interface (AWS CLI) 在本地工作站上配置。
- AWS Resource Access Manager (AWS RAM) 用于在账户之间共享 Route 53 规则。因此，必须在 AWS Organizations 环境中启用共享，如长篇故事部分所述。

限制

- AWS Managed Microsoft AD 标准版的股票上限为 5 股。
- AWS Managed Microsoft AD 企业版的股票上限为 125 股。
- 这种模式下的解决方案仅限于通过 AWS 区域 该支持共享 AWS RAM。

产品版本

- 微软 Active Directory 在 Windows Server 2008、2012、2012 R2 或 2016 上运行。

架构

目标架构

在此设计中 AWS Managed Microsoft AD ，安装在共享服务中 AWS 账户。尽管这不是必需的，但此模式假设此配置。如果您以不同的 AWS Managed Microsoft AD 方式进行配置 AWS 账户，则可能需要相应地修改 Epics 部分中的步骤。

此设计使用 Route 53 解析器，通过使用 Route 53 规则来支持名称解析。如果本地 DNS 解决方案使用 Microsoft DNS，则为 AWS 命名空间 (`aws.company.com`) (公司 DNS 命名空间 (`company.com`) 的子域) 创建条件转发规则并不简单。如果您尝试创建传统的条件转发器，将会导致错误。这是因为 Microsoft Active Directory 已经被认为对的任何 `company.com` 子域具有权威性。要解决此错误，必须首先为创建一个 `aws.company.com` 委托，委托该命名空间的权限。然后，您可创建条件转发器。

每个分支账户的虚拟私有云 (VPC) 可以根据根命名空间拥有自己的唯一 DNS AWS 命名空间。在此设计中，每个分支账户将账户名称的缩写附加到基本 AWS 命名空间。在分支账户中创建私有托管区域后，这些区域将与分支账户中的本地 VPC 以及中央 AWS 网络账户中的 VPC 相关联。这样，中央 AWS 网络账户就可以回答与分支账户相关的 DNS 查询。这样，Route 53 和 Route 就可以 AWS Managed Microsoft AD 共同承担管理 AWS 命名空间 (`aws.company.com`) 的责任。

自动化和扩缩

此设计使用 Route 53 解析器端点在本地环境 AWS 之间扩展 DNS 查询。每个 Route 53 Resolver 端点包含多个弹性网络接口（分布在多个可用区），每个网络接口每秒最多可处理 10,000 个查询。Route 53 解析器支持每个端点最多 6 个 IP 地址，因此该设计总共支持每秒多达 60,000 个 DNS 查询，分布在多个可用区中，以实现高可用性。

此外，这种模式会自动考虑内部未来的增长 AWS。无需修改本地配置的 DNS 转发规则，即可支持新增的私有托管区域 VPCs 及其关联的私有托管区域 AWS。

工具

Amazon Web Services

- AWS Directory Service for Microsoft Active Directory允许你的目录感知工作负载和 AWS 资源使用中的 Microsoft Active Directory。 AWS Cloud
- AWS Organizations是一项账户管理服务，可帮助您将多个账户整合 AWS 账户 到一个由您创建和集中管理的组织中。
- AWS Resource Access Manager (AWS RAM) 可帮助您安全地共享资源， AWS 账户 从而减少运营开销并提供可见性和可审计性。
- Amazon Route 53 是一种可用性高、可扩展性强的 DNS Web 服务。

工具

- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行外壳中的命令进行交互。在此模式中， AWS CLI 用于配置 Route 53 授权。

操作说明

创建和共享 AWS Managed Microsoft AD 目录

Task	描述	所需技能
部署 AWS Managed Microsoft AD。	1. 创建和配置新的目录。有关详细步骤，请参阅《AWS Directory Service 管理指南》 AWS Managed Microsoft AD中的 “创建”。 2. 记录 AWS Managed Microsoft AD 域控制器的 IP 地址。 您将在后面的步骤中涉及。	AWS 管理员
共享您的目录	建立目录后，与 AWS 组织 AWS 账户 中的其他人共享。有关说明，请参阅《AWS Directory Service 管理指南》中的 “共享您的目录”。 **Note** AWS Managed Microsoft AD 标准版的股票上限为 5 股。 企业版的股票上限为 125 股。	AWS 管理员

配置 Route 53

Task	描述	所需技能
创建 Route 53 解析器。	Route 53 解析器简化了本地数据中心 AWS 之间的 DNS 查询解析。 1. 按照 Route 53 开发人员指南中的说明 安装 Route 53 解析器。 2. 在中央 AWS 网络账户 VPC 内至少两个可用区的私有子网中配置 Route 53 解析器，以实现高可用性。 **Note** 尽管不需要使用中央 AWS 网络账户 VPC，但其余步骤假设此配置。	AWS 管理员
创建 Route 53 规则。	您的特定使用案例可能需要大量 Route 53 规则，但您需要配置以下规则作为基准： • 本地命名空间 (`company.com`) 的传出规则，使用中央网络账户出站 Route 53 Resolvers。目标 IP 地址是本地 DNS 服务器。 • 将此规则与中央网络账户 VPC 关联。 • AWS 命名空间 (`aws.company.com`) 的传出规则，使	AWS 管理员

Task	描述	所需技能
	用中央网络账户出站 Route 53 Resolvers。目标 IP 地址是中央网络账户的入站 Route 53 解析器 IP 地址。 • 请勿将此规则与中央 AWS 网络账户 VPC（其中包含 Route 53 解析器）相关联。 • AWS 命名空间 (`aws.company.com`) 的第二条传出规则，它指向 AWS Managed Microsoft AD 域控制器（使用前一篇长篇故事中的）。IPs • 将此规则与中央 AWS 网络账户 VPC（其中包含 Route 53 解析器）相关联。 • 请勿将此规则与其他规则共享或关联 AWS 账户。 有关更多信息，请参阅 Route 53 开发人员指南中的托管转发规则。	

Task	描述	所需技能
配置 Route 53 配置文件。	Route 53 配置文件用于与分支账户共享规则。 1. 按照《Route 53 开发者指南》中的说明在中央网络账户中创建新的 Route 53 配置文件。 2. 将本地命名空间 (`company.com`) 的规则添加到配置文件中。 3. 将 AWS 命名空间 (`aws.company.com`) 的第一条规则添加到配置文件中，该规则以 Route 53 入站解析器的 IP 地址为目标。 4. 与 AWS 组织共享 Route 53 简介。 5. 在每个分支账户中接受 Route 53 配置文件资源共享。 6. 将 Route 53 配置文件与每个分支账户 VPC 相关联。	AWS 管理员

配置本地 Active Directory DNS

Task	描述	所需技能
创建委托。	使用 Microsoft DNS 管理单元 (`dnsmgmt.msc`) 在 Active Directory 中为 `company.com` 命名空间创建新的委托。委托域的名称应为 `aws`。这使得 `aws.company.com` 委	Active Directory

Task	描述	所需技能
	托的完全限定域名 (FQDN) 成为可能。使用 AWS Managed Microsoft AD 域控制器的 IP 地址作为名称服务器 IP 值，并使用`server.aws.company.com` 作为名称。（此委托仅用于冗余，因为将为此命名空间创建一个优先于委托的条件转发器。）	
创建条件转发器。	使用 Microsoft DNS 管理单元 (`dnsmgmt.msc`) 为 `aws.company.com` 创建新条件转发器。 使用中央 DNS 中 AWS 入站 Route 53 解析器的 IP 地址 AWS 账户 作为条件转发器的目标。	Active Directory

为辐条创建 Route 53 私有托管区域 AWS 账户

Task	描述	所需技能
创建 Route 53 私有托管区：	在每个分支账户中创建 Route 53 私有托管区域。将此私有托管区域与分支账户 VPC 相关联。有关详细步骤，请参阅 Route 53 开发人员指南中的创建私有托管区。	AWS 管理员
创建授权。	使用 AWS CLI 为中央 AWS 网络账户 VPC 创建授权。在每个分支的上下文中运行以下命令 AWS 账户：	AWS 管理员

Task	描述	所需技能
	`aws route53 create-vpc-association-authorization --hosted-zone-id <hosted-zone-id> \ --vpc VPCRegion=<region>,VPCId=<vpc-id>` 其中： • <hosted-zone-id> 是分支账户中的 Route 53 私有托管区域。 • <region> AWS 区域和<vpc-id>是中央 AWS 网络账户 VPC 的和 VPC ID。	

Task	描述	所需技能
创建关联。	使用为中央 AWS 网络账户 VPC 创建 Route 53 私有托管区域关联 AWS CLI。在中央 AWS 网络帐户的上下文中运行以下命令： `aws route53 associate-vpc-with-hosted-zone --hosted-zone-id <hosted-zone-id> \ --vpc VPCRegion=<region>,VPCId=<vpc-id>` 其中： • `<hosted-zone-id>` 是分支账户中的 Route 53 私有托管区域。 • `<region>`并且`<vpc-id>`是中央 AWS 网络账户的 AWS 区域 和 VPC ID。	AWS 管理员

相关资源

- 使用 Route 53 解析器简化多账户环境中的 DNS 管理（AWS 博客文章）
- 创建你的 AWS Managed Microsoft AD（AWS Directory Service 文档）
- 共享 AWS Managed Microsoft AD 目录（AWS Directory Service 文档）
- 什么是 Amazon Route 53 Resolver？（亚马逊 Route 53 文档）
- 创建私有托管区域（亚马逊 Route 53 文档）
- 什么是 Amazon Route 53 个人资料？（亚马逊 Route 53 文档）

使用 Amazon CloudWatch 可观测性访问管理器进行集中监控

由 Anand Krishna Varanasi (AWS)、Jimmy Morgan (AWS)、Ashish Kumar (AWS)、Balaji Vedagiri (AWS)、JAGDISH KOMAKULA (AWS)、Sarat Chandra Pothula (AWS) 和 Vivek Thangamuthu (AWS) 创作

摘要

可观测性对于监控、理解应用程序和排除应用程序故障至关重要。跨多个账户的应用程序（如 AWS Control Tower 或 landing zone 实现）会生成大量日志和跟踪数据。为了快速解决问题或了解用户分析或业务分析，您需要一个跨所有账户的通用可观测性平台。Amazon CloudWatch Observability 访问管理器允许您从一个中心位置访问和控制多个账户日志。

您可使用可观测性访问管理器来查看和管理源账户生成的可观测性数据日志。源账户是为其资源生成 AWS 账户 可观测性数据的个人。源账户与监控账户共享其可观测性数据。共享的可观察性数据可以包括 Amazon 中的指标 CloudWatch、Amazon 日志中的 CloudWatch 日志以及中的 AWS X-Ray跟踪。有关更多信息，请参阅 Observability Access Manager 文档。

这种模式适用于拥有多个应用程序或基础架构 AWS 账户 并需要在公共位置查看日志的用户。它解释了如何使用 Terraform 设置 Observability Access Manager，以监控这些应用程序或基础设施的状态和运行状况。您可以通过多种方式安装此解决方案：

- 作为您手动设置的独立 Terraform 模块
- 通过使用持续集成和持续交付 (CI/CD) 管道
- 通过与其他解决方案集成，例如 Acco AWS Control Tower unt Factory for Terraform (AFT)

操作说明部分中的说明涵盖了手动实现。有关 AFT 的安装步骤，请参阅 Obs GitHub ervability Access Manager 存储库的自述文件。

先决条件和限制

先决条件

- 在您的系统或自动管道中安装或引用 Terraform。（我们建议您使用最新版本。）
- 可用作中央监控账户的账户。其他账户创建到中央监控账户的链接以便查看日志。
- （可选）源代码存储库，例如 GitHub、 AWS CodeCommit、Atlassian Bitbucket 或类似系统。如果您使用的是自动化 CI/CD 管道，则不需要源代码存储库。
- （可选）在中创建用于代码审查和代码协作的拉取请求 (PRs) 的权限 GitHub。

限制

Observability Access Manager 具有以下服务限额，且该限额不可更改。部署此功能之前，请考虑这些配额。有关更多信息，请参阅 CloudWatch 文档中的CloudWatch 服务配额。

- 源账户关联：您最多可以将每个源账户关联到五个监控账户。
- S ink AWS 区域 s：您可以为一个账户构建多个接收器，但每个账户只允许生成一个接收器。

此外：

- 汇和链接必须在同一位置创建 AWS 区域；它们不能是跨区域的。

跨区域和跨账户监控

对于跨区域、跨账户监控，您可以选择以下选项之一：

- 为警报和指标创建跨账户和跨区域 CloudWatch 仪表板。此选项不支持日志和跟踪。
- 使用 Amazon OpenSearch 服务实现集中日志。
- 从所有租户账户中为每个区域创建一个接收器，将指标推送到集中式监控账户（如本模式中所述），然后使用CloudWatch 指标流将数据发送到公共外部目标或第三方监控产品，例如 Datadog、Dynatrace、Sumo Logic、Splunk 或 New Relic。

架构

组成部分

CloudWatch Observability Access Manager 由两个主要组件组成，可实现跨账户可观察性：

- 接收器使源账户能够将可观测性数据发送到中央监控账户。接收器基本上为源账户提供了一个连接的网关连接。只能有一个接收器网关或连接，并且可以有多个账户连接到它。
- 每个源账户都有一个到接收器网关连接的链接，可观测性数据通过此链接发送。必须先创建接收器，然后才能从每个源账户创建链接。

架构

下图说明了 Observability Access Manager 及其组件。

工具

AWS 服务

- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和运行的应用程序 AWS 的指标。
- AWS Organizations是一项账户管理服务，可帮助您将多个账户整合 AWS 账户 到一个由您创建和集中管理的组织中。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。

工具

- Terraform 是一款基础设施即代码 (IaC) 工具 HashiCorp ，可帮助您创建和管理云和本地资源。
- AWS Control Tower Account F@@ actory for Terraform (AFT) 设置了 Terraform 管道来帮助你在中配置和自定义账户。 AWS Control Tower您可以选择使用 AFT 跨多个账户大规模设置 Observability Access Manager。

代码存储库

此模式的代码可在 Obs GitHub ervability 访问管理器存储库中找到。

最佳实践

- 在 AWS Control Tower 环境中，将日志帐户标记为中央监控帐户（接收器）。
- 如果您有多个组织中有多个账户 AWS Organizations，我们建议您在配置策略中加入组织而不是个人账户。如果您的账户数量较少，或者这些账户不属于接收器配置策略中的组织，则您可能会决定包含个人账户。

操作说明

设置接收器模块

Task	描述	所需技能
克隆存储库。	克隆 GitHub 可观测性访问管理器存储库：	AWS DevOps、云管理员、AWS 管理员

Task	描述	所需技能
	`git clone https://github.com/aws-samples/cloudwatch-observability-access-manager-terraform`	
为接收器模块指定属性值。	在 `main.tf` 文件中 (在存储库的 `deployments/aft-account-customizations/LOGGING/terraform/` 文件夹中)，为以下属性指定值： • `sink_name` ：CloudWatch 水槽的名称。 • `allowed_oam_resource_types` ：可观察性访问管理器目前支持 CloudWatch 指标、日志组和 AWS X-Ray 跟踪。 • `allowed_source_accounts` ：允许向中央 CloudWatch 接收器帐户发送日志的源帐户。 • `allowed_source_organizations` ：允许向中央 CloudWatch 接收器账户发送日志的来源 AWS Control Tower 组织。 有关更多信息，请参阅 AWS CloudFormation 文档AWS::Oam::Sink中的。	AWS DevOps、云管理员、AWS 管理员

Task	描述	所需技能

安装接收器模块。	导出您选择作为监控账户的凭据 AWS 账户 ，然后安装 Observability Access Manager 接收器模块：
	`Terraform Init`
	`Terrafom Plan`
	`Terraform Apply`
		AWS DevOps、云管理员、AWS 管理员

设置链接模块

Task	描述	所需技能

为链接模块指定属性值。	在 `main.tf` 文件中（在存储库的 `deployments/aft-account-customizations/LOGGING/terraform/` 文件夹中），为以下属性指定值：
	• `account_label` ：使用下列值之一：
	  • `$AccountName` ：账户的名称。
	  • `$AccountEmail` ：全球唯一的电子邮件地址，包括电子邮件域(例如 `hello@example.com`)
	  • `$AccountEmailNoDomain` ：没有域名的电子邮件地址。
	• `allowed_oam_resource_types` ：可观察性访问
		AWS DevOps、云管理员、云架构师

Task	描述	所需技能
	管理器目前支持 CloudWatch 指标、日志组和 AWS X-Ray 跟踪。 有关更多信息，请参阅 AWS CloudFormation 文档AWS::Oam::Link中的。	

为个人账户安装链接模块。	导出个人账户凭证，并安装 Observability 访问管理器链接模块：
	
	`Terraform Plan`
	`Terraform Apply`
	
	您可为每个账户单独设置链接模块，也可以使用 AFT 在大量账户中自动安装此模块。
		AWS DevOps、云管理员、云架构师

批准 sink-to-link连接

Task	描述	所需技能
状态消息。	1. 登录到监控账户。 2. 打开 CloudWatch 管理控制台。 3. 在左侧导航窗格中，选择 设置。 在右侧，您应该会看到带有绿色勾号的监控账户已启用状态消息。这意味着监控账户有一个 Observability Access	

Task	描述	所需技能
	Manager 接收器，其他账户的链接将连接到该接收器。	
批准 link-to-sink连接。	1. 选择状态消息下方的关联账户资源选项。该信息确认这是监控账户，列出了从租户源帐户共享的数据（日志、指标、跟踪），并将账户标签显示为 $ AccountName。 此屏幕提供了两个用于将租户账户链接到监控账户的选项：组织级批准或账户级批准。对于每个选项，您可以选择下载审批 AWS CloudFormation 模板或单独批准每个账户。 2. 为简单起见，在每个账户级别选择要批准的任意账户。此选项为账户提供批准的链接。 3. 选择复制 URL 以复制链接。 4. 登录到源账户。 5. 在浏览器窗口，粘贴链接，然后选择批准接收器链接。 6. 对于其他源账户，重复此操作。 有关更多信息，请参阅 CloudWatch 文档中的将监控账户与源账户关联起来。	AWS DevOps、云管理员、云架构师

验证跨账户可观测性数据

Task	描述	所需技能
查看跨账户数据。	1. 登录到监控账户。 2. 打开 CloudWatch 管理控制台。 3. 在左侧导航窗格，选择选项以查看跨账户日志、指标和跟踪。	AWS DevOps、云管理员、云架构师

（可选）允许源账户信任监控账户

Task	描述	所需技能
查看其他账户的指标、控制面板、日志、小部件和警报。	作为一项附加功能，您可以与其他账户共享 CloudWatch 指标、仪表板、日志、小组件和警报。每个账户都使用名为 CloudWatch- 的 IAM 角色 CrossAccountSharingRole 来访问这些数据。 与中央监控账户有信任关系的源账户可以承担该角色并查看监控账户的数据。 CloudWatch 提供了用于创建角色的示例 CloudFormation 脚本。选择管理 IAM 角色，然后在要查看数据的账户中运行此脚本。 `{ "Version": "2012-10-17", "Statement": [`	AWS DevOps、云管理员、云架构师

Task	描述	所需技能

	<pre>{
	"Effect": "Allow",
	"Principal": {
	"AWS": [
	"arn:aws:iam::XXXXXXXXXX:root",
	"arn:aws:iam::XXXXXXXXXX:root",
	"arn:aws:iam::XXXXXXXXXX:root",
	"arn:aws:iam::XXXXXXXXXX:root"
	]
	},
	"Action": "sts:AssumeRole"
	}
	]
	}</pre> 有关更多信息，请参阅 CloudWatch 文档 CloudWatch 中的启用跨账户功能。	

（可选）从监控账户查看跨账户跨 Region

Task	描述	所需技能
设置跨账户、跨区域访问。	在中央监控账户中，您可以选择添加账户选择器，以便在账户之间轻松切换并查看其数据，而无需进行身份验证。	AWS DevOps、云管理员、云架构师

Task	描述	所需技能
	1. 登录到监控账户。 2. 打开 CloudWatch 管理控制台。 3. 在左侧导航窗格中，选择 设置。 4. 在查看跨账户跨区域部分中，选择配置。 5. 选择启用，然后选中显示控制台选择器复选框。 6. 选择以下选项之一： • 账户 ID 输入：当您想要更改账户以查看跨账户数据时，此选项会提示您手动输入账户 ID。 • AWS 组织账户选择器：如果您已 CloudWatch 与集成 AWS Organizations，则此选项将提供一个下拉选择器，其中包含组织中账户的完整列表。 • 自定义账户选择器：此选项允许您手动输入账户列表 IDs 以填充选择器。 7. 选择 Save changes（保存更改）。 有关更多信息，请参阅文档中的跨账户跨区域 CloudWatch 控制台。 CloudWatch	

相关资源

- CloudWatch 跨账户可观察性（Ama CloudWatch zon 文档）
- 亚马逊 CloudWatch 可观察性访问管理器 API 参考（亚马逊 CloudWatch 文档）
- 资源：aws_oam_sink（Terraform 文档）
- 数据来源：aws_oam_link（Terraform 文档）
- CloudWatchObservabilityAccessManager（AWS Boto3 文档）

在启动时检查 EC2 实例是否有必需的标签

由 Susanne Kangnoh (AWS) 和 Archit Mathur (AWS) 创建

摘要

亚马逊弹性计算云 (Amazon EC2) 在亚马逊网络服务 (AWS) 云中提供可扩展的计算容量。使用 Amazon，您 EC2 无需预先投资硬件，因此您可以更快地开发和部署应用程序。

您可以使用标签以不同的方式对您的 AWS 资源进行分类。 EC2 当您的账户中有许多资源并且想要根据标签快速识别特定资源时，实例标记非常有用。您可以使用标签为 EC2 实例分配自定义元数据。每个标签都由用户定义的键和值组成。我们建议您创建一组一致的标签以满足您的组织要求。

此模式提供了一个 AWS CloudFormation 模板来帮助您监控 EC2 实例中是否存在特定标签。该模板创建一个 Amazon Events CloudWatch 事件，用于监视 AWS CloudTrail TagResource或UntagResource事件，以检测新的 EC2 实例标记或标签移除。如果缺少预定义标签，它会调用 AWS Lambda 函数，该函数会使用 Amazon Simple Notification Service (Amazon SNS) 向您提供的电子邮件地址发送违规消息。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 用于上传提供的 Lambda 代码的 Amazon Simple Storage Service （ Amazon S3 ）存储桶。
- 您希望接收违规通知的电子邮件地址。

限制

- 此解决方案支持 CloudTrail TagResource或UntagResource活动。它不会为任何其他事件创建通知。
- 此解决方案仅检查标签键。它并不监视键值。

架构

工作流程架构

自动化和扩缩

- 您可以针对不同的 AWS 区域和账户多次使用 AWS CloudFormation 模板。您只需在每个区域或账户中运行该模板一次。

工具

Amazon Web Services

- 亚马逊 EC2 — Amazon Elastic Compute Cloud (Amazon EC2) 是一项网络服务，可在云中提供安全、可调整大小的计算容量。该服务旨在降低开发人员进行网络规模级云计算的难度。
- AWS CloudTrail — CloudTrail 是一项 AWS 服务，可帮助您对 AWS 账户进行治理、合规以及运营和风险审计。用户、角色或 AWS 服务采取的操作在中记录为事件 CloudTrail。
- Amazon CloudWatch Events — Amazon CloudWatch Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。 CloudWatch 通过发送消息以响应环境、激活功能、进行更改和捕获状态信息，事件会在操作变化发生时意识到这些变化，并在必要时采取纠正措施。
- AWS Lambda — AWS Lambda 是一项计算服务，使您无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项高度可扩展的对象存储服务，可用于各种存储解决方案，包括网站、移动应用程序、备份和数据湖。
- Amazon SNS — Amazon Simple Notification Service (Amazon SNS) 是一项 Web 服务，可让应用程序、终端用户和设备即时发送和接收来自云端的通知。

代码

此模式包括一个包含两个文件的附件：

- `index.zip` 是压缩文件，其中包含此模式的 Lambda 代码。
- `ec2-require-tags.yaml`是部署 Lambda 代码的 CloudFormation 模板。

有关如何使用这些文件的信息，请参阅操作说明部分。

操作说明

部署 Lambda 代码

Task	描述	所需技能
将代码上传到 S3 存储桶。	创建新的 S3 存储桶或使用现有 S3 存储桶上传附加的 `index.zip` 文件（Lambda 代码）。此存储桶必须与您要监控的资源（EC2 实例）位于同一 AWS 区域。	云架构师
部署 CloudFormation 模板。	在与 S3 存储桶相同的 AWS 区域中打开 Cloudformation 控制台，然后部署附件中提供的 `ec2-require-tags.yaml` 文件。在下一个操作说明中，提供模板参数的值。	云架构师

填写 CloudFormation 模板中的参数

Task	描述	所需技能
提供 S3 存储桶名称。	输入您在第一篇操作说明中创建或选择的 S3 存储桶的名称。此 S3 存储桶包含 Lambda 代码的.zip 文件，并且必须与模板和 EC2 要监控 CloudFormation 的实例位于相同的 AWS 区域。	云架构师
提供 S3 密钥。	提供 Lambda 代码 .zip 文件在 S3 存储桶中的位置，不带前导斜杠（例如，`index.zip` 或 `controls/index.zip`）。	云架构师

Task	描述	所需技能
提供电子邮箱地址。	提供要接收违规通知的活动电子邮件地址。	云架构师
定义日志记录级别。	指定日志记录级别和详细程度。`Info` 指定有关应用程序进度的详细信息消息，应仅用于调试。`Error` 指定仍允许应用程序继续运行的错误事件。`Warning` 表示潜在的有害情况。	云架构师
输入所需标签密钥。	输入要检查的标签键。如果要指定多个密钥，请用逗号分隔，不要使用空格。（例如，`ApplicationId,CreatedBy,Environment,Organization` 搜索四个密钥。）Events CloudWatch 事件搜索这些标签密钥，如果找不到则发送通知。	云架构师

确认订阅

Task	描述	所需技能
确认电子邮件订阅。	成功部署 CloudFormation 模板后，它会向您提供的电子邮件地址发送一封订阅电子邮件。要接收通知，您必须确认此电子邮件订阅。	云架构师

相关资源

- 创建存储桶（Amazon S3 文档）

- 上传对象（Amazon S3 文档）
- 标记您的亚马逊 EC2 资源（亚马逊 EC2 文档）
- 使用 AWS 创建在 AWS API 调用时触发 CloudWatch 的事件规则 CloudTrail（亚马逊 CloudWatch 文档）

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用会话管理器连接到 Amazon EC2 实例

由 Jason Cornick (AWS)、Abhishek Bastikoppa (AWS) 和 Yaniv Ron (AWS) 创建

摘要

此模式描述了如何使用会话管理器（AWS Systems Manager 的一项功能 EC2）连接到亚马逊弹性计算云 (Amazon) 实例。使用这种模式，您可以通过 Web 浏览器在 EC2 实例上运行 bash 命令。会话管理器不需要您打开入站端口，也不需要公有 IP 地址作为 EC2 实例。此外，它还消除了使用不同 Secure Shell (SSH) 密钥维护堡垒主机的需要。您可以使用 AWS Identity and Access Management (IAM) 策略管理对 Session Manager 的访问，并配置日志记录，记录重要信息，例如实例访问和操作。

在此模式中，您可以配置 IAM 角色并将其关联到您使用亚马逊系统映像 (AMI) 预配置的 Linux EC2 实例。然后，您可以在 Amazon CloudWatch Logs 中配置日志并使用会话管理器启动与该实例的会话。

尽管此模式连接到 Amazon Web Services (AWS) 云中的 Linux EC2 实例，但您可以使用这种方法使用会话管理器与其他服务器（例如本地服务器或其他虚拟机）进行连接。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 访问托管节点的权限。有关说明，请参阅控制用户会话对托管节点的访问权限。
- `ssm`、`ec2`、`ec2messages`、`ssmmessages` 和 `s3` 的 VPC 端点。有关说明，请参阅 Systems Manager 文档中的创建 VPC 端点。

架构

目标技术堆栈

- 会话管理器
- Amazon EC2
- CloudWatch 日志

目标架构

1. 用户通过 IAM 验证其身份和凭证。
2. 用户通过会话管理器启动 SSH 会话，并向 EC2 实例发送 API 调用。
3. 安装在 EC2 实例上的 AWS Systems Manager SSM 代理连接到会话管理器并运行命令。
4. 出于审计和监控目的，会话管理器将日志数据发送到 CloudWatch 日志。或者，您可以将日志数据发送到 Amazon Simple Storage Service (Amazon S3) 存储桶。有关更多信息，请参阅使用 Amazon S3 记录会话数据（Systems Manager 文档）。

工具

Amazon Web Services

- Amazon CloudWatch Lo gs 可帮助您集中管理来自所有系统、应用程序和 AWS 服务的日志，以便您可以监控它们并安全地将其存档。
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。此模式使用亚马逊系统映像 (AMI) 来配置 Linux EC2 实例。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Systems Manager 可帮助您管理在 Amazon Web Services Cloud 中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。此模式使用 Session Manager，这是 Systems Manager 的一项功能。

最佳实践

我们建议您阅读有关 AWS Well-Architected Framework 的安全支柱的更多信息，并探索加密选项并应用设置 Session Manager（Systems Manager 文档）中的安全建议。

操作说明

设置基础架构

Task	描述	所需技能
创建 IAM 角色。	为 SSM 代理创建 IAM 角色。按照为 Amazon Web Services	AWS 系统管理员

Task	描述	所需技能
	创建角色（IAM 文档）中的说明操作，并注意以下事项： 1. 对于 AWS 服务，请选择EC2。 2. 对于权限策略，选择 `AmazonSSMManagedInstanceCore` 。 3. 在角色名称 中，输入 `EC2_SSM_Role` 。	

Task	描述	所需技能
创建实 EC2 例。	1. 创建实 EC2 例。按照启动实例（Amazon EC2 文档）中的说明进行操作，并注意以下几点： a. 在名称和标签部分中，选择添加其他标签。在 Key (键) 中输入 Name，在 Value (值) 中输入 Productio n_Server_One 。 b. 选择预安装了 SSM 代理的 Amazon Linux AMI。有关完整列表，请参阅预安装 SSM 代理AMIs的情况（Systems Manager 文档）。 c. 在高级详细信息部分的 IAM 实例配置文件中，选择 EC2_ssm_Role。 2. 打开 Systems Manager 控制台，网址为https://console.aws.amazon.com/systems-manager/。 3. 在导航窗格中，选择 Fleet Manager。 4. 验证实例是否显示在托管式节点列表中。	AWS 系统管理员

Task	描述	所需技能
设置日志记录。	1. 在日志中创建 CloudWatch 日志组。按照创建日志组（CloudWatch 日志文档）中的说明进行操作。命名新日志组 `SessionManager` 。 2. 为 Session Manager 配置日志记录。按照使用 Amazon CloudWatch Logs 记录会话数据（Systems Manager 文档）中的说明进行操作，并注意以下几点： a. 不要选择 “仅允许加密的 CloudWatch 日志组”。 b. 在从列表中选择日志组中，选择SessionManager。	AWS 系统管理员

连接到实例

Task	描述	所需技能
Connect 连接到 EC2 实例。	1. 在 Systems Manager 控制台中启动会话。有关说明，请参阅启动会话（Systems Manager 文档）。对于目标实例，选择 Production_Server_One 实例左侧的选项按钮。 2. 建立连接后，运行多个 bash 命令。 3. 在 Systems Manager 控制台中，结束会话。有	AWS 系统管理员

Task	描述	所需技能
	关说明，请参阅结束会话（Systems Manager 文档）。	
验证日志记录。	1. 在 CloudWatch 日志中，打开日志组的日志流。有关说明，请参阅查看日志数据（CloudWatch 日志文档）。 2. 在日志数据中，确认列出了在上一个情景中运行的命令。	AWS 系统管理员

故障排除

事务	解决方案
IAM 问题	如需支持，请参阅故障排除（IAM 文档）。

相关资源

- 完成 Session Manager 先决条件（Systems Manager 文档）
- 使用亚马逊设计和实施日志记录和监控 CloudWatch（AWS Prescriptive Guidence）

在不支持 AWS 的 AWS 区域创建管道 CodePipeline

由 Anand Krishna Varanasi (AWS) 编写

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

AWS CodePipeline 是一项持续交付 (CD) 编排服务，是亚马逊网络服务 (AWS) 的一组 DevOps 工具的一部分。它与各种来源（例如版本控制系统和存储解决方案）、AWS 和 AWS Partners 提供的持续集成 (CI) 产品和服务以及开源产品集成，为应用程序和基础设施的快速部署提供 end-to-end工作流程服务。

但是， CodePipeline 并非所有 AWS 区域都支持，因此使用隐形协调器来连接 AWS、AWS 和 AWS 等 AW CodeCommit S CI/CD services. This pattern describes how to implement an end-to-end workflow pipeline in AWS Regions where CodePipeline isn’t yet supported by using AWS CI/CD CodeBuild 服务会很有用。 CodeDeploy

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- AWS Cloud Development Kit (AWS CDK) CLI 版本 2.28 或更高版本

架构

目标技术堆栈

下图显示了在不支持的 CodePipeline区域（例如非洲（开普敦）区域）中创建的管道。开发者将 CodeDeploy 配置文件（也称为部署生命周期挂钩脚本）推送到托管的 Git 存储库 CodeCommit。（请参阅此模式提供的GitHub 存储库。）Amazon EventBridge 规则会自动启动。 CodeBuild

CodeDeploy 配置文件 CodeCommit 作为管道源阶段的一部分从中提取并传输到 CodeBuild。

在下一阶段， CodeBuild 执行以下任务：

1. 下载应用程序源代码 TAR 文件。您可以使用参数存储（AWS Systems Manager 的一项功能）来配置此文件的名称。

2. 下载 CodeDeploy 配置文件。
3. 创建应用程序源代码和特定于应用程序类型的 CodeDeploy 配置文件的组合存档。
4. 使用组合存档启动 CodeDeploy 部署到亚马逊弹性计算云 (Amazon EC2) 实例。

工具

Amazon Web Services

- AWS CodeBuild 是一项完全托管的构建服务，可帮助您编译源代码、运行单元测试和生成可随时部署的项目。
- AWS CodeCommit 是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。
- AWS CodeDeploy 可自动部署到亚马逊 EC2 或本地实例、AWS Lambda 函数或亚马逊弹性容器服务 (Amazon ECS) Container Service 服务。
- AWS CodePipeline 可帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件变更所需的步骤。
- AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，可帮助您在代码中定义并预置 Amazon Web Services Cloud 基础设施。

代码

此模式的代码可在 “ GitHub CodePipeline 不支持的区域” 存储库中找到。

操作说明

设置开发人员工作站

Task	描述	所需技能
安装 AWS CDK CLI。	有关说明，请参阅 AWS CDK 文档。	AWS DevOps
安装 Git 客户端。	要创建提交，您可以使用安装在本地计算机上的 Git 客	AWS DevOps

Task	描述	所需技能
	户端，然后将提交推送到 CodeCommit 存储库。要使用 Git 客户端 CodeCommit 进行设置，请参阅CodeCommit 文档。	
安装 npm。	安装 npm 包管理器。有关更多信息，请参阅 npm 文档。	AWS DevOps

设置管道

Task	描述	所需技能
克隆代码存储库。	运行以下命令，将 “ GitHub CodePipeline 不支持的区域” 存储库克隆到您的本地计算机。 `git clone https://github.com/aws-samples/invisible-codepipeline-unsupported-regions`	DevOps 工程师
在 cdk.json 中设置参数。	打开 `cdk.json`文件并为以下参数赋值： `"pipeline_account":"XXXXXXXXXXXX", "pipeline_region":"us-west-2", "repo_name": "app-dev-repo", "ec2_tag_key": "test-vm", "configName" : "cbdeployconfig",`	AWS DevOps

Task	描述	所需技能

	<pre>"deploymentGroupNa
	me": "cbdeploygroup",
	"applicationName" :
	"cbdeployapplicati
	on",
	"projectName" :
	"CodeBuildProject"</pre>
	其中：
	• `pipeline_account` 是将在其中构建管道的 Amazon Web Services account。
	• `pipeline_region` 是将在其中构建管道的 AWS 区域。
	• `repo_name` 是 CodeCommit 存储库的名称。
	• `ec2_tag_key` 是附加到您要将代码部署到的 EC2 实例的标签。
	• `configName` 是 CodeDeploy 配置文件的名称。
	• `deploymentGroupNam e` 是 CodeDeploy 部署组的名称。
	• `applicationName` 是 CodeDeploy 应用程序的名称。
	• `projectName` 是 CodeBuild 项目名称。	

Task	描述	所需技能

设置 AWS CDK 构造库。	在克隆的 GitHub 存储库中，使用以下命令安装 AWS CDK 构造库、构建您的应用程序并进行合成以生成应用程序的 AWS CloudFormation 模板。
	`npm i aws-cdk-lib`
	`npm run build`
	`cdk synth`
		AWS DevOps

部署示例 AWS CDK 应用程序	在不支持的区域(例如 `af-south-1`)中运行以下命令来部署代码。 `cdk deploy`	AWS DevOps

为以下内容设置 CodeCommit 存储库 CodeDeploy

Task	描述	所需技能
为应用程序设置 CI/CD。	克隆您在`cdk.json`文件中指定的 CodeCommit 存储库（默认为该存储库`app-dev-repo` ），为应用程序设置 CI/CD 管道。 `git clone https://git-codecommit.us-west-2.amazonaws.com/v1/repos/app-dev-repo` 其中，存储库名称和区域取决于您在 `cdk.json` 文件中提供的值。	AWS DevOps

测试管道

Task	描述	所需技能
根据部署说明测试管道。	“ GitHub CodePipeline 不支持的区域” 存储库的CodeDeploy_Files 文件夹包含指示部署应用程序 CodeDeploy 的示例文件。该appspec.yml 文件是一个 CodeDeploy 配置文件，其中包含用于控制应用程序部署流程的挂钩。您可以使用示例文件 index.html 、start_server.sh 、stop_server.sh 和 install_dependencies.sh 来更新托管在 Apache 上的网站。这些都是示例，您可以使用 GitHub 存储库中的代码来部署任何类型的应用程序。当文件被推送到 CodeCommit 存储库时，会自动启动不可见的管道。有关部署结果，请在 CodeBuild 和 CodeDeploy 控制台中查看各个阶段的结果。	AWS DevOps

相关资源

- 入门（AWS CDK 文档）
- 云开发工具包（CDK）简介(AWS 研习会参与平台 Workshop Studio)
- AWS CDK 研讨会

使用 AWS CDK 外观和逃生舱口自定义默认角色名称

由 SANDEEP SINGH (AWS) 和詹姆斯·雅各布 (AWS) 创作

摘要

此模式演示如何自定义由 AWS Cloud Development Kit (AWS CDK) 构造创建的角色的默认名称。如果您的组织有基于命名惯例的特定限制，则通常需要自定义角色名称。例如，您的组织可能会设置 AWS Identity and Access Management (IAM) 权限边界或服务控制策略 (SCPs)，要求在角色名称中使用特定的前缀。在这种情况下，由 AWS CDK 构造生成的默认角色名称可能不符合这些约定，因此可能需要更改。此模式通过在中使用逃生舱口和方面来满足这些要求。 AWS CDK您可以使用逃生舱口来定义自定义角色名称，并使用方面将自定义名称应用于所有角色，以确保遵守组织的政策和限制。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- AWS CDK 文档中指定的先决条件

限制

- Aspects 根据资源类型筛选资源，因此所有角色共享相同的前缀。如果您需要为不同的角色添加不同的角色前缀，则需要根据其他属性进行额外筛选。例如，要为与 AWS Lambda 函数关联的角色分配不同的前缀，您可以按特定的角色属性或标签进行筛选，为与 Lambda 相关的角色应用一个前缀，为其他角色应用不同的前缀。
- IAM 角色名称的最大长度为 64 个字符，因此必须修剪修改后的角色名称才能满足此限制。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额页面，然后选择服务的链接。

架构

目标技术堆栈

- AWS CDK
- AWS CloudFormation

目标架构

- 一个 AWS CDK 应用程序由一个或多个 AWS CloudFormation 堆栈组成，这些堆栈经过合成和部署以管理 AWS 资源。
- 要修改第 2 层 (L2) 构造未公开的 AWS CDK托管资源的属性，您可以使用逃生舱口来覆盖底层 CloudFormation 属性（在本例中为角色名称），并在 AWS CDK 堆栈合成过程中使用一个方面将该角色应用于 AWS CDK 应用程序中的所有资源。

工具

AWS 服务

- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，可帮助您在代码中定义和配置 AWS Cloud 基础架构。
- AWS CDK 命令行接口 (AWS CDK CLI)（也称为 AWS CDK Toolkit）是一个命令行云开发套件，可帮助您与 AWS CDK 应用程序进行交互。CLI `cdk` 命令是与您的 AWS CDK 应用程序交互的主要工具。它运行您的应用程序，查询您定义的应用程序模型，并生成和部署由生成的 CloudFormation 模板。 AWS CDK
- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在资源的整个生命周期中跨地区对其 AWS 账户 进行管理。

代码存储库

此模式的源代码和模板可在 GitHub CDK Aspects Override 存储库中找到。

最佳实践

请参阅AWS 规范性指南网站上使用 AWS CDK 中的 TypeScript 创建 IaC 项目的最佳实践。

操作说明

安装 AWS CDK CLI

Task	描述	所需技能
安装 C AWS CDK LI。	要全局安装 AWS CDK CLI，请运行以下命令：	AWS DevOps

Task	描述	所需技能
	`npm install -g aws-cdk`	
验证 版本。	运行命令： `cdk --version` 确认您使用的是 AWS CDK CLI 的版本 2。	AWS DevOps
引导 AWS CDK 环境。	在部署 AWS CloudFormation 模板之前，请准备好要 AWS 区域 使用的账户和账户。运行命令： `cdk bootstrap <account> /<Region>` 有关更多信息，请参阅AWS CDK 文档中的引导。 AWS	AWS DevOps

部署 AWS CDK 应用程序以演示各方面的用法

Task	描述	所需技能
设置项目。	1. 将此模式的 GitHub 存储库克隆到您的本地计算机： `git clone https://github.com/aws-samples/cdk-aspects-override` 2. 导航到本地计算机上的项目目录。 3. 安装项目依赖项：	AWS DevOps

Task	描述	所需技能
	`npm ci`	
使用由分配的默认角色名称部署堆栈。 AWS CDK	部署两个包含 Lambda 函数及其关联角色的 CloudFormation 堆栈（`ExampleStack1` 和`ExampleStack2` ）： `npm run deploy:ExampleAppWithoutAspects` 该代码未明确传递角色属性，因此角色名称将由 AWS CDK. 有关输出示例，请参阅 “其他信息” 部分。	AWS DevOps

Task	描述	所需技能

部署包含各个方面的堆栈。	在此步骤中，您可以通过为 AWS CDK 项目中部署的所有 IAM 角色添加前缀来应用强制执行角色名称约定的方面。该方面在`lib/aspects.ts` 文件中定义。该方面使用逃生舱口通过添加前缀来覆盖角色名称。该坡向应用于`bin/app-with-aspects.ts` 文件中的堆栈。本示例中使用的角色名称前缀是`dev-unicorn` 。
	1. 编辑 `bin/app-with-aspects.ts` 文件。
	2. 在文件中，`ROLE_NAME_PREFIX` 使用`dev-unicorn` 以下前缀更新变量：
	`const app = new cdk.App();`
	`// Define a prefix for the role names`
	`const ROLE_NAME_PREFIX = 'dev-unicorn';`
	`// Instantiate the RoleNamingConventionAspect with the desired prefix`
	`const roleNamingConventionAspect = new RoleNamingConventionAspect(ROLE_NAME_PREFIX);`
		AWS DevOps

Task	描述	所需技能
	3. 通过以下方面部署 AWS CDK 应用程序： `npm run deploy:ExampleAppWithAspects` 有关输出示例，请参阅 “其他信息” 部分。	

清理资源

Task	描述	所需技能
删除您的 AWS CloudFormation 堆栈。	使用完此模式后，运行以下命令清理资源以避免产生额外成本： `cdk destroy --all -f && cdk --app npx ts-node bin/app-with-aspects.ts' destroy --all -f`	AWS DevOps

故障排除

事务	解决方案
您在使用时遇到问题 AWS CDK。	请参阅 AWS CDK 文档中的常见 AWS CDK 问题疑难解答。

相关资源

- AWS Cloud Development Kit (AWS CDK)

- AWS CDK 文档
- AWS CDK on GitHub
- 逃生舱口
- 方面和 AWS CDK

其他信息

由 AWS CloudFormation 不带任何方面创建的角色名称

```
Outputs:
ExampleStack1WithoutAspects.Function1RoleName = example-stack1-without-as-
Function1LambdaFunctionSe-y7FYTY6FXJXA
ExampleStack1WithoutAspects.Function2RoleName = example-stack1-without-as-
Function2LambdaFunctionSe-dDZV4rkWqWnI
...

Outputs:
ExampleStack2WithoutAspects.Function3RoleName = example-stack2-without-as-
Function3LambdaFunctionSe-ygMv49iTyMq0
```

由 AWS CloudFormation 使用方面创建的角色名称

```
Outputs:
ExampleStack1WithAspects.Function1RoleName = dev-unicorn-
Function1LambdaFunctionServiceRole783660DC
ExampleStack1WithAspects.Function2RoleName = dev-unicorn-
Function2LambdaFunctionServiceRole2C391181
...

Outputs:
ExampleStack2WithAspects.Function3RoleName = dev-unicorn-
Function3LambdaFunctionServiceRole4CAA721C
```

使用私有静态在 Amazon EC2 上部署 Cassandra 集群 IPs 以避免重新平衡

由 Dipin Jain (AWS) 编写

摘要

亚马逊弹性计算云 (Amazon EC2) 实例的私有 IP 将在其整个生命周期中保留。然而，私有 IP 可能会在计划内或计划外的系统崩溃期间发生变化，例如在亚马逊机器映像（AMI）升级期间。在某些情况下，保留私有静态 IP 可以提高工作负载的性能并缩短恢复时间。例如为 Apache Cassandra 种子节点使用静态 IP 可以防止集群产生再平衡开销。

此模式描述了如何将辅助弹性网络接口连接到 EC2 实例，以便在重新托管期间保持 IP 静态。该模式侧重于 Cassandra 集群，但你可以将此实现用于任何受益于私有静态 IPs的架构。

先决条件和限制

先决条件

- 有效 Amazon Web Service (AWS) 账户

产品版本

- DataStax 版本 5.11.1
- 操作系统：Ubuntu 16.04.6 LTS

架构

源架构

源可以是本地虚拟机 (VM) 上的 Cassandra 集群，也可以是 AWS 云中的 EC2 实例。下图阐明了第二场景。此示例包括 4 个集群节点：3 个种子节点和 1 个管理节点。在源架构中，每个节点都连接有一个网络接口。

目标架构

目标集群托管在每个节点上都连接了辅助弹性网络 interface 的 EC2 实例上，如下图所示。

自动化和扩缩

您还可以自动将第二个弹性网络接口连接到 A EC2 uto Scaling 群组，如 A WS 知识中心视频中所述。

操作说明

在亚马逊上配置 Cassandra 集群 EC2

Task	描述	所需技能
启动 EC2 节点以托管 Cassandra 集群。	在亚马逊 EC2 控制台上，为你的 AWS 账户中的 Ubuntu 节点启动四个 EC2 实例。三个（种子）节点用于 Cassandra 集群，第四个节点用作集群管理节点，您将在其中安装 DataStax 企业版 (DSE) OpsCenter。有关说明，请参阅 Amazon EC2 文档。	云工程师
确认节点通信。	确保四个节点可通过数据库和集群管理端口相互通信。	网络工程师
在管理节点 OpsCenter 上安装 DSE。	从管理节点上的 Debian 软件包中安装 DSE OpsCenter 6.1。有关说明，请参阅DataStax 文档。	数据库管理员
创建辅助网络接口。	Cassandra 根据每个节点的 EC2 实例 IP 地址为该节点生成一个通用唯一标识符 (UUID)。此 UUID 用于在环上分发虚拟节点 (vnode)。在 EC2 实例上部署 Cassandra 时，会在创建实例时自动为其分配 IP 地址。 如果发生计划内或计划外停机，则新 EC2 实例的 IP 地址会发生变化，数据分布会发生变化，并且必须重新平衡整个环路。这是不可取	云工程师

Task	描述	所需技能
	的。要保留分配的 IP 地址，请使用具有固定 IP 地址的辅助弹性网络接口。 1. 在 Amazon EC2 控制台上，选择网络接口，创建网络接口。 2. 对于子网，选择您在其中创建 EC2 实例的子网。 3. 对于“私有 IPv4 地址”，选择“自动分配”。 4. 在安全组，选择一个安全组，然后选择创建网络接口。 有关创建网络接口的更多信息，请参阅 Amazon EC2 文档。	
将辅助网络接口连接至集群节点。	1. 在 Amazon EC2 控制台上，选择实例。 2. 选中您之前创建的 EC2 实例的复选框。 3. 依次选择操作、联网、附加网络接口。 4. 选择在上一步中创建的网络接口，然后选择附加。 有关连接网络接口的更多信息，请参阅 Amazon EC2 文档。	云工程师

Task	描述	所需技能
在 Amazon 中添加路由 EC2 以解决非对称路由问题。	当您连接第二个网络接口时，网络很可能执行非对称路由。为避免这种情况，您可为新的网络接口添加路由。 有关非对称路由的深入解释和补救措施，请参阅 AWS 知识中心视频或克服多宿主服务器上的非对称路由（Patrick 在 Linux 杂志上发表的文章 McManus，2004 年 4 月 5 日）。	网络工程师
更新 DNS 条目，以指向辅助网络接口 IP。	指定节点的完全限定域名 (FQDN) 指向辅助网络接口的 IP。	网络工程师
使用 DSE OpsCenter 安装和配置 Cassandra 集群。	当集群节点准备好使用辅助网络接口时，您可安装和配置 Cassandra 集群。	数据库管理员

从节点故障中恢复集群

Task	描述	所需技能
为集群种子节点创建 AMI。	对节点进行备份，以便在节点发生故障时可以使用数据库二进制文件恢复它们。有关说明，请参阅 Amazon EC2 文档中的创建 AMI。	备份管理员
从节点故障中恢复	将故障节点替换为从 AMI 启动的新 EC2 实例，并连接故障节点的辅助网络接口。	备份管理员

Task	描述	所需技能
验证 Cassandra 集群是否正常运行。	替换节点启动后，在 DSE 中验证集群运行状况 OpsCenter。	数据库管理员

相关的资源

- 从 Debian 软件包中安装 DSE OpsCenter 6.1（DataStax 文档）
- 如何在 Ubuntu EC2 实例中使用辅助网络接口（AWS 知识中心视频）
- 在@@ 亚马逊上运行 Apache Cassandra 的最佳实践（ EC2AWS 博客文章）

使用 AWS Transit Gateway Connect 扩展 VRFs 到 AWS

由 Adam Till (AWS)、Yashar Araghi (AWS)、Vikas Dewangan (AWS) 和 Mohideen (AWS) 创作 HajaMohideen

摘要

虚拟路由转发（VRF）是传统网络的一个特征。它使用路由表形式的隔离逻辑路由域来分隔同一物理基础设施内的网络流量。当您将本地网络连接到 AWS 时，您可以配置 AWS Transit Gateway 以支持 VRF 隔离。此模式使用示例架构将本地 VRFs 连接到不同的公交网关路由表。

此模式使用 AWS Direct Connect 中的传输虚拟接口 (VIFs) 和传输网关 Connect 附件来扩展 VRFs。中转 VIF 用于访问与 Direct Connect 网关关联的一个或多个 Amazon VPC 中转网关。中转网关 Connect 连接将中转网关与在 VPC 中运行的第三方虚拟设备连接起来。中转网关 Connect 连接支持通用路由封装（GRE）隧道协议以实现高性能，支持边界网关协议（BGP）以实现动态路由。

此模式中描述的方法具有以下优点：

- 使用 Transit Gateway Connect，您可以向 Transit Gateway Connect 对等方通告最多 1,000 条路由，并从中接收最多 5,000 条路由。在不使用 Transit Gateway Connect 的情况下使用 Direct Connect 中转 VIF 功能时，每个中转网关最多可使用 20 个前缀。
- 无论您的客户使用何种 IP 地址架构，您都可以保持流量隔离并使用 Transit Gateway Connect 在 AWS 上提供托管服务。
- VRF 流量不需要遍历公共虚拟接口。这使得许多组织更容易遵守合规性和安全要求。
- 每个 GRE 隧道支持高达 5 Gbps，每个中转网关 Connect 连接最多可以有四个 GRE 隧道。这比许多其他连接类型都要快，例如支持高达 1.25 Gbps 的 AWS Site-to-Site VPN 连接。

先决条件和限制

先决条件

- 所需 Amazon Web Services account 已创建（有关详细信息，请参阅架构）
- 在每个账户中代入 AWS Identity and Access Management (IAM) 角色。
- 每个账户的 IAM 角色必须有权预调配 AWS Transit Gateway 和 AWS Direct Connect 资源。有关更多信息，请参阅中转网关的身份验证和访问控制和 Direct Connect 的身份和访问管理。
- Direct Connect 连接已成功创建。有关更多信息，请参阅使用连接向导创建连接。

限制

- 生产、质量保证和开发账户 VPCs 中的公交网关附件存在限制。有关更多信息，请参阅 VPC 的中转网关连接。
- 创建和使用 Direct Connect 网关是有限制的。有关更多信息，请参阅 AWS Direct Connect 限额。

架构

目标架构

以下示例架构提供了一种可重复使用的解决方案，用于 VIFs 通过公交网关 Connect 附件部署公交。该架构通过使用多个 Direct Connect 位置来提供弹性。有关更多信息，请参阅 Direct Connect 文档中的最大弹性。本地网络包含生产、质量保证和开发，这些内容扩展到 AWS VRFs ，并使用专用路由表进行隔离。

在 AWS 环境中，有两个账户专门用于扩展 VRFs：一个 Direct Connect 账户和一个网络中心账户。Direct Connect 账户包含每台路由器的连接和传输 VIFs 。您可以 VIFs 从 Direct Connect 账户创建传输，但将其部署到网络中心帐户，这样您就可以将它们与网络中心账户中的 Direct Connect 网关关联。网络中心账户包含 Direct Connect 网关和中转网关。AWS 资源连接方式如下：

1. Transit 将直接 VIFs 连接地点的路由器与 Direct Connect 账户中的 AWS Direct Connect 连接起来。
2. 中转 VIF 将 Direct Connect 与网络中心账户中的 Direct Connect 网关连接。
3. 中转网关关联将 Direct Connect 网关与网络中心账户中的中转网关连接起来。
4. Transit gateway Connect 附件将公交网关与生产、质量保证和开发账户连接起来。 VPCs

中转 VIF 架构

下图显示了公交的配置详细信息 VIFs。此示例架构使用 VLAN 为隧道源，但您也可以使用环回。

以下是公交的配置详细信息，例如自治系统编号 (ASNs) VIFs。

资源	Item	Detail

路由器-01	ASN	65534
路由器-02	ASN	65534
路由器-03	ASN	65534
路由器-04	ASN	65534
Direct Connect 网关	ASN	64601
Transit Gateway	ASN	64600
	CIDR 块	10.100.254.0/24

中转网关 Connect 架构

下图和表格介绍了如何通过中转网关 Connect 连接配置单个 VRF。此外 VRFs，请在 CIDR 块内分配唯一的隧道 IDs、传输网关 GRE IP 地址和 BGP。对等 GRE IP 地址与中转 VIF 中的路由器对等 IP 地址相匹配。

下表包含路由器配置详细信息。

路由器	隧道	IP 地址	来源	目标
路由器-01	隧道 1	169.254.101.17	VLAN 60	10.100.254.1
			169.254.100.1	
路由器-02	隧道 11	169.254.101.81	VLAN 61	10.100.254.11
			169.254.100.5	
路由器-03	隧道 21	169.254.101.145	VLAN 62	10.100.254.21
			169.254.100.9	
路由器-04	隧道 31	169.254.101.209	VLAN 63	10.100.254.31
			169.254.100.13	

下表包含中转网关配置详细信息。

隧道	中转网关 GRE IP 地址	对等 GRE IP 地址	CIDR 块内的 BGP
隧道 1	10.100.254.1	VLAN 60 169.254.100.1	169.254.101.16/29
隧道 11	10.100.254.11	VLAN 61 169.254.100.5	169.254.101.80/29
隧道 21	10.100.254.21	VLAN 62 169.254.100.9	169.254.101.144/29
隧道 31	10.100.254.31	VLAN 63 169.254.100.13	169.254.101.208/29

部署

操作说明部分介绍如何在多台客户路由器上部署单个 VRF 的示例配置。步骤 1–5 完成后，您可使用步骤 6–7 为要扩展到 AWS 的每个新 VRF 创建新的中转网关 Connect 连接：

1. 创建中转网关。
2. 为每个 VRF 创建中转网关路由表。
3. 创建中转虚拟接口。
4. 创建 Direct Connect 网关。
5. 使用允许的前缀创建 Direct Connect 网关虚拟接口和网关关联。
6. 创建中转网关 Connect 连接。
7. 创建中转网关 Connect 对等节点。
8. 将中转网关 Connect 连接与路由表相关联。
9. 向路由器传播路由。

工具

Amazon Web Services

- AWS Direct Connect 通过标准的以太网光纤电缆将内部网络链接到 Direct Connect 位置。通过此连接，您可以直接创建连接到公有 Amazon Web Services 的虚拟接口，同时绕过网络路径中的互联网服务提供商。
- AWS Transit Gateway 是一个连接虚拟私有云 (VPCs) 和本地网络的中心枢纽。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。

操作说明

规划架构

Task	描述	所需技能
创建自定义架构图。	1. 在连接部分中，下载逻辑示意图模板。 2. 在微软 Office 中打开所附的图表 PowerPoint。 3. 在架构概述幻灯片上，为您的环境自定义架构图。确定需要扩展到您的 AWS 环境的本地 VRFs 。 4. 在中转 VIF 幻灯片上，自定义架构图。识别路由器、Direct Connect 网关以及中转网关的 AS 号。识别中转 VIF 每一端的 IP 地址。 5. 在中转网关 Connect 幻灯片上，为每个 VRF 自定义架构图。确定配置路由器和中转网关 Connect 对等点所需所有 IP 地址。	云架构师、网络管理员

创建中转网关资源

Task	描述	所需技能
创建中转网关。	1. 登录至网络中心账户。 2. 按照创建中转网关中的说明进行操作。请注意此示例以下几点： • 对于 Amazon 端自治系统号（ASN），请输入唯一的 ASN。就本示例而言，ASN 是 64600。 • 选择 DNS 支持。 • 对于此示例架构，不需要 VPN ECMP 支持、默认路由表关联、默认路由表延迟和组播支持。 • 对于公交网关 CIDR 块，请输入您的传输网关的 IPv4 CIDR 块。就本示例而言，CIDR 块为 `10.100.254.0/24` 。	网络管理员、云架构师
创建中转网关路由表。	按照创建中转网关路由表中的说明进行操作。请注意此示例以下几点： • 对于名称标签，请提供中转网关路由表的名称。我们建议使用与 VRF 对应的名称，例如 `routetable-dev-vrf` 。 • 对于中转网关 ID，请选择您之前创建的中转网关。	云架构师、网络管理员

创建中转虚拟接口

Task	描述	所需技能
创建中转虚拟接口。	1. 登录 Direct Connect 账户。 2. 按照创建到 Direct Connect 网关的中转虚拟接口中的说明进行操作。请注意此示例以下几点： • 对于虚拟接口名称，请输入中转 VIF 的名称。我们建议使用与路由器对应的名称，例如 `transit-vif-router01` 。 • 对于连接，请选择路由器，例如 `router-01` 。 • 对于虚拟接口所有者，请输入网络中心账户的账户 ID。有关说明，请参阅查看您的 Amazon Web Services account ID。 • 对于 Direct Connect 网关，请勿进行任何选择。您将在后续步骤中连接 Direct Connect 网关。 • 对于 VLAN，请输入路由器的 VLAN，例如 60。 • 对于 BGP ASN，请输入路由器的 ASN，例如 65534。 • 在附加设置下，执行以下操作： • 选择 IPv4。 • 对于您的路由器对等 IP，请输入路由器对	云架构师、网络管理员

Task	描述	所需技能
	等体 IP 地址，例如 `169.254.100.1` 。 • 对于 Amazon 路由器对等 IP，请输入 Amazon 路由器对等 IP，例如 `169.254.100.2` 。 • 对于 BGP 身份验证密钥，需要提供密码。如果将此留空，AWS 将创建一个只能在此账户中访问的密钥。 3. 重复这些说明为 VRF 创建所有传输 VIFs 。	

创建 Direct Connect 资源

Task	描述	所需技能
创建 Direct Connect 网关。	1. 登录至网络中心账户。 2. 按照创建 Direct Connect 网关中的说明进行操作。请注意此示例以下几点： • 对于 Amazon 端 ASN，请输入 Direct Connect 网关的 ASN，例如 `64601`。 • 不选择虚拟私有网关。	云架构师、网络管理员
将 Direct Connect 网关连接到公交 VIFs。	1. 在网络中心账户中，在 https://console.aws.amazon.com/directconnect/v2/上打开 AWS Direct Connect 控制台。	云架构师、网络管理员

Task	描述	所需技能
	2. 在导航窗格中，选择 Virtual Interfaces。 3. 选择新的中转 VIF，然后选择接受。 4. 选择您创建的 Direct Connect 网关。 5. 对每个中转 VIF 重复这些说明。	
使用允许的前缀创建 Direct Connect 网关关联。	在网络中心账户中，按照关联中转网关中的说明进行操作。请注意此示例以下几点： • 对于网关，请选择您之前创建的中转网关。 • 对于允许的前缀，请输入分配给中转网关的 CIDR 块，例如 `10.100.254.0/24` 。 创建此关联会自动创建具有 Direct Connect 网关资源类型的中转网关连接。此连接不需要与中转网关路由表关联。	云架构师、网络管理员

Task	描述	所需技能
创建中转网关 Connect 连接。	1. 在网络中心账户中，打开 Amazon VPC 控制台，网址为https://console.aws.amazon.com/vpc/。 2. 在导航窗格中，选择“中转网关连接”。 3. 选择 Create Transit Gateway Attachment（创建中转网关连接）。 4. 对于名称标签，请为连接输入名称。我们建议使用与 VRF 对应的名称，例如 PROD-VRF。 5. 对于中转网关 ID，选择您之前创建的中转网关。 6. 对于 Attachment type（连接类型），选择 Connect（连接）。 7. 对于传输连接 ID，请选择您之前创建的 Direct Connect 网关。 8. 选择 Create Transit Gateway Attachment（创建中转网关连接）。 9. 对要扩展的每个 VRF 重复该步骤。	云架构师、网络管理员

Task	描述	所需技能
创建中转网关 Connect 对等节点。	1. 在网络中心账户中，按照创建 Transit Gateway Connect 对等节点（GRE 隧道）中的说明进行操作。请注意此示例以下几点： • 对于名称标签，为中转网关 Connect 对等节点输入名称。我们建议使用与路由器对应的名称，例如 `connectpeer-router01` 。 • 对于中转网关 GRE 地址，请输入中转网关 CIDR 块中分配的 IP 地址，例如 `10.100.254.1` 。 • 对于对等 GRE 地址，请输入分配给在路由器上为中转 VIF 创建的 VLAN 的 IP 地址，例如 `169.254.100.1` 。如果 AWS 可以访问 IP 地址，您可以使用任何接口（例如 VLAN 或环回）作为对等 GRE 地址。 • 对于 BGP 内部 CIDR 块 (IPv4)，请输入 BGP 内部的 CIDR 块 IP 地址，例如。`169.254.101.16/29` • 对于对等 ASN，请输入路由器的 ASN，例如 65534。	

Task	描述	所需技能
	2. 重复这些说明，为每台路由器创建 GRE 隧道。	

将路由传播到路由器

Task	描述	所需技能
传播路由。	将新的中转网关 Connect 连接与您之前为此 VRF 创建的路由表相关联。例如，将生产中转网关 Connect 连接与 `Production-VRF` 路由表关联。 为通告到路由器的前缀创建静态路由。 1. 登录至网络中心账户。 2. 打开位于 https://console.aws.amazon.com/vpc/ 的 Amazon VPC 控制台。 3. 在导航窗格中，在中转网关下方选择中转网关路由表。 4. 选择 `Production-VRF` 路由表。 5. 在操作菜单上，选择创建静态路由。 6. 对于 CIDR，请输入通往目标 VPC 中中转网关连接的通告路由的 CIDR 块，例如 `10.100.1.0/24` 。 7. 对于选择连接，请选择相关的中转网关 Connect 连接。	网络管理员、云架构师

Task	描述	所需技能
	8. 选择 Create static route（创建静态路由）。	

相关资源

AWS 文档

- Direct Connect 文档
  - 使用 Direct Connect 网关
  - 中转网关关联
  - AWS Direct Connect 虚拟接口
- Transit Gateway 文档
  - 使用中转网关
  - 将中转网关连接到 Direct Connect 网关
  - 中转网关 Connect 连接和中转网关 Connect 对等节点
  - 创建中转网关 Connect 连接

AWS Blog 文章

- 使用 AWS Transit Gateway 连接对混合网络执行分段
- 使用 AWS Transit Gateway 连接来扩展 VRFs 和增加 IP 前缀广告

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

当 AWS KMS 密钥的密钥状态发生变化时获取 Amazon SNS 通知

创建者：Shubham Harsora (AWS)、Aromal Raj Jayarajan (AWS) 和 Navdeep Pareek (AWS)

摘要

删除与 AWS Key Management Service（AWS KMS）密钥关联的数据和元数据后，该密钥将丢失。删除是不可逆的，您无法恢复丢失的数据（包括加密数据）。您可以设置通知系统，提醒您 AWS KMS 密钥的密钥状态发生更改，从而防止数据丢失。

此模式向您展示如何监控 AWS KMS 密钥的状态变化，方法是使用亚马逊 EventBridge 和亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 在 AWS KMS 密钥的密钥状态更改为或时自动发出通知。`Disabled` `PendingDeletion`例如，如果用户尝试禁用或删除 AWS KMS 密钥，您将收到一封电子邮件通知，其中包含有关尝试更改状态的详细信息。您也可以使用此模式安排删除 AWS KMS 密钥。

先决条件和限制

先决条件

- 一个拥有 AWS Identity and Access Management (IAM) 用户的有效 Amazon Web Services account
- 一个 AWS KMS 密钥

架构

技术堆栈

- Amazon EventBridge
- AWS Key Management Service（AWS KMS）
- Amazon Simple Notification Service (Amazon SNS)

目标架构

下图显示了一种架构，用于构建自动监控和通知流程，以检测 AWS KMS 密钥状态的任何更改。

图表显示了以下工作流：

1. 用户禁用或计划删除 AWS KMS 密钥。
2. EventBridge 规则评估计划`Disabled`或`PendingDeletion`事件。
3. 该 EventBridge 规则调用了 Amazon SNS 主题。
4. Amazon SNS 会向用户发送一封电子邮件通知消息。

> **Note**
>
> 您可以自定义电子邮件以满足组织的需求。我们建议包括有关使用 AWS KMS 密钥的实体的信息。这可以帮助用户了解删除 AWS KMS 密钥的影响。您还可以安排在 AWS KMS 密钥删除前一两天发送提醒电子邮件通知。

自动化和扩缩

AWS CloudFormation 堆栈部署了所有必要的资源和服务，以使这种模式发挥作用。您可以在单个账户中独立实施该模式，也可以将 AWS 用 CloudFormation StackSets于 AWS Org anizations 中的多个独立账户或组织单位。

工具

- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和 AWS 区域的整个生命周期中对其进行管理。此模式的 CloudFormation 模板描述了您需要的所有 AWS 资源，并 CloudFormation 为您预置和配置这些资源。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。 EventBridge 提供来自您自己的应用程序和 AWS 服务的实时数据流，并将这些数据路由到目标，例如 AWS Lambda。 EventBridge 简化了构建事件驱动架构的过程。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥，以帮助保护您的数据。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。

代码

此模式的代码可在 GitHub 监控 AWS KMS 密钥禁用和计划删除存储库中找到。

操作说明

部署 CloudFormation 模板

Task	描述	所需技能
克隆存储库。	运行以下命令，将 Mon GitHub itor AWS KMS 密钥禁用和计划删除存储库克隆到本地计算机： `git clone https://github.com/aws-samples/aws-kms-deletion-notification`	AWS 管理员、云架构师
更新模板的参数。	在代码编辑器中，打开您从存储库中克隆的`Alerting-KMS-Events.yaml` CloudFormation 模板，然后更新以下参数： • 对于 `DestinationEmailAddress`，请输入您计划用于接收 SNS 通知的有效电子邮件地址。 • 对于 `SNSTopicName`，请输入 SNS 主题的名称。	AWS 管理员、云架构师
部署 CloudFormation 模板。	1. 登录 AWS 管理控制台并打开CloudFormation 控制台。 2. 在导航窗格中，选择创建堆栈，然后选择使用新资源（标准）。 3. 在标识资源页面上，选择下一步。	AWS 管理员、云架构师

Task	描述	所需技能
	4. 在指定模板页面，对于模板来源，选择上传模板文件。 5. 选择 “选择文件”，从克隆的 GitHub 存储库中选择`Alerting-KMS-Events.yaml` 文件，然后选择 “下一步”。 6. 在堆栈名称中，输入堆栈名称。 7. 选择提交。	

确认订阅

Task	描述	所需技能
确认订阅电子邮件。	CloudFormation 模板成功部署后，Amazon SNS 会向您在模板中 CloudFormation 提供的电子邮件地址发送订阅确认消息。 要接收通知，您必须确认此电子邮件订阅。有关更多信息，请参阅 Amazon SNS 开发人员指南中的确认订阅。	AWS 管理员、云架构师

测试订阅通知

Task	描述	所需技能
禁用 AWS KMS 密钥。	1. 登录 AWS 管理控制台，然后打开 AWS KMS 控制台。	AWS 管理员

Task	描述	所需技能
	2. 要更改区域，请选择当前显示的区域的名称，然后选择要切换到的区域。 3. 在导航窗格中，选择客户托管密钥。 4. 勾选要启用或禁用的 AWS KMS 密钥对应的复选框。 5. 要禁用 AWS KMS 密钥，请依次选择密钥操作、禁用。	
验证订阅。	确认您收到 Amazon SNS 通知电子邮件。	AWS 管理员

清理资源

Task	描述	所需技能
删除 CloudFormation 堆栈。	1. 登录 AWS 管理控制台并打开CloudFormation 控制台。 2. 在导航窗格中，选择 Stacks（堆栈）。 3. 选择您之前创建的堆栈，然后选择删除。	AWS 管理员

相关资源

- AWS CloudFormation（AWS 文档）
- 在 AWS CloudFormation 控制台上创建堆栈（AWS CloudFormation 文档）
- 在 AWS 上构建事件驱动型架构（AWS 研习会参与平台 Workshop Studio 文档）
- AWS Key Management Service（AWS KMS）最佳实践（AWS 白皮书）
- AWS Key Management Service（AWS KMS）的安全最佳实践（AWS KMS 开发人员指南）

其他信息

默认情况下，Amazon SNS 会在传输过程中提供加密。为了与安全最佳实践保持一致，您还可以使用 AWS KMS 客户托管密钥为 Amazon SNS 启用服务器端加密。

在非工作负载子网的多账户 VPC 设计中保留可路由的 IP 空间

由 Adam Spicer (AWS) 编写

摘要

Amazon Web Services (AWS) 已发布最佳实践，建议在虚拟私有云（VPC）中为中转网关连接和网关负载均衡器端点（以支持 AWS Network Firewall 或第三方设备）使用专用子网。这些子网用于包含这些服务的弹性网络接口。如果您同时使用 AWS Transit Gateway 和网关负载均衡器，则会在 VPC 的每个可用区中创建两个子网。由于设计方式 VPCs ，这些额外的子网不能小于 /28 掩码，并且会消耗宝贵的可路由 IP 空间，而这些空间本来可以用于可路由的工作负载。此模式演示了如何为这些专用子网使用辅助的、不可路由的无类别域间路由 (CIDR) 范围，以帮助保留可路由的 IP 空间。

先决条件和限制

先决条件

- 可路由 IP 空间的多 VPC 策略
- 您正在使用的服务（中转网关连接和网关负载均衡器或 Network Firewall 端点）的不可路由 CIDR 范围

架构

目标架构

此模式包括两个参考架构：一种架构具有用于传输网关 (TGW) 连接的子网和一个 Gateway Load Balancer 端点 (GWLBe)，第二个架构具有仅用于 TGW 附件的子网。

架构 1 - 连接 TGW 的 VPC，具有到设备的入口路由

下图展示了跨两个可用区的 VPC 的参考架构。在入口处，VPC 使用入口路由模式将发往公有子网的流量引导到bump-in-the-wire 设备进行防火墙检查。TGW 连接支持从私有子网到单独的 VPC 的出口。

此模式对 TGW 连接子网和子网使用不可路由的 CIDR 范围。 GWLBe 在 TGW 路由表中，使用一组更具体的路由，将此不可路由的 CIDR 配置为黑洞（静态）路由。如果路由要传播到 TGW 路由表，则将应用这些更具体的黑洞路由。

在此示例中，/23 可路由 CIDR 被划分并完全分配给可路由子网。

架构 2 — 连接 TGW 的 VPC

下图展示了另一个跨可用区的 VPC 的参考架构。TGW 连接支持从私有子网到单独的 VPC 的出站。它仅对 TGW 连接子网使用不可路由 CIDR 范围。在 TGW 路由表中，使用一组更具体的路由，将此不可路由的 CIDR 配置为黑洞路由。如果路由要传播到 TGW 路由表，则将应用这些更具体的黑洞路由。

在此示例中，/23 可路由 CIDR 被划分并完全分配给可路由子网。

工具

Amazon Web Services 和资源

- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。在这种模式中，VPC 辅助 CIDRs 用于在工作负载 CIDRs中保留可路由 IP 空间。
- 互联网网关入口路由（边缘关联）可以与网关负载均衡器端点一起使用，用于专用的不可路由子网。
- AWS Transit Gateway 是一个连接 VPCs 本地网络的中心枢纽。在这种模式中 VPCs ，集中连接到传输网关，传输网关附件位于不可路由的专用子网中。
- 网关负载均衡器可帮助您部署、扩展和管理虚拟设备，例如防火墙、入侵检测和防御系统以及深度数据包检测系统。网关充当所有流量的单一入口和出口点。在这种模式下，网关负载均衡器的端点可以在不可路由的专用子网中使用。
- AWS Network Fire wall 是一项有状态的托管式网络防火墙以及入侵检测和防御服务，适用于 AWS 云 VPCs 中。在此模式中，防火墙的端点可以在专用的不可路由子网中使用。

代码存储库

此模式的运行手册和 AWS CloudFormation 模板可在 GitHub 不可路由的辅助 CIDR 模式存储库中找到。您可使用示例文件在您的环境中设置工作实验室。

最佳实践

AWS Transit Gateway

- 为每个中转网关 VPC 附件使用单独的子网。
- 从辅助不可路由 CIDR 范围中为中转网关连接子网分配 /28 子网。
- 在每个中转网关路由表中，为不可路由的 CIDR 范围添加一条更具体的静态路由作为黑洞。

网关负载均衡器与入口路由

- 使用入口路由将流量从互联网引导至网关负载均衡器端点。
- 为每个网关负载均衡器端点使用单独子网。
- 从网关负载均衡器端点子网辅助不可路由 CIDR 范围中分配一个 /28 子网。

操作说明

创建 VPCs

Task	描述	所需技能
确定不可路由的 CIDR 范围。	确定一个不可路由的 CIDR 范围，该范围将用于中转网关连接子网以及（可选）用于任何网关负载均衡器或 Network Firewall 端点子网。此 CIDR 范围用作 VPC 的辅助 CIDR。它不得从 VPC 的主要 CIDR 范围或更大的网络进行路由。	云架构师
确定可路由的 CIDR 范围。VPCs	确定一组将用于您的可路由 CIDR 范围。 VPCs此 CIDR 范围将用作您的主 CIDR。 VPCs	云架构师
创建 VPCs。	创建您的 VPCs 并将它们连接到传输网关。根据您在前两个步骤中确定的范围，每个 VPC 都应具有可路由的主要 CIDR 范围和不可路由的辅助 CIDR 范围。	云架构师

配置中转网关黑洞路由

Task	描述	所需技能
创建更具体的不可路由的黑洞 CIDRs 。	每个中转网关路由表都需要为不可 CIDRs路由的路由创建一	云架构师

Task	描述	所需技能
	组黑洞路由。这些配置可确保来自辅助 VPC CIDR 的任何流量都不可路由，并且不会泄漏到更大的网络中。这些路由应比 VPC 上设置为辅助 CIDR 的不可路由 CIDR 更具体。例如，如果辅助不可路由的 CIDR 为 100.64.0.0/26，则中转网关路由表中的黑洞路由应为 100.64.0.0/27 和 100.64.0.32/27。	

相关资源

- 部署网关负载均衡器的最佳实践
- 使用网关负载均衡器的分布式检查架构
- Networking Immersion Day – 互联网到 VPC 防火墙实验室
- 中转网关设计最佳实践

其他信息

在处理需要大量 IP 地址的大规模容器部署时，不可路由辅助 CIDR 范围也很有用。您可将此模式与私有 NAT 网关配合使用，从而使用不可路由的子网来托管您的容器部署。有关更多信息，请参阅博客帖子如何使用私有 NAT 解决方案解决私有 IP 耗尽问题。

使用代码存储库在中 AWS Service Catalog 配置 Terraform 产品

由 Rahul Sharad Gaikwad 博士 (AWS) 和 Tamilselvan P (AWS) 编写

摘要

AWS Service Catalog 支持自助服务配置，并管理您的 HashiCorp Terraform 配置。如果您使用 Terraform，则可以使用 Service Catalog 作为单一工具，在其中大规模组织、管理和分发 Terraform 配置。 AWS 您可以访问 Service Catalog 的主要功能，包括对标准化和预先批准的基础架构即代码 (IaC) 模板进行编目、访问控制、使用最低权限访问的云资源配置、版本控制、与数 AWS 账户千人共享以及标记。最终用户（例如工程师、数据库管理员和数据科学家）会看到他们有权访问的产品和版本列表，他们可以通过一个操作进行部署。

此模式可帮助您使用 Terraform 代码部署 AWS 资源。 GitHub 存储库中的 Terraform 代码可通过 Service Catalog 进行访问。使用这种方法，您可以将产品与现有的 Terraform 工作流程集成。管理员可以使用 Terraform 创建 Service Catalog AWS Launch Wizard 产品组合并向其中添加产品。

以下是此解决方案的好处：

- 由于 Service Catalog 中具有回滚功能，因此如果在部署过程中出现任何问题，则可以将产品恢复到以前的版本。
- 您可以轻松识别产品版本之间的差异。这可以帮助您在部署期间解决问题。
- 您可以在 Service Catalog 中配置存储库连接，例如到 GitHub 或 GitLab。您可以直接通过存储库进行产品更改。

有关总体优势的信息 AWS Service Catalog，请参阅什么是 Service Catalog。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- GitHub BitBucket、或其他包含 ZIP 格式的 Terraform 配置文件的存储库。
- AWS Serverless Application Model 已安装命令行界面 (AWS SAM CLI)。
- AWS Command Line Interface (AWS CLI)，已安装并配置。
- 去吧，安装好了。
- Python 版本 3.9，已安装。 AWS SAM CLI 需要这个版本的 Python。
- 编写和运行 AWS Lambda 函数的权限以及访问和管理 Service Catalog 产品和产品组合的权限。

架构

图表显示了以下工作流：

1. 当 Terraform 配置准备就绪后，开发人员会创建一个包含所有 Terraform 代码的.zip 文件。开发者将.zip 文件上传到连接到 Service Catalog 的代码存储库中。
2. 管理员将 Terraform 产品与 Service Catalog 中的产品组合相关联。管理员还会创建启动约束，允许最终用户配置产品。
3. 在 Service Catalog 中，最终用户使用 Terraform 配置启动 AWS 资源。他们可以选择要部署的产品版本。

工具

AWS 服务

- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Service Catalog帮助您集中管理已获批准的 IT 服务目录。 AWS最终用户可在遵循组织设定约束的情况下快速部署他们所需已获得批准的 IT 服务。

其他服务

- Go 是谷歌支持的开源编程语言。
- Python 是通用的计算机编程语言。

代码存储库

如果您需要可通过 Service Catalog 部署的 Terraform 示例配置，则可以使用 Amazon Mac GitHub ie 组织设置使用 Terraform 存储库中的配置。不需要使用此存储库中的代码示例。

最佳实践

- 在通过 Service Catalog 启动产品时配置变量值，而不是在 Terraform 配置文件 (`terraform.tfvars`) 中为变量提供值。
- 仅向特定用户或管理员授予对产品组合的访问权限。

- 遵循最低权限原则，授予执行任务所需的最低权限。有关更多信息，请参阅 AWS Identity and Access Management (IAM) 文档中的授予最低权限和安全最佳实践。

操作说明

设置本地工作站

Task	描述	所需技能
（可选）安装 Docker。	如果要在开发环境中运行这些 AWS Lambda 函数，请安装 Docker。有关说明，请参阅 Docker 文档中的安装 Docker 引擎。	DevOps 工程师

为 Terraform 安装 AWS Service Catalog 引擎。	1. 输入以下命令来克隆 Terraform AWS Service Catalog 引擎存储库。
	`git clone https://github.com/aws-samples/service-catalog-engine-for-terraform-os.git`
	2. 导航到克隆存储库的根目录。
	3. 输入以下命令。这将安装引擎。
	`run ./bin/bash/deploy-tre.sh -r`
	自动安装期间不使用默认配置文件中 AWS 区域 设置的内容。相反，您可以在运行此命令时提供区域。
		DevOps 工程师，AWS 管理员

Connect GitHub 存储库

Task	描述	所需技能
创建与 GitHub 存储库的连接。	1. 登录 AWS Management Console，然后打开开发者工具控制台。您可以通过选择诸如 AWS CodePipeline 或之类的服务来访问开发者工具控制台 AWS CodeDeploy。 2. 在左侧导航窗格中，选择“设置”，然后选择“连接”。 3. 选择创建连接。 4. 选择用于维护 Terraform 源代码的存储库。例如，您可以选择 Bitbucket GitHub、或GitHub 企业服务器。 5. 输入连接的名称，然后选择 Connect。 6. 当系统提示您时，请对存储库进行身份验证。 身份验证完成后，连接即创建完毕，状态更改为活动。	AWS 管理员

在 Service Catalog 中创建 Terraform 产品

Task	描述	所需技能
创建 Service Catalog 产品。	1. 打开 AWS Service Catalog 管理控制台。 2. 导航到管理部分，然后选择产品列表。 3. 选择创建产品。	AWS 管理员

Task	描述	所需技能
	4. 在产品详情部分的创建产品页面上，选择外部产品类型。Service Catalog 使用此产品类型来支持 Terraform 社区版产品。 5. 输入 Service Catalog 产品的名称和所有者。 6. 选择 “使用 CodeStar 提供程序指定您的代码存储库”。 7. 为您的存储库输入以下信息： • 使用连接您的提供商 AWS CodeConnections — 选择您之前创建的连接。 • 存储库-选择存储库。 • 分支-选择分支。 • 模板文件路径-选择存储代码模板文件的路径。文件名应以结尾`tar.gz`。 8. 在 “版本名称和描述” 下，提供有关产品版本的信息。 9. 选择创建产品。	

Task	描述	所需技能
创建产品组合。	1. 打开 AWS Service Catalog 管理控制台。 2. 导航到 “管理” 部分，然后选择，选择 “电子档案夹”。 3. 选择创建产品组合。 4. 输入以下值： • 产品组合名称 – `Sample terraform` • 产品组合描述 — `Sample portfolio for Terraform configurations` • 所有者-您的联系信息，例如电子邮件地址 5. 选择创建。	AWS 管理员
将 Terraform 产品添加到产品组合中。	1. 打开 AWS Service Catalog 管理控制台。 2. 导航到管理部分，然后选择产品列表。 3. 选择您之前创建的 Terraform 产品。 4. 选择 “操作”，然后选择 “将产品添加到产品组合”。 5. 选择`Sample terraform` 投资组合。 6. 选择将产品添加到产品组合。	AWS 管理员

Task	描述	所需技能
创建访问策略。	1. 打开 AWS Identity and Access Management (IAM) 控制台。 2. 在导航窗格中，选择策略。 3. 在内容窗格中，选择创建策略。 4. 选择 JSON 选项。 5. 在此模式的 “其他信息” 部分的 “访问策略” 中输入示例 JSON 策略。 6. 选择下一步。 7. 在 “查看并创建” 页面的 “策略名称” 框中，输入`TerraformResourceCreationAndArtifactAccessPolicy` 。 8. 选择创建策略。	AWS 管理员

Task	描述	所需技能
创建自定义信任策略。	1. 打开 IAM 控制台。 2. 在导航窗格中，选择角色。 3. 选择 Create role（创建角色）。 4. 在“可信实体类型”下，选择“自定义信任策略”。 5. 在 JSON 策略编辑器中，在此模式的“其他信息”部分的“信任策略”中输入示例 JSON 策略。 6. 选择下一步。 7. 在“权限策略”下`TerraformResourceCreationAndArtifactAccessPolicy`，选择您之前创建的。 8. 选择下一步。 9. 在“角色详细信息”下的“角色名称”框中输入`SCLaunch-product`。 **Important** 角色名称必须以开头`SCLaunch`。 10选择 Create role（创建角色）。	AWS 管理员

Task	描述	所需技能
向 Service Catalog 产品添加启动约束。	1. 以具有管理权限的用户身份登录。 AWS Management Console 2. 打开 AWS Service Catalog 管理控制台。 3. 在导航窗格中，选择投资组合。 4. 选择您之前创建的投资组合。 5. 在产品组合详细信息页面上，选择约束选项卡，然后选择创建约束。 6. 对于产品，请选择您之前创建的 Terraform 产品。 7. 在 “启动约束” 下，对于 “方法”，选择 “输入角色名称”。 8. 在 “角色名称” 框中输入`SCLaunch-product` 。 9. 选择创建。	AWS 管理员

Task	描述	所需技能
授予对产品的访问权限。	1. 打开 AWS Service Catalog 管理控制台。 2. 在导航窗格中，选择投资组合。 3. 选择您之前创建的投资组合。 4. 选择 “访问权限” 选项卡，然后选择 “授予访问权限”。 5. 选择 “角色” 选项卡，然后选择应有权部署此产品的角色。 6. 选择 Grant access (授予访问权限)。	AWS 管理员
启动该产品。	1. 以有权部署 Service Catalog 产品的用户 AWS Management Console 身份登录。 2. 打开 AWS Service Catalog 管理控制台。 3. 在导航窗格中，选择产品。 4. 选择您之前创建的农产品，然后选择 Launch 产品。 5. 输入产品名称并定义任何必需的参数。 6. 选择启动产品。	DevOps 工程师

验证部署

Task	描述	所需技能
验证部署。	Service Catalog 配置工作流程有两台 AWS Step Functions 状态机： • `ManageProvisionedProductStateMachine` — 在配置新的 Terraform 产品和更新现有 Terraform 预配置产品时，Service Catalog 会调用此状态机。 • `TerminateProvisionedProductStateMachine` — 在终止现有 Terraform 预配置产品时，Service Catalog 会调用此状态机。 您可以检查`ManageProvisionedProductStateMachine` 状态机的日志，以确认产品已配置。 1. 登录 AWS Management Console，然后打开AWS Step Functions 控制台。 2. 在左侧导航窗格中，选择状态机。 3. 选择 `ManageProvisionedProductStateMachine` 。 4. 在 “执行” 列表中，输入预配置的产品 ID 以定位执行。	DevOps 工程师

Task	描述	所需技能
	Note 状态文件后端存储桶名称以开头`sc-terraform-engine-state-`。 5. 验证账户中是否已创建所有必需的资源。	

清理基础设施

Task	描述	所需技能
删除预配置的产品。	1. 以有权部署 Service Catalog 产品的用户 AWS Management Console 身份登录。 2. 打开 AWS Service Catalog 管理控制台。 3. 在左侧导航栏中，选择预配置产品。 4. 选择您创建的产品。 5. 在 “操作” 列表中，选择 “终止”。 6. 在确认文本框中，输入`terminate`，然后选择终止预配置的产品。 7. 重复这些步骤以终止所有预配置的产品。	DevOps 工程师

Task	描述	所需技能
移除 Terraform 的 AWS Service Catalog 引擎。	1. 以具有管理权限的用户身份登录。 AWS Management Console 2. 打开亚马逊简单存储服务 (Amazon S3) S ervice 控制台。 3. 在导航窗格中，选择存储桶。 4. 选择`sc-terraform-engine-logging-XXXX`存储桶。 5. 选择 “空”。 6. 对以下存储桶重复步骤 4—5： • `sc-terraform-engine-state-XXXX` • `terraform-engine-bootstrap-XXXX` 7. 打开AWS CloudFormation 控制台，然后验证您的操作是否正确 AWS 区域。 8. 在左侧导航栏中，选择堆栈。 9. 选择`SAM-TRE`，然后选择 “删除”。等到堆栈被删除。 10选择`Bootstrap-TRE` ，然后选择 “删除”。等到堆栈被删除。	AWS 管理员

相关资源

AWS 文档

- 开始使用 Terraform 产品

Terraform 文档

- Terraform installation（Terraform 安装）
- Terraform 后端配置
- Terraform AWS 提供者文档

其他信息

访问政策

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:ExistingObjectTag/servicecatalog:provisioning": "true"
                }
            }
        },
        {
            "Action": [
                "s3:CreateBucket*",
                "s3:DeleteBucket*",
                "s3:Get*",
                "s3:List*",
                "s3:PutBucketTagging"
            ],
            "Resource": "arn:aws:s3:::*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "resource-groups:CreateGroup",
                "resource-groups:ListGroupResources",
```

```
                "resource-groups:DeleteGroup",
                "resource-groups:Tag"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "tag:GetResources",
                "tag:GetTagKeys",
                "tag:GetTagValues",
                "tag:TagResources",
                "tag:UntagResources"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

信任策略

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "GivePermissionsToServiceCatalog",
            "Effect": "Allow",
            "Principal": {
                "Service": "servicecatalog.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        },
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::account_id:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringLike": {
                    "aws:PrincipalArn": [
```

```
                        "arn:aws:iam::accounti_id:role/TerraformEngine/
TerraformExecutionRole*",
                        "arn:aws:iam::accounti_id:role/TerraformEngine/
ServiceCatalogExternalParameterParserRole*",
                        "arn:aws:iam::accounti_id:role/TerraformEngine/
ServiceCatalogTerraformOSParameterParserRole*"
                    ]
                }
            }
        }
    ]
}
```

使用 Amazon SES 通过单个电子邮件地址注册多个 Amazon Web Services account

由 Joe Wozniak(AWS) 和 Shubhangi Vishwakarma(AWS) 编写

摘要

此模式描述了如何将真实的电子邮件地址与与之关联的电子邮件地址分开。 AWS 账户 AWS 账户 要求在创建账户时提供唯一的电子邮件地址。在某些组织中，管理团队 AWS 账户 必须承担与其邮件团队一起管理许多唯一电子邮件地址的负担。对于管理许多企业的大型组织来说，这可能很困难 AWS 账户。此外，如果您的电子邮件系统不允许 Sieve Email Filtering：子地址扩展 (RFC 5233) 中定义的加号寻址或子寻址，则通过在电子邮件地址的本地部分的末尾添加加号 (+) 和标识符，例如，此模式可以帮助克服`admin+123456789123@example.com`此限制。

这种模式提供了一种独特的电子邮件地址自动售货解决方案，使 AWS 账户 所有者能够将一个电子邮件地址与多个电子邮件地址相关联 AWS 账户。然后， AWS 账户 所有者的真实电子邮件地址与表格中生成的这些电子邮件地址相关联。该解决方案处理唯一电子邮件账户的所有传入电子邮件，查找每个账户的拥有者，然后将收到的所有邮件转发给拥有者。

先决条件和限制

先决条件

- 对的管理访问权限 AWS 账户。
- 可以进入开发环境。
- （可选）熟悉 AWS Cloud Development Kit (AWS CDK) 工作流程和 Python 编程语言将帮助您解决任何问题或进行修改。

限制

- 出售的电子邮件地址的总长度为 64 个字符。有关详细信息，请参阅 AWS Organizations API 参考CreateAccount中的。

产品版本

- Node.js 版本 12.7.0 或更高版本
- Python 3.9 或更高版本
- Python 软件包 pip 和 virtualenv

- AWS CDK 版本 2.23.0 或更高版本
- Docker 20.10.x 或更高版本

架构

目标技术堆栈

- AWS CloudFormation 堆栈
- AWS Lambda 函数
- 亚马逊简单电子邮件服务 (Amazon SES) Service 规则和规则集
- AWS Identity and Access Management (IAM) 角色和策略
- Amazon Simple Storage Service (Amazon S3) 存储桶和存储桶策略
- AWS Key Management Service (AWS KMS) 密钥和密钥策略
- Amazon Simple Notification Service(Amazon SNS) 主题和主题策略
- Amazon DynamoDB 表

目标架构

此图显示了两个流程：

- 电子邮件地址自动售货流程：在图中，电子邮件地址自动售货流程（下部分）通常从账户自动售货解决方案或外部自动化开始，或者手动调用。在请求中，使用包含所需元数据的负载调用 Lambda 函数。该函数使用此信息生成唯一的账户名和电子邮件地址，将其存储在 DynamoDB 数据库，然后将值返回给调用方。然后，可以使用这些值来创建新值 AWS 账户 （通常通过使用 AWS Organizations）。
- 电子邮件转发流程：此流程如上图的上半部分所示。使用电子邮件 AWS 账户 地址自动售货流程生成的账户电子邮件创建时， AWS 会向该电子邮件地址发送各种电子邮件，例如账户注册确认和定期通知。按照此模式中的步骤操作，您可以将 with Amazon SES 配置为接收整个域的电子邮件。AWS 账户 此解决方案配置了转发规则，允许 Lambda 处理所有传入的电子邮件，检查该 T0 地址是否在 DynamoDB 表中，然后将邮件转发到账户拥有者的电子邮件地址。使用此流程，账户拥有者可以将多个账户与一个电子邮件地址相关联。

自动化和扩缩

此模式使用 AWS CDK 来完全自动化部署。该解决方案使用 AWS 托管服务，这些服务将（或可以配置为）自动扩展以满足您的需求。Lambda 函数可能需要额外配置才能满足您的扩缩需求。有关更多信息，请参阅 Lambda 文档中的了解 Lambda 函数缩放。

工具

Amazon Web Services

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在资源的整个生命周期中跨地区对其 AWS 账户 进行管理。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您通过命令行外壳中的命令与 AWS 服务进行交互。
- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Key Management Service (AWS KMS) 帮助您创建和控制加密密钥以帮助保护您的数据。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Email Service (Amazon SES) 帮助您通过使用您自己的电子邮件地址和域发送和接收电子邮件。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

部署所需工具

- 具有 AWS CLI 和 IAM 访问权限的开发环境 AWS 账户。 有关详细信息，请参阅 “相关资源” 部分中的链接。
- 在您的开发系统上，安装以下内容：
  - Git 命令行工具，可从 Git 下载网站上获得。
  - 用于 AWS CLI 为配置访问凭证 AWS CDK。有关更多信息，请参阅 AWS CLI 文档。
  - Python 版本 3.9 或更高版本，可从 Python 下载网站上获得。

- Python 打包 pip 和 vir tualenv。有关安装说明，请参阅 pip 文档和 virtualen v 文档。
- Node.js 版本 12.7.0 或更高版本。有关安装说明，请参阅 Node.js 文档。
- AWS CDK 版本 2.23.0 或更高版本。有关安装说明，请参阅AWS CDK 文档。
- Docker 版本 20.10.x 或更高版本。有关安装说明，请参阅 Docker 文档。

代码

此模式的代码可在 GitHub AWS 账户 工厂电子邮件存储库中找到。

操作说明

分配目标部署环境

Task	描述	所需技能
识别或创建 AWS 账户.	确定您拥有完全管理权限 AWS 账户 的现有或新的电子邮件解决方案，以部署电子邮件解决方案。	AWS 管理员、云管理员
设置部署环境。	按照以下步骤配置易于使用的部署环境并设置依赖项： 1. 使用 “工具” 部分中列出的工具设置您的开发环境。 2. 使用以下命令将 GitHub AWS 账户 工厂电子邮件存储库代码库克隆到您的开发环境中： `git clone https://github.com/aws-samples/aws-account-factory-email` 3. 在`requirements.txt` 文件中（在存储库的根目录中），更新`aws-cdk-l`	AWS DevOps，应用程序开发者

Task	描述	所需技能
	ib== 以开头的行，使其与您的环境中运行的版本相匹配。 AWS CDK 要识别版本，请使用 cdk --version 命令。	

设置验证的域

Task	描述	所需技能
识别和分配域。	电子邮件转发功能需要专用域。确定并分配一个可以通过 Amazon SES 验证的域或子域。此域应可用于在部署电子邮件转发解决方案 AWS 账户的地方接收传入的电子邮件。 域要求： • 该域应该是标准域或子域。 • 该域名应可在外部 DNS 解析，因为它将用于接收来自组织外部的电子邮件。	云管理员、网络管理员、DNS 管理员
验证域。	验证所识别的域是否可用于接受传入电子邮件。 完成 Amazon SES 文档中的验证您的域以接收 Amazon SES 电子邮件中的说明。这需要与负责域的 DNS 记录人员或团队进行协调。	AWS 应用程序开发人员 DevOps
设置 MX 记录。	使用指向您和地区的 Amazon SES 终端节点的 MX 记录来设置您的 AWS 账户 域名。有	云管理员、网络管理员、DNS 管理员

Task	描述	所需技能
	关更多信息，请参阅 Amazon SES 文档中的发布 Amazon SES 电子邮件接收的 MX 记录。	

部署电子邮件自动售卖和转发解决方案

Task	描述	所需技能
修改中的默认值cdk.json。	编辑 cdk.json 文件中的一些默认值（位于存储库的根目录），以便解决方案在部署后能够正常运行。 1. 修改 SES_DOMAIN_NAME 值以匹配您之前验证的域名。 2. 修改 ADDRESS_FROM 值以包含中的相同域 SES_DOMAIN_NAME 。地址的本地部分应由您的云团队决定。该地址将成为通过解决方案转发的每封电子邮件的 FROM 地址。 3. 修改 ADDRESS_ADMIN 值，以匹配任何不匹配的传入邮件将被转发到的电子邮件地址。此值必须是有效的电子邮件地址。	AWS 应用程序开发人员 DevOps
部署电子邮件自动售卖与转发解决方案。	1. 创建 Python 虚拟环境： `python -m venv .venv` 2. 激活 Python 虚拟环境：	AWS 应用程序开发人员 DevOps

Task	描述	所需技能

	`source .venv/bin/activate`
	
	或者在 Windows 平台上，使用：
	
	`% .venv\Scripts\activate.bat`
	
	3. 准确无误安装所有 Python 要求：
	
	`pip install -r requirements.txt`
	
	4. 合成 CloudFormation 模板：
	
	`cdk synth`
	
	确认没有错误，并且完整 CloudFormation 模板包含预期的输出。
	
	5. （可选）如果您是首次将 AWS CDK 代码部署到当前 AWS 账户 或区域，请引导环境。有关更多信息，请参阅AWS CDK 文档中的引导。 AWS CDK
	
	`cdk bootstrap aws://AWS-ACCOUNT-NUMBER/REGION`	

Task	描述	所需技能

	将 AWS-ACCOUNT-NUMBER 和 REGION 替换为实际值。
	6. 部署解决方案：
	`cdk bootstrap`
	`cdk deploy`
	命令应该没有错误地完成。	

验证解决方案是否已部署。	开始测试之前，验证解决方案是否成功部署： 1. 打开AWS CloudFormation 控制台并查找包含该名称的 CloudFormation 堆栈AwsMailFwdStack 。 2. 确认 AwsMailFwdStack 堆栈具有以下资源： • Lambda 函数 • Amazon SES 规则和规则集 • IAM 角色和策略 • Amazon S3 存储桶和存储桶策略 • AWS KMS 密钥和密钥策略 • Amazon SNS 主题和主题策略 • DynamoDB 表	AWS 应用程序开发人员 DevOps

验证电子邮件自动售卖和转发是否按预期运行

Task	描述	所需技能
验证该 API 是否正常运行。	在此步骤中，您将测试数据提交到解决方案的 API，并确认解决方案产生预期输出并且后端操作已按预期执行。 使用测试输入，手动运行 Vend Email Lambda 函数。(有关示例，请参阅 sample_vend_request.json 文件。) 对于 `OwnerAddress` ，请使用有效的电子邮件地址。API 应返回账户名和账户电子邮件以及预期的值。	AWS 应用程序开发人员 DevOps
确认电子邮件已转发。	在此步骤中，您通过系统发送测试电子邮件并验证电子邮件是否转发给预期收件人。 1. 获取上一步中的账户电子邮件。 2. 向此地址发送一封包含测试主题和正文的电子邮件。 3. 确认您已通过账户拥有者的电子邮件地址收到电子邮件。 4. 确认您收到的电子邮件中的 FROM 地址与 `cdk.json` 中的 `ADDRESS_FROM` 设置相匹配。 5. 确认收到的电子邮件的主题和正文与原始发送邮件相同。	AWS 应用程序开发人员 DevOps

故障排除

事务	解决方案
系统无法按预期转发电子邮件。	验证您的设置是否正确： 1. 您应该已经完成域 Amazon SES 验证流程。 2. 您的域名应正确设置，MX 记录指向您 AWS 账户 和地区的 Amazon SES 终端节点。有关更多信息，请参阅 Amazon SES 文档中的发布 Amazon SES 电子邮件接收的 MX 记录。 验证域设置后，请按下列步骤操作： 1. 打开您部署解决方案的账户和地区的 Amazon CloudWatch 控制台，然后在导航窗格中导航到 CloudWatch 日志组。 2. 在日志组列表中搜索 `SesMailForwardLogGroup` 。 3. 调查此组中的日志，以查看在电子邮件销售和转发过程中是否生成了任何错误。
当你尝试部署 AWS CDK 堆栈时，你会收到类似于以下内容的错误： “模板格式错误：无法识别的资源类型”	在大多数情况下，此错误消息意味着您所定位的区域没有所有可用的 Amazon Web Services。如果您使用 Amazon EC2 实例来部署解决方案，则您的目标区域可能与实例运行的区域不同。 **Note** 默认情况下，会 AWS CDK 部署到您在中配置的区域和账户。 AWS CLI 可能的解决方案： 1. 查看 AWS 按地区划分的服务，调查此解决方案所需的所有服务（参见此模式前面的 “目标

事务	解决方案
	技术堆栈” 部分）是否都在 AWS 区域 您的目标中。 2. 如果您使用的 EC2 实例的目标区域与实例运行的区域不同，请务必在部署解决方案 AWS CLI 之前设置`AWS_DEFAULT_REGION` 环境变量或使用设置区域。有关更多信息，请参阅 AWS CLI 文档AWS CLI中的配置环境变量。或者，您可以按照环境AWS CDK 文档中的说明修改存储库根目录中的`app.py`文件，使其包含硬编码的账户 ID 和区域。
部署此解决方案时，您会收到错误消息： “部署失败：错误: AwsMailFwdStack: 找 parameter /cdk-bootstrap/hnb659fds/version 不到 SSM。是否已引导环境？ 请运行 'cdk bootstrap'	如果您从未将任何 AWS CDK 资源部署到目标 AWS 账户 和区域，则必须先按照错误指示运行该`cdk bootstrap` 命令。如果您在运行 bootstrapping 命令后继续收到此错误，则您可能正在尝试将解决方案部署到与运行开发环境的区域不同的区域。 要解决此问题，请在部署解决方案 AWS CLI 之前设置`AWS_DEFAULT_REGION` 环境变量或使用设置区域。或者，您可以按照环境AWS CDK 文档中的说明修改存储库根目录中的`app.py`文件，使其包含硬编码的账户 ID 和区域。

相关资源

- 有关安装的帮助 AWS CLI，请参阅安装或更新到最新版本的 AWS CLI。
- 有关 AWS CLI 使用 IAM 访问凭证设置的帮助，请参阅配置设置 AWS CLI。
- 有关方面的帮助 AWS CDK，请参阅入门 AWS CDK。

其他信息

成本

部署此解决方案时， AWS 账户 持有者可能会产生与使用以下服务相关的费用。 对您来说，了解这些服务的计费方式非常重要，这样您就可以了解任何潜在的费用。有关定价信息，请参阅以下页面：

- Amazon SES 定价
- Amazon S3 定价
- AWS KMS 定价
- AWS Lambda 定价
- Amazon DynamoDB 定价

在单账户 AWS 环境中为混合网络设置 DNS 解析

由 Abdullahi Olaoye (AWS) 编写

摘要

此模式描述了如何设置完全混合的域名系统 (DNS) 架构，该架构支持本地资源、AW end-to-end S 资源和互联网 DNS 查询的 DNS 解析，而无需管理开销。该模式描述了如何设置 Amazon Route 53 Resolver 转发规则，以根据域名确定应将源自 AWS 的 DNS 查询发送到何处。对本地资源的 DNS 查询将转发到本地 DNS 解析程序。AWS 资源的 DNS 查询和互联网 DNS 查询由 Route 53 解析程序解析。

此模式涵盖 AWS 单账户环境中的混合 DNS 解析。有关在 AWS 多账户环境中设置出站 DNS 查询的信息，请参阅在多账户 AWS 环境中为混合网络设置 DNS 解析。

先决条件和限制

先决条件

- 一个 Amazon Web Services account
- 您的 Amazon Web Services account 中的虚拟私有云（VPC）
- 通过 AWS 虚拟专用网络 (AWS VPN) 或 AWS Direct Connect 在本地环境和 VPC 之间建立网络连接
- 您的本地 DNS 解析程序的 IP 地址（可从您的 VPC 访问）
- 要转发给本地解析程序的域名/子域名（例如 onprem.mydc.com）
- AWS 私有托管区的域名/子域名（例如 myvpc.cloud.com）

架构

目标技术堆栈

- Amazon Route 53 私有托管区
- Amazon Route 53 Resolver
- Amazon VPC
- AWS VPN 或 Direct Connect

目标架构

工具

- Amazon Route 53 Resolver 通过在整个混合云实现无缝 DNS 查询解析，让企业客户更轻松地使用混合云。您可以创建 DNS 终端节点和条件转发规则，以解析本地数据中心与您的数据中心之间的 DNS 命名空间。 VPCs
- Amazon Route 53 私有托管区域是一个容器，其中包含有关您希望 Route 53 如何响应您使用 Amazon VPC 服务创建的一个或多个 VPCs 域中的域及其子域的 DNS 查询的信息。

操作说明

配置私有托管区

Task	描述	所需技能
为 AWS 保留域名（例如 myvpc.cloud.com）创建 Route 53 私有托管区。	该区域保存应从本地环境解析的 AWS 资源 DNS 记录。有关说明，请参阅 Route 53 文档中的创建私有托管区。	网络管理员、系统管理员
将此私有托管区与分支账户 VPC 相关联。	要使 VPC 中的资源能够解析此私有托管区中的 DNS 记录，您必须将您的 VPC 与托管区关联。有关说明，请参阅 Route 53 文档中的创建私有托管区。	网络管理员、系统管理员

设置 Route 53 解析程序 端点

Task	描述	所需技能
创建入站端点。	Route 53 Resolver 使用入站端点接收来自本地 DNS 解析程序的 DNS 查询。有关说明，请参阅 Route 53 文档 VPCs中的将入站 DNS 查询转发给您。记下入站端点 IP 地址。	网络管理员、系统管理员

Task	描述	所需技能
创建出站端点。	Route 53 解析程序使用出站端点向本地 DNS 解析程序发送 DNS 查询。有关说明，请参阅 Route 53 文档中的将出站 DNS 查询转发到您的网络。记下输出端点 ID。	网络管理员、系统管理员

设置转发规则并将其与您的 VPC 关联

Task	描述	所需技能
为本地域创建转发规则	此规则将指示 Route 53 解析程序将本地域（例如 onprem.mydc.com）的任何 DNS 查询转发给本地 DNS 解析程序。要创建此规则，您需要本地 DNS 解析程序的 IP 地址和 Route 53 解析程序的出站端点 ID。有关说明，请参阅 Route 53 文档中的管理转发规则。	网络管理员、系统管理员
将转发规则与 VPC 关联。	若要使转发规则生效，您必须将该规则与您的 VPC 关联起来。然后，Route 53 解析程序在解析域名时会考虑该规则。有关说明，请参阅 Route 53 文档中的管理转发规则。	网络管理员、系统管理员

配置本地 DNS 解析程序

Task	描述	所需技能
在内部部署 DNS 解析程序中配置条件转发。	要从本地环境向 Route 53 私有托管区发送 DNS 查询，您必	网络管理员、系统管理员

Task	描述	所需技能
	须在本地 DNS 解析程序中配置条件转发。这指示 DNS 解析程序将 AWS 域（例如 myvpc.cloud.com）的所有 DNS 查询转发到 Route 53 解析程序的入站端点 IP 地址。	

测试 end-to-end DNS 解析度

Task	描述	所需技能
测试 AWS 到本地环境的 DNS 解析。	在 VPC 中的服务器上，对本地域（例如 server1.onprem.mydc.com）执行 DNS 查询。	网络管理员、系统管理员
测试从本地环境到 AWS 的 DNS 解析。	在本地服务器上，对 AWS 域（例如 serv1.myvpc.cloud.com）执行 DNS 解析。	网络管理员、系统管理员

相关资源

- 使用 Amazon Route 53 和 AWS Transit Gateway 对混合云进行集中 DNS 管理（AWS 网络与内容交付博客）
- 使用 Route 53 Resolver 简化多账户环境中的 DNS 管理（AWS Security 博客文章）
- 使用私有托管区（Route 53 文档）
- Route 53 解析程序 入门（Route 53 文档）

使用 AWS 在亚马逊 EC2 上自动设置 UiPath RPA 机器人 CloudFormation

由 Rahul Sharad Gaikwad 博士 (AWS) 和 Tamilselvan P (AWS) 编写

摘要

此模式说明了如何在亚马逊弹性计算云 (Amazon) 实例上部署机器人流程自动化 (RPA EC2) 机器人。它使用 I EC2 mage Builder 管道来创建自定义的亚马逊系统映像 (AMI)。AMI 是一种预配置的虚拟机 (VM) 映像，其中包含用于部署 EC2 实例的操作系统 (OS) 和预安装的软件。此模式使用 AWS CloudFormation 模板在自定义 AMI 上安装 UiPath Studio 社区版。 UiPath 是一款 RPA 工具，可帮助您设置机器人来自动执行任务。

作为该解决方案的一部分，使用基本 AMI 启动 EC2 Windows 实例，并在实例上安装 UiPath Studio 应用程序。该模式使用 Microsoft 系统准备（Sysprep）工具复制自定义的 Windows 安装。之后，它删除主机信息并从实例创建最终 AMI。然后，您可以使用最终的 AMI 以及您自己的命名约定和监控设置来按需启动实例。

> **Note**
>
> 此模式不提供有关使用 RPA 机器人的任何信息。有关该信息，请参阅UiPath 文档。您也可使用此模式根据您的要求自定义安装步骤来设置其他 RPA 机器人应用程序。

此模式提供以下自动化和优点：

- 应用程序部署和共享：您可以构建 Amazon EC2 AMIs 以进行应用程序部署，并通过 I EC2 mage Builder 管道在多个账户之间共享它们，该管道使用 AWS CloudFormation 模板作为基础设施即代码 (IaC) 脚本。
- Amazon EC2 配置和扩展： CloudFormation IaC 模板提供自定义计算机名称序列和 Active Directory 自动加入功能。
- 可观察性和监控：该模式设置了亚马逊 CloudWatch 控制面板，以帮助您监控亚马逊 EC2 指标（例如 CPU 和磁盘使用率）。
- RPA 为您的企业带来的好处：RPA 提高了准确性，因为机器人可以自动、一致地执行分配的任务。RPA 还可以提高速度和生产力，因为它消除了不增加价值的操作并处理重复的活动。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 用于部署 CloudFormation 模板@@ 的 AWS Identity and Access Management (IAM) 权限
- 用于@@ 使用 Im EC2 age Builder 设置跨账户 AMI 分发的 IAM 策略

架构

1. 管理员在`ec2-image-builder.yaml`文件中提供基本 Windows AMI，并在 CloudFormation 控制台中部署堆栈。
2. CloudFormation 堆栈部署 EC2 Image Builder 管道，其中包括以下资源：
   - `Ec2ImageInfraConfiguration`
   - `Ec2ImageComponent`
   - `Ec2ImageRecipe`
   - `Ec2AMI`
3. EC2 Image Builder 管道使用基本 AMI 启动临时 Windows EC2 实例并安装所需的组件（在本例中为 UiPath Studio）。
4. EC2 Image Builder 会删除所有主机信息并从 Windows 服务器中创建 AMI。
5. 您可以使用自定义 AMI 更新`ec2-provisioning yaml`文件并根据您的要求启动多个 EC2 实例。
6. 您可以使用 CloudFormation 模板部署 Count 宏。此宏为 CloudFormation 资源提供了 C ount 属性，因此您可以轻松地指定多个相同类型的资源。
7. 更新 CloudFormation `ec2-provisioning.yaml`文件中宏的名称并部署堆栈。
8. 管理员根据要求更新 `ec2-provisioning.yaml` 文件并启动堆栈。
9. 该模板使用 UiPath Studio 应用程序部署 EC2 实例。

工具

Amazon Web Services

- AWS CloudFormation 可帮助您以自动且安全的方式对基础设施资源进行建模和管理。
- Amazon CloudWatch 可帮助您观察和监控 AWS、本地和其他云上的资源和应用程序。

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 云中提供安全且可调整大小的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- EC2 Image Builder 简化了在 AWS 上或本地使用的虚拟机和容器镜像的构建、测试和部署。
- Amazon EventBridge 可帮助您在 AWS、现有系统或软件即服务 (SaaS) 应用程序中大规模构建事件驱动型应用程序。
- AWS Identity and Access Management (IAM) 可帮助您安全地控制对您 AWS 资源的访问。借助 IAM，您可以集中管理控制用户可访问哪些 AWS 资源的权限。可以使用 IAM 来控制谁通过了身份验证(准许登录)并获得授权(拥有权限)来使用资源。
- AWS Lambda 是一项无服务器、事件驱动计算服务，让您能够为几乎任何类型的应用程序或后端服务运行代码，而无需预调配或管理服务器。您可以从 200 多种 Amazon Web Services 和 SaaS 应用程序中调用 Lambda 函数，并且只需按实际使用量付费。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Systems Manager 代理 (SSM 代理) 可帮助 Systems Manager 更新、管理和配置 EC2 实例、边缘设备、本地服务器和虚拟机 (VMs)。

代码存储库

此模式的代码可在使用 CloudFormation存储库的 GitHub UiPath RPA 机器人设置中找到。该模式还使用 A WS Mac CloudFormation ros 存储库中提供的宏。

最佳实践

- AWS 每月 AMIs都会发布新的 Windows。其中包含最新的操作系统补丁、驱动程序和启动代理。您应该在启动新实例或构建自己的自定义镜像时利用最新的 AMI。
- 在镜像生成期间应用所有可用的 Windows 或 Linux 安全补丁。

操作说明

为基础镜像部署镜像管道

Task	描述	所需技能
设置 EC2 Image Builder 管道。	1. 使用 CloudFormation存储库克隆 UiPath RPA 机器人设置，或者从存储库下载	AWS DevOps

Task	描述	所需技能
	`ec2-image-builder.yaml` 模板。 2. 登录 AWS 管理控制台，然后打开 AWS CloudFormation 控制台。 3. 选择创建堆栈。 4. 在指定模板部分，选择上传模板文件。 5. 在您的计算机上找到并上传 `ec2-image-builder.yaml` 模板，然后选择下一步。 6. 为堆栈提供输入参数或者接受默认值。选择下一步。 **Note** 参数的数量和值可能会因您的输入值而异。 7. （可选）配置堆栈选项，然后选择下一步。 8. 查看堆栈详细信息。 9. 在屏幕末尾，选中确认权能的复选框，然后选择提交。 10监控堆栈的进度。如果状态为 `CREATE_COMPLETE`，则部署已准备就绪。	

Task	描述	所需技能
查看 EC2 Image Builder 设置。	EC2 Image Builder 设置包括基础架构配置、分发设置和安全扫描设置。若要查看设置，请执行以下操作： 1. 打开 EC2 Image Builder 控制台。 2. 在导航窗格中导航到各种 Image Builder 设置。 **Note** 最佳做法是，应仅通过 CloudFormation 模板对 EC2 Image Builder 进行任何更新。	AWS DevOps
查看镜像管道。	要查看已部署的镜像管道，请执行以下操作： 1. 在 EC2 Image Builder 控制台上，从导航窗格中选择图像管道。 2. 选择您所创建的镜像管道。 3. 查看输出映像、图像配方、基础设施配置、分发设置、Amazon EventBridge 规则和标签的配置详情。	AWS DevOps

Task	描述	所需技能
查看 Image Builder 日志。	EC2 Image Builder 日志按 CloudWatch 日志组进行聚合。要查看日志，请执行 CloudWatch以下操作： 1. 打开 CloudWatch 管理控制台。 2. 在导航窗格中，依次选择日志和日志组。 3. 选择日志组名称。 EC2 Image Builder 日志聚合到日志组中`/aws/imagebuilder/XXX` 。 4. 检查相应日志流中的最新日志，了解运行镜像管道时是否遇到任何错误。 EC2 Image Builder 日志也存储在 S3 存储桶中。若要查看存储桶中的日志，请执行以下操作： 1. 打开 Amazon S3 控制台。 2. 在存储桶列表中，请选择桶名称。日志聚合在 S3 存储桶 `<stack-name>-XXXXXX` 中。	AWS DevOps

Task	描述	所需技能
将 UiPath 文件上传到 S3 存储桶。	1. 从位置 https://download.uipath.com/UiPathStudioCommunity.msi 下载适用于 UiPath Studio 的 `.msi`文件。 2. 将文件上传到 S3 存储桶。 3. 更新 `ec2-image-builder.yaml` 模板中用户数据部分第 310 行中的存储桶名称和文件密钥。	AWS DevOps

部署并测试 Count 宏

Task	描述	所需技能
部署计数宏。	1. 克隆或下载计数 CloudFormation 宏。 2. 导航到 `Count`文件夹。 3. 您将需要一个 S3 存储桶来存储 CloudFormation 项目。如果您还没有 S3 存储桶，请使用名称 `aws s3 mb s3://<bucket name>` 创建一个存储桶。 4. 将 Count 宏模板打包。该模板使用 AWS 无服务器应用程序模型 (SAM)，在部署之前必须对其进行转换。 `aws cloudformation package \ --template-file template.yaml \`	DevOps 工程师

Task	描述	所需技能
	--s3-bucket <your bucket name here> \ --output-template-file packaged.yaml 例如： aws cloudformation package \ --template-file template.yaml \ --s3-bucket count-macro-ec2 \ --output-template-file packaged.yaml 5. 部署打包的模板以创建 CloudFormation 堆栈。 aws cloudformation deploy \ --stack-name Count-macro \ --template-file packaged.yaml \ --capabilities CAPABILITY_IAM 如果要使用控制台，请按照上一篇长篇故事或CloudFormation 文档中的说明进行操作。	

Task	描述	所需技能
测试计数宏。	若要测试宏的功能，请尝试启动宏附带的示例模板。 `aws cloudformation deploy \ --stack-name Count-test \ --template-file test.yaml \ --capabilities CAPABILITY_IAM`	DevOps 工程师

部署 CloudFormation 堆栈以使用自定义映像配置实例

Task	描述	所需技能
部署 Amazon EC2 配置模板。	要部署 EC2 映像管道，请使用 CloudFormation以下方法： 1. 从GitHub 存储库下载`ec2-provisioning.yaml` 模板，或者如果您克隆了存储库，则可以在计算机上找到该模板。 2. 打开 CloudFormation 管理控制台。 3. 重复第一个长篇故事中的步骤（或按照CloudFormation 文档中的说明进行操作）进行部署`ec2-provisioning.yaml` 。	AWS DevOps
查看 Amazon EC2 设置。	Amazon EC2 设置包括安全、联网、存储、状态检查、监控	AWS DevOps

Task	描述	所需技能
	和标签配置。若要查看这些配置，请执行以下操作： 1. 打开 Amazon EC2 控制台。 2. 在导航窗格中，选择实例，然后选择由 Amazon EC2 配置模板创建的 EC2 实例。 3. 在实例摘要中，选择选项卡以查看相应的 Amazon EC2 设置。	
查看 CloudWatch 控制面板。	1. 打开 CloudWatch 管理控制台。 2. 在导航窗格中，选择控制面板。 3. 选择包含堆栈名称的控制面板。 **Note** 配置堆栈后，需要一段时间才能在仪表板中填充指标。 控制面板提供以下指标：`CPUUtilization`、`DiskUtilization`、`MemoryUtilization`、`NetworkIn`、`NetworkOut`、`StatusCheckFailed`。	AWS DevOps

Task	描述	所需技能
查看关于内存和磁盘使用情况的自定义指标。	1. 在CloudWatch 控制台上，选择仪表板。 2. 在导航窗格中，依次选择指标、所有指标。 3. 选择 “自定义命名空间”。CWAgent	AWS DevOps
查看内存与磁盘使用情况警报。	1. 在CloudWatch 控制台的导航窗格中，选择仪表板。 2. 选择所有警报。	AWS DevOps
验证快照生命周期规则。	1. 打开 Amazon EC2 控制台。 2. 在导航窗格中，选择生命周期管理器。 3. 验证 AMI 生命周期设置。	AWS DevOps

删除环境（可选）

Task	描述	所需技能
删除堆栈。	PoC 或试点项目完成后，我们建议您删除创建的堆栈，以确保您无需为这些资源付费。 1. 打开 A WS CloudFormation 控制台。 2. 在导航窗格中，选择堆栈，然后选择您之前创建的要删除的一个或两个堆栈。该堆栈当前必须处于运行状态。 3. 在堆栈详细信息窗格中，选择删除。	AWS DevOps

Task	描述	所需技能
	4. 当系统提示时，选择 Delete stack（删除堆栈）。 **Important** 堆栈删除操作开始后无法停止。堆栈进入 DELETE_IN_PROGRESS 状态。 如果删除失败，则堆栈将处于 DELETE_FAILED 状态。有关解决方案，请参阅 AWS CloudFormation 疑难解答文档中的删除堆栈失败。 有关保护堆栈不被意外删除的信息，请参阅 AWS CloudFormation 文档中的保护堆栈不被删除。	

故障排除

事务	解决方案
当你部署 Amazon EC2 配置模板时，你会收到错误消息：收到来自转换 123xxxx:: Count 的格式错误的响应。	这是一个已知问题。（参见 AW CloudFormation S 宏存储库中的自定义解决方案和 PR。） 要修复此问题，请打开 AWS Lambda 控制台并index.py使用存储库中的GitHub 内容进行更新。

相关资源

GitHub 存储库

- UiPath 使用 RPA 机器人设置 CloudFormation
- 计数 CloudFormation 宏

AWS 参考

- 在 AWS CloudFormation 控制台上创建堆栈（CloudFormation 文档）
- 故障排除 CloudFormation（CloudFormation 文档）
- 监控 Amazon EC2 实例的内存和磁盘指标（亚马逊 EC2 文档）
- 如何使用 CloudWatch 代理在 Windows 服务器上查看 “性能监视器” 的指标？（AWS re: Post 文章）

其他参考资料

- UiPath 文档
- 在 SysPreped AMI 中设置主机名（Brian Beach 的博客文章）
- 当参数发生变化时，如何让 Cloudformation 通过宏重新处理模板？ 堆栈溢出

在 AWS 上设置高度可用的 PeopleSoft 架构

由 Ramanathan Muralidhar (AWS) 编写

摘要

当您将 PeopleSoft 工作负载迁移到 AWS 时，弹性是一个重要的目标。它可确保您的 PeopleSoft 应用程序始终保持高可用性，并能够快速从故障中恢复。

此模式为您在 AWS 上的 PeopleSoft 应用程序提供了一个架构，可确保网络、应用程序和数据库层的高可用性 (HA)。它将 Amazon Relational Database Service (Amazon RDS) 用于 Oracle，或将 Amazon RDS for SQL Server 数据库用于数据库层。该架构还包括亚马逊 Route 53、亚马逊弹性计算云 (亚马逊 EC2) Linux 实例、亚马逊弹性块存储 (亚马逊 EBS)、亚马逊弹性文件系统 (亚马逊 EFS) 和应用程序负载均衡器等 AWS 服务，并且具有可扩展性。

Oracle PeopleSoft 为劳动力管理和其他业务运营提供了一套工具和应用程序。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 具有在 AWS 上进行设置所需的许可证的 PeopleSoft 环境
- 您的 Amazon Web Services account 中设置的虚拟私有云（VPC）包含以下资源：
  - 至少两个可用区
  - 每个可用区中有 1 个公有子网和 3 个私有子网
  - NAT 网关和互联网网关
  - 每个子网的路由表用于路由流量
  - 根据贵组织的标准定义的网络访问控制列表（网络 ACLs）和安全组，以帮助确保 PeopleSoft 应用程序的安全

限制

- 这种模式提供高可用性 (HA) 解决方案。它不支持灾难恢复 (DR) 方案。在极少数情况下，用于实现 HA 的整个 AWS 区域出现故障，应用程序将不可用。

产品版本

- PeopleSoft 运行 PeopleTools 8.52 及更高版本的应用程序

架构

目标架构

PeopleSoft 生产应用程序的停机或中断会影响应用程序的可用性，并对您的业务造成重大中断。

我们建议您在设计 PeopleSoft 生产应用程序时使其始终保持高可用性。您可通过消除单点故障、添加可靠的交叉点或失效转移点以及检测故障来实现这一点。下图说明了 AWS PeopleSoft 上的 HA 架构。

此架构部署使用适用于 Oracle 的 Amazon RDS 作为 PeopleSoft 数据库，使用在红帽企业 Linux (RHEL) 上运行的 EC2 实例。您还可以使用 Amazon RDS for SQL Server 作为 Peoplesoft 数据库。

该架构包含以下组件：

- Amazon Route 53 用作域名服务器 (DNS)，用于将来自互联网的请求路由到 PeopleSoft 应用程序。
- AWS WS WAF 帮助您防范可能影响可用性、损害安全性或消耗过多资源的常见 Web 漏洞和机器人。AWS Shield Advanced（未图示）提供更广泛的保护。
- 应用程序负载均衡器通过针对 Web 服务器的高级请求路由对 HTTP 和 HTTPS 流量进行负载平衡。
- 支持应用程序的 Web 服务器、应用程序服务器、进程调度器服务器和 Elasticsearch 服务器在多个可用区中运行并使用 Amazon A uto Scaling EC2 。 PeopleSoft
- PeopleSoft 应用程序使用的数据库以多可用区配置在 Amazon RDS 上运行。
- PeopleSoft 应用程序使用的文件共享在 Amazon EFS 上配置，用于跨实例访问文件。
- Amazon A EC2 uto Scaling 使用亚马逊系统映像 (AMI) 来确保在需要时快速克隆 PeopleSoft 组件。
- NAT 网关将私有子网中的实例连接到 VPC 外部的服务，并确保外部服务无法启动与这些实例的连接。
- 互联网网关是一种横向扩展、冗余且高度可用的 VPC 组件，支持在 VPC 和互联网之间进行通信。
- 公有子网中的堡垒主机提供从外部网络（例如互联网或本地网络）访问私有子网中的服务器的权限。堡垒主机提供对私有子网中服务器的受控且安全的访问。

架构详情

该 PeopleSoft 数据库位于采用多可用区配置的 Amazon RDS for Oracle（或适用于 SQL Server 的 Amazon RDS）数据库中。Amazon RDS Multi-AZ 功能可跨两个可用区复制数据库更新，以提高持久性和可用性。Amazon RDS 会自动失效转移到备用数据库，以进行计划内维护和计划外中断。

PeopleSoft Web 层和中间层安装在 EC2 实例上。这些实例分布在多个可用区中，并由自动扩缩组绑定。这确保了这些组件始终具有高可用性。维护最低数量的所需实例，以确保应用程序始终可用并且可以在需要时进行扩展。

我们建议您为 OEM EC2 实例使用最新一代的 EC2 实例类型。当前一代的实例类型，例如在 AWS Nitro 系统上构建的实例，支持硬件虚拟机 (HVMs)。HVM AMIs 需要利用增强的联网，而且它们还能提供更高的安全性。属于每个 Auto Scaling 组的 EC2 实例在替换或扩展实例时使用自己的 AMI。我们建议您根据希望应用程序处理的负载以及 Oracle 为您的 PeopleSoft PeopleSoft 应用程序和 PeopleTools 版本推荐的最低值来选择 EC2 实例类型。有关硬件和软件要求的更多信息，请参见 Oracle 支持网站。

PeopleSoft Web 层和中间层共享 Amazon EFS 挂载以共享报告、数据文件和（如果需要）`PS_HOME`目录。出于性能和成本考虑，Amazon EFS 在每个可用区配置挂载目标。

Application Load Balancer 的配置是为了支持访问 PeopleSoft 应用程序的流量，并对不同可用区之间的 Web 服务器之间的流量进行负载平衡。应用程序负载均衡器是在至少两个可用区内提供 HA 的网络设备。Web 服务器使用负载平衡配置将流量分发到不同的应用程序服务器。Web 服务器和应用程序服务器间的负载平衡可确保负载在实例间均匀分布，并有助于避免因实例过载而导致瓶颈和服务中断。

Amazon Route 53 提供 DNS 服务，用于将流量从互联网路由到应用程序负载均衡器。Route 53 是一种可用性高、可扩展性强的 DNS Web 服务。

HA 详情

- 数据库：Amazon RDS 的多可用区功能通过同步复制在多个可用区中运行两个数据库。这将创建一个具有自动失效转移功能的高可用性环境。Amazon RDS 具有失效转移事件检测功能，这些事件发生时启动自动失效转移。您也可通过 Amazon RDS API 启动手动失效转移。有关详细说明，请参见博客文章 Amazon RDS 幕后：多可用区。失效转移是无缝的，并且应用程序会在发生故障时自动重新连接到数据库。但是，失效转移期间的任何进程调度程序作业都会生成错误，并且必须重新提交。
- PeopleSoft 应用程序服务器：应用程序服务器分布在多个可用区中，并为其定义了一个 Auto Scaling 组。如果实例发生故障，自动扩缩组会立即将其替换为从应用程序服务器模板的 AMI 克隆的正常实例。具体而言，Jolt pooling 已启用，当应用程序服务器实例出现故障时，会话会自动失效转移到另一个应用程序服务器，并且自动扩缩组会自动启动另一个实例、启动应用程序服务器并将其注册到 Amazon EFS 挂载中。使用 Web 服务器中的 `PSSTRSETUP.SH` 脚本，将新创建的应用程序

服务器自动添加到 Web 服务器中。这可确保应用程序服务器始终处于高可用性并能快速从故障中恢复。

- 流程调度器：流程调度器服务器分布在多个可用区，并为其定义自动扩缩组。如果实例发生故障，自动扩缩组会立即将其替换为从进程调度程序服务器模板的 AMI 克隆的运行状况良好的实例。具体来说，当进程调度程序实例出现故障时，自动扩缩组会自动启动另一个实例并启动进程调度程序。实例失败时正在运行的任何作业都必须重新提交。这确保了进程调度程序始终具有高可用性并能够快速从故障中恢复。
- Elasticsearch 服务器：Elasticsearch 服务器具有为其定义的自动扩缩组。如果实例发生故障，自动扩缩组会立即将其替换为从 Elasticsearch 服务器模板的 AMI 克隆的运行状况良好的实例。具体来说，当 Elasticsearch 实例出现故障时，向其提供请求的应用程序负载均衡器会检测到故障并停止向其发送流量。自动扩缩组会自动启动另一个实例且启动 Elasticsearch 实例。当 Elasticsearch 实例备份时，应用程序负载均衡器检测到它运行状况良好并开始再次向其发送请求。这可确保 Elasticsearch 服务器始终保持高可用性并快速从故障中恢复。
- Web 服务器：Web 服务器具有为其定义的自动扩缩组。如果实例发生故障，自动扩缩组会立即将其替换为从 Web 服务器模板的 AMI 克隆的正常实例。具体来说，当 Web 服务器实例出现故障时，向其提供请求的应用程序负载均衡器会检测到故障并停止向其发送流量。自动扩缩组会自动启动另一个实例并启动 Web 服务器实例。备份 Web 服务器实例时，应用程序负载均衡器检测到其运行状况良好，并再次开始向其发送请求。这可确保 Web 服务器始终保持高可用性并快速从故障中恢复。

工具

Amazon Web Services

- 应用程序负载均衡器将传入的应用程序流量分发到多个可用区域中的多个目标（例如 EC2 实例）。
- 亚马逊弹性区块存储 (Amazon EBS) Block Store 提供块级存储卷，用于亚马逊弹性计算云 (Amazon) 实例。 EC2
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- Amazon Elastic File System (Amazon EFS) 可帮助您在 Amazon Web Services Cloud 中创建和配置共享文件系统。
- Amazon Relational Database Service (Amazon RDS) 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展关系数据库。
- Amazon Route 53 是一种可用性高、可扩展性强的 DNS Web 服务。

最佳实践

运营最佳实践

- PeopleSoft 在 AWS 上运行时，使用 Route 53 路由来自互联网和本地的流量。如果主数据库实例不可用，则使用失效转移选项 将流量重新路由到灾难恢复 (DR) 站点。
- 始终在 PeopleSoft 环境前使用 Application Load Balancer。这样可确保以安全的方式将流量负载均衡到 Web 服务器。
- 在应用程序负载均衡器目标组设置中，确保使用负载均衡器生成的 Cookie 开启粘性。

> **Note**
>
> 如果您使用外部单点登录 (SSO)，则可能需要使用基于应用程序的 Cookie。这样可确保 Web 服务器和应用程序服务器之间的连接保持一致。

- 对于 PeopleSoft 生产应用程序，Application Load Balancer 空闲超时必须与您使用的网络配置文件中设置的值相匹配。这样可防止用户会话在负载均衡器层过期。
- 对于 PeopleSoft 生产应用程序，请将应用程序服务器的回收次数设置为可最大限度地减少内存泄漏的值。
- 如果您将 Amazon RDS 数据库用于 PeopleSoft 生产应用程序（如本模式所述），请以多可用区格式运行该数据库以实现高可用性。
- 如果您的数据库在 PeopleSoft 生产应用程序的 EC2 实例上运行，请确保备用数据库在另一个可用区上运行以实现高可用性。
- 对于灾难恢复，请确保您的 Amazon RDS 数据库或 EC2 实例在与生产数据库不同的 AWS 区域中配置了备用数据库。这样可以确保在该地区发生灾难时，您可将应用程序割接到另一个区域。
- 对于灾难恢复，请使用 Amazon Elastic Disaster Recovery在与生产组件不同的区域中设置应用程序级组件。这样可以确保在该地区发生灾难时，您可将应用程序割接到另一个区域。
- 使用 Amazon EFS（满足中等 I/O 要求）或亚马逊 FSx（用于高 I/O 要求）来存储您的 PeopleSoft 报告、附件和数据文件。这样可以确保内容存储在一个中心位置，并且可从基础设施中的任何地方进行访问。
- 使用 Amazon CloudWatch（基本和详细）近乎实时地监控您的 PeopleSoft 应用程序正在使用的 AWS 云资源。这样可确保您立即收到问题警报，并在问题影响环境可用性之前快速解决问题。
- 如果您使用 Amazon RDS 数据库作为 PeopleSoft 数据库，请使用增强监控。此功能提供对 50 多个指标的访问，包括 CPU、内存、文件系统 I/O 和磁盘 I/O。

- 使用 AWS CloudTrail 监控您的 PeopleSoft 应用程序正在使用的 AWS 资源上的 API 调用。这可以帮助您执行安全分析、资源更改跟踪和合规性审核。

安全最佳实践

- 要保护您的 PeopleSoft 应用程序免受 SQL 注入或跨站脚本 (XSS) 等常见漏洞攻击，请使用 AWS WAF。考虑使用 AWS Shield Advanced 提供量身定制的检测和缓解服务。
- 向 Application Load Balancer 添加一条规则，自动将流量从 HTTP 重定向到 HTTPS，从而帮助保护您的 PeopleSoft 应用程序。
- 为应用程序负载均衡器设置单独安全组。此安全组应仅允许 HTTPS/HTTP 入站流量，不允许出站流量。这样可确保仅允许预期流量，并有助于保护您的应用程序。
- 对应用程序服务器、Web 服务器和数据库使用私有子网，对出站 Internet 流量使用 NAT 网关。这样可确保无法公开访问支持该应用程序的服务器，同时仅向需要该应用程序的服务器提供公共访问权限。
- 使用不同的 VPCs 方法来运行 PeopleSoft 生产和非生产环境。使用 AWS Transit Gateway ACLs、VPC 对等互连、网络和安全组来控制 VPC 与您的本地数据中心之间的流量（如有必要）。
- 遵循最低权限原则。仅向绝对需要的用户授予访问 PeopleSoft 应用程序所使用的 AWS 资源的权限。仅授予执行任务所需最低权限。有关更多信息，请参阅 AWS Well-Architected Framework 的安全支柱。
- 尽可能使用 AWS Systems Manager 访问 PeopleSoft 应用程序使用的 EC2 实例。

可靠性最佳实践

- 使用应用程序负载均衡器时，请为每个所启用的可用区注册一个目标。这使得负载均衡器最有效。
- 我们建议您 URLs 为每个 PeopleSoft 生产环境设置三个不同的网址：一个用于访问应用程序的 URL，一个用于提供集成代理的 URL，以及一个用于查看报告的 URL。如果可能，每个 URL 都应有自己的专用 Web 服务器和Application Server。这种设计有助于提高 PeopleSoft 应用程序的安全性，因为每个 URL 都具有不同的功能和受控的访问权限。它还可最大限度地减少底层服务失败时的影响范围。
- 我们建议您在 PeopleSoft 应用程序的负载均衡器目标组上配置运行状况检查。运行状况检查应在 Web 服务器上执行，而不是在运行这些服务器的 EC2 实例上执行。这样可以确保在 Web 服务器崩溃或托管 Web 服务器的 EC2 实例出现故障时，Application Load Balancer 会准确反映该信息。
- 对于 PeopleSoft 生产应用程序，我们建议您将 Web 服务器分布在至少三个可用区中。这样可以确保即使其中一个可用区出现故障， PeopleSoft 应用程序也始终保持高可用性。

- 对于 PeopleSoft 生产应用程序，启用 jolt pooling ()。`joltPooling=true`这样可以确保在服务器因修补目的或虚拟机故障而停机时，您的应用程序可以失效转移到另一台应用程序服务器。
- 对于 PeopleSoft 生产应用程序，`DynamicConfigReload` 请设置为 1。8.52 及更高 PeopleTools 版本支持此设置。它可以动态地向 Web 服务器添加新的应用程序服务器，而无需重新启动服务器。
- 要最大限度地减少应用 PeopleTools 补丁时的停机时间，请使用蓝/绿部署方法为 Web 和应用程序服务器的 Auto Scaling 组启动配置。有关更多信息，请参见 AWS 部署选项概述白皮书。
- 使用 AWS Backup 在 AWS 上备份您的 PeopleSoft 应用程序。AWS Backup 是一项经济实惠、完全托管、基于策略服务，可大规模简化数据保护。

性能最佳实践

- 在 Application Load Balancer 终止 SSL 以获得最佳 PeopleSoft 环境性能，除非您的业务需要整个环境中的加密流量。
- 为亚马逊简单通知服务 (Amazon SNS) Simple N CloudWatchotification Service 等 AWS 服务创建接口 VPC 终端节点，以便流量始终位于内部。这有成本效益，有助于保护您的应用程序安全。

成本优化最佳实践标准

- 标记您的 PeopleSoft 环境使用的所有资源，并启用成本分配标签。这些标签可帮您查看和管理资源成本。
- 对于 PeopleSoft 生产应用程序，请为 Web 服务器和应用程序服务器设置 Auto Scaling 组。这样可以维护最少数量的 Web 和应用程序服务器来支持您的应用程序。您可使用自动扩缩组策略根据需要向上和向下扩展服务器。
- 使用账单警报，在费用超过您指定的预算阈值时收到提醒。

可持续发展最佳实践标准

- 使用基础设施即代码 (IaC) 来维护您的 PeopleSoft 环境。这可以帮您构建一致的环境并保持变更控制。

操作说明

将您的 PeopleSoft 数据库迁移到 Amazon RDS

Task	描述	所需技能
创建数据库子网组。	在 Amazon RDS 控制台 导航窗格，选择子网组，然后创建子网位于多个可用区域的 Amazon RDS 数据库子网组。Amazon RDS 数据库需要这样做，才能使 Amazon RDS 数据库在多可用区配置中运行。	云管理员
创建 Amazon RDS 数据库。	在您为 PeopleSoft HA 环境选择的 AWS 区域的可用区中创建 Amazon RDS 数据库。创建 Amazon RDS 数据库时，请务必选择多可用区选项（创建备用实例）和您在上一步中创建的数据库子网组。有关更多信息，请参阅 Amazon RDS 文档。	云管理员、Oracle Database 管理员
将您的 PeopleSoft 数据库迁移到 Amazon RDS。	使用 AWS PeopleSoft 数据库迁移服务 (AWS DMS) 将您的现有数据库迁移到 Amazon RDS 数据库。有关更多信息，请参阅 AWS DMS 文档和 AWS Blog 文章使用 AWS DMS 以近乎零停机时间迁移 Oracle 数据库。	云管理员、数据库 PeopleSoft 管理员

设置您的 Amazon EFS 文件系统

Task	描述	所需技能
创建文件系统。	在 Amazon EFS 控制台，为每个可用区创建文件系统并挂载目标。有关说明，请参阅 Amazon EFS 文档。创建文件系统后，请记下 DNS 名称。您在挂载文件系统时使用此信息。	云管理员

设置您的 PeopleSoft 应用程序和文件系统

Task	描述	所需技能
启动实 EC2 例。	为您的 PeopleSoft 应用程序启动一个 EC2 实例。有关说明，请参阅 Amazon EC2 文档。 • 对于名称，请输入 APP_TEMPLATE 。 • 对于操作映象，请选择 Red Hat。 • 对于实例类型，选择适合您的 PeopleSoft 应用程序的实例类型。有关更多信息，请参阅架构部分中的架构详细信息。	云管理员、 PeopleSoft 管理员
在实例 PeopleSoft 上安装。	在您创建的 EC2 实例 PeopleTools 上安装您的 PeopleSoft 应用程序和。有关说明，请参阅 Oracle 文档。	云管理员、 PeopleSoft 管理员

Task	描述	所需技能
创建应用程序服务器。	为 AMI 模板创建 Application Server，并确保其成功连接到 Amazon RDS 数据库。	云管理员、 PeopleSoft 管理员

挂载 Amazon EFS 文件系统。	以 root 用户身份登录 EC2 实例并运行以下命令将 Amazon EFS 文件系统挂载到服务器PSFTMNT上名为的文件夹中。
	`sudo su -`
	`mkdir /psftmnt`
	`cat /etc/fstab`
	将以下行附加到 `/etc/fstab` 文件中。使用您在创建文件系统时记录的 DNS 名称。
	`fs-09e064308f1145388.efs.us-east-1.amazonaws.com:/ /psftmnt nfs4 nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport,_netdev 0 0`
	`mount -a`
		云管理员、 PeopleSoft 管理员

检查权限。	确保该PSFTMNT文件夹具有适当的权限，以便 PeopleSoft 用户可以正确访问该文件夹。	云管理员、 PeopleSoft 管理员

Task	描述	所需技能
创建额外实例。	重复本操作说明中的前几个步骤，为流程调度器、Web 服务器和 Elasticsearch 服务器创建模板实例。将这些实例命名为 PRCS_TEMPLATE 、WEB_TEMPLATE 和 SRCH_TEMPLATE 。对于 Web 服务器，设置 joltPooling=true 和 DynamicConfigReload=1 。	云管理员、 PeopleSoft 管理员

创建脚本以设置服务器

Task	描述	所需技能

创建用于安装应用服务器的脚本。	在 Amazon EC2 APP_TEMPLATE 实例中，以 PeopleSoft 用户身份创建以下脚本。给它命名为 appstart.sh ，并将其放在 PS_HOME 目录中。您将使用此脚本启动应用程序服务器，并在 Amazon EFS 挂载上记录服务器名称。
	`#!/bin/ksh`
	`. /usr/homes/hcmdemo/.profile.`
	`psadmin -c configure -d HCMDEMO`
	`psadmin -c parallelboot -d HCMDEMO`
	`touch /psftmnt/`echo $HOSTNAME`
		PeopleSoft 管理员

Task	描述	所需技能

创建用于安装流程调度服务器的脚本。	在 Amazon EC2 PRCS_TEMPLATE 实例中，以 PeopleSoft 用户身份创建以下脚本。给它命名为 `prcsstart.sh` ，并将其放在 PS_HOME 目录中。您将使用此脚本启动进程调度器服务器。
	
	`#!/bin/ksh`
	`. /usr/homes/hcmdemo/.profile`
	`/* The following line ensures that the process scheduler always has a unique name during replacement or scaling activity. */`
	`sed -i "s/.*PrcsServerName.*/`hostname -I | awk -F. '{print "PrcsServerName=PSUNX"$3$4}'`/" $HOME/appserv/prcs/*/psprcs.cfg`
	`psadmin -p configure -d HCMDEMO`
	`psadmin -p start -d HCMDEMO`
		PeopleSoft 管理员

Task	描述	所需技能

创建用于安装 Elasticsearch 服务器的脚本。	在亚马逊 EC2 SRCH_TEMPLATE 实例中，以 Elasticsearch 用户的身份创建以下脚本。给它命名为 `srchstart.sh` ，并将其放在 HOME 目录中。
	
	`#!/bin/ksh`
	`/* The following line ensures that the correct IP is indicated in the elasticsearch.yaml file. */`
	`sed -i "s/.*network.host.*/`hostname -I
		awk '{print "host:"$0}'`/" $ES_HOME_DIR/config/elasticsearch.yaml`
		`nohup $ES_HOME_DIR/bin/elasticsearch &`
			PeopleSoft 管理员

Task	描述	所需技能
创建安装 Web 服务器脚本。	在 Amazon EC2 WEB_TEMPLATE 实例中，以 Web 服务器用户的身份在HOME目录中创建以下脚本。 renip.sh：此脚本可确保 Web 服务器在从 AMI 克隆时有正确的 IP。 `#!/bin/ksh` `hn=`hostname`` `/* On the following` `line, change the IP` `with the hostname with` `the hostname of the web` `template. */` `for text_file in `find` `* -type f -exec grep` `-l '<hostname-of-the-` `web-template>' {} \;` `do` `sed -e 's/<hostn` `ame-of-the-web-tem` `plate>/'$hn'/g'` `$text_file > temp` `mv -f temp $text_file` `done`  psstrsetup.sh ：此脚本可确保 Web 服务器使用当前正在运行 IPs 的正确应用程序服务器。它尝试通过 jolt 端口连接到每台应用程序服务器，并将其添加到配置文件中。  `#!/bin/ksh` `c2=""`	PeopleSoft 管理员

Task	描述	所需技能
	for ctr in `ls -1 /psftmnt/*.internal` do c1=`echo $ctr | awk -F "/" '{print $3}'` /* In the following lines, 9000 is the jolt port. Change it if necessary. */ if nc -z $c1 9000 2> /dev/null; then if [[$c2 = ""]]; then c2="psserver="`echo $c1`":9000" else c2=`echo $c2`","`echo $c1`":9000" fi fi done webstart.sh ：此脚本运行前两个脚本并、和启动 Web 服务器。 #!/bin/ksh /* Change the path in the following if necessary. */ cd /usr/homes/hcmdemo ./renip.sh ./psstrsetup.sh webserv/peoplesoft/bin/startPIA.sh	

Task	描述	所需技能
添加 crontab 条目。	在亚马逊 EC2 WEB_TEMPLATE 实例中，以网络服务器用户的身份将以下行添加到 cronta b。更改时间和路径以反映所需值。此条目可确保 Web 服务器 configuration.properties 文件中始终包含正确的应用程序服务器条目。 `* * * * * /usr/homes/hcmdemo/psstrsetup.sh`	PeopleSoft 管理员

创建 AMIs 和 Auto Scaling 组模板

Task	描述	所需技能
为应用服务器模板创建 AMI。	在亚马逊 EC2 控制台上，创建亚马逊 EC2 APP_TEMPLATE 实例的 AMI 映像。命名 AMI PSAPPSRV-SCG-VER1。有关说明，请参阅 Amazon EC2 文档。	云管理员、 PeopleSoft 管理员
AMIs 为其他服务器创建。	重复上一步 AMIs 为流程调度器、Elasticsearch 服务器和 Web 服务器创建。	云管理员、 PeopleSoft 管理员
为应用程序服务器自动扩缩组创建启动模板。	为应用程序服务器自动扩缩组创建启动模板。将模板命名为 PSAPPSRV_TEMPLATE. 在模板中，选择您为 APP_TEMPLATE 实例创建的 AMI。有关	云管理员、 PeopleSoft 管理员

Task	描述	所需技能

	说明，请参阅 Amazon EC2 文档。
	• 在启动模板中，根据要求选择实例类型。
	• 在高级详细信息部分的用户数据字段，添加以下条目。确保路径和用户信息准确无误。您在上一步中创建了 `appstart.sh` 脚本。
	`#! /bin/ksh`
	`su -c "/usr/homes/hcmdemo/appstart.sh" - hcmdemo`	

为流程调度程序服务器自动扩缩组创建启动模板。	重复上一步的操作，为流程调度器服务器自动扩缩组创建启动模板。将模板命名为 `PSPRCS_TEMPLATE` 。在模板中，选择您为流程调度器所创建的 AMI。
	• 在高级详细信息部分的用户数据字段，添加以下条目。确保路径和用户信息准确无误。您在上一步中创建了 `prcsstart.sh` 脚本。
	`#! /bin/ksh`
	`su -c "/usr/homes/hcmdemo/prcsstart.sh" - hcmdemo`
		云管理员、 PeopleSoft 管理员

Task	描述	所需技能

为 Elasticsearch 服务器自动扩缩组创建启动模板。	重复上述步骤，为 Elasticsearch 服务器自动扩缩组创建启动模板。将模板命名为 SRCH_TEMPLATE 。在模板中，选择您为搜索服务器所创建的 AMI。
	• 在高级详细信息部分的用户数据字段，添加以下条目。确保路径和用户信息准确无误。您在上一步中创建了 srchstart.sh 脚本。
	`#! /bin/ksh`
	`su -c “/usr/homes/essearch/srchstart.sh” - essearch`
		云管理员、 PeopleSoft 管理员

为 Web 服务器自动扩缩组创建启动模板。	重复上述步骤，为 Web 服务器自动扩缩组创建启动模板。将模板命名为 WEB_TEMPLATE 。在模板中，选择您为 Web 服务器所创建的 AMI。
	• 在高级详细信息部分的用户数据字段，添加以下条目。确保路径和用户信息准确无误。您在上一步中创建了 webstart.sh 脚本。
	`#! /bin/ksh`
	`su -c “/usr/homes/hcmdemo/webstart.sh” - hcmdemo`
		云管理员、 PeopleSoft 管理员

创建自动扩缩组

Task	描述	所需技能
为应用程序服务器创建自动扩缩组。	在 Amazon EC2 控制台上，使用PSAPPSRV_TEMPLATE 模板创建名PSAPPSRV_ASG 为应用程序服务器的 Auto Scaling 组。有关说明，请参阅 Amazon EC2 文档。 • 在选择实例启动选项页面，选择正确的 VPC，然后从不同的可用区域中选择多个子网。 • 在配置高级选项页面，不要选择负载均衡器。 • 在配置组大小和扩展策略页面，根据您要架构系统的负载量以及是否要使用扩展策略来选择设置。我们建议您将所需容量和最小容量设置为 2，以在任何时间点都至少有一个实例可用于处理流量。有关 Auto Scaling 策略的更多信息，请参阅亚马逊 EC2 文档。	云管理员、 PeopleSoft 管理员
为其他服务器创建自动扩缩组。	重复上述步骤，为进程计划程序、Elasticsearch 服务器和 Web 服务器创建自动扩缩组。	云管理员、 PeopleSoft 管理员

创建和配置目标组

Task	描述	所需技能
为 Web 服务器创建目标组。	在 Amazon EC2 控制台上，为 Web 服务器创建目标组。有关说明，请参阅弹性负载均衡文档。将端口设置为 Web 服务器侦听端口。	云管理员
配置运行状况检查。	确认运行状况检查值是否正确，以反映您的业务需求。有关更多信息，请参阅弹性负载均衡文档。	云管理员
为 Elasticsearch 服务器创建目标组。	重复前面的步骤，为 Elasticsearch 服务器创建一个名为 PSFTSRCH 的目标组，并设置正确的 Elasticsearch 端口。	云管理员
将目标组添加到自动扩缩组。	打开您之前创建的名为 PSPIA_ASG 的 Web 服务器自动扩缩组。在负载平衡选项卡，选择编辑，然后将 PSFTWEB 目标组添加至自动扩缩组。 对 Elasticsearch 自动扩缩组 PSSRCH_ASG 重复此步骤，添加您之前创建的目标组 PSFTSRCH。	云管理员
设置会话粘性。	在目标组 PSFTWEB，选择属性选项卡，选择编辑，设置会话粘性。对于粘性类型，选择负载均衡器生成 cookie，并将持续时间设置为 1。有关更	云管理员

Task	描述	所需技能
	多信息，请参阅弹性负载均衡文档。 为目标组 PSFTSRCH 重复此步骤。	

创建并配置应用程序负载均衡器。

Task	描述	所需技能
为 Web 服务器创建负载均衡器。	创建名为 PSFTLB 的应用程序负载均衡器，以对流向 Web 服务器的流量进行负载平衡。有关说明，请参阅弹性负载均衡文档。 • 提供负载均衡器名称。 • 对于 Scheme，选择 Internet-facing。 • 在网络映射部分，选择正确的 VPC，以及来自不同可用区的至少两个公有子网。 • 在侦听器和路由部分，选择目标组 PSFTWEB，并指定正确的协议和端口号。	云管理员
为 Elasticsearch 服务器创建负载均衡器。	创建名为 PSFTSCH 的应用程序负载均衡器，以对流向 Elasticsearch 服务器的流量进行负载平衡。 • 提供负载均衡器名称。 • 对于方案，选择内部。 • 在网络映射，选择正确的 VPC 和私有子网。	云管理员

Task	描述	所需技能
	• 在侦听器和路由部分，选择目标组 PSFTSRCH，并指定正确的协议和端口号。	
配置 Route 53。	在 Amazon Route 53 控制台上，在将为 PeopleSoft 应用程序提供服务的托管区域中创建一条记录。有关说明，请参阅 Amazon Route 53 文档。这样可以确保所有流量都通过 PSFTLB 负载均衡器。	云管理员

相关资源

- 甲骨文 PeopleSoft 网站
- AWS 文档

使用 AWS 弹性灾难恢复为 Oracle JD Edwar EnterpriseOne ds 设置灾难恢复

创建者：Thanigaivel Thirumalai (AWS)

摘要

由自然灾害、应用程序故障或服务中断引发的灾难会损害收入，导致企业应用程序停机。为了减少此类事件的影响，灾难恢复（DR）计划对于采用JD Edwards EnterpriseOne 企业资源规划（ERP）系统和其他关键任务和业务关键型软件的公司至关重要。

这种模式解释了企业如何使用 AWS Elastic 灾难恢复作为其 JD Edwards EnterpriseOne 应用程序的灾难恢复选项。 它还概述了使用弹性灾难恢复故障转移和故障恢复为托管在 AWS 云中的亚马逊弹性计算云 (Amazon EC2) 实例上的数据库构建跨区域灾难恢复策略的步骤。

> **Note**
>
> 这种模式要求跨区域灾难恢复实施的主要和次要区域托管在 AWS 上。

Oracle JD Edwards EnterpriseOne 是一款集成式 ERP 软件解决方案，适用于各行各业的大中型公司。

AWS Elastic Daser Recovery 使用经济实惠的存储、最少的计算和恢复，通过快速、可靠地 point-in-time恢复本地和基于云的应用程序，最大限度地减少停机时间和数据丢失。

AWS 提供了四种核心灾难恢复架构模式。本文档重点介绍如何使用指示灯策略进行设置、配置和优化。此策略可帮助您创建低成本灾难恢复环境，在该环境中，您可以先配置复制服务器，以从源数据库复制数据，而只有在开始灾难恢复演练和恢复时才配置实际的数据库服务器。此策略省去了灾难恢复区域维护数据库服务器的成本。相反，您需要为用作复制服务器的小型 EC2 实例付费。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 在 Oracle 数据库或 Microsoft SQL Server 上运行的 JD Edwards EnterpriseOne 应用程序，受支持的数据库在托管 EC2 实例上处于运行状态。此应用程序应包括安装在一个 AWS 区域中的所有 JD Edwards EnterpriseOne 基础组件（企业服务器、HTML 服务器和数据库服务器）。
- 用于设置弹性灾难恢复服务的 AWS Identity and Access Management (IAM) 角色。

- 根据所需连接设置配置用于运行 Elastic Disaster Recovery 的网络。

限制

- 您可以使用此模式复制所有层，除非数据库托管在 Amazon Relational Database Service (Amazon RDS) 上，在这种情况下，我们建议您使用 Amazon RDS 的跨区域复制功能。
- Elastic 灾难恢复与 CloudEndure 灾难恢复不兼容，但您可以从 CloudEndure 灾难恢复中升级。有关更多信息，请参阅 Elastic Disaster Recovery 文档中的常见问题。
- Amazon Elastic Block Store (Amazon EBS) 限制您的快照拍摄速率。使用 Elastic Disaster Recovery，您可在单个 Amazon Web Services account 中最多复制 300 台服务器。若要复制更多服务器，您可以使用多个 Amazon Web Services account 或多个目标 AWS 区域。（您必须为每个账户和地区分别设置 Elastic Disaster Recovery。）有关更多信息，请参阅 Elastic Disaster Recovery 文档中的最佳实践。
- 源工作负载（JD Edwards EnterpriseOne 应用程序和数据库）必须托管在 EC2 实例上。这种模式不支持本地或其他云环境的工作负载。
- 这种模式侧重于JD Edwards的 EnterpriseOne 组件。完整的灾难恢复和业务连续性计划 (BCP) 应包含其他核心服务，包括：
  - 网络（虚拟私有云、子网和安全组）
  - Active Directory
  - Amazon WorkSpaces
  - Elastic Load Balancing
  - 托管式数据库服务，比如 Amazon Relational Database Service（Amazon RDS）

有关先决条件、配置和限制的更多信息，请参阅 Elastic Disaster Recovery 文档。

产品版本

- Oracle JD Edwards EnterpriseOne （基于甲骨文最低技术要求的 Oracle 和 SQL Server 支持的版本）

架构

目标技术堆栈

- 单个区域和单个虚拟私有云（VPC）用于生产和非生产，第二区域用于灾难恢复

- 单一可用区可以确保服务器之间的低延迟
- 应用程序负载均衡器用于分配网络流量，以提高应用程序在多个可用区的可扩展性和可用性
- Amazon Route 53，用于提供域名系统（DNS）配置
- 亚马逊 WorkSpaces 将为用户提供云端桌面体验
- Amazon Simple Storage Service (Amazon S3)，用于存储备份、文件和对象
- Amazon CloudWatch 用于应用程序日志、监控和警报
- Amazon Elastic Disaster Recovery，用于灾难恢复

目标架构

下图显示了 EnterpriseOne 使用 Elastic 灾难恢复的 JD Edwards 跨区域灾难恢复架构。

过程

以下是对该进程的高度回顾。有关详细信息，请参阅操作说明部分。

- Elastic Disaster Recovery 复制从初始同步开始。在初始同步期间，AWS Replication Agent 会将所有数据从源磁盘复制至暂存区域子网中的相应资源。
- 初始同步完成后，连续复制将无限期继续。
- 安装代理并开始复制后，您可以查看 EC2 启动参数，包括服务特定的配置和 Amazon 启动模板。当源服务器被指示为准备恢复时，即可启动实例。
- 当 Elastic Disaster Recovery 发出一系列 API 调用，以开始启动操作时，将根据您的启动设置立即在 AWS 上启动恢复实例。该服务会在启动期间自动启动转换服务器。
- 转换完成和可供使用后，新实例将在 AWS 上启动。启动时的源服务器状态通过与已启动实例关联的卷表示。转换过程包括更改驱动程序、网络和操作系统许可证，确保实例在 AWS 上以原生方式启动。
- 启动后，新建卷将不再与源服务器保持同步。AWS Replication Agent 会继续定期将对源服务器的更改复制到暂存区域卷，但启动的实例并未反映这些更改。
- 启动新的演练或恢复实例时，数据始终反映在源服务器复制到暂存区域子网的最新状态中。
- 当源服务器被标记为准备恢复时，您可启动实例。

> **Note**
>
> 该过程是双向的：从主 AWS 区域故障转移到灾难恢复区域，以及在主站点恢复后故障恢复到主站点。您可通过以完全编排的方式，将数据从目标计算机复制回源计算机的方向，反过来为失效自动恢复做好准备。

此模式中描述的此进程的好处包括：

- 灵活性：复制服务器根据数据集和复制时间进行横向扩展和横向缩减，因此您可在不中断源工作负载或复制的情况下执行灾难恢复测试。
- 可靠性：复制功能强大、无中断且可持续。
- 自动化：此解决方案为测试、恢复和失效自动恢复提供了统一、自动化流程。
- 成本优化：您只能复制所需卷并为其付费，并且只有在灾难恢复站点的计算资源被激活后，才可以为这些资源付费。您可将成本优化的复制实例（我们建议您使用计算优化型实例类型）用于多个数据源，也可以使用具有大 EBS 卷的单个源。

自动化和扩缩

当您大规模执行灾难恢复时，JD Edwards EnterpriseOne 服务器将依赖环境中的其他服务器。例如：

- 启动时连接到 JD Edwards EnterpriseOne 支持的数据库的 JD Edwards EnterpriseOne 应用程序服务器依赖该数据库。
- 需要身份验证且需要在启动时连接到域控制器才能启动服务的 JD Edwards EnterpriseOne 服务器依赖于域控制器。

因此，我们建议自动执行失效转移任务。例如，您可以使用 AWS Lambda 或 AWS Step Functions 自动执行 JD Edwards EnterpriseOne 启动脚本和负载均衡器更改，以自动执行故障转移过程。 end-to-end有关更多信息，请参阅博客文章使用 AWS Elastic Disaster Recovery 创建可扩展的灾难恢复计划。

工具

Amazon Web Services

- Amazon Elastic Block Store (Amazon EBS) 提供用于实例的块级存储卷。 EC2

- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- AWS Elastic Disaster Re covery 使用经济实惠的存储、最少的计算和恢复，可以快速、可靠地point-in-time恢复本地和基于云的应用程序，从而最大限度地减少停机时间和数据丢失。
- 借助 Amazon Virtual Private Cloud(Amazon VPC) ，您可完全控制自己的虚拟网络环境，包括资源放置、连接和安全。

最佳实践

一般最佳实践

- 制定书面计划，说明在发生真正恢复事件时该怎么做。
- 正确设置 Elastic 灾难恢复后，创建一个 AWS CloudFormation 模板，以便在需要时按需创建配置。确定服务器和应用程序的启动顺序，并将其记录至恢复计划。
- 定期进行演练（适用标准的 Amazon EC2 费率）。
- 使用 Elastic Disaster Recovery 控制台或以编程方式监控正在进行的复制的运行状况。
- 保护 point-in-time快照并在终止实例之前进行确认。
- 为 AWS Replication Agent 安装创建 IAM 角色。
- 在真实灾难恢复场景中为恢复实例启动终止保护。
- 对于启动恢复实例的服务器，请勿在 Elastic Disaster Recovery控制台中使用断开与 AWS 连接操作，即使是真实的恢复事件也是如此。执行断开连接会终止与这些源服务器相关的所有复制资源，包括您的 point-in-time (PIT) 恢复点。
- 更改 PIT 策略，以更改快照保留天数。
- 在 Elastic Disaster Recovery 启动设置中编辑启动模板，为目标服务器设置正确的子网、安全组和实例类型。
- 使用 Lambda 或 Step Functions 自动更改 JD Edwards EnterpriseOne 启动脚本和负载均衡器，从而自动执行 end-to-end故障转移流程。

JD Edwards 的 EnterpriseOne 优化和注意事项

- 移PrintQueue入数据库。
- 移MediaObjects入数据库。
- 从批处理服务器和逻辑服务器中排除日志和临时文件夹。
- 从 Oracle WebLogic 中排除临时文件夹。

- 为失效转移后启动创建脚本。
- 排除 SQL Server 中的 tempdb。
- 排除 Oracle 临时文件。

操作说明

执行初始任务与配置

Task	描述	所需技能
设置复制网络。	在主 AWS 区域实施您的 JD Edwards EnterpriseOne 系统并确定用于灾难恢复的 AWS 区域按照 Elastic 灾难恢复文档中复制网络要求部分中的步骤来规划和设置您的复制和灾难恢复网络。	AWS 管理员
确定 RPO 和 RTO。	确定应用程序服务器与数据库的恢复时间目标（RTO）和恢复点目标（RPO）。	云架构师、灾难恢复架构师
为 Amazon EFS 启用复制。	如果适用，请使用 AWS、rs y nc 或其他适当的工具为共享文件系统（例如亚马逊弹性文件系统 (Amazon EFS)）启用从 AWS DataSync 主区域复制到灾难恢复区域。	云管理员
在灾难恢复情况下管理 DNS	确定进程，以在灾难回复演练或实际灾难恢复过程中更新域名系统（DNS）。	云管理员
为设置创建 IAM 角色。	按照 Elastic Disaster Recovery 文档的 Elastic Disaster Recovery 初始化和权限部分中的说明，创建 IAM 角色以	云管理员

Task	描述	所需技能
	初始化和管理 Amazon Web Services。	
设置 VPC 对等连接。	确保源和目标之间 VPCs 是对等的，并且可以互相访问。有关配置说明，请参阅 Amazon VPC 文档。	AWS 管理员

配置 Elastic Disaster Recovery 复制设置

Task	描述	所需技能
初始化 Elastic Disaster Recovery。	打开 Elastic Disaster Recovery 控制台，选择目标 AWS 区域（您将在其中复制数据并启动恢复实例），然后选择设置默认复制。	AWS 管理员
设置复制服务器。	1. 在设置复制服务器窗格，输入暂存区域子网和复制服务器实例类型。默认情况下选择 `t3.small`实例类型。根据您的要求配置该设置，并记得考虑实例定价。有关更多信息，请参阅 Amazon EC2 定价。 2. 在服务访问权限部分，选择查看详细信息以查看服务关联角色和在服务初始化期间创建的其他策略。 3. 选择下一步。	AWS 管理员
配置卷与安全组。	1. 在卷和安全组窗格中，为复制服务器选择 EBS 卷类	AWS 管理员

Task	描述	所需技能
	型，并将 Amazon EBS 加密设置为默认。 2. 选择始终使用 AWS Elastic Disaster Recovery 安全组，这样 Elastic Disaster Recovery 就会自动连接和监控默认安全组。 3. 选择下一步。	
配置其他设置。	1. 在其他设置窗格，配置数据路由和节流、PIT 策略和标签。 • 数据路由和节流控制数据如何从外部的服务器流向复制服务器。选择使用私有 IP 执行数据复制。否则，复制服务器将自动分配公有 IP，并且数据将通过公共 Internet 流动。 • 在时间点 (PIT) 策略部分，配置保留策略，以确定不需要快照的持续时间。原定设置的保留期为七天。 • 在标签部分，为由 Elastic Disaster Recovery 在您的 Amazon Web Services account 中创建的资源添加自定义标签。 2. 选择下一步，在下一个窗格中查看设置，然后选择创建默认值以创建默认模板。	AWS 管理员

安装 AWS Replication Agent

Task	描述	所需技能
创建一个 IAM 角色。	创建一个包含 `AWSElasticDisasterRecoveryAgentInstallationPolicy` 策略的 IAM 角色。在选择 AWS 访问类型部分中，启用程序访问。记下访问密钥 ID 和秘密访问密钥。在安装 AWS Replication Agent 期间，您将需要这些信息。	AWS 管理员
查看要求。	查看并完成安装 AWS Replication Agent 的 Elastic Disaster Recovery 文档中的先决条件。	AWS 管理员
安装 AWS Replication Agent。	按照操作系统的安装说明操作，然后安装 AWS Replication Agent。 • 对于 Microsoft Windows：下载安装文件并以管理员身份运行 .exe 文件。 按照提示完成安装。 • 对于 Linux：复制以下命令（按显示的顺序）并将其粘贴到您的 Secure Shell (SSH) 会话中。第一个命令为下载安装程序，第二个命令运行它。 `wget -O ./aws-replication-installer-init.py https://aws-elastic-disaster-recovery-us-west`	AWS 管理员

Task	描述	所需技能

	`-2.s3.amazonaws.com/latest/linux/aws-replication-installer-init.py`
	
	`sudo python3 aws-replication-installer-init.py`
	
	按照提示完成安装。
	
	Note
	更改 URL 以反映您所在的地区。
	
	对其余服务器重复上述步骤。	

监控复制。	返回 Elastic Disaster Recovery源服务器窗格，以监控复制状态。初始同步将花费一定的时间，具体取决于数据传输的大小。 源服务器完全同步后，服务器状态将更新为就绪。这意味着已在暂存区创建复制服务器，并且 EBS 卷已从源服务器复制到暂存区。	AWS 管理员

配置启动设置

Task	描述	所需技能
编辑启动设置。	若要更新演练和恢复实例的启动设置，请在 Elastic Disaster Recovery 控制台，选择源服务器，然后选择操作、编辑启动设置。或者，您可以从源服务器页面中选择要复制的源计算机，然后选择启动设置选项卡。此选项卡分为两个部分：“常规启动设置” 和 “EC2 启动模板”。	AWS 管理员
配置常规启动设置。	根据您的要求修改常规启动设置。 • 正确调整实例类型：如果您选择基本，Elastic Daser Recovery 将绕过您在亚马逊 EC2 启动模板中选择的实例类型，并根据源服务器的操作系统、CPU 和 RAM 自动选择实例类型。 • 复制私有 IP：选择是否希望 Elastic Daser Recovery 确保演练或恢复实例的私有 IP 与源服务器的私有 IP 相匹配。如果您选择 “是”，请确保您在 Amazon EC2 启动模板中设置的子网的 IP 范围包含私有 IP 地址。	AWS 管理员

Task	描述	所需技能
	有关更多信息，请参阅 Elastic Disaster Recovery 文档中的常规启动设置。	
配置 Amazon EC2 启动模板。	Elastic 灾难恢复使用 Amazon EC2 启动模板为每台源服务器启动演练和恢复实例。在安装 AWS Replication Agent 之后，系统会自动为您添加至 Elastic Disaster Recovery 的每台源服务器创建启动模板。 如果要将 Amazon EC2 启动模板与 Elastic 灾难恢复配合使用，则必须将其设置为默认启动模板。 有关更多信息，请参阅 Elastic 灾难恢复文档中的EC2 启动模板。	AWS 管理员

启动灾难恢复演练与失效转移

Task	描述	所需技能
启动演习	1. 在 Elastic Disaster Recovery 控制台，打开源服务器页面，验证源服务器的状态是否为就绪。 2. 选择要执行灾难恢复演练的所有源服务器。 3. 从“启动恢复作业”菜单中，选择“启动演练”，然后选择相应的 point-in-time快照。这将启动选定源服务器恢复	AWS 管理员

Task	描述	所需技能
	作业。您可以在恢复作业历史记录选项卡上监控作业的状态。 启动的演练实例还会显示在恢复实例页面。 **Note** 对源服务器的进一步更改将同步到复制服务器，而不是演练实例。 4. 测试与验证灾难演练实例。 5. 在恢复实例页面，选择演练实例，然后选择操作，断开与 AWS 连接。这将从恢复实例中删除 AWS Replication Agent，并从 Elastic Disaster Recovery 中删除与恢复实例关联的所有资源。 6. 选择删除恢复实例。这将从 Elastic Disaster Recovery 控制台中删除该实例的表示形式，并完全取消该实例与 Elastic Disaster Recovery 服务的关联。它不会删除底层 EC2 实例。 7. 从 Amazon EC2 控制台终止灾难恢复演练实例。	

Task	描述	所需技能
	有关更多信息，请参阅 Elastic Disaster Recovery 文档中的准备失效转移。	
验证演练。	在上一步中，您在灾难恢复区域启动了新目标实例。目标实例是源服务器副本，具体取决于您启动启动时拍摄的快照。 在此过程中，您将连接到 Amazon EC2 目标计算机以确认它们按预期运行。 1. 打开 Amazon EC2 控制台。 2. 选择实例（正在运行）。 3. 选择目标实例并记下其私有 IPv4 地址。 4. 确保您可以连接到 EC2 实例，并确保按预期复制 JD Edwards EnterpriseOne 和相关组件。	

Task	描述	所需技能
启动失效转移。	失效转移是指将流量从主系统重定向至辅助系统。Elastic Disaster Recovery通 过在 AWS 上启动恢复实例帮助您执行失效转移。启动恢复实例后，您可将流量从主系统重定向到这些实例。 1. 在 Elastic Disaster Recovery 控制台，打开源服务器页面，确认源服务器的准备恢复列显示为就绪，数据复制状态列是否显示正常。 2. 选择源服务器。从启动恢复作业菜单，选择启动恢复。 3. 选择要从中启动恢复实例的point-in-time快照，然后选择启动恢复。 这将启动恢复作业。您可在恢复实例页面上监控任务的状态。 4. 测试与验证恢复实例。如果需要，请调整 DNS 配置并将您的 JD Edwards EnterpriseOne 应用程序连接到数据库。 5. 现在，您可以断开源 JD Edwards EnterpriseOne 服务器的连接并停用该服务器，因为所有更改都已写入新的恢复实例。	AWS 管理员

Task	描述	所需技能
	6. 按照安装 AWS Replication Agent 操作说明中所述的过程，将恢复实例注册为灾难恢复区域的源服务器。 有关更多信息，请参阅 Elastic Disaster Recovery 文档中的执行失效转移。	

Task	描述	所需技能
启动失效自动恢复。	启动失效自动恢复的过程与启动失效转移的过程类似。 1. 在主区域中打开 Elastic Disaster Recovery 控制台。导航到恢复实例页面，选择演练实例，然后选择操作、断开与 AWS 连接、删除恢复实例。 2. 在灾难恢复区域中打开 Elastic Disaster Recovery 控制台。通过安装 AWS 复制代理，将您的新 JD Edwards EnterpriseOne 服务器注册为灾难恢复区域的源服务器。数据将与在新暂存子网中配置的新复制服务器同步。 **Note** 将新的 JD Edwards EnterpriseOne 服务器注册为源服务器后，您可能会在 Elastic 灾难恢复控制台中看到两个源服务器：一台服务器是从主 EC2 实例创建的，另一台是从恢复实例创建的新服务器。我们建议您正确标记服务器以避免混淆，并且最好将新	AWS 管理员

Task	描述	所需技能
	服务器添加至启动模板。 3. 要从主区域重启灾难恢复复制，请取消灾难恢复区域中启动的恢复实例与 Elastic Disaster Recovery 控制台的关联，并将该主机注册为主区域中的源服务器。 有关更多信息，请参阅 Elastic Disaster Recovery 文档中的执行失效自动恢复。	

Task	描述	所需技能
启动 JD Edwards EnterpriseOne 组件。	1. 通过登录 EnterpriseOne 数据库服务器启动 JD Edwards 数据库。 2. 数据库运行时，启动 JD Edwards EnterpriseOne 逻辑和批处理服务器。 3. 在 Web 服务器 WebLogic 上启动，然后在 JAS 服务器上启动 JAS 实例。 4. 从 WebLogic 配置服务器和 SM 控制台的服务器上启动。 5. 在服务上启动 SM 代理。 6. 确认登录 JD Edwards 的操作是否正 EnterpriseOne 常。 你需要在 Route 53 和 Application Load Balancer 中合并更改才能让 JD Edwards EnterpriseOne 链接起作用。 您可使用 Lambda、Step Functions 和 Systems Manager（运行命令）自动执行这些步骤。 **Note** Elastic 灾难恢复对托管操作系统和文件系统的源 EC2 实例 EBS 卷执行块级复制。使用 Amazon EFS 创建的共	京东爱德华兹数控系统 EnterpriseOne

Task	描述	所需技能
	享文件系统不属于此复制的组成部分。您可以使用 AWS 将共享文件系统复制到灾难恢复区域 DataSync，如第一篇故事中所述，然后将这些复制的文件系统挂载到灾难恢复系统中。	

故障排除

事务	解决方案
源服务器数据复制状态为已停止，复制延迟。如果您查看详细信息，则数据复制状态将显示未看到代理。	检查以确认停顿的源服务器是否在运行。 **Note** 如果源服务器出现故障，则复制服务器将自动终止。 有关延迟问题的更多信息，请参阅 Elastic Disaster Recovery 文档中的复制延迟问题。
扫描磁盘后，在 RHEL 8.2 中在源 EC2 实例中安装 AWS 复制代理失败。 `aws_replication_agent_installer.log` 显示缺少内核标头。	在 RHEL 8、CentOS 8 或 Oracle Linux 8 上安装 AWS Replication Agent 之前，请运行： `sudo yum install elfutils-libelf-devel` 有关更多信息，请参阅 Elastic Disaster Recovery 文档中的 Linux 安装要求。

事务	解决方案
在 Elastic Disaster Recovery 控制台，您会看到源服务器为就绪，但存在延迟，数据复制状态为已停止。 视 AWS Replication Agent 不可用的时间长短而定，状态可能表示延迟时间过长，但问题仍然存在。	使用操作系统命令确认 AWS Replication Agent 是否在源 EC2 实例中运行，或者确认该实例正在运行。 更正所有问题后，Elastic Disaster Recovery 将重新开始扫描。等到所有数据都已同步并且复制状态为正常，再开始灾难恢复演练。
初始复制延迟较高。在 Elastic Disaster Recovery 控制台，您可以看到源服务器的初始同步状态非常慢。	查看 Elastic Disaster Recovery 文档的复制延迟问题部分中记录的复制延迟问题。 因内部计算操作，复制服务器可能无法处理负载。在这种情况下，请在咨询 AWS Technical Support 团队后尝试升级实例类型。

相关资源

- AWS Elastic Disaster Recovery 用户指南
- 使用 AWS Elastic Disaster Recovery 创建可扩展的灾难恢复计划（AWS Blog 文章）
- AWS Elastic Disaster Recovery - 技术简介（AWS Skill Builder 课程；需要登录）
- AWS Elastic Disaster Recovery 快速入门指南

在多区域、多账户组织中设置 AWS CloudFormation 偏差检测

创建者：Ram Kandaswamy (AWS)

摘要

Amazon Web Services (AWS) 用户通常会寻找一种有效的方法来检测资源配置不匹配情况，包括 AWS CloudFormation 堆栈中的偏差，并尽快对其进行修复。使用时 AWS Control Tower 尤其如此。

此模式提供了一种规范的解决方案，通过使用整合的资源配置更改并根据这些更改生成结果来有效地解决问题。该解决方案专为在多个 AWS 区域、多个账户或两者组合中创建多个 AWS CloudFormation 堆栈的场景而设计。该解决方案目标包括以下各项：

- 简化偏差检测过程
- 设置通知和警报
- 设置合并报告

先决条件和限制

先决条件

- AWS Config 已在必须监控的所有地区和账户中启用

限制

- 生成的报告仅支持逗号分隔值 (CSV) 和 JSON 输出格式。

架构

下图显示了使用多个账户的 AWS Organizations 设置。 AWS Config 规则在账户之间进行通信。

该工作流程包括以下步骤：

1. 该 AWS Config 规则可检测偏差。
2. 在其他账户中发现的漂移检测结果将发送到管理账户。
3. Amazon CloudWatch 规则调用一个 AWS Lambda 函数。
4. Lambda 函数查询 AWS Config 规则以获取聚合结果。

5. Lambda 函数通知亚马逊简单通知服务 (Amazon SNS) Simple Notification Service，后者会发送有关偏差的电子邮件通知。

自动化和扩缩

此处介绍的解决方案可以针对其他区域和账户进行扩展。

工具

AWS 服务

- AWS Config提供了中 AWS 资源配置的详细视图 AWS 账户。这些信息包括资源之间的关联方式以及资源以前的配置方式，让您了解资源的配置和关系如何随着的时间的推移而更改。
- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和运行的应用程序 AWS 的指标。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。

操作说明

自动进行漂移检测 AWS CloudFormation

Task	描述	所需技能
创建聚合器。	1. 登录 AWS Management Console 并通过 https://console.aws.amazon.com/config 打开 AWS Config 控制台。 2. 在管理账户中创建聚合器。 3. 确保已开启数据复制，以便 AWS Config 可以从源帐户获取数据。 4. 选择所有适用的地区和账户。您可以根据选择帐户	云架构师

Task	描述	所需技能
	AWS Organizations。我们建议采用这种方法，因为组织中的新账户会自动成为聚合器的一部分。	
创建 AWS 托管规则。	添加`cloudformation-stack-drift-detection-check` AWS托管规则。该规则需要一个参数值：`cloudformationArn` 。 输入具有堆栈偏差检测权限的 IAM 角色 Amazon 资源名称（ARN）。该角色必须具有能够代入该角色 AWS Config 的信任策略。	云架构师
创建聚合器高级查询部分。	若要从多个来源获取偏差堆栈，请创建以下查询： `SELECT resourceId, configuration.driftInformation.stackDriftStatus WHERE resourceType = 'AWS::CloudFormation::Stack' AND configuration.driftInformation.stackDriftStatus IN ('DRIFTED')`	云架构师、开发人员

Task	描述	所需技能
自动运行查询和发布。	1. 使用附加的代码创建 Lambda 函数。Lambda 会将结果发布到 Lambda 函数中作为环境变量提供的 SNS 主题。 2. 要接收提醒，请创建 SNS 主题的电子邮件订阅。	云架构师、开发人员
创建 CloudWatch 规则。	创建基于计划的 CloudWatch 规则来调用 Lambda 函数，该函数负责发出警报。	云架构师

相关资源

资源

- 什么是 AWS Config？
- 多账户多区域数据聚合
- 检测堆栈和资源的非托管配置更改
- IAM：将 IAM 角色传递给特定角色 AWS 服务
- 什么是 Amazon SNS？

其他信息

注意事项

我们建议使用此模式中提供的解决方案，而不是使用涉及按特定时间间隔调用 API 的自定义解决方案，以便在每个 CloudFormation 堆栈或堆栈集上启动漂移检测。按特定时间间隔使用 API 调用的自定义解决方案可能会导致大量 API 调用并影响性能。由于 API 调用的次数较多，可能会出现限制。另一潜在问题是，如果仅根据计划识别资源更改，则检测会出现延迟。

由于堆栈集是由堆栈组成的，因此您可以使用此解决方案。堆栈实例详细信息也作为解决方案的一部分提供。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

成功导入 S3 存储桶作为 AWS CloudFormation 堆栈

创建者：Ram Kandaswamy (AWS)

摘要

如果您使用亚马逊网络服务 (AWS) 资源，例如亚马逊简单存储服务 (Amazon S3) Simple Service 存储桶，并且想要使用基础设施即代码 (IaC) 方法，则可以将资源导入 AW CloudFormation S 并将其作为堆栈进行管理。

此模式提供了成功将 S3 存储桶导入 AWS CloudFormation 堆栈的步骤。通过使用此模式的方法，您可以避免在单个操作中导入 S3 存储桶时可能出现的错误。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 现有的 S3 存储桶与 S3 存储桶策略。有关这方面的更多信息，请参阅 AWS 知识中心中的 “我应该使用哪个 S3 存储桶策略来遵守 AWS Config 规则 s3-bucket-ssl-requests-only”。
- 现有 AWS Key Management Service (AWS KMS) 密钥及其别名。有关这方面的更多信息，请参阅 AWS KMS 文档中的使用别名。
- `CloudFormation-template-S3-bucket`AWS CloudFormation 模板示例（附后），已下载到您的本地计算机。

架构

图表显示了以下工作流：

1. 用户创建一个 JSON 或 YAML 格式的 AWS CloudFormation 模板。
2. 该模板创建一个 AWS CloudFormation 堆栈来导入 S3 存储桶。
3. AWS CloudFormation 堆栈管理您在模板中指定的 S3 存储桶。

技术堆栈

- AWS CloudFormation

- AWS Identity and Access Management (IAM)
- AWS KMS
- Amazon S3

工具

- AWS CloudFormation — AWS CloudFormation 可帮助您以可预测的方式重复创建和配置 AWS 基础设施部署。
- AWS Identity and Access Management (IAM) – IAM 是一项 Web 服务，用于安全地控制对 AWS 资源的访问。
- AWS KMS – AWS Key Management Service (AWS KMS) 是一项扩展到云的加密和密钥管理服务。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项面向互联网的存储服务。

操作说明

将具有 AWS KMS key基于加密功能的 S3 存储桶作为 AWS CloudFormation 堆栈导入

Task	描述	所需技能

创建用于导入 S3 存储桶和 KMS 密钥的模板。	在您的本地计算机上，使用以下示例模板创建用于导入 S3 存储桶和 KMS 密钥的模板：
	`AWSTemplateFormatVersion: 2010-09-09`
	`Parameters:`
	`  bucketName:`
	`    Type: String`
	`Resources:`
	`  S3Bucket:`
	`    Type: 'AWS::S3::Bucket'`
		AWS DevOps

Task	描述	所需技能
	DeletionPolicy: Retain Properties: BucketName: !Ref bucketName BucketEncryption: ServerSideEncryptionConfiguration: - ServerSideEncryptionByDefault: SSEAlgorithm: 'aws:kms' KMSMasterKeyID: !GetAtt - KMSS3Encryption - Arn KMSS3Encryption: Type: 'AWS::KMS::Key' DeletionPolicy: Retain Properties: Enabled: true	

Task	描述	所需技能

	KeyPolicy: !Sub
	|-
	{
	"Id": "key-consolepolicy-3",
	"Version": "2012-10-17",
	"Statement": [
	{
	"Sid": "Enable IAM User Permissions",
	"Effect": "Allow",
	"Principal": {
	"AWS": ["arn:aws:iam::${AWS::AccountId}:root"]
	},
	"Action": "kms:*",
	"Resource": "*"
	}
	}	

Task	描述	所需技能

	<pre>]
	
	}
	
	EnableKey
	Rotation: true</pre>	

创建堆栈。	1. 登录 AWS 管理控制台，打开 AWS CloudFormation 控制台，选择查看堆栈，选择创建堆栈，然后选择使用现有资源（导入资源）。 2. 选择上传模板文件，然后上传您之前创建的模板文件。 3. 输入堆栈的名称，并根据您的要求配置其余选项。 4. 选择创建堆栈，等待堆栈的状态变为 IMPORT_COMPLETE 。	AWS DevOps

Task	描述	所需技能

创建 KMS 密钥别名。	1. 在 AWS CloudFormation 控制台上，选择堆栈，选择您之前创建的堆栈的名称，选择模板窗格，然后选择在设计器中查看。
	2. 将以下代码段添加到您的模板的 `Resource` 部分，然后选择创建堆栈并完成向导：
	
	`KMSS3EncryptionAlias:`
	
	`  Type: 'AWS::KMS ::Alias'`
	
	`  DeletionPolicy: Retain`
	
	`  Properties:`
	
	`  AliasName: alias/ S3BucketKey`
	
	`  TargetKeyId: !Ref KMSS3Encryption`
	
	有关这方面的更多信息，请参阅 AWS CloudFormation 文档中的 AWS CloudFormation 堆栈更新。
		AWS DevOps

Task	描述	所需技能

更新堆栈以包含 S3 存储桶策略。	1. 在 AWS CloudFormation 控制台上，选择堆栈，选择您之前创建的堆栈的名称，选择模板窗格，然后选择在设计器中查看。
	2. 将以下代码段添加到模板的 `Resource` 部分，然后选择创建堆栈并完成向导：
	
	`S3BucketPolicy:`
	
	`  Type: 'AWS::S3::BucketPolicy'`
	
	`  Properties:`
	
	`    Bucket: !Ref S3Bucket`
	
	`    PolicyDocument: !Sub |-`
	
	`      {`
	
	`  "Version": "2008-10-17",`
	
	`  "Id": "restricthttp",`
	
	`  "Statement": [`
	
	`  {`
	
	`  "Sid": "denyhttp",`
		AWS DevOps

Task	描述	所需技能

	"Effect": "Deny",
	"Principal": {
	"AWS": "*"
	},
	"Action": "s3:*",
	"Resource": ["arn:aws:s3:::${S3Bucket}","arn:aws:s3:::${S3Bucket}/*"],
	"Condition": {
	"Bool": {
	"aws:SecureTransport": "false"
	}
	}
	}
	]	

Task	描述	所需技能
	} **Note** 此 S3 存储桶策略有一个拒绝声明，用于限制不安全的 API 调用。	
更新密钥政策。	1. 在 AWS CloudFormation 控制台上，选择堆栈，选择您之前创建的堆栈的名称，选择模板窗格，然后选择在设计器中查看。 2. 修改模板的 KMS 资源，使其包含允许管理员管理 KMS 密钥的密钥策略。 3. 选择创建堆栈，选择下一步，然后根据您的要求完成向导。 有关更多信息，请参阅 AWS KMS 文档AWS KMS中的密钥策略。	AWS 管理员

Task	描述	所需技能

添加资源级标签。	1. 在 AWS CloudFormation 控制台上，选择堆栈，选择您之前创建的堆栈的名称，选择模板窗格，然后选择在设计器中查看。
	2. 将以下代码段添加到模板的 Amazon S3 资源 `Properties` 部分，然后选择创建堆栈并完成向导：
	`Tags:`
	`  - Key: createdBy`
	`    Value: Cloudformation`
		AWS DevOps

相关资源

- 将现有资源引入 AWS CloudFormation 管理
- AWS re: Invent 2017：深入了解 AWS CloudFormation（视频）

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS 在不同 AWS 区域的 Amazon EFS 文件系统之间同步数据 DataSync

由 Sarat Chandra Pothula (AWS) 和 Aditya Ambati (AWS) 创作

摘要

该解决方案提供了一个强大的框架，用于在不同 AWS 区域的 Amazon Elastic File System (Amazon EFS) 实例之间进行高效、安全的数据同步。这种方法具有可扩展性，可提供受控的跨区域数据复制。该解决方案可以增强您的灾难恢复和数据冗余策略。

通过使用 AWS Cloud Development Kit (AWS CDK)，此模式使用基础设施即代码 (IaC) 方法来部署解决方案资源。AWS CDK 应用程序部署了 AWS、 DataSync亚马逊 EFS、亚马逊虚拟私有云（亚马逊 VPC）和亚马逊弹性计算云（亚马逊 VPC EC2）的基本资源。此 IaC 提供了一个完全符合 AWS 最佳实践的可重复且受版本控制的部署流程。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- AWS 命令行界面 (AWS CLI) Line 2.9.11 或更高版本，已安装并配置
- AWS CDK 版本 2.114.1 或更高版本，已安装并已启动
- 已安装 NodeJS 版本 20.8.0 或更高版本

限制

- 该解决方案继承了 Amazon EFS 的限制，例如数据传输速率、大小限制和区域可用性。 DataSync 有关更多信息，请参阅 AWS DataSync 配额和 A mazon EFS 配额。
- 此解决方案仅支持 Amazon EFS。 DataSync 支持其他 AWS 服务，例如亚马逊简单存储服务 (Amazon S3) Service 和 Amazon for Lust FSx re。但是，此解决方案需要修改才能与其他服务同步数据。

架构

此解决方案部署了以下 AWS CDK 堆栈：

- Amazon VPC 堆栈 — 此堆栈在主要和次要 AWS 区域中设置虚拟私有云 (VPC) 资源，包括子网、互联网网关和 NAT 网关。
- Amazon EFS 堆栈 — 此堆栈将 Amazon EFS 文件系统部署到主区域和次要区域，并将它们连接到各自 VPCs的区域。
- Amazon EC2 堆栈 — 此堆栈在主要和次要区域启动 EC2 实例。这些实例配置为挂载 Amazon EFS 文件系统，从而允许它们访问共享存储。
- DataSync 位置堆栈 — 此堆栈使用名`DataSyncLocationConstruct`为的自定义构造在主要和次要区域中创建 DataSync 位置资源。这些资源定义了数据同步的端点。
- DataSync 任务堆栈 — 此堆栈使用名`DataSyncTaskConstruct`为的自定义结构在主区域中创建 DataSync 任务。此任务配置为使用 DataSync 源位置和目标位置在主区域与次要区域之间同步数据。

工具

Amazon Web Services

- AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，可帮助您在代码中定义和预调配 Amazon Web Services Cloud 基础设施。
- AWS DataSync 是一项在线数据传输和发现服务，可帮助您在 AWS 存储服务之间移动文件或对象数据。
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- Amazon Elastic File System (Amazon EFS) 可帮助您在 Amazon Web Services Cloud 中创建和配置共享文件系统。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。

代码存储库

此模式的代码可在 GitHub Amazon EFS 跨区域 DataSync 项目存储库中找到。

最佳实践

遵循中使用 AWS CDK 创建 IaC 项目的最佳实践中描述 TypeScript 的最佳实践。

操作说明

部署 AWS CDK 应用程序

Task	描述	所需技能
克隆项目存储库。	输入以下命令以克隆 Amazon EFS 跨区域 DataSync 项目存储库。 `git clone https://github.com/aws-samples/aws-efs-crossregion-datasync.git`	AWS DevOps
安装 npm 依赖项。	输入以下命令。 `npm ci`	AWS DevOps

选择主要和次要区域。	在克隆的存储库中，导航到该`src/infa`目录。在`Launcher.ts` 文件中，更新`PRIMARY_AWS_REGION` 和`SECONDARY_AWS_REGION` 值。使用相应的地区代码。
	`const primaryRegion = { account: account, region: '<PRIMARY_AWS_REGION>' };`
	`const secondaryRegion = { account: account, region: '<SECONDARY_AWS_REGION>' };`
		AWS DevOps

引导环境。	输入以下命令引导您要使用的 AWS 账户和 AWS 区域。	AWS DevOps

Task	描述	所需技能
	`cdk bootstrap <aws_account>/<aws_region>` 有关更多信息，请参阅 AWS CDK 文档中的引导。	
列出 AWS CDK 堆栈。	输入以下命令以查看应用程序中的 AWS CDK 堆栈列表。 `cdk ls`	AWS DevOps
合成 AWS CDK 堆栈。	输入以下命令，为 AWS CDK 应用程序中定义的每个堆栈生成一个 AWS CloudFormation 模板。 `cdk synth`	AWS DevOps
部署 AWS CDK 应用程序。	输入以下命令将所有堆栈部署到您的 AWS 账户，无需手动批准即可进行任何更改。 `cdk deploy --all --require-approval never`	AWS DevOps

验证部署

Task	描述	所需技能
登录主区域中的 EC2 实例。	1. 使用会话管理器（AWS Systems Manager 的一项功能）登录主区域中的 EC2 实例。有关说明，请参阅使用 AWS Systems Manager 会	AWS DevOps

Task	描述	所需技能
	话管理器连接到您的 Linux 实例。 2. 将目录更改为 Amazon EFS 挂载路径。 `cd /mnt/efs`	

创建临时文件。	输入以下命令在 Amazon EFS 挂载路径中创建临时文件。
	`sudo dd if=/dev/zero \`
	`of=tmptst.dat \`
	`bs=1G \`
	`seek=5 \`
	`count=0`
	
	`ls -lrt tmptst.dat`
		AWS DevOps

启动 DataSync 任务。	输入以下命令将临时文件从主区域复制到辅助区域，其中<ARN-task> 是您的 DataSync 任务的 Amazon 资源名称 (ARN)。
	`aws datasync start-tas`
	`k-execution \`
	`--task-arn <ARN-`
	`task>`
	
	该命令按以下格式返回任务执行的 ARN。
	`arn:aws:datasync:<`
	`region>:<account-I`
	`D>:task/task-execu`
	`tion/<exec-ID>`
		AWS DevOps

Task	描述	所需技能
检查数据传输的状态。	输入以下命令来描述 DataSync 执行任务，其中<ARN-task-execution> 是任务执行的 ARN。 `aws datasync describe-task-execution \ --task-execution-arn <ARN-task-execution>` 当、和VerifyStatus 都具有值时 PrepareStatus TransferStatus ，DataSync 任务就完成了SUCCESS。	AWS DevOps
登录辅助区域中的 EC2 实例。	1. 使用会话管理器（AWS Systems Manager 的一项功能）登录辅助区域中的 EC2 实例。有关说明，请参阅使用 AWS Systems Manager 会话管理器连接到您的 Linux 实例。 2. 将目录更改为 Amazon EFS 挂载路径。 `cd /mnt/efs`	AWS DevOps
验证复制。	输入以下命令以验证临时文件是否存在于 Amazon EFS 文件系统中。 `ls -lrt tmptst.dat`	AWS DevOps

相关资源

AWS 文档

- AWS CDK API 参考
- 使用 Amazon EFS 配置 AWS DataSync 转账
- 排除 AWS DataSync 转账问题

其他 AWS 资源

- AWS DataSync FAQs

使用 LocalStack 和 Terraform 测试来测试 AWS 基础架构

由 Ivan Girardi (AWS) 和 Ioannis Kalyvas (AWS) 创作

摘要

此模式可帮助您在本地测试 Terraform AWS 中的基础设施即代码 (IaC)，而无需在环境中配置基础架构。 AWS 它将 Terraform Tests 框架与。LocalStackD LocalStack ocker 容器提供了一个模拟各种本地开发环境。 AWS 服务这可以帮助您测试和迭代基础架构部署，而不会产生成本。 AWS Cloud

此解决方案具有以下优势：

- 成本优化 — 运行测试 LocalStack 无需使用 AWS 服务。这样可以防止您产生与创建、操作和修改这些 AWS 资源相关的成本。
- 速度和效率 — 本地测试通常也比部署 AWS 资源更快。这种快速反馈回路加快了开发和调试速度。由于在本地 LocalStack 运行，因此无需互联网连接即可开发和测试 Terraform 配置文件。您可以在本地调试 Terraform 配置文件并立即获得反馈，从而简化开发过程。
- 一致性和可重复性 — LocalStack 为测试提供一致的环境。这种一致性有助于确保无论外部 AWS 变化或网络问题如何，测试都会产生相同的结果。
- 隔离 — 使用进行测试 LocalStack 可防止意外影响实时 AWS 资源或生产环境。这种隔离可以安全地进行各种配置的实验和测试。
- 自动化 — 与持续集成和持续交付 (CI/CD) 管道集成可帮助您自动测试 Terraform 配置文件。在部署之前，管道会对 IaC 进行全面测试。
- 灵活性-您可以模拟不同的和服务配置 AWS 区域 AWS 账户，以更紧密地匹配您的生产环境。

先决条件和限制

先决条件

- 安装 Docker
- 启用对默认 Docker 套接字的访问权限 (`/var/run/docker.sock`)。有关更多信息，请参阅 LocalStack 文档。
- 安装 Docker Comp ose
- 安装 Terraform 版本 1.6.0 或更高版本
- 安装 Terraform CLI

- 配置 Terraform AWS 提供者
- （可选）安装并配置 AWS Command Line Interface (AWS CLI)。有关如何 AWS CLI 与一起使用的示例 LocalStack，请参阅使用 GitHub 测试 AWS 基础架构 LocalStack 和 Terraform Te sts 存储库。

限制

- 此模式为测试亚马逊简单存储服务 (Amazon S3) Simple Service AWS Lambda、 AWS Step Functions、和亚马逊 DynamoDB 资源提供了明确的示例。但是，您可以扩展此解决方案以包括其他 AWS 资源。
- 此模式提供了在本地运行 Terraform Tests 的说明，但是您可以将测试集成到任何 CI/CD 管道中吗。
- 此模式提供了使用 LocalStack 社区图片的说明。如果您使用的是 LocalStack Pro 镜像，请参阅 LocalStack Pro 文档。
- LocalStack 为不同 AWS APIs的人提供仿真服务。有关完整列表，请参阅AWS 服务功能覆盖范围。某些高级功能可能需要订阅 LocalStack Pro。

架构

下图显示了此解决方案的架构。主要组件是源代码存储库、CI/CD 管道和 LocalStack Docker 容器。LocalStack Docker 容器在 AWS 服务 本地托管以下内容：

- 用于存储文件的 Amazon S3 存储桶
- Amazon CloudWatch 用于监控和记录
- 用于运行无服务器代码的 AWS Lambda 函数
- 一台用于编排多步骤工作流程的 AWS Step Functions 状态机
- 用于存储 NoSQL 数据的 Amazon DynamoDB 表

图表显示了以下工作流：

1. 您可以将 Terraform 配置文件添加并提交到源代码存储库。
2. CI/CD 管道会检测更改并启动生成过程以进行静态 Terraform 代码分析。管道构建并运行 LocalStack Docker 容器。然后，管道开始测试过程。
3. 管道将对象上传到 LocalStack Docker 容器中托管的 Amazon S3 存储桶。

4. 上传对象会调用一个函数。 AWS Lambda
5. Lambda 函数将亚马逊 S3 事件通知存储在日志中。 CloudWatch
6. Lambda 函数启动 AWS Step Functions 状态机。
7. 状态机将 Amazon S3 对象的名称写入 DynamoDB 表中。
8. CI/CD 管道中的测试过程会验证上传对象的名称是否与 DynamoDB 表中的条目相匹配。它还会验证 S3 存储桶是否已使用指定名称部署，并且该 AWS Lambda 函数已成功部署。

工具

AWS 服务

- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和运行的应用程序 AWS 的指标。
- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Storage Service (Amazon S3)是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Step Functions是一项无服务器编排服务，可帮助您组合 AWS Lambda 功能和其他功能 AWS 服务 来构建关键业务应用程序。

其他工具

- Docker 是一组平台即服务（PaaS）产品，它们使用操作系统级别的虚拟化技术在容器中交付软件。
- Docker Compose 是一款用于定义和运行多容器应用程序的工具。
- LocalStack是在单个容器中运行的云服务模拟器。通过使用 LocalStack，您可以在使用的本地计算机上运行工作负载 AWS 服务，而无需连接到 AWS Cloud。
- Terraform 是一款 IaC 工具 HashiCorp ，可帮助您创建和管理云和本地资源。
- Terraform Tests 可帮助您通过类似于集成或单元测试的测试来验证 Terraform 模块的配置更新。

代码存储库

此模式的代码可在使用 LocalStack 和 Terraform Tests 存储库的 GitHub 测试 AWS 基础架构中找到。

最佳实践

- 此解决方案测试在 Terraform 配置文件中指定的 AWS 基础架构，并且不会在中部署这些资源。AWS Cloud如果要部署资源，请遵循最低权限原则（IAM 文档）并正确配置 Terraform 后端（Terraform 文档）。
- LocalStack 在 CI/CD 管道中集成时，我们建议您不要在权限模式下运行 LocalStack Docker 容器。有关更多信息，请参阅运行时权限和 Linux 功能（Docker 文档）和自管理运行器的安全性（GitLab 文档）。

操作说明

部署解决方案

Task	描述	所需技能
克隆存储库。	在 bash Shell 中输入以下命令：这将使用以下内容克隆测试 AWS 基础架构 LocalStack 和 Terraform Te sts 存储库：GitHub `git clone https://github.com/aws-samples/localstack-terraform-test.git`	DevOps 工程师

运行 LocalStack 容器。	1. 输入以下命令以导航到克隆的存储库：
	`cd localstack-terraform-test`
	2. 输入以下命令以分离模式启动 LocalStack Docker 容器：
	`docker-compose up -d`
		DevOps 工程师

Task	描述	所需技能
	3. 等待 LocalStack Docker 容器开始运行。	
初始化 Terraform。	输入以下命令来初始化 Terraform： `terraform init`	DevOps 工程师

运行 Terraform 测试。	1. 输入以下命令来运行 Terraform 测试：
	`terraform test`
	2. 确认所有测试都成功完成。该输出应该类似于以下内容：
	`Success! 3 passed, 0 failed.`
		DevOps 工程师

清理资源。	输入以下命令销毁 LocalStack 容器： `docker-compose down`	DevOps 工程师

故障排除

事务	解决方案

`Error: reading DynamoDB Table Item (Files|README.md): empty` 运行`terraform test`命令时的结果。	1. 重新输入`terraform test`命令。
	2. 如果这不能解决错误，请编辑 main.tf 文件以将睡眠超时时间增加到大于 15 秒的值：
	`resource "time_sleep" "wait" {`
	`  create_duration = "15s"`

事务	解决方案

	triggers = {
	s3_object = local.key_json
	}
	}

相关资源

- Terraform 入门： AWS CloudFormation 专家指南（AWS 规范性指导）AWS CDK
- 使用 Terraform AWS 提供程序的最佳实践（AWS 规范性指南）
- Terraform CI/CD 并 AWS 使用新的 Terraform 测试框架进行测试（博客文章）AWS
- 使用来自 AWS Marketplace（AWS 博客文章）的 LocalStack 云仿真器加速软件交付

其他信息

与 GitHub 操作集成

你可以使用 Actions 在 CI/CD 管道中集成 LocalStack 和 Terraform 测试。 GitHub 有关更多信息，请参阅GitHub 操作文档。以下是 GitHub 操作配置文件示例：

```
name: LocalStack Terraform Test

on:
  push:
    branches:
      - '**'

  workflow_dispatch: {}

jobs:
  localstack-terraform-test:
    runs-on: ubuntu-latest

    steps:
    - uses: actions/checkout@v4

    - name: Build and Start LocalStack Container
      run: |
```

```
        docker compose up -d

    - name: Setup Terraform
      uses: hashicorp/setup-terraform@v3
      with:
        terraform_version: latest

    - name: Run Terraform Init and Validation
      run: |
        terraform init
        terraform validate
        terraform fmt --recursive --check
        terraform plan
        terraform show

    - name: Run Terraform Test
      run: |
        terraform test

    - name: Stop and Delete LocalStack Container
      if: always()
      run: docker compose down
```

将 SAP Pacemaker 集群从升级到 ENSA1 ENSA2

由 Gergely Cserdi (AWS) 和 Balazs Sandor Skublics (AWS) 编写

摘要

此模式说明了将基于独立入队服务器 () ENSA1 的 SAP Pacemaker 群集升级到的步骤和注意事项。ENSA2此模式中的信息适用于 SUSE Linux Enterprise Server (SLES) 和 Red Hat Enterprise Linux (RHEL) 操作系统。

SAP NetWeaver 7.52 或 S/4HANA 1709 及更早版本上的 Pacemaker 集群在 ENSA1 架构上运行，并且经过专门配置。 ENSA1如果你在 Amazon Web Services (AWS) 上运行 SAP 工作负载 ENSA2，并且有兴趣迁移到，你可能会发现 SAP、SUSE 和 RHEL 文档没有提供全面的信息。此模式描述了重新配置 SAP 参数和 Pacemaker 集群以从升级到所需的技术步骤。 ENSA1 ENSA2它提供了 SUSE 系统的示例，但对于 RHEL 集群，概念是相同的。

注意： ENSA1 和 ENSA2 是仅适用于 SAP 应用程序的概念，因此此模式中的信息不适用于 SAP HANA 或其他类型的集群。

严格来说， ENSA2 可以搭配或不使用 Enqueue Replicator 2 一起使用。但是，高可用性 (HA) 和故障转移自动化（通过集群解决方案）需要 Enqueue Replicator 2。此模式使用ENSA2 集群一词来指具有独立入队服务器 2 和 Enqueue Replicator 2 的集群。

先决条件和限制

先决条件

- 一个 ENSA1基于工作的集群，在 SLES 或 RHEL 上使用 Pacemaker 和 Corosync。
- 至少有两个亚马逊弹性计算云 (Amazon EC2) 实例在其中运行 (ABAP) SAP 中央服务 (ASCS/SCS) 和 Enqueue 复制服务器 (ERS) 实例。
- 了解管理 SAP 应用程序和集群的知识。
- 以根用户身份访问 Linux 环境。

限制

- ENSA1基于群集仅支持双节点架构。
- ENSA2基于集群不能部署到 7.52 之前 NetWeaver 的 SAP 版本。

- EC2 集群中的实例应位于不同的 AWS 可用区中。

产品版本

- SAP NetWeaver 版本 7.52 或更高版本
- 从 S/4HANA 2020 开始，仅支持 ENSA2 集群
- 内核 7.53 或更高版本，它支持 ENSA2 和 Enqueue Replicator 2
- 适用于 SAP 应用程序的 SLES 版本 12 或更高版本
- 适用于 SAP 的高可用性 (HA) RHEL 版本 7.9 或更高版本

架构

源技术堆栈

- 带有 SAP 内核 NetWeaver 7.53 或更高版本的 SAP 7.52
- SLES 或 RHEL 操作系统

目标技术堆栈

- 搭载 SAP 内核 NetWeaver 7.53 或更高版本的 SAP 7.52，包括搭载 ABAP 平台的 S/4HANA 2020
- SLES 或 RHEL 操作系统

目标架构

下图显示了基于集群的 ASCS/SCS 和 ERS 实例的高可用性配置。 ENSA2

ENSA1 和 ENSA2 集群的比较

SAP ENSA2 作为继任者推出 ENSA1。 ENSA1基于集群的集群支持双节点架构，由于集群中没有其他节点可供故障切换，因此该ASCS/SCS instance fails over to ERS when an error occurs. This limitation stems from how the ASCS/SCS instance regains the lock table information from the shared memory of the ERS node after failover. ENSA2-based clusters with Enqueue Replicator 2 eliminate this limitation, because the ASCS/SCS instance can collect the lock information from the ERS instance over the network. ENSA2-based clusters can have more than two nodes, because the ASCS/SCS instance is no longer required to fail over to the ERS node. (However, in a two-

node ENSA2 cluster environment, the ASCS/SCS实例仍将故障转移到 ERS 节点。 ） ENSA2 从 SAP 内核 7.50 开始受支持，但有一些限制。对于支持 Enqueue Replicator 2 的 HA 设置，最低要求为 NetWeaver 7.52（参见 SAP OSS Note 2630416）。默认情况下，S/4HANA 1809 自带推荐的 ENSA2 架构，而 S/4HANA 仅 ENSA2 支持 2020 版开始。

自动化和扩缩

目标架构中的 HA 集群可使 ASCS 自动故障转移到其他节点。

迁移到 ENSA2基于基础的集群的场景

升级到 ENSA2基于基础的集群有两种主要方案：

- 场景 1：假设你的 SAP 版本和内核版本支持，则选择在不进行 SAP 升级或 S/4HANA 转换 ENSA2 的情况下升级到。 ENSA2
- 场景 2：在升级或转换（例如，升级到 S/4HANA 1809 或更高版本）的过程中，您使用 SUM 迁移到。 ENSA2

操作说明部分涵盖了这两种场景的步骤。第一种情况要求您在更改集群配置之前手动设置与 SAP 相关的参数。 ENSA2在第二种场景中，二进制文件和 SAP 相关参数由 SUM 部署，您唯一剩下的任务是更新 HA 的集群配置。我们仍然建议您在使用 SUM 后验证 SAP 参数。在大多数情况下，S/4HANA 转换是集群升级的主要原因。

工具

- 对于操作系统包管理器，我们建议使用 Zypper（适用于 SLES）或 YUM（适用于 RHEL）工具。
- 对于集群管理，我们建议使用 crm (适用于 SLES) 或 pcs (适用于 RHEL) shell。
- SAP 实例管理工具，例如 SAPControl。
- （可选）用于 S/4HANA 转换升级的 SUM 工具。

最佳实践

- 有关在 AWS 上使用 SAP 工作负载的最佳实践，请参阅 AWS Well-Architected Framework 的 SAP Lens。
- 考虑一下您的 ENSA2 多节点架构中的群集节点数量（奇数或偶数）。
- 按照 SAP S/4-HA-CLU 1.0 认证标准为 SLES 15 设置 ENSA2 集群。
- 在升级到之前，请务必保存或备份现有的集群和应用程序状态 ENSA2。

操作说明

手动配置 SAP 参数 ENSA2（仅限方案 1）

Task	描述	所需技能

配置默认配置文件中的参数。	如果要在 ENSA2 保持相同 SAP 版本的情况下升级到，或者目标版本默认为 ENSA1，请将默认配置文件（DEFAULT.PFL 文件）中的参数设置为以下值。
	`enq/enable=TRUE`
	`enq/serverhost=sapascsvirt`
	`enq/serverinst=10`
	`(instance number of ASCS/SCS instance)`
	`enque/process_location=REMOTESA`
	`enq/replicatorhost=sapersvirt`
	`enq/replicatorinst=11`
	`(instance number of ERS instance)`
	其中 `sapascsvirt` 是 ASCS 实例的虚拟主机名，`sapersvirt` 是 ERS 实例的虚拟主机名。您可以更改这些设置以适合您的目标环境。
	Note
	要使用此升级选项，您的 SAP 版本和内核版本必须支持 ENSA2
		SAP

Task	描述	所需技能
	和 Enqueue Replicato r 2。	

Task	描述	所需技能

配置 ASCS/SCS 实例配置文件。	如果您想在 ENSA2 保持相同 SAP 版本的情况下升级到，或者目标版本默认为 ENSA1，请在 ASCS/SCS 实例配置文件中设置以下参数。
	
	配置文件中定义 ENSA1 的部分如下所示。
	
	`#----------`
	`----------`
	`----------`
	`---------`
	`Start SAP enqueue server`
	`#----------`
	`----------`
	`----------`
	`-------`
	`_EN = en.sap$(S`
	`APSYSTEMNAME)$(INS`
	`TANCE_NAME)`
	`Execute_04 = local rm -`
	`f $(_EN)`
	`Execute_05 = local ln -`
	`s -f $(DIR_EXECUTABLE)/`
	`enserver$(FT_EXE)`
	` $(_EN)`
	`Start_Program_01 =`
	` local $(_EN) pf=$(_PF)`
	
	要重新配置此部分，请执行以下操作： ENSA2
	
	1. 根据 SAP 的最新信息，将 _EN 程序前缀更改为 _ENQ（OSS Note 2501860；需要一个 SAP
		SAP

Task	描述	所需技能
	ONE Support Launchpad 用户账户）。 2. 将排队服务器的二进制文件从 `enserver` 更改为 `enq_server` 。 3. 将新参数 `enq/server/replication/enable` 设置为 TRUE。 4. 确保 `Autostart = 0`。 更改后，此配置文件部分将如下所示。 `#------------------ ------------------ ------------------ --------- Start SAP enqueue server #------------------ ------------------ ------------------ ------- _ENQ = enq.sap$(SAPSYSTEMNAME)$(IN STANCE_NAME) Execute_04 = local rm - f $(_ENQ) Execute_05 = local ln - s -f $(DIR_EXECUTABLE)/ enq_server$(FT_EXE) $(_ENQ) Start_Program_01 = local $(_ENQ) pf= $(_PF) ... enq/server/replic ation/enable = TRUE`	

Task	描述	所需技能
	`Autostart = 0` **⚠ Important** _ENQ不得启用重新启动选项。如果 _ENQ 设置为 RestartProgram_01 ，则将其更改为 StartProgram_01 。这可以防止 SAP 重新启动服务或干扰集群管理的资源。	

Task	描述	所需技能
配置 ERS 配置文件。	如果您想在 ENSA2 保持相同 SAP 版本的情况下升级到，或者您的目标版本默认为 ENSA1，请在 ERS 实例配置文件中设置以下参数。 查找定义排队复制器的部分。它类似于以下内容。 `#----------------- ----------------- ------------------- Start enqueue replicati on server #------------------- ----------------- ----------------- _ER = er.sap$(S APSYSTEMNAME)$(INS TANCE_NAME) Execute_03 = local rm - f $(_ER) Execute_04 = local ln - s -f $(DIR_EXECUTABLE)/ enrepserver$(FT_EXE) $(_ER) Start_Program_00 = local $(_ER) pf=$(_PF) NR=$(SCSID)` 要为 Enqueue Replicator 2 重新配置此部分，请执行以下操作： 1. 根据 SAP 的最新说明，将 _ER 程序前缀更改为 _ENQR（OSS Note	SAP

Task	描述	所需技能
	2501860；需要一个 SAP ONE Support Launchpad 用户账户）。 2. 将排队复制器的二进制文件更改为 enq_replicator 而不是 enrepserver 。 3. 确保 Autostart = 0。 更改后，此配置文件部分应如下所示。 `#------------------ ------------------ ------------------- Start enqueue replicati on server #------------------- ------------------ ----------------- _ENQR = enqr.sap$ (SAPSYSTEMNAME)$(I NSTANCE_NAME) Execute_01 = local rm - f $(_ENQR) Execute_02 = local ln - s -f $(DIR_EXECUTABLE)/ enq_replicator$(FT _EXE) $(_ENQR) Start_Program_00 = local $(_ENQR) pf= $(_PF) NR=$(SCSID) … Autostart = 0` **Important** _ENQR不得启用重新启动选项。如果 _ENQR	

Task	描述	所需技能
	设置为 `RestartProgram_01` ，则将其更改为 `StartProgram_01` 。这可以防止 SAP 重新启动服务或干扰集群管理的服务。	

重新启动 SAP 启动服务。	更改此操作说明前面所述的配置文件后，请重新启动 ASCS/SCS 和 ERS 的 SAP 启动服务。
	`sapcontrol -nr 10 -function RestartService SCT`
	`sapcontrol -nr 11 -function RestartService SCT`
	其中 SCT 是指 SAP 系统 ID，并假设 10 和 11 分别是 ASCS/SCS 和 ERS 实例的实例编号。
		SAP

重新配置群集 ENSA2（两种场景都需要）

Task	描述	所需技能
验证 SAP 资源代理中的版本号。	当您使用 SUM 将 SAP 升级到 S/4HANA 1809 或更高版本时，SUM 会处理 SAP 配置文件中的参数更改。只有集群需要手动调整。不过，我们建议	AWS 系统管理员

Task	描述	所需技能

	您在对集群进行任何更改之前先验证参数设置。
	
	Note
	此长篇故事中的示例假设您使用的是 SUSE 操作系统。如果您使用的是 RHEL，则需要使用诸如 YUM 和 pcs shell 之类的工具，而不是 Zypper 和 crm。
	
	检查架构中的两个节点，确认 `resource-agents` 包与 SAP 推荐的最低版本匹配。对于 SLES，请查看 SAP OSS Note 2641019。对于 RHEL，请查看 SAP OSS Note 2641322。(SAP Notes 需要 SAP ONE Support Launchpad 用户账户。)
	
	`sapers:sctadm 23>`
	`zypper search -s -i`
	`resource-agents`
	`Loading repository`
	`data...`
	`Reading installed`
	`packages...`
	`S | Name | Type |`
	`Version | Arch |`
	`Repository`
	`--+-------------`
	`----+---------+---`
	`------------------`	

Task	描述	所需技能
	``` ---------------+-- ------+---------- ----------------- i | resource-agents | package | 4.8.0+git 30.d0077df0-150300 .8.28.1 | x86_64 | SLE-Product-HA15-SP3- Updates ``` 如有必要，请更新 `resource-agents` 版本。	
备份集群配置。	按如下方式备份 CRM 集群配置。 `crm configure show > /tmp/cluster_config_backup.txt`	AWS 系统管理员
设置维护模式。	将集群设置为维护模式。 `crm configure property maintenance-mode="true"`	AWS 系统管理员

Task	描述	所需技能

检查集群配置。	检查当前集群配置。
	
	`crm configure show`
	
	以下是完整输出的摘录：
	
	`node 1: sapascs`
	`node 2: sapers`
	`...`
	`primitive rsc_sap_S`
	`CT_ASCS10 SAPInstance`
	` \`
	`operations $id=rsc_s`
	`ap_SCT_ASCS10-oper`
	`ations \`
	`op monitor interval=120`
	` timeout=60 on-fail=r`
	`estart \`
	`params InstanceN`
	`ame=SCT_ASCS10_sap`
	`ascsvirt START_PRO`
	`FILE="/sapmnt/SCT/`
	`profile/SCT_ASCS10`
	`_sapascsvirt" \`
	`  AUTOMATIC_RECOVER=`
	`false \`
	`meta resource-stickines`
	`s=5000 failure-t`
	`imeout=60 migration-`
	`threshold=1 priority=`
	`10`
	`primitive rsc_sap_S`
	`CT_ERS11 SAPInstance \`
	`operations $id=rsc_s`
	`ap_SCT_ERS11-opera`
	`tions \`
	`op monitor interval=120`
	` timeout=60 on-fail=r`
	`estart \`
	`params InstanceN`
	`ame=SCT_ERS11_sape`
		AWS 系统管理员

Task	描述	所需技能

	<pre>rsvirt START_PRO
	FILE="/sapmnt/SCT/
	profile/SCT_ERS11_
	sapersvirt" \
	   AUTOMATIC_RECOVER=
	false IS_ERS=true \
	meta priority=1000
	...
	colocation col_sap_S
	CT_no_both -5000:
	 grp_SCT_ERS11
	 grp_SCT_ASCS10
	location loc_sap_S
	CT_failover_to_ers
	 rsc_sap_SCT_ASCS10 \
	rule 2000: runs_ers_SCT
	 eq 1
	order ord_sap_S
	CT_first_start_asc
	s Optional: rsc_sap_S
	CT_ASCS10:start
	 rsc_sap_SCT_ERS11:
	stop symmetrical=false
	...</pre>
	其中 `sapascsvirt` 是指 ASCS 实例的虚拟主机名，`sapersvirt` 是指 ERS 实例的虚拟主机名，SCT 是指 SAP 系统 ID。	

Task	描述	所需技能
移除故障转移主机托管限制。	在前面的示例中，位置约束`loc_sap_SCT_failover_to_ers` 指定 ASCS 的 ENSA1 功能在故障转移时应始终跟随 ERS 实例。使用 ENSA2，ASCS 应该能够自由地故障转移到任何参与的节点，因此您可以移除此限制。 `crm configure delete loc_sap_SCT_failover_to_ers`	AWS 系统管理员

Task	描述	所需技能

调整原语。	您还需要对 ASCS 和 ERS SAPInstance 原语进行细微的更改。
	
	以下是为配置的 ASCS SAPInstance 原语的示例。ENSA1
	
	`primitive rsc_sap_SCT_ASCS10 SAPInstance \`
	`operations $id=rsc_sap_SCT_ASCS10-operations \`
	`op monitor interval=120 timeout=60 on-fail=restart \`
	`params InstanceName=SCT_ASCS10_sapascsvirt START_PROFILE="/sapmnt/SCT/profile/SCT_ASCS10_sapascsvirt" \`
	`AUTOMATIC_RECOVER=false \`
	`meta resource-stickiness=5000 failure-timeout=60 migration-threshold=1 priority=10`
	
	要升级到 ENSA2，请将此配置更改为以下内容。
	
	`primitive rsc_sap_SCT_ASCS10 SAPInstance \`
		AWS 系统管理员

Task	描述	所需技能
	operations $id=rsc_sap_SCT_ASCS10-operations \ op monitor interval=120 timeout=60 on-fail=restart \ params InstanceName=SCT_ASCS10_sapascsvirt START_PROFILE="/sapmnt/SCT/profile/SCT_ASCS10_sapascsvirt" \ AUTOMATIC_RECOVER=false \ meta resource-stickiness=3000 这是为配置的 ERS SAPInstance 基元的示例。 ENSA1 primitive rsc_sap_SCT_ERS11 SAPInstance \ operations $id=rsc_sap_SCT_ERS11-operations \ op monitor interval=120 timeout=60 on-fail=restart \ params InstanceName=SCT_ERS11_sapersvirt START_PROFILE="/sapmnt/SCT/profile/SCT_ERS11_sapersvirt" \ AUTOMATIC_RECOVER=false IS_ERS=true \ meta priority=1000 要升级到 ENSA2，请将此配置更改为以下内容。	

Task	描述	所需技能

	`primitive rsc_sap_SCT_ERS11 SAPInstance \ operations $id=rsc_sap_SCT_ERS11-operations \ op monitor interval=120 timeout=60 on-fail=restart \ params InstanceName=SCT_ERS11_sapersvirt START_PROFILE="/sapmnt/SCT/profile/SCT_ERS11_sapersvirt" \ AUTOMATIC_RECOVER=false IS_ERS=true`
	
	您可以通过多种方式更改原语。例如，您可以在 vi 等编辑器中对其进行修改，如下例所示。
	
	`crm configure edit rsc_sap_SCT_ERS11`	

禁用维护模式。	在集群上禁用维护模式。 `crm configure property maintenance-mode="false"` 当集群退出维护模式时，它会尝试使用新 ENSA2 设置使 ASCS 和 ERS 实例联机。	AWS 系统管理员

（可选）添加集群节点

Task	描述	所需技能
查看最佳实践。	在添加更多节点之前，请务必了解最佳实践，例如使用奇数还是偶数节点。	AWS 系统管理员
添加节点。	添加更多节点涉及一系列任务，例如更新操作系统、安装与现有节点匹配的软件包以及使附加可用。您可以使用 SAP Software Provisioning Manager (SWPM) 中的准备其他主机选项来创建主机的 SAP 特定基准。有关更多信息，请参阅下一部分中列出的 SAP 指南。	AWS 系统管理员

相关资源

SAP 和 SUSE 参考资料

要访问 SAP Notes，您必须拥有 SAP ONE Support Launchpad 用户账户。有关详细信息，请参阅 SAP 支持网站。

- SAP Note 2501860 — 适用于 ABAP 7.52 的 SAP NetWeaver 应用程序服务器文档
- SAP Note 2641019 – 在 SUSE HA 环境 ENSA2 中安装 ENSA2 并从更新 ENSA1 到
- SAP Note 2641322 — 使用适用于 SAP 的红帽 HA 解决方案 ENSA2 时的安装 ENSA2 和更新从 ENSA1 到
- SAP Note 2711036 — 在 HA 环境中使用独立排队服务器 2
- 独立排队服务器 2（SAP 文档）
- SAP S/4 HANA — Enqueue Replication 2 高可用性集群 — 设置指南（SUSE 文档）

AWS 参考

- SAP HANA on AWS：适用于 SLES 和 RHEL 的高可用性配置指南
- SAP Lens – AWS Well-Architected Framework

在不同的 AWS 账户中 VPCs 使用一致的可用区

由 Adam Spicer (AWS) 编写

摘要

在 Amazon Web Services (AWS) Cloud 上，可用区的名称可能因您的 Amazon Web Services account 和标识其位置的可用区 ID (AZ ID) 而异。如果您使用 AWS CloudFormation 创建虚拟私有云 (VPCs)，则在创建子网时必须指定可用区的名称或 ID。如果您 VPCs 在多个账户中创建，则可用区名称是随机分配的，这意味着子网在每个账户中使用不同的可用区。

要在您的账户中使用相同的可用区，您必须将每个账户中的可用区名称映射至相同的可用区 ID。例如，下图显示：`use1-az6` AZ ID 在 Amazon Web Services account A 名为 `us-east-1a`，在 Amazon Web Services account Z 名为 `us-east-1c`。

此模式通过提供跨账户、可扩展的解决方案来在子网中使用相同的可用区，从而有助于确保区域一致性。区域一致性可确保您的跨账户网络流量避免跨可用区网络路径，这有助于降低数据传输成本并降低工作负载之间的网络延迟。

这种模式是 AWS CloudFormation AvailabilityZoneId 属性的另一种方法。

先决条件和限制

先决条件

- 位于相同 AWS 区域 中的两个有效 Amazon Web Services account。
- 评估需要多少个可用区来支持您在该区域中的 VPC 要求。
- 识别并记录您需要支持的每个可用区的可用区 ID。有关这方面的更多信息，请参阅 AWS Resource Access Manager 文档中的您的 AWS 资源的可用区 IDs 。
- 以逗号分隔的按顺序排列的可用区列表。 IDs例如，列表中的第一个可用区映射为 `az1`，第二个可用区映射为 `az2`，此映射结构将一直持续到以逗号分隔的列表完全映射为止。可以映射的可用区 IDs 数量没有上限。
- GitHub 多账户可用区映射存储库中的`az-mapping.yaml`文件已复制到您的本地计算机

架构

下图显示了在账户中部署和创建 AWS Systems Manager Parameter Store 值的架构。当您在账户中创建 VPC 时，会使用这些 Parameter Store 值。

图表显示了以下工作流：

1. 此模式的解决方案部署到需要 VPC 区域一致性的所有账户。
2. 该解决方案为每个可用区 ID 创建 Parameter Store 值并存储新的可用区名称。
3. AWS CloudFormation 模板使用存储在每个 Parameter Store 值中的可用区名称，这样可以确保区域一致性。

下图显示了使用此模式的解决方案创建 VPC 的工作流。

图表显示了以下工作流：

1. 向 AWS 提交用于创建 VPC 的模板 CloudFormation。
2. AWS CloudFormation 解析每个可用区的参数存储值，并返回每个可用区 ID 的可用区名称。
3. 使用区域一致性 IDs 所需的正确可用区创建 VPC。

部署此模式的解决方案后，您可创建引用参数存储值的子网。如果您使用 AWS CloudFormation，则可以引用以下 YAML 格式的示例代码中的可用区映射参数值：

```
Resources:
    PrivateSubnet1AZ1:
        Type: AWS::EC2::Subnet
        Properties:
            VpcId: !Ref VPC
            CidrBlock: !Ref PrivateSubnetAZ1CIDR
            AvailabilityZone:
                !Join
                    - ''
                    - - '{{resolve:ssm:/az-mapping/az1:1}}'
```

此示例代码包含在 GitHub 多账户可用区映射存储库`vpc-example.yaml` 的文件中。它向您展示如何创建与 Parameter Store 值一致的 VPC 和子网以实现区域一致性。

技术堆栈

- AWS CloudFormation
- AWS Lambda
- AWS Systems Manager Parameter Store

自动化和扩缩

您可以使用 AWS CloudFormation StackSets 或 AWS Control Tower 定制解决方案将此模式部署到您的所有 AWS 账户。有关更多信息，请参阅 AWS Cloudformation 文档 CloudFormation StackSets中的“使用 AWS” 和 AWS 解决方案库中的 AWS Control Tower 自定义设置。

部署 AWS CloudFormation 模板后，您可以将其更新为使用 Parameter Store 值，并在管道 VPCs 中或根据您的要求进行部署。

工具

Amazon Web Services

- AWS CloudFormation 可帮助您建模和设置 AWS 资源，快速一致地配置这些资源，并在资源的整个生命周期中对其进行管理。您可以使用模板来描述资源及其依赖关系，然后将它们作为堆栈一起启动和配置，而不必单独管理资源。您可以跨多个 Amazon Web Services account 和 AWS 区域管理和预置堆栈。
- AWS Lambda 是一项计算服务，支持无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。您只需为消耗的计算时间付费 - 代码未运行时不产生费用。
- AWS Systems Manager Parameter Store 是 AWS Systems Manager 的其中一项功能。它可提供安全的分层存储，用于配置数据管理和密钥管理。

代码

此模式的代码在 GitHub 多账户可用区映射存储库中提供。

操作说明

部署 az-mapping.yaml 文件

Task	描述	所需技能
确定该区域所需可用区。	1. 确定您所在地区 IDs 必须持续使用的可用区。 2. 将这些 AZ 记录 IDs 在逗号分隔的列表中，并按照您想要的应用顺序进行记录。例如，列表中的第一个可用区映射为 `az1`，第二个可用区映射为 `az2`。可以映射的可用区 IDs 数量没有上限。	云架构师
部署 az-mapping.yaml 文件。	使用该`az-mapping.yaml` 文件在所有必需的 AWS 账户中创建 AWS CloudFormation 堆栈。在 `AZIds` 参数中，使用您之前创建的以逗号分隔的列表。 我们建议您使用 AWS CloudFormation StackSets 或 AWS Cont rol Tower 定制解决方案。	云架构师

VPCs 在您的账户中部署

Task	描述	所需技能
自定义 AWS CloudFormation 模板。	使用 AWS 创建子网时 CloudFormation，请自定义模板以使用您之前创建的参数存储值。	云架构师

Task	描述	所需技能
	有关示例模板，请参阅 GitHub 多账户可用区映射存储库中的`vpc-example.yaml` 文件。	
部署 VPCs。	将自定义 AWS CloudFormation 模板部署到您的账户。然后，该区域中的每个 VPC 在用于子网可用区中都具有区域一致性	云架构师

相关资源

- 您 IDs 的 AWS 资源的可用区（AWS Resource Access Manager 文档）
- AWS::EC2::Subnet（AWS CloudFormation 文档）

IDs 在 IAM 策略中使用用户进行访问控制和自动化

由 Srinivas Ananda Babu (AWS) 和 Ram Kandaswamy (AWS) 创作

摘要

这种模式解释了在 AWS Identity and Access Management (IAM) 中使用基于用户名的策略的潜在缺陷、使用用户的 IDs好处以及如何将这种方法与 AWS CloudFormation 集成以实现自动化。

在中 AWS Cloud，IAM 服务可帮助您精确地管理用户身份和访问控制。但是，在创建 IAM 策略时依赖用户名可能会导致不可预见的安全风险和访问控制问题。例如，考虑以下场景：新员工 John Doe 加入您的团队，您使用该用户名创建一个 IAM 用户账户`j.doe`，该账户通过引用用户名的 IAM 策略向他们授予权限。当 John 离开公司时，该帐户将被删除。当新员工 Jane Doe 加入你的团队并重新创建`j.doe`用户名时，麻烦就开始了。现有策略现在授予 Jane Doe 与 John Doe 相同的权限。这会带来一场潜在的安全和合规性噩梦。

手动更新每项策略以反映新的用户详细信息是一个耗时且容易出错的过程，尤其是在组织成长过程中。解决方案是使用唯一且不可变的用户 ID。创建 IAM 用户账户时，会为 IAM 用户 AWS 分配一个唯一的用户 ID（或委托人 ID）。您可以在 IAM 策略 IDs 中使用这些用户来确保一致可靠的访问控制，且不会受到用户名更改或重复使用的影响。

例如，使用用户 ID 的 IAM 策略可能如下所示：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::example-bucket",
            "Principal": { "AWS": "arn:aws:iam::123456789012:user/abcdef01234567890" }
        }
      ]
}
```

在 IAM 策略 IDs 中使用用户的好处包括：

- 独一无二。用户 IDs 在所有方面都是独一无二的 AWS 账户，因此他们提供正确且一致的权限申请。
- 不变性。用户 IDs 无法更改，因此它们为在策略中引用用户提供了稳定的标识符。
- 审计与合规。 AWS 服务 通常 IDs 在日志和审计跟踪中包含用户，这样可以轻松地将操作追溯到特定用户。

- 自动化和集成。 IDs 在 AWS APIs SDKs、或自动化脚本中使用用户可以确保流程不受用户名更改的影响。
- 经得起未来考验。从一开始就 IDs 在策略中使用用户可以防止潜在的访问控制问题或大量的策略更新。

自动化

当您使用诸如之类的基础设施即代码 (IaC) 工具时 AWS CloudFormation，基于用户名的 IAM 策略的陷阱仍可能导致问题。当您调用Ref内部函数时，IAM 用户资源会返回用户名。随着组织基础设施的发展，如果您重复使用用户名，则创建和删除资源（包括 IAM 用户账户）的周期可能会导致意想不到的访问控制问题。

要解决此问题，我们建议您将用户 IDs 纳入您的 CloudFormation 模板中。但是，为此目的 IDs 获取用户可能具有挑战性。这正是自定义资源可以提供帮助的地方。您可以使用 CloudFormation 自定义资源通过与 AWS APIs 外部服务集成来扩展服务的功能。通过创建用于获取给定 IAM 用户的用户 ID 的自定义资源，您可以在 CloudFormation 模板中提供该用户 ID。这种方法简化了引用用户的流程， IDs 并确保您的自动化工作流程保持稳健且面向未来。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 云管理员运行 AWS CloudFormation 模板的 IAM 角色

限制

- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额页面，然后选择服务的链接。

架构

目标架构

下图显示了如何 AWS CloudFormation 使用支持的自定义资源 AWS Lambda 来检索 IAM 用户 ID。

自动化和扩缩

您可以为不同的 AWS 区域 账户多次使用该 CloudFormation 模板。您只需在每个区域或账户中运行一次。

工具

Amazon Web Services

- IAM — AWS Identity and Access Management (IAM) 是一项网络服务，可帮助您安全地控制对 AWS 资源的访问。可以使用 IAM 来控制谁通过了身份验证（准许登录）并获得授权（具有相应权限）来使用资源。
- AWS CloudFormation— AWS CloudFormation 帮助您对 AWS 资源进行建模和设置，这样您就可以花更少的时间管理这些资源，而将更多的时间集中在运行的应用程序上 AWS。您可以创建一个描述 AWS 所需资源的模板，并 CloudFormation 负责为您配置和配置这些资源。
- AWS Lambda— AWS Lambda 是一项计算服务，无需预置或管理服务器即可支持运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。

最佳实践

如果您是从头开始或计划新建部署，我们强烈建议您使用集中AWS IAM Identity Center式用户管理。IAM Identity Center 可与您现有的身份提供商（例如 Active Directory 或 Okta）集成 AWS，从而无需直接创建和管理 IAM 用户。这种方法不仅可以确保一致的访问控制，还可以简化用户生命周期管理，并有助于增强整个 AWS 环境的安全性和合规性。

操作说明

验证权限

Task	描述	所需技能
验证您的 AWS 账户 和 IAM 角色。	确认您拥有的 IAM 角色有权在您的中部署 CloudFormation 模板 AWS 账户。 如果您计划在本过程的最后一步中使用 AWS CLI 而不是 CloudFormation 控制台来部署模板，则还应设置临时证书以	云架构师

Task	描述	所需技能
	运行 AWS CLI 命令。有关说明，请参阅 I AM 文档。	

生成一个 CloudFormation 模板

Task	描述	所需技能
创建 CloudFormation 模板。	1. 按照 CloudFormation 文档中的说明创建 CloudFormation 模板。你可以使用 JSON 或 YAML 格式。此模式假设您使用的是 YAML 格式。 2. 使用名称保存模板`get_unique_user_id.yaml` 。	AWS DevOps，云架构师

为用户名添加输入参数。	将以下代码添加到 CloudFormation 模板的`Parameters` 部分：
	`Parameters:`
	`  NewIamUserName:`
	`  Type: String`
	`  Description: Unique username for the new IAM user`
	此参数提示用户输入用户名。
		AWS DevOps，云架构师

添加自定义资源以创建 IAM 用户。	将以下代码添加到 CloudFormation 模板的`Resources` 部分：
	`Resources:`
	`  rNewIamUser:`
		AWS DevOps，云架构师

Task	描述	所需技能
	`Type: 'AWS::IAM ::User' Properties: UserName: !Ref NewIamUserName` 此代码添加了一个 CloudFormation 资源，该资源使用`NewIamUserName` 参数提供的名称创建 IAM 用户。	
为 Lambda 函数添加执行角色。	在此步骤中，您将创建一个 IAM 角色，该角色向 AWS Lambda 函数授予获取 IAM 的权限`UserId`。指定 Lambda 运行所需的最低权限如下： • `logs:CreateLogStream` • `logs:PutLogEvents` • `CreateLogGroup` • `iam:GetUser` • 适用于 `lambda.amazonaws.com` 的 `AssumeRole` 有关创建执行角色的说明，请参阅 Lambda 文档。您将在下一步创建 Lambda 函数时引用此角色。	AWS 管理员、云架构师

Task	描述	所需技能

添加 Lambda 函数以获取唯一的 IAM。UserId	在此步骤中，您将使用 Python 运行时定义一个 Lambda 函数以获取唯一的 IAM。UserId为此，请将以下代码添加到 CloudFormation 模板的Resources 部分。<<ROLENAME>> 替换为您在上一步中创建的执行角色的名称。
	
	`GetUserLambdaFunction:`
	`Type: 'AWS::Lambda::Function'`
	`Properties:`
	`Handler: index.handler`
	`Role: <<ROLENAME>>`
	`Timeout: 30`
	`Runtime: python3.11`
	`Code:`
	`ZipFile: |`
	`import cfnresponse, boto3`
	`def handler(event, context):`
	`try:`
	`print(event)`
	`user = boto3.client('iam').get_user(UserName=event['ResourceProperties']['NewIamUserName'])['User']`
	`cfnresponse.send(event,`
		AWS DevOps，云架构师

Task	描述	所需技能

	<pre>context, cfnrespon
	se.SUCCESS, {'NewIamU
	serId': user['Use
	rId'], 'NewIamUs
	erPath': user['Pat
	h'], 'NewIamUserArn':
	user['Arn']})
	except
	Exception as e:
	cfnrespon
	se.send(event,
	context, cfnrespon
	se.FAILED, {'NewIamU
	ser': str(e)})</pre>	

添加自定义资源。	将以下代码添加到 CloudFormation 模板的`Resources` 部分：
	<pre>rCustomGetUniqueUs
	erId:
	Type: 'Custom::
	rCustomGetUniqueUs
	erIdWithLambda'
	Properties:
	ServiceToken: !
	GetAtt GetUserLa
	mbdaFunction.Arn
	NewIamUserName: !
	Ref NewIamUserName</pre>
	此自定义资源调用 Lambda 函数来获取 IAM。`UserID`
		AWS DevOps，云架构师

Task	描述	所需技能

定义 CloudFormation 输出。	将以下代码添加到 CloudFormation 模板的`Outputs`部分：
	
	`Outputs:`
	`  NewIamUserId:`
	`    Value: !GetAtt rCustomGetUniqueUserId.NewIamUserId`
	
	这将显示新 IAM `UserID` 用户的 IAM。
		AWS DevOps，云架构师

保存模板。	保存对 CloudFormation 模板所做的更改。	AWS DevOps，云架构师

部署 CloudFormation 模板

Task	描述	所需技能

部署 CloudFormation 模板。	要使用 CloudFormation 控制台部署`get_unique_user_id.yaml` 模板，请按照CloudFormation 文档中的说明进行操作。
	
	或者，您可以运行以下 AWS CLI 命令来部署模板：
	
	`aws cloudformation create-stack \`
	`--stack-name DemoNewUser \`
	`--template-body file://get_unique_user_id.yaml \`
		AWS DevOps，云架构师

Task	描述	所需技能
	`--parameters Parameter Key=NewIamUserName ,ParameterValue=de mouser \ --capabilities CAPABILITY_NAMED_IAM`	

相关资源

- 从 CloudFormation 控制台创建堆栈（CloudFormation 文档）
- Lambda 支持的自定义资源（文档）CloudFormation
- 唯一标识符（IAM 文档）
- 将临时证书与 AWS 资源配合使用（IAM 文档）

在本地验证 Account Factory for Terraform (AFT) 代码

由 Alexandru Pop (AWS) 和 Michal Gorniak (AWS) 编写

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

此模式显示了如何在本地测试由 Acco AWS Control Tower unt Fact HashiCorp ory for Terraform（AFT）管理的 Terraform 代码。Terraform 是一款基础设施即代码 (IaC) 工具，可帮助您使用代码来配置和管理云基础架构和资源。AFT 设置了一个 Terraform 管道，可以帮助你在其中配置和自定义多个 AWS 账户 管道。 AWS Control Tower

在代码开发过程中，它可能有助于在 AFT 管道之外在本地测试 Terraform 基础设施即代码（IaC）。该模式说明了如何执行以下操作：

- 检索存储在您的 AFT 管理账户存储 AWS CodeCommit 库中的 Terraform 代码的本地副本。
- 使用检索代码在本地模拟 AFT 管道。

此过程还可以用于运行不属于普通 AFT 管道的 Terraform 命令。例如，您可使用此方法来运行 `terraform validate`、`terraform plan`、`terraform destroy` 和 `terraform import` 等命令。

先决条件和限制

先决条件

- 一个活跃的 AWS 多账户环境，使用 AWS Control Tower
- 全面部署 AFT 环境
- AWS Command Line Interface (AWS CLI)，已安装并配置
- AWS CLI 已安装并@@ 配置的 AWS CodeCommit凭证助手
- Python 3.x
- Git，已在本地计算机上安装并配置
- `git-remote-commit`实用程序，已安装和配置
- Terraform，已安装和配置（本地 Terraform 软件包版本必须与 AFT 部署中使用的版本相匹配）

限制

- 此模式不包括 AFT 或任何特定 Terraform 模块所需的 AWS Control Tower部署步骤。
- 在此过程中本地生成的输出不保存在 AFT 管道运行时日志中。

架构

目标技术堆栈

- 部署在部署中部署的 AFT 基础架构 AWS Control Tower
- Terraform
- Git
- AWS CLI 第 2 版

自动化和扩缩

此模式显示了如何在单个 AFT 管理的单个 AFT 管理中本地调用 Terraform 代码进行 AFT 全局账户自定义。 AWS 账户验证您的 Terraform 代码后，您可将其应用于多账户环境中的其余账户。有关更多信息，请参阅文档中的 “重新调用自定义” AWS Control Tower 。

您还可使用类似的过程在本地终端中运行 AFT 账户自定义。要从 AFT 账户自定义项中本地调用 Terraform 代码，请在 AFT 管理账户中克隆aft-account-customizations存储库，而不是从 AFT aft-global-account-customizations管理账户 CodeCommit 中克隆存储库。

工具

Amazon Web Services

- AWS Control Tower按照规范性最佳实践，帮助您设置和管理 AWS 多账户环境。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行外壳中的命令进行交互。

其他服务

- HashiCorp Terraform 是一款基础设施即代码 (IaC) 工具，可帮助您使用代码来配置和管理云基础架构和资源。
- Git 是开源分布式版本控制系统。

代码

以下是 bash 脚本示例，可在本地运行由 AFT 管理的 Terraform 代码。若要使用此脚本，请按照此模式操作说明部分中的说明操作。

```
#! /bin/bash
# Version: 1.1 2022-06-24 Unsetting AWS_PROFILE since, when set, it interferes with
 script operation
#          1.0 2022-02-02 Initial Version
#
# Purpose: For use with AFT: This script runs the local copy of TF code as if it were
 running within AFT pipeline.
#        * Facilitates testing of what the AFT pipline will do
#           * Provides the ability to run terraform with custom arguments (like 'plan'
 or 'move') which are currently not supported within the pipeline.
#
# © 2021 Amazon Web Services, Inc. or its affiliates. All Rights Reserved.
# This AWS Content is provided subject to the terms of the AWS Customer Agreement
# available at http://aws.amazon.com/agreement or other written agreement between
# Customer and either Amazon Web Services, Inc. or Amazon Web Services EMEA SARL or
 both.
#
# Note: Arguments to this script are passed directly to 'terraform' without parsing nor
 validation by this script.
#
# Prerequisites:
#    1. local copy of ct GIT repositories
#    2. local backend.tf and aft-providers.tf filled with data for the target account
 on which terraform is to be run
#       Hint: The contents of above files can be obtain from the logs of a previous
 execution of the AFT pipeline for the target account.
#    3. 'terraform' binary is available in local PATH
#    4. Recommended: .gitignore file containing 'backend.tf', 'aft_providers.tf' so the
 local copy of these files are not pushed back to git

readonly credentials=$(aws sts assume-role \
    --role-arn arn:aws:iam::$(aws sts get-caller-identity --query "Account" --output
 text ):role/AWSAFTAdmin \
    --role-session-name AWSAFT-Session \
    --query Credentials )

unset AWS_PROFILE
export AWS_ACCESS_KEY_ID=$(echo $credentials | jq -r '.AccessKeyId')
export AWS_SECRET_ACCESS_KEY=$(echo $credentials | jq -r '.SecretAccessKey')
```

```
export AWS_SESSION_TOKEN=$(echo $credentials | jq -r '.SessionToken')
terraform "$@"
```

操作说明

将示例代码保存为本地文件

Task	描述	所需技能
将示例代码保存为本地文件。	1. 复制此模式的代码 部分中的示例 bash 脚本并将其粘贴到代码编辑器中。 2. 命名该文件`ct_terraform.sh` ，然后将该文件保存在本地的专用文件夹中，例如`~/scripts` 或`~/bin`。	AWS 管理员
使示例代码运行。	打开终端窗口，通过执行以下任一操作 AWS 在您的 AFT 管理账户中进行身份验证： • 使用已配置访问AFT管理账户所需权限的现有配置AWS CLI 文件。要使用配置文件，您可以运行以下命令： `export AWS_PROFILE=<aft account profile name>` • 如果您的组织使用 SSO 进行访问 AWS，请在组织的 SSO 页面上输入 AFT 管理账户的凭据。	AWS 管理员

Task	描述	所需技能
	Note 您的组织可能还有一个自定义工具，可以为您的 AWS 环境提供身份验证凭证。	

验证对 AFT 管理账户的访问权限是否正确 AWS 区域。	**Important**
	确保使用与 AFT 管理账户进行身份验证时使用的终端会话相同。
	
	1. 运行以下命令导航到您 AWS 区域 的 AFT 部署：
	`export AWS_REGIO N=<aft_region>`
	2. 确保您使用的是正确的帐户。
	a. 运行以下命令：
	`aws code-commit list-repositories`
	b. 验证输出中列出的存储库是否与您的 AFT 管理账户中存储库的名称相匹配。
		AWS 管理员

Task	描述	所需技能

创建新的本地目录来存储 AFT 存储库代码。	从相同的终端会话中，运行以下两个命令：
	`mkdir my_aft`
	`cd my_aft`
		AWS 管理员

克隆远程 AFT 存储库代码。	1. 从您的本地终端运行以下命令：
	`git clone codecommit::$AWS_REGION://aft-global-customizations`
	Note
	为简单起见，此过程和 AFT 仅使用主代码分支。要使用代码分支，也可在此处输入代码分支命令。但是，当 AFT 自动化应用来自主分支的代码时，来自非主分支的任何已应用更改都将被回滚。
	2. 导航到克隆的目录：
	`cd aft-global-customizations/terraform`
		AWS 管理员

创建 AFT 管道本地运行所需 Terraform 配置文件

Task	描述	所需技能
打开之前运行的 AFT 管道，将 Terraform 配置文件复制中本地文件夹。	**Note** 要让 AFT 管道在本地运行，需要在此长篇故事中创建的`backend.tf` 和`aft-providers.tf` 配置文件。这些文件是在基于云 AFT 管道中自动创建，但必须手动创建才能使管道在本地运行。在本地运行 AFT 管道需要一组文件，这些文件表示在单个管道中运行管道 AWS 账户。 1. 使用您的 AWS Control Tower 管理账户凭据登录并打开AWS CodePipeline 控制台。 AWS Management Console确保您与部署 AFT AWS 区域 的地方相同。 2. 在左侧导航窗格中，选择管道。 3. 选择 ############-customizations-pipeline。（ ############ 是你用来在本地运行 Terraform 代码的 AWS 账户 ID。 ）	AWS 管理员

Task	描述	所需技能

	4. 确保最近执行标记为显示成功值。如值不同，则必须在 AFT 管道中重新调用自定义设置。有关更多信息，请参阅文档中的 “重新调用自定义” AWS Control Tower 。
	5. 选择最新的运行时系统，以显示其详细信息。
	6. 在 Apply-AFT-Global-Customizations部分中，找到 Apply-Terraform 阶段。
	7. 选择 Apply-Terraform 阶段的详细信息部分。
	8. 查找 Apply-Terraform 阶段的运行时日志。
	9. 在运行时日志中，查找以以下几行开头和结尾的部分：
	`"\n\n aft-providers.tf ... "\n \n backend.tf"`
	10. 在这两个标签之间复制输出，并将它们保存为本地 Terraform 文件夹（终端会话的当前工作目录）中命名为 `aft-providers.tf` 的本地文件。
	自动生成 providers.tf 语句示例
	`## Autogenerated providers.tf ##`	

Task	描述	所需技能

	``
	## Updated on:
	 2022-05-31 16:27:45
	 ##
	provider "aws" {
	  region = "us-east-
	2"
	  assume_role {
	  role_arn =
	 "arn:aws:iam::####
	########:role/AWSA
	FTExecution"
	  }
	  default_tags {
	  tags = {
	  managed_by
	  = "AFT"
	  }
	  }
	}
	``	

	11.在运行时日志中，查找以以下几行开头和结尾的部分： ``` "\n\n tf ... "\n \n backend.tf" ```	
	12.在这两个标签之间复制输出，并将它们保存为本地 Terraform 文件夹（终端会话的当前工作目录）中命名为 `tf` 的本地文件。 自动生成的 backend.tf 语句示例 ``` ## Autogenerated backend.tf ## ```	

Task	描述	所需技能

	``
	## Updated on: 2022-05-3
	1 16:27:45 ##
	terraform {
	  required_version =
	 ">= 0.15.0"
	  backend "s3" {
	    region =
	 "us-east-2"
	    bucket =
	 "aft-backend-#####
	#######-primary-re
	gion"
	    key =
	 "############-aft-
	global-customizati
	ons/terraform.tfst
	ate"
	    dynamodb_table =
	 "aft-backend-#####
	#######"
	    encrypt =
	 "true"
	    kms_key_id =
	 "########-####-####-
	####-############"
	    role_arn =
	 "arn:aws:iam::####
	#########:role/AWS
	AFTExecution"
	  }
	}
	``
	
	Note
	
	backend.tf 和aft-providers.tf 文件与特定 AWS 账户的 AFT 部署和文件夹相关联。这些文件也有所不同，具体取决于它们是	

Task	描述	所需技能
	否位于同一 AFT 部署中的aft-global-customizations和aft-account-customizations存储库中。确保使用相同的运行时列表生成两项文件。	

使用示例 bash 脚本，在本地运行 AFT 管道

Task	描述	所需技能
实施要验证的 Terraform 配置更改。	1. 运行以下命令导航到克隆的aft-global-customizations存储库： `cd aft-global-customizations/terraform` **Note** 文件`backend.tf` 和`aft-providers.tf` 位于此目录中。该目录还包含存储库中的 Terraform 文件。aft-global-customizations 2. 将要在本地测试的 Terraform 代码更改合并至配置文件中。	AWS 管理员

Task	描述	所需技能

运行`ct_terraform.sh` 脚本并查看输出。	1. 导航至包含 sh 脚本的本地文件夹。
	2. 要验证修改后的 Terraform 代码，请通过运行以下命令运行 `ct_terraform.sh` 脚本：
	`~/scripts/ct_terraform.sh apply`
	`terraform --help`
	Note
	在此步骤中，你可以运行任何 Terraform 命令。若要查看 Terraform 命令的完整列表，请运行以下命令：
	3. 查看命令输出，然后在本地调试代码更改，然后再提交更改并将其推送回 AFT 存储库。
	Important
	• 任何在本地进行但未推送回远程存储库的更改都是临时
		AWS 管理员

Task	描述	所需技能
	的，可以通过运行的 AFT 管道自动化随时撤消。 • AFT 自动化可以随时运行，因为它可以由其他用户与 AFT 自动化触发器调用。 • AFT 将始终应用存储库主分支中代码，撤销所有未提交的更改。	

将您的本地代码更改推送回 AFT 存储库

Task	描述	所需技能

将对backend.tf 和aft-providers.tf 文件的引用添加到.gitignore 文件中。	通过运行以下命令将您创建的 backend.tf 和 aft-providers.tf 文件添加至 .gitignore 文件中：
	`echo backend.tf >> .gitignore`
	`echo aft-providers.tf >>.gitignore`
	Note
	将文件移动到.gitignore 文件中可确保它们不会被提交并推回远程 AFT 存储库。
		AWS 管理员

提交您的代码更改并将其推送至远程 AFT 存储库。	1. 要向存储库中添加任何新 Terraform 配置文件，请运行以下命令：	AWS 管理员

Task	描述	所需技能

	`git add <filename>`
	2. 要提交您的更改并将其推送到中的远程 AFT 存储库 CodeCommitt，请运行以下命令：
	`git commit -a`
	`git push`
	Important
	在此之前，您通过执行此过程引入的代码更改 AWS 账户 仅适用于一个代码更改。	

将更改推广到多个账户

Task	描述	所需技能
对所有由 AFT 管理的账户进行更改。	要将更改推广到由 AFT 管理 AWS 账户 的多个项目，请按照文档中 “重新调用自定义” 中的 AWS Control Tower 说明进行操作。	AWS 管理员

更多模式

- 使用只读副本 PeopleSoft 在 Amazon RDS Custom 上将 HA 添加到 Oracle
- 自动审核允许从公有 IP 地址进行访问 AWS 的安全组
- 使用着陆区域加速器自动创建账户 AWS
- 通过 AWS Systems Manager 自动添加或更新 Windows 注册表项
- 自动执行 AWS 资源评测
- 使用 AWS CDK 自动部署 AWS Service Catalog 产品组合与产品
- 使用 DR Orchestrator 框架自动执行跨区域故障转移和故障恢复
- 自动删除 AWS CloudFormation 堆栈和相关资源
- 使用 Terraform 在亚马逊托管 Grafana 上自动提取和可视化亚马逊 MWAA 自定义指标
- 在 Amazon MQ 中自动化 RabbitMQ 配置
- 在跨区域自动复制 Amazon RDS 实例 AWS 账户
- 使用云托管人和 AWS CDK 将适用于 Systems Manager 的 AWS 托管策略自动附加到 EC2 实例配置文件
- 自动使用 AWS CDK 为微服务构建 CI/CD 管道与 Amazon ECS 集群
- 自动检测变化并为 monorepo 启动不同的 CodePipeline 管道 CodeCommit
- 使用 AWS Confi CloudTrail g 中的自定义补救规则自动重新启用 AWS
- 使用 AWS DataOps 开发套件构建数据管道，以提取、转换和分析 Google Analytics（分析）数据
- 使用 Amazon A EC2 uto Scaling 和 Systems Manager 构建 Micro Focus 企业服务器 PAC
- 使用 GitHub Actions 和 Terraform 构建 Docker 镜像并将其推送到 Amazon ECR
- 使用 Terraform 在 AWS Organizations 中集中管理 IAM 访问密钥
- 使用 Terraform 在 AWS Organizations 中集中分发软件包
- 使用无服务器方法将 Amazon Web Services 串在一起
- 使用混合链接模式将数据中心扩展配置为 VMware Cloud on AWS
- 使用在 Amazon Bedrock 中配置模型调用日志 AWS CloudFormation
- 在 AWS 上的 SQL Server 的“始终打开”可用性组中配置只读路由
- 将 VMware vRealize Automation 配置为在 A VMs W VMware S 上的云端上
- 使用 AWS Amplify、Angular 和模块联合为微前端创建门户
- 在组织中创建跨账户 Amazon EventBridge 连接
- 自动为 Java 和 Python 项目创建动态 CI 管道

- 使用 AWS 上的 VMware C VMware loud 在 AWS 上部署 SDDC
- 使用私有端点和应用程序负载均衡器在内部网站上部署 Amazon API Gateway API
- 部署和调试 Amazon EKS 集群
- 使用 AWS CDK 和来部署和管理 AWS Control Tower 控件 CloudFormation
- 使用 Terraform 部署和管理 AWS Control Tower 控件
- 使用 Terraf CloudWatch orm 部署 Synthetics 加那利群岛
- 使用 Terraform 和 DRA 部署 Lustre 文件系统以实现高性能数据处理
- 使用 Terraform 和 Amazon AWS Bedrock 部署 RAG 用例
- 使用 Terraform 在 AWS Wavelength 区域中部署资源
- 使用 Terraform 为 AWS WAF 解决方案部署安全自动化
- 检测已过期的 CA 证书的 Amazon RDS 和 Aurora 数据库实例
- 记录您的 AWS 着陆区设计
- 确保 IAM 配置文件与 EC2 实例关联
- 将 AWS Organizations 中整个组织的 AWS Backup 报告导出为 CSV 文件
- 使用 Amazon Personalize 生成个性化和重新排名的推荐
- 在 Amazon Data Firehose 资源未使用密钥加密时进行识别并发出警报 AWS KMS
- 使用引导管道实现 Account Factory for Terraform (AFT)
- 在 Amazon API Gateway 中使用自定义域名实现基于路径的 API 版本控制
- 使用 Kubernetes 在亚马逊 EKS 工作节点上安装 SSM 代理 DaemonSet
- 使用在 Amazon EKS 工作节点上安装 SSM CloudWatch 代理和代理 preBootstrapCommands
- 在 AWS 上将 VMware vRealize 网络洞察与 VMware 云集成
- 使用以下 AWS IAM Identity Center 命令将权限集作为代码进行管理 AWS CodePipeline
- 管理多个 Amazon Web Services account 和 AWS 区域中的 AWS Service Catalog 产品
- 通过使用 AWS CDK 设置 Amazon ECS Anywhere 来管理本地容器应用程序
- 将 DNS 记录批量迁移至 Amazon Route 53 私有托管区
- 将 Oracle 电子商务套件迁移到 Amazon RDS Custom
- 将 Oracle 迁移 PeopleSoft 到亚马逊 RDS 定制版
- 使用 AWS MGN 将 RHEL BYOL 系统迁移至 AWS License-Included 实例
- 使用 H VMware CX 将 SDDC 迁移到 AWS VMware 上的 VMware 云端
- 设置最小可行数据空间，以便在组织之间共享数据

- 监控 Amazon ElastiCache 集群的静态加密
- 监控 ElastiCache 集群中的安全组
- 使用 AWS 服务监控 SAP RHEL Pacemaker 集群
- 使用 AWS CDK 和 GitHub 操作工作流程优化多账户无服务器部署
- 从多个私密访问中央 AWS 服务终端节点 VPCs
- 使用 GitHub 操作基于 AWS CloudFormation 模板配置 AWS Service Catalog 产品
- 通过部署角色自动售货机解决方案来配置权限最低的 IAM 角色
- 使用 AWS Lambda 自动化功能 AWS 账户 从 AWS Managed Microsoft AD 正面移除 Amazon EC2 条目
- 使用 AWS Lambda 自动化功能 AWS 账户 从 AWS Managed Microsoft AD 中移除 Amazon EC2 条目
- 在不重启容器的情况下轮换数据库凭证
- 在创建 IAM 用户时发送通知
- 使用 VMware Aria 日志操作 AWS 将日志从 VMware Cloud 发送到 Splunk
- 为基于蜂窝的架构设置无服务器蜂窝路由器
- 使用 AWS CDK 在 Amazon ECS Anywhere 上为混合工作负载设置 CI/CD 管道和 GitLab
- 使用 NICE EnginFrame 和 NICE DCV 会话管理器设置自动缩放虚拟桌面基础架构
- 使用有效备用数据库为 Amazon RDS Custom 上的 Oracle 电子商务套件设置 HA/DR 架构
- 在多账户 AWS 环境中为混合网络设置 DNS 解析
- 使用 Amazon 为 SQL Server Always On FCI 设置多可用区基础架构 FSx
- 在 Aurora PostgreSQL-Compatible 上设置 Oracle UTL_FILE 功能
- 使用应用程序负载均衡器在 Amazon ECS 中使用双向 TLS 简化应用程序身份验证
- 使用 AWS Private CA 和 AWS RAM 简化私有证书管理
- 使用 SageMaker AI 和 Hydra 简化从本地开发到可扩展实验的机器学习工作流程
- 使用 AWS Organizations 自动标记中转网关连接
- 在 Amazon RDS 上为 Oracle PeopleSoft 应用程序过渡角色适用于 Oracle 定制
- 使用 Amazon Q Developer 作为编码助手来提高工作效率

网络和移动应用程序

主题

- 使用亚马逊 Cognito 和 AWS Amplify 用户界面对现有 React 应用程序用户进行身份验证
- 从 AWS 存储库持续部署现代 AWS Amplify 网络应用程序 CodeCommit
- 使用 AWS Amplify 创建 React 应用程序，并使用 Amazon Cognito 添加身份验证
- 使用 AWS Amplify、Angular 和模块联合为微前端创建门户
- 将基于 React 的单页应用程序部署到 Amazon S3 CloudFront
- 使用私有端点和应用程序负载均衡器在内部网站上部署 Amazon API Gateway API
- 在本地 Angular 应用程序中嵌入亚马逊 QuickSight 控制面板
- 使用 Green Boost 探索全栈云原生 Web 应用程序开发
- 使用 AWS 对来自 GitHub 的 Node.js 应用程序运行单元测试 CodeBuild
- 使用 AWS Lambda 以六边形架构构建 Python 项目
- 更多模式

使用亚马逊 Cognito 和 AWS Amplify 用户界面对现有 React 应用程序用户进行身份验证

由 Daniel Kozhemyako (AWS) 创作

摘要

此模式展示了如何使用 AWS Amplify用户界面库和 Amazon Cognito 用户池向现有的前端 React 应用程序添加身份验证功能。

该模式使用 Amazon Cognito 为应用程序提供身份验证、授权和用户管理。它还使用 A mplify UI 中的组件，这是一个开源库，可将的功能扩展 AWS Amplify 到用户界面 (UI) 开发。Authenticator UI 组件管理登录会话，并运行连接云的工作流程，通过 Amazon Cognito 对用户进行身份验证。

实现此模式后，用户可使用以下任何凭证登录：

- 用户名和密码
- 社交身份提供商，例如 Apple、Facebook、Google 和 Amazon
- SAML 2.0 或 OpenID Connect (OIDC) 兼容的企业身份提供程序

> **Note**
>
> 要创建自定义身份验证 UI 组件，可以在无头模式下运行 Authenticator UI 组件。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- React 18.2.0 或更高版本的 Web 应用程序
- Node.js 和 npm 6.14.4 或更高版本，已安装

限制

- 此模式仅适用于 React Web 应用程序。
- 此模式使用预先构建的 Amplify UI 组件。该解决方案不包括实现自定义 UI 组件所需步骤。

产品版本

- Amplify UI 6.1.3 或更高版本（第 1 代）
- Amplify 6.0.16 或更高版本（第 1 代）

架构

目标架构

下图显示了一种架构，该架构使用 Amazon Cognito 对 React 网络应用程序的用户进行身份验证。

工具

AWS 服务

- Amazon Cognito 为您的 Web 和移动应用程序提供身份验证、授权和用户管理。

其他工具

- Amplify UI 是一个开源用户界面库，它提供了可以连接到云端的可自定义组件。
- Node.js 是一个事件驱动的 JavaScript 运行时环境，专为构建可扩展的网络应用程序而设计。
- npm 是在 Node.js 环境中运行的软件注册表，用于共享或借用软件包以及管理私有软件包的部署。

最佳实践

如果您正在构建新应用程序，我们建议您使用第二代 Amplify。

操作说明

创建 Amazon Cognito 用户池

Task	描述	所需技能
创建用户池。	创建 Amazon Cognito 用户池。配置用户池的登录选项和安全要求以适应您的用例。	应用程序开发人员

Task	描述	所需技能
添加应用程序客户端。	配置用户池应用程序客户端。您的应用程序需要使用此客户端才能与 Amazon Cognito 用户池进行交互。	应用程序开发人员

将你的 Amazon Cognito 用户池与身份验证器用户界面组件集成

Task	描述	所需技能
安装依赖项。	要安装 `aws-amplify` 和 `@aws-amplify/ui-react` 软件包，请从应用程序的根目录运行以下命令： `npm i @aws-amplify/ui-react aws-amplify`	应用程序开发人员

配置用户池。	根据以下示例，创建一个`aws-exports.js` 文件并保存在该`src`文件夹中。该文件应包含以下信息：
	• AWS 区域 你的 Amazon Cognito 用户池所在的
	• 亚马逊 Cognito 用户池 ID
	• 应用程序客户端 ID
	`// replace the user pool region, id, and app client id details`
	`const awsmobile = { "aws_project_region": "put_your_region_here",`
		应用程序开发人员

Task	描述	所需技能

	<pre>"aws_cognito_regio
	n": "put_your_region_h
	ere",
	"aws_user_pools_id
	": "put_your_user_poo
	l_id_here",
	"aws_user_pools_we
	b_client_id":
	"put_your_user_poo
	l_app_id_here"
	}
	
	export default
	awsmobile;</pre>	

Task	描述	所需技能

导入与配置 Amplify 服务。	1. 在应用程序的入口点文件（例如App.js）中，通过输入以下代码行来导入和加载该aws-exports.js 文件：
	`import { Amplify } from 'aws-amplify'; import awsExports from './aws-exports';`
	2. 根据以下示例，使用文件配置 Amplify 客户端：aws-exports.js
	`// Configure Amplify in index file or root file Amplify.configure({ ...awsExports });`
	有关更多信息，请参阅 Amplify 文档中的配置 Amplify 类别。
		应用程序开发人员

Task	描述	所需技能

添加身份验证器 UI 组件。	若要显示 Authenticator UI 组件，请在应用程序的入口点文件 (App.js) 中添加以下代码行：
	
	`import { Authenticator } from '@aws-amplify/ui-react';`
	`import '@aws-amplify/ui-react/styles.css';`
	
	Note
	示例代码片段导入Authenticator 用户界面组件和 Amplify UI styles.css 文件，这是使用该组件的预构建主题时所必需的。
	
	U Authenticator I 组件提供两个返回值：
	
	• 用户详情
	• 一个可以调用以注销用户的函数
	
	请参见以下示例组件：
	
	`function App() {`
	`return (`
	`<Authenticator>`
	`{({ signOut, user }) => (`
		应用程序开发人员

Task	描述	所需技能
	<pre> Welcome {user.use rname} <button onClick={ signOut}>Sign out</ button>)} </Authent icator>); }</pre> **Note** 有关示例App.js文件，请参阅此模式的“其他信息”部分。	

Task	描述	所需技能

（可选）检索会话信息。	用户通过身份验证后，您可从 Amplify 客户端检索有关其会话的数据。例如，您可从用户的会话中检索 JSON Web 令牌 (JWT)，这样您就可以对来自其会话的请求进行身份验证，然后再向后端 API 发送请求。
	
	参见以下含 JWT 请求标头示例：
	
	`import { fetchAuthSession } from 'aws-amplify/auth';`
	`(await fetchAuthSession()).tokens?.idToken?.toString();`
		应用程序开发人员

故障排除

事务	解决方案
新用户无法注册此应用程序。	如下所示，请确保您的 Amazon Cognito 用户池已配置为允许用户注册加入用户池： • 登录 AWS Management Console，然后打开 Amazon Cognito 控制台。 • 在左侧导航窗格中，选择用户池。 • 从列表中选择您的用户池。 • 在常规设置下，选择策略。 • 选择允许用户自己注册。
从 v5 升级到 v6 后，身份验证组件停止运行。	该`Auth`类别已移至函数方法，并在 Amplify v6 中命名了参数。现在，必须 APIs 直接从`aws-`

事务	解决方案
	amplify/auth 路径导入函数。有关更多信息，请参阅 Amplify 文档中的从 v5 迁移到 v6。

相关资源

- 亚马逊 Cognito（AWS 网站）入门
- 创建新 React 应用程序（React 文档）
- 什么是 Amazon Cognito？（亚马逊 Cognito 文档）
- Amplify 用户界面库（Amplify 文档）

其他信息

该App.js文件应包含以下代码：

```
import './App.css';
import { Amplify } from 'aws-amplify';
import awsExports from './aws-exports';
import { fetchAuthSession } from 'aws-amplify/auth';
import { Authenticator } from '@aws-amplify/ui-react';
import '@aws-amplify/ui-react/styles.css';
Amplify.configure({ ...awsExports });
let token = (await fetchAuthSession()).tokens?.idToken?.toString();
function App() {
  return (
      <Authenticator>
        {({ signOut, user }) => (
            <div>
              <p>Welcome {user.username}</p>
                <p>Your token is: {token}</p>
              <button onClick={signOut}>Sign out</button>
            </div>
        )}
      </Authenticator>
  );
}

export default App;
```

从 AWS 存储库持续部署现代 AWS Amplify 网络应用程序 CodeCommit

由 Deekshitulu Pentakota (AWS) 与 Sai Katakam (AWS) 编写

摘要

注意： CodeCommit AWS 不再向新客户开放。AWS 的现有客户 CodeCommit 可以继续照常使用该服务。了解更多。

现代 Web 应用程序构造为单页应用程序 (SPAs)，它将所有应用程序组件打包成静态文件。通过使用 AWS Amplify Hosting，您可构建持续集成和持续部署 (CI/CD) 管道，用于构建、部署和托管在基于 Git 的存储库中管理的现代 Web 应用程序。当您将 Amplify Hosting 连接至代码存储库时，每次提交都会启动一个工作流来部署应用程序的前端和后端。这种方法的好处是，Web 应用程序只有在部署成功完成后才会更新，这可以防止前端和后端之间的不一致。

在这种模式中，您可以使用 AWS CodeCommit 存储库来管理您的现代 Web 应用程序。这些说明中的示例 Web 应用程序采用 React SPA 框架。但是，Amplify Hosting 支持众多其他 SPA 框架，例如 Angular、Vue、Next.js；它还支持单站点生成器，例如 Gatsby、Hugo 和 Jekyll。

此模式适用于在以下服务与概念方面有经验的 AWS 构建者：

- AWS CodeCommit
- AWS Amplify Hosting
- React
- JavaScript
- Node.js
- npm
- Git

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 在 Amplify 中创建资源的权限和. CodeCommit 有关更多信息，请参阅 A mplify 的身份和访问管理以及适用于 AWS 的身份和访问管理。 CodeCommit
- 已安装和配置 AWS 命令行界面（AWS CLI）。
- 文本编辑器或者代码编辑器。

- CodeCommit，为使用 Git 凭据的 HTTPS 用户进行设置。
- Amplify 的IAM 服务角色。
- npm 和 Node.js，已安装(npm 文档)。

限制

- 这种模式不介绍 Amplify 应用程序后端 (例如 API、身份验证或数据库) 的开发和集成。有关后端的更多信息，请参阅 Amplify 文档中的创建后端。

产品版本

- AWS CLI 版本 2.0
- Node.js 版本 16.x 或更高版本

架构

目标技术堆栈

- 包含 React SPA 的 AWS CodeCommit 存储库
- AWS Amplify Hosting 工作流

目标架构

工具

Amazon Web Services

- AWS Amplify Hosting提供了基于 Git 的工作流，用于托管持续部署的全栈无服务器 Web 应用程序。
- AWS CodeCommit 是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。

其他工具

- Node.js 是一个事件驱动的 JavaScript 运行时环境，专为构建可扩展的网络应用程序而设计。
- npm 是在 Node.js 环境中运行的软件注册表，用于共享或借用软件包以及管理私有软件包的部署。

操作说明

创建存储 CodeCommit 库

Task	描述	所需技能
创建存储库。	有关说明，请参阅 CodeCommit 文档中的创建 AWS CodeCommit 存储库。	AWS DevOps
克隆存储库。	有关说明，请参阅 CodeCommit 文档中的通过克隆 CodeCommit 存储库来连接存储库。如果系统提示，请提供 Git 凭证。	应用程序开发人员

创建 React 应用程序

Task	描述	所需技能

创建新 React 应用程序。	1. 输入以下命令导航到克隆的存储库。`<repo name>`替换为 CodeCommit 存储库的名称。
	`$ cd <repo name>`
	2. 输入以下命令，以在克隆的存储库中创建新的 React 应用程序。
	`$ npx create-react-app .`
		应用程序开发人员

Task	描述	所需技能
	3. 编写应用程序代码，然后输入以下命令来启动它。 `$ npm start` 有关创建自定义 React 应用程序的更多信息，请参阅创建 React 应用程序文档中的创建 React 应用程序说明。您也可按照 Amplify 文档中部署前端中的说明，将示例 React 应用程序部署至 Amplify 账户。	

创建分支并推送代码。	1. 输入以下命令在本地创建新分支，其中 <branch> 是您要分配给新分支的名称。
	
	`$ git checkout -b <branch>`
	
	2. 输入以下命令将分支推送到 CodeCommit 存储库，其中<branch>是您在上一步中分配的名称。有关更多信息，请参阅使用 Commit。
	
	`$ git push --set-upstream origin <branch>`
		应用程序开发人员

在 AWS Amplify Hosting 中部署应用程序

Task	描述	所需技能
将 Amplify 连接至存储库。	有关说明，请参阅 Amplify Hosting 文档中的连接存储库。选择 AWS CodeCommit 以及您之前创建的存储库和分支。	应用程序开发人员

定义前端构建设置。	有关说明，请参阅 Amplify Hosting 文档中的确认前端版本设置。接受默认值或者输入以下内容。
	`Build settings:`
	`version: 0.1`
	`frontend:`
	`  phases:`
	`  preBuild:`
	`  commands:`
	`  - npm ci`
	`  build:`
	`  commands:`
	`  - npm run build`
	`  artifacts:`
	`  baseDirectory:`
	`build`
	`  files:`
	`  - '**/*'`
	`  cache:`
	`  paths:`
	`  - node_modules/`
	`**/*`
		应用程序开发人员

审核与部署。	有关说明，请参阅 Amplify Hosting 文档中的保存和部署。等待部署过程完成。	应用程序开发人员

验证持续部署

Task	描述	所需技能
验证初始部署。	部署过程完成后，在域下选择链接。验证应用程序是否按照预期运行。	应用程序开发人员
将更改推送至代码存储库。	在本地工作站上编辑代码并将更改推送到 CodeCommit 存储库。Amplify Hosting 会检测仓库中的更改，并自动启动构建和部署过程。确认应用程序更新在域内可见。	应用程序开发人员

相关资源

AWS CodeCommit 文档

- 为 AWS 做准备 CodeCommit
  - 使用 Git 凭证设置 HTTPS 用户
  - 使用 AWS CLI 凭证助手设置在 Linux、macOS 或 Unix 上与 AWS CodeCommit 存储库的 HTTPS 连接的步骤
- 开始使用 AWS CodeCommit

AWS Amplify Hosting 文档

- 现有代码入门
- 设置自定义域

React 资源

- 创建 React 应用程序网站
- 创建 React 应用程序文档
- 创建 React 应用程序存储库 (GitHub)

使用 AWS Amplify 创建 React 应用程序，并使用 Amazon Cognito 添加身份验证

由 Rishi Singla (AWS) 创建

摘要

此模式演示了如何使用 AWS Amplify 创建基于 React 的应用程序，以及如何使用 Amazon Cognito 为前端添加身份验证。AWS Amplify 由一组工具(开源框架、可视化开发环境、控制台)和服务(网络应用程序和静态网站托管)组成，用于加速 AWS 上移动和网络应用程序的开发。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 计算机上已安装 Node.js 和 npm

产品版本

- Node.js 版本 10.x 或更高版本（要验证您的版本，请在终端窗口中运行 `node -v`）
- npm 6.x 或更高版本（要验证您的版本，请在终端窗口中运行 `npm -v`）

架构

目标技术堆栈

- AWS Amplify
- Amazon Cognito

工具

- Amplify 命令行界面（CLI）
- Amplify Libraries（开源客户端库）
- Amplify Studio（可视化界面）

操作说明

安装 AWS Amplify CLI

Task	描述	所需技能

安装 Amplify CLI	Amplify CLI 是一个统一的工具链，用于为您的 React 应用程序创建 Amazon Web Services Cloud 服务。若要安装 Amplify CLI，请运行：
	`npm install -g @aws-amplify/cli`
	如果有新的主要版本可用，npm 将通知您。如果是这样，请使用以下命令升级您的 npm 版本：
	`npm install -g npm@9.8.0`
	其中 9.8.0 指的是您要安装的版本。
		应用程序开发人员

创建 React 应用程序

Task	描述	所需技能
创建 React 应用程序。	若要创建新的 React 应用程序，请使用命令： `npx create-react-app amplify-react-application`	应用程序开发人员

Task	描述	所需技能
	其中 `ampify-react-application` 是应用程序的名称。 应用程序创建成功后，您将看到提示信息： `Success! Created amplify-react-application` 将为 React 应用程序创建一个包含多个子文件夹的目录。	
在本地计算机上启动应用程序。	转到上一步中创建的目录 `amplify-react-application` 并运行以下命令： `amplify-react-application% npm start` 这将在您的本地计算机上启动 React 应用程序。	应用程序开发人员

配置 Amplify CLI

Task	描述	所需技能
配置 Amplify 以连接到您的 Amazon Web Services account。	通过运行以下命令配置 Amplify： `amplify-react-application % amplify configure` Amplify CLI 要求您按照以下步骤设置对您的 Amazon Web	常规 AWS，应用程序开发人员

Task	描述	所需技能
	Services account 的访问权限： 1. 登录您的 AWS 管理员账户。 2. 指定您要使用的 AWS 区域。 3. 创建具有编程访问权限的 AWS Identity and Access Management（AWS IAM）用户，并将 `AdministratorAccess-Amplify` 权限策略附加给该用户。 4. 复制访问密钥 ID 和秘密访问密钥。 5. 在终端中输入这些详细信息。 6. 创建配置文件名称或使用默认配置文件。 **Warning** 这种情况需要具有编程访问权限和长期证书的 IAM 用户，这会带来安全风险。为帮助减轻这种风险，我们建议仅向这些用户提供执行任务所需的权限，并在不再需要这些用户时将其移除。必要时可以更新访问密钥。有关更多信息，请参阅《IAM 用户	

Task	描述	所需技能

	指南》中的 更新访问密钥。
	
	这些步骤在终端中显示如下。
	
	`Follow these steps to set up access to your AWS account:`
	`Sign in to your AWS administrator account: https://console.aws.amazon.com/`
	`Press Enter to continue`
	`Specify the AWS Region`
	`? region:  us-east-1`
	`Follow the instructions at https://docs.amplify.aws/cli/start/install/#configure-the-amplify-cli`
	`to complete the user creation in the AWS console`
	`https://console.aws.amazon.com/iamv2/home#/users/create`
	`Press Enter to continue`
	`Enter the access key of the newly created user:`
	`? accessKeyId: ********************`
	`? secretAccessKey: ****************************************`
	`This would update/create the AWS Profile in your local machine`
	`? Profile Name:  new`	

Task	描述	所需技能
	``` Successfully set up the new user. ``` 有关这些步骤的更多信息，请参阅 Amplify 开发人员中心中的文档。	

初始化 Amplify

Task	描述	所需技能
初始化 Amplify。	1. 要在新目录中初始化 Amplify，请运行： ``` amplify init ``` Amplify 会提示您输入项目名称和配置参数 2. 指定所有参数，然后按 Y 以使用指定的配置初始化项目。 ``` Project information | Name: amplifyreactproject | Environment: dev | Default editor: Visual Studio Code | App type: javascript | Javascript framework: react ```	应用程序开发人员，常规 AWS

Task	描述	所需技能
	``` | Source Directory Path: src | Distribution Directory Path: build | Build Command: npm run-script build | Start Command: npm run-script start ```	
	3. 选择在上一步中创建的配置文件。这些资源将部署到您创建的 Amplify 项目中的 dev 环境中。 4. 要确认资源已创建，您可以打开 AWS Amplify 控制台并查看用于创建资源的 AWS CloudFormation 模板和详细信息。	
	``` Deploying root stack amplifyreactproject [================== ==---------------- ----] 2/4 amplify-amplif yreactproject-d… AWS::CloudFormatio n::Stack CREATE_IN_PROGRESS UnauthRole AWS::IAM: :Role CREATE_COMPLETE ```	

Task	描述	所需技能

	<pre>DeploymentBucket AWS::S3::Bucket CREATE_IN_PROGRESS
	
	AuthRole AWS::IAM::Role CREATE_COMPLETE</pre>	

向前端添加身份验证

Task	描述	所需技能
添加身份验证	您可以使用 `amplify add <category>` 命令添加诸如用户登录或后端 API 之类的功能。在此步骤中，您将使用该命令添加身份验证。 Amplify 提供包含 Amazon Cognito、前端库和嵌入式身份验证器 UI 组件的后端身份验证服务。功能包括用户注册、用户登录、多重身份验证、用户注销和无密码登录。您还可以通过与 Amazon、Google 和 Facebook 等联合身份提供商集成来对用户进行身份验证。Amplify 身份验证类别与其他 Amplify 类别（如 API、分析和存储）无缝集成，因此您可以为经过身份验证和未经身份验证的用户定义授权规则。	应用程序开发人员，常规 AWS

Task	描述	所需技能

	1. 若要为 React 应用配置身份验证，请运行以下命令：
	
	`amplify-react-application1 % amplify add auth`
	
	这将显示以下信息和提示。您可以根据您的业务和安全需求选择适当的配置。
	
	`Using service: Cognito, provided by: awscloudformation The current configured provider is Amazon Cognito. Do you want to use the default authentication and security configuration? (Use arrow keys) # Default configuration Default configuration with Social Provider (Federation) Manual configuration I want to learn more.`
	
	2. 举一个简单的例子，选择默认配置，然后选择用户的登	

Task	描述	所需技能

	录机制（在本例中为电子邮件）：
	
	`How do you want users to be able to sign in?`
	`Username`
	
	`# Email`
	
	`Phone Number`
	
	`Email or Phone Number`
	
	`I want to learn more.`
	
	3. 绕过高级设置完成身份验证资源的添加：
	
	`Do you want to configure advanced settings? (Use arrow keys)`
	`# No, I am done.`
	
	`Yes, I want to make some additional changes.`
	
	4. 构建您的本地后端资源，并在云中进行配置：
	
	`amplify-react-application1 % amplify push`	

Task	描述	所需技能

	此命令会对您账户中的 Congito 用户池进行适当更改。
	5. 按 Y 使用配置auth资源 CloudFormation。
	这将配置以下资源：
	`UserPool AWS::Cognito::UserPool CREATE_COMPLETE`
	`UserPoolClientWeb AWS::Cognito::UserPoolClient CREATE_COMPLETE`
	`UserPoolClientWeb AWS::Cognito::UserPoolClient CREATE_COMPLETE`
	`UserPoolClientRole AWS::IAM::Role CREATE_COMPLETE`
	`UserPoolClientLambda AWS::Lambda::Function CREATE_COMPLETE`
	`UserPoolClientLambdaPolicy AWS::IAM::Policy CREATE_COMPLETE`
	`UserPoolClientLogPolicy AWS::IAM::Policy`	

Task	描述	所需技能

	<pre>CREATE_IN
	_PROGRESS</pre>
	您还可以使用 AWS Cognito 控制台查看这些资源(查找 Cognito 用户池和身份池)。
	
	此步骤使用 Cognito 用户池和身份池配置更新 React 应用程序的 `src` 文件夹中的 `aws-exports.js` 文件。	

更改 App.js 文件

Task	描述	所需技能

更改 App.js 文件。	在 `src` 文件夹中，打开并修订 `App.js` 文件。修改后的文件应如下所示：
	
	<pre>{ App.Js File after
	modifications:
	import React from
	'react';
	import logo from './
	logo.svg';
	import './App.css';
	import { Amplify } from
	'aws-amplify';
	import { withAuthe
	nticator, Button,
	Heading } from '@aws-
	amplify/ui-react';
	import awsconfig from
	'./aws-exports';
	Amplify.configure(a
	wsconfig);</pre>
		应用程序开发人员

Task	描述	所需技能
	`function App({ signOut }) { return ( <div> <h1>Thankyou for doing verification</h1> <h2>My Content</h2> <button onClick={signOut}>Sign out</button> </div> ); } export default withAuthenticator(App);`	
导入 React 程序包。	App.js 文件导入了两个 React 程序包。使用命令安装这些程序包： `amplify-react-application1 % npm install --save aws-amplify @aws-amplify/ui-react`	应用程序开发人员

启动 React 应用程序并检查身份验证

Task	描述	所需技能
启动应用程序。	在本地计算机上启动 React 应用程序： `amplify-react-application1 % npm start`	应用程序开发人员，常规 AWS

Task	描述	所需技能
检查身份验证。	检查应用程序是否提示输入身份验证参数。（在我们的示例中，我们已将电子邮件配置为登录方式。） 前端用户界面应提示您输入登录凭证，并提供用于创建账户的选项。 您还可以配置 Amplify 构建流程，将后端添加为持续部署工作流的一部分。然而，此模式不涵盖该选项。	应用程序开发人员，常规 AWS

相关资源

- 入门（npm 文档）
- 创建独立 Amazon Web Services account（Amazon Web Services account 管理文档）
- AWS Amplify 文档
- Amazon Cognito 文档

使用 AWS Amplify、Angular 和模块联合为微前端创建门户

由 Milena Godau (AWS) 和 Pedro Garcia (AWS) 创作

摘要

微前端架构使多个团队能够独立处理前端应用程序的不同部分。每个团队都可以在不干扰应用程序其他部分的情况下开发、构建和部署前端的片段。从最终用户的角度来看，它似乎是一个单一的、有凝聚力的应用程序。但是，他们正在与多个由不同团队发布的独立应用程序进行交互。

本文档介绍如何使用 Angular 前端框架和模块AWS Amplify联合来创建微前端架构。在这种模式中，微前端通过外壳（或父级）应用程序在客户端进行组合。shell 应用程序充当检索、显示和集成微前端的容器。shell 应用程序处理全局路由，它加载不同的微前端。@angular-architects/模块联合插件将模块联邦与 Angular 集成在一起。您可以通过使用部署 shell 应用程序和微前端。 AWS Amplify最终用户通过基于 Web 的门户访问该应用程序。

门户是垂直分割的。这意味着微前端是整个视图或一组视图，而不是同一视图的一部分。因此，shell 应用程序一次只能加载一个微前端。

微前端是作为远程模块实现的。shell 应用程序会延迟加载这些远程模块，这会将微前端初始化推迟到需要时再进行初始化。这种方法通过仅加载必要的模块来优化应用程序性能。这缩短了初始加载时间并改善了整体用户体验。此外，您可以通过 webpack 配置文件 (webpack.config.js) 在模块之间共享常见的依赖关系。这种做法促进了代码的重用，减少了重复，并简化了捆绑过程。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- Node.js 和 npm，已安装
- Amplify CLI，已安装
- Angular CLI，已安装
- 使用@@ 权限 AWS Amplify
- 熟悉 Angular

产品版本

- Angular CLI 版本 13.1.2 或更高版本

- @angular-architects/模块联合版本 14.0.1 或更高版本
- webpack 版本 5.4.0 或更高版本
- AWS Amplify 第 1 代

限制

微前端架构是构建可扩展且具有弹性的 Web 应用程序的强大方法。但是，在采用这种方法之前，了解以下潜在挑战至关重要：

- 集成 — 与整体式前端相比，关键挑战之一是复杂性可能会增加。编排多个微前端、处理它们之间的通信以及管理共享依赖关系可能更加复杂。此外，微前端之间的通信可能会带来性能开销。这种通信会增加延迟并降低性能。这需要通过有效的消息传递机制和数据共享策略来解决。
- 代码重复 — 由于每个微前端都是独立开发的，因此存在为通用功能或共享库复制代码的风险。这可能会增加应用程序的总体规模并带来维护难题。
- 协调和管理 — 跨多个微前端协调开发和部署流程可能具有挑战性。在分布式架构中，确保版本控制一致、管理依赖关系和维护组件之间的兼容性变得更加重要。建立明确的治理、指导方针以及自动测试和部署管道对于实现无缝协作和交付至关重要。
- 测试 — 测试微前端架构可能比测试单片前端更为复杂。它需要付出额外的努力和专门的测试策略来执行跨组件集成测试和 end-to-end测试，并验证跨多个微前端的一致用户体验。

在承诺使用微前端方法之前，我们建议您先查看 “了解和实现微前端”。 AWS

架构

在微前端架构中，每个团队都独立开发和部署功能。下图显示了多个 DevOps 团队如何协同工作。门户网站团队开发 shell 应用程序。shell 应用程序充当容器。它检索、显示和集成其他团队发布的微前端应用程序。 DevOps 您用于发布 AWS Amplify shell 应用程序和微前端应用程序。

架构图显示了以下工作流程：

1. 门户网站团队开发和维护 shell 应用程序。shell 应用程序协调了微前端的集成和渲染，以构成整个门户。
2. A 和 B 团队开发并维护一个或多个集成到门户中的微前端或功能。每个团队都可以在各自的微前端上独立工作。

3. 最终用户使用 Amazon Cognito 进行身份验证。
4. 最终用户访问门户，并加载 shell 应用程序。当用户导航时，shell 应用程序会处理路由并检索请求的微前端，加载其捆绑包。

工具

AWS 服务

- AWS Amplify是一组专门构建的工具和功能，可帮助前端 Web 和移动开发人员快速构建全栈应用程序。 AWS在这种模式下，你可以使用 Amplify CLI 来部署 Amplify 微前端应用程序。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行 shell 中的命令进行交互。

其他工具

- @angular-architects/module-federation 是一款将 Angular 与模块联合集成在一起的
- Angular 是一个开源 Web 应用程序框架，用于构建现代、可扩展和可测试的单页应用程序。它遵循基于组件的模块化架构，可促进代码的重用和维护。
- Node.js 是一个事件驱动的 JavaScript 运行时环境，专为构建可扩展的网络应用程序而设计。
- npm 是在 Node.js 环境中运行的软件注册表，用于共享或借用软件包以及管理私有软件包的部署。
- Webpack Module Federation 可帮助您将独立编译和部署的代码（例如微前端或插件）加载到应用程序中。

代码存储库

此模式的代码可在微前端门户中使用 Angular 和 Module Federation 存储库获得。 GitHub 此存储库包含以下两个文件夹：

- `shell-app`包含 shell 应用程序的代码。
- `feature1-app`包含一个微前端示例。shell 应用程序获取此微前端并将其显示为门户应用程序中的一个页面。

最佳实践

微前端架构具有许多优点，但也带来了复杂性。以下是实现流畅开发、高质量代码和卓越用户体验的一些最佳实践：

- 规划和沟通 — 为了简化协作，请在前期规划、设计和清晰的沟通渠道上进行投资。
- 设计一致性 — 使用设计系统、样式指南和组件库，在微前端强制实现一致的视觉风格。这提供了有凝聚力的用户体验并加快了开发速度。
- 依赖关系管理 — 由于微前端是独立发展的，因此采用标准化的合同和版本控制策略来有效地管理依赖关系并防止出现兼容性问题。
- 微前端架构 — 为了实现独立的开发和部署，每个微前端都应该对封装的功能负有明确而明确的责任。
- 集成和通信 — 为了促进平稳集成并最大限度地减少冲突，请在微前端（包括 APIs事件和共享数据模型）之间定义明确的合同和通信协议。
- 测试和质量保证 — 为微前端实施测试自动化和持续集成管道。这提高了整体质量，减少了手动测试工作量，并在微前端交互之间验证了功能。
- 性能优化- 持续监控性能指标并跟踪微前端之间的依赖关系。这可以帮助您识别瓶颈并保持最佳的应用程序性能。为此，请使用性能监控和依赖关系分析工具。
- 开发者体验 — 通过提供清晰的文档、工具和示例，专注于开发者体验。这可以帮助您简化开发流程并加入新的团队成员。

操作说明

创建 shell 应用程序

Task	描述	所需技能

创建 shell 应用程序。	1. 在 Angular CLI 中，输入以下命令：
	`ng new shell --routing`
	2. 输入以下命令以导航到项目文件夹：
	`cd shell`
		应用程序开发人员

Task	描述	所需技能
	Note 外壳和微前端应用程序的文件夹和项目结构可以完全独立。它们可以作为独立的 Angular 应用程序来处理。	
安装 插件。	在 Angular CLI 中，输入以下命令安装 @angular-architects/module-Federations 插件： `ng add @angular-architects/module-federation --project shell --port 4200`	应用程序开发人员
将微前端 URL 添加为环境变量。	1. 打开环境.ts 文件。 2. `mfe1URL: 'http://localhost:5000'` 添加到`environment` 对象： `export const environment = { production: false, mfe1URL: 'http://localhost:5000', };` 3. 保存并关闭环境.ts 文件。	应用程序开发人员

Task	描述	所需技能

定义路由。	1. 打开 app-routin g.module.ts 文件。
	2. 在 Angular CLI 中，输入以下命令从 @angular-architects/module-Federations 插件中导入`loadRemoteModule` 模块：
	`import { loadRemoteModule } from '@angular-architects/module-federation';`
	3. 按以下方式设置默认路由：
	`{ path: '', pathMatch: 'full', redirectTo: 'mfe1' },`
	4. 为微前端设置路由：
	`{ path: 'mfe1', loadChildren: () => loadRemoteModule({ type: 'module', remoteEntry: `${environment.mfe1URL}/remoteEntry.js`, exposedModule: './Module'`
		应用程序开发人员

Task	描述	所需技能

	``
	      })
	      .then(m =>
	m.Mfe1Module)
	 },
	``
	5. 保存并关闭 app-routing.module.ts 文件。	

声明mfe1模块。	1. 在该src文件夹中，创建一个名为 d ecl.d.ts 的新文件。 2. 打开 d ecl.d.ts 文件。 3. 将以下内容添加到文件中： ``` declare module 'mfe1/Module'; ``` 4. 保存并关闭 d ecl.d.ts 文件。	应用程序开发人员

Task	描述	所需技能

为微前端做好预加载准备。	预加载微前端有助于 webpack 正确协商共享库和软件包。
	1. 打开主.ts 文件。
	2. 将内容替换为以下内容：
	`import { loadRemoteEntry } from '@angular-architects/module-federation';`
	`Promise.all([ loadRemoteEntry(`${environment.mfe1URL}/remoteEntry.js`, 'mfe1'), ]) .catch(err => console.error('Error loading remote entries', err)) .then(() => import('./bootstrap')) .catch(err => console.error(err) );`
	3. 保存并关闭主.ts 文件。
		应用程序开发人员

Task	描述	所需技能
调整 HTML 内容。	1. 打开 app.component.html 文件。 2. 将内容替换为以下内容： `<h1>Shell application is running!</h1><router-outlet></router-outlet>` 3. 保存并关闭 app.component.html 文件。	应用程序开发人员

创建微前端应用程序

Task	描述	所需技能

创建微前端。	1. 在 Angular CLI 中，输入以下命令：
	`ng new mfe1 --routing`
	2. 输入以下命令以导航到项目文件夹：
	`cd mfe1`
		应用程序开发人员

安装 插件。	输入以下命令安装 @angular-architects/module-Federation 插件： `ng add @angular-architects/module-federation --project mfe1 --port 5000`	应用程序开发人员

Task	描述	所需技能

创建模块和组件。	输入以下命令来创建模块和组件并将其导出为远程输入模块：
	`ng g module mfe1 --routing`
	`ng g c mfe1`
		应用程序开发人员

设置默认路由路径。	1. 打开 mfe-routing.module.ts 文件。 2. 按以下方式设置默认路由： `{ path: '', component: Mfe1Component },` 3. 保存并关闭 mfe-routing.module.ts 文件。	应用程序开发人员

Task	描述	所需技能

添加路mfe1线。	1. 打开 app-routin g.module.ts 文件。
	2. 按以下方式设置默认路由：
	`{ path: '', pathMatch: 'full', redirectTo: 'mfe1' },`
	3. 添加以下mfe1路线：
	`{ path: 'mfe1', loadChildren: () => import('./mfe1/mfe1.module').then((m) => m.Mfe1Module), },`
	4. 保存并关闭 app-routin g.module.ts 文件。
		应用程序开发人员

Task	描述	所需技能

编辑 webpack.config.js 文件。	1. 打开 webpack.config.js 文件。
	2. 编辑该For remotes部分以匹配以下内容：
	`// For remotes (please adjust)`
	`name: "mfe1",`
	`filename: "remoteEntry.js",`
	`exposes: {`
	`'./Module': './src/app/mfe1/mfe1.module.ts',`
	`},`
	3. 在该shared部分中，添加应用程序与 shell mfe1 应用程序共享的所有依赖关系：
	`shared: share({`
	`"@angular/core": { singleton: true, strictVersion: true, requiredVersion: 'auto' },`
	`"@angular/common": { singleton: true, strictVersion: true, requiredVersion: 'auto' },`
	`"@angular/common/http": { singleton: true, strictVersion: true, requiredVersion: 'auto' },`
	`"@angular/router": { singleton: true, strictVer`
		应用程序开发人员

Task	描述	所需技能
	sion: true, requiredV ersion: 'auto' }, ...shared Mappings.getDescri ptors() })	
	4. 保存并关闭 webpack.c onfig.js 文件。	
调整 HTML 内容。	1. 打开 app.component.html 文件。 2. 将内容替换为以下内容： <router-outlet></r outer-outlet> 3. 保存并关闭 app.compo nent.html 文件。	应用程序开发人员

在本地运行应用程序

Task	描述	所需技能
运行mfe1应用程序。	1. 输入以下命令启动mfe1应用程序： npm start 2. 在 Web 浏览器中，访问http://localhost:5000 。 3. 验证微前端是否可以独立运行。mfe1应用程序应正确呈现，不会出现任何错误。	应用程序开发人员

Task	描述	所需技能
运行 shell 应用程序。	1. 输入以下命令启动 shell 应用程序： `npm start` 2. 在 Web 浏览器中，访问`http://localhost:4200/mfe1` 。 3. 验证`mfe1`微前端是否嵌入在 shell 应用程序中。门户应用程序应正确呈现，不会出现任何错误，并且该`mfe1`应用程序应嵌入其中。	应用程序开发人员

重构 shell 应用程序以处理微前端加载错误

Task	描述	所需技能

创建模块和组件。	在 shell 应用程序的根文件夹中，输入以下命令为错误页面创建模块和组件：
	`ng g module error-page --routing`
	`ng g c error-page`
		应用程序开发人员

调整 HTML 内容。	1. 打开 error-pag e.component.html 文件。 2. 将内容替换为以下内容： `<p>Sorry, this page is not available.</p>` 3. 保存并关闭 error-pag e.component.html 文件。	应用程序开发人员

Task	描述	所需技能
设置默认路由路径。	1. 打开 error-page-routing .module.ts 文件。 2. 按以下方式设置默认路由： `{ path: '', component: ErrorPageComponent },` 3. 保存并关闭 error-page-routing.module.ts 文件。	应用程序开发人员

Task	描述	所需技能

创建用于加载微前端的函数。	1. 打开 app-routin g.module.ts 文件。
	2. 创建以下函数：
	`function loadMFE(url: string) { return loadRemoteModule({ type: 'module', remoteEntry: `${url}/remoteEntry.js`, exposedModule: './Module' }) .then(m => m.Mfe1Module) .catch( () => import('./error-page/error-page.module').then(m => m.ErrorPageModule) ); }`
	3. 将mfe1路径修改为以下内容：
	`{ path: 'mfe1', loadChildren: () => loadMFE(environment.mfe1URL) },`
	4. 保存并关闭 app-routin g.module.ts 文件。
		应用程序开发人员

Task	描述	所需技能

测试错误处理。	1. 如果它尚未运行，请输入以下命令来启动 shell 应用程序：
	`npm start`
	2. 在 Web 浏览器中，访问`http://localhost:4200/mfe1` 。
	3. 验证错误页面是否已呈现。您应该会看到以下文本：
	`Sorry, this page is not available.`
		应用程序开发人员

使用以下方法部署应用程序 AWS Amplify

Task	描述	所需技能
部署微前端。	1. 在 Amplify CLI 中，导航到微前端应用程序的根文件夹。 2. 输入以下命令来初始化 Amplify： `amplify init` 3. 当系统提示你输入 Amplify 项目的名称时，请按 Enter。这会重复使用 package.json 文件中的名称。	AWS 应用程序开发人员 DevOps

Task	描述	所需技能

	4. 当系统提示您使用上述配置初始化项目时，请输入`Yes`。
	5. 当系统提示您选择身份验证方法时，选择`AWS Profile`。
	6. 选择要使用的配置文件。
	7. 等待 Amplify 初始化项目。此过程完成后，您将在终端中收到一条确认消息。
	8. 输入以下命令将 Amplify 托管类别添加到微前端：
	`amplify add hosting`
	9. 当系统提示您选择插件模块时，选择`Hosting with Amplify Console`。
	10. 当系统提示您选择类型时，选择`Manual deployment`。
	11. 输入以下命令安装项目 npm 依赖项：
	`npm install`
	12. 输入以下命令，将应用程序发布到 Amplify 控制台：
	`amplify publish -y`
	发布完成后，Amplify 会返回微前端的网址。	

Task	描述	所需技能
	13复制 URL。你需要这个值来更新 shell 应用程序。	

Task	描述	所需技能

部署 shell 应用程序。	1. 在该src/app/environments文件夹中，打开 environments.prod.ts 文件。
	2. 将该mfe1URL值替换为已部署的微前端的 URL：
	`export const environment = { production: true, mfe1URL: 'https://<env>.<Amplify-app-ID>.amplifyapp.com' };`
	3. 保存并关闭 environments.prod.ts 文件。
	4. 在 Amplify CLI 中，导航到外壳应用程序的根文件夹。
	5. 输入以下命令来初始化 Amplify：
	`amplify init`
	6. 当系统提示你输入 Amplify 项目的名称时，请按 Enter。这会重复使用 package.json 文件中的名称。
	7. 当系统提示您使用上述配置初始化项目时，请输入Yes。
	8. 当系统提示您选择身份验证方法时，选择AWS Profile。
	9. 选择要使用的配置文件。
		应用程序开发人员、应用程序所有者

Task	描述	所需技能

	10.等待 Amplify 初始化项目。此过程完成后，您将在终端中收到一条确认消息。
	11.在外壳应用程序中添加 Amplify 托管类别：
	`amplify add hosting`
	12.当系统提示您选择插件模块时，选择`Hosting with Amplify Console`。
	13.当系统提示您选择类型时，选择`Manual deployment`。
	14.输入以下命令安装项目 npm 依赖项：
	`npm install`
	15.输入以下命令，将外壳应用程序发布到 Amplify 控制台：
	`amplify publish -y`
	发布完成后，Amplify 会返回已部署的外壳应用程序的网址。
	16.记下外壳应用程序的 URL。	

Task	描述	所需技能

启用 CORS。	由于 shell 和微前端应用程序独立托管在不同的域上，因此您必须在微前端启用跨源资源共享 (CORS)。这允许 shell 应用程序从不同的来源加载内容。要启用 CORS，您需要添加自定义标头。
	1. 在 Amplify CLI 中，导航到微前端的根文件夹。
	2. 输入以下命令：
	`amplify configure hosting`
	3. 当系统提示您配置自定义设置时，请输入Y。
	4. 登录 AWS Management Console，然后打开 Amplify 控制台。
	5. 选择微前端。
	6. 在导航窗格中，选择托管，然后选择自定义标题。
	7. 选择编辑。
	8. 在编辑自定义标题窗口中，输入以下内容：
	`customHeaders:`
	`  - pattern: '*.js'`
	`    headers:`
	`      - key: Access-Control-Allow-Origin`
	`        value: '*'`
	`      - key: Access-Control-Allow-Methods`
		AWS 应用程序开发人员 DevOps

Task	描述	所需技能

	<pre>value: 'GET, OPTIONS'
	- key: Access-Control-Allow-Headers
	value: '*'</pre>
	9. 选择保存。
	10重新部署微前端以应用新的自定义标头。	

Task	描述	所需技能

在 shell 应用程序上创建重写规则。	Angular 外壳应用程序配置为使用 HTML5 路由。如果用户执行硬刷新，Amplify 会尝试从当前网址加载页面。这会生成 403 错误。为避免这种情况，您可以在 Amplify 控制台中添加一条重写规则。
	
	要创建重写规则，请执行以下步骤：
	
	1. 在 Amplify CLI 中，导航到外壳应用程序的根文件夹。
	2. 输入以下命令：
	
	`amplify configure hosting`
	
	3. 当系统提示您配置自定义设置时，请输入Y。
	4. 打开 Amplify 控制台。
	5. 选择外壳应用程序。
	6. 在导航窗格中，依次选择托管、重写和重定向。
	7. 在重写和重定向页面上选择管理重定向。
	8. 选择 “打开文本编辑器”。
	9. 在 JSON 编辑器中，输入以下重定向：
	
	`[ { "source": "/<*>",`
		AWS 应用程序开发人员 DevOps

Task	描述	所需技能

	<pre>"target": "/index.html",
	"status": "404-200",
	"condition": null
	}
	]</pre>
	10选择保存。	

测试门户网站。	1. 在 Web 浏览器中，输入已部署的 shell 应用程序的 URL。 2. 验证 shell 应用程序和微前端是否正确加载。	应用程序开发人员

清理资源

Task	描述	所需技能
删除应用程序。	如果您不再需要 shell 和微前端应用程序，请将其删除。这有助于防止对您未使用的资源收费。 1. 登录 AWS Management Console，然后打开 Amplify 控制台。 2. 选择微前端。 3. 在导航窗格中，选择应用程序设置，然后选择常规设置。 4. 选择 “删除应用程序”。	常规 AWS

Task	描述	所需技能
	5. 在确认窗口中，输入 `delete`，然后选择删除应用程序。 6. 重复这些步骤以删除 shell 应用程序。	

故障排除

事务	解决方案
运行`amplify init`命令时没有可用的 AWS 配置文件	如果您没有配置 AWS 配置文件，您仍然可以继续执行该`amplify init`命令。但是，当系统提示您输入身份验证方法时，您需要选择该`AWS access keys`选项。准备好您的 AWS 访问密钥和私有密钥。 或者，您可以为配置命名的配置文件 AWS CLI。有关说明，请参阅 AWS CLI 文档中的配置和凭据文件设置。
加载远程条目时出错	如果在 shell 应用程序的 main.ts 文件中加载远程条目时遇到错误，请确保该`environment.mfe1URL` 变量设置正确。此变量的值应为微前端的 URL。
访问微前端时出现 404 错误	如果您在尝试访问本地微前端（例如 at）时出现 404 错误`http://localhost:4200/mfe1`，请检查以下内容： • 对于 shell 应用程序，请确保 app-routing.module.ts 文件中的路由配置设置正确，并确保该函数正确调用了微前端。`loadRemoteModule`

事务	解决方案
	• 对于微前端，请验证 webpack.config.js 文件的**exposes**配置是否正确，并确保正确生成 remoteEntry.js 文件。

其他信息

AWS 文档

- 在 AWS（AWS 规范性指导）上理解和实现微前端
- Amplify CLI（Amplify 文档）
- Amplify 托管（Amplify 文档）

其他参考资料

- 模块联合
- Node.js
- 有角度
- @angular-建筑师/模块联盟

将基于 React 的单页应用程序部署到 Amazon S3 CloudFront

由 Jean-Baptiste Guillois (AWS) 编写

摘要

单页应用程序 (SPA) 是一种网站或 Web 应用程序，它使用 JavaScript APIs动态更新所显示网页的内容。这种方法增强了网站的用户体验和性能，因为它仅更新新数据，而不是从服务器重新加载整个网页。

这种模式提供了 step-by-step一种在亚马逊简单存储服务 (Amazon S3) Simple Service 和亚马逊上用 React 编写的 SPA 编码和托管的方法。 CloudFront这种模式中的 SPA 使用在 Amazon API Gateway 中配置并通过亚马逊 CloudFront 分发公开的 REST API 来简化跨域资源共享 (CORS) 管理。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- Node.js 和 npm，已安装并配置。有关更多信息，请参阅 Node.js 文档的下载部分。
- Yarn，已安装和配置。有关更多信息，请参阅 Yarn 文档。
- Git，已安装和配置。有关更多信息，请参阅 Git 文档。

架构

此架构是使用 AWS CloudFormation （基础架构即代码）自动部署的。它使用诸如亚马逊 S3 之类的区域服务来存储静态资产，使用 CloudFront 带有 Amazon API Gateway 的亚马逊来公开区域 API (REST) 终端节点。应用程序日志是使用 Amazon 收集的 CloudWatch。所有 AWS API 调用均在中进行审计 AWS CloudTrail。所有安全配置（例如身份和权限）均在 AWS Identity and Access Management (IAM) 中管理。静态内容通过亚马逊 CloudFront 内容分发网络 (CDN) 传送，DNS 查询由亚马逊 Route 53 处理。

工具

Amazon Web Services

- Amazon API Gateway 可帮助您创建、发布、维护、监控和保护任何规模的 RES WebSocket APIs T、HTTP。

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在资源的整个生命周期中跨地区对其 AWS 账户 进行管理。
- Amazon 通过全球数据中心网络交付您的网页内容，从而降低延迟并提高性能，从而 CloudFront加快网络内容的分发。
- AWS CloudTrail帮助您审计您的治理、合规和运营风险 AWS 账户。
- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和运行的应用程序 AWS 的指标。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- Amazon Route 53 是一种可用性高、可扩展性强的 DNS Web 服务。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

代码

此模式的示例应用程序代码可在 GitHub 基于 React 的 CORS 单页应用程序存储库中找到。

最佳实践

通过使用 Amazon S3 对象存储，您可以以安全、高弹性、高性能且经济实惠的方式存储应用程序的静态资产。此任务无需使用专用容器或亚马逊弹性计算云 (Amazon EC2) 实例。

通过使用 Amazon CloudFront 内容分发网络，您可以减少用户在访问您的应用程序时可能遇到的延迟。您还可以连接 Web 应用程序防火墙 (AWS WAF)，以保护您的资产免受恶意攻击。

操作说明

本地构建和部署您的应用程序

Task	描述	所需技能
克隆存储库。	运行以下命令以克隆示例应用程序的存储库： `git clone https://github.com/aws-samples/react-cors-spa react-cors-spa && cd react-cors-spa`	AWS 应用程序开发人员 DevOps

Task	描述	所需技能
本地部署应用程序。	1. 在项目目录中，运行 `npm install` 命令，以启动应用程序依赖项。 2. 运行 `yarn dev` 命令在本地启动应用程序。	AWS 应用程序开发人员 DevOps
在本地访问应用程序。	打开浏览器窗口并输入访问应用程序的 `http://localhost:3000` URL。	AWS 应用程序开发人员 DevOps

部署应用程序

Task	描述	所需技能
部署 AWS CloudFormation 模板。	1. 登录 AWS Management Console，然后打开AWS CloudFormation 控制台。 2. 选择 Create stack（创建堆栈），然后选择 With new resources (standard)（使用新资源（标准））。 3. 选择上传模板文件。 4. 选择选择文件，从克隆的存储库中选择 `react-cors-spa-stack.yaml` 文件，然后选择下一步。 5. 输入您堆栈的名称，然后选择下一步。 6. 保留所有默认选项，然后选择下一步。 7. 检查堆栈的最终设置，然后选择创建堆栈。	AWS 应用程序开发人员 DevOps

Task	描述	所需技能
自定义应用程序源文件。	1. 部署堆栈后，打开输出选项卡并确定`Bucket`名称和`APIDomain` 值。 2. 复制 REST API 的 CloudFront 分发域。 3. 导航到`index.tsx` 文件第 13 行的变量值`<project_root>/src/pages/index.tsx` ，然后将此域插入或粘贴到`APIEndPoint` 变量值中。	应用程序开发人员
构建应用程序包。	在项目目录中，运行 `yarn build` 命令以生成应用程序包。	应用程序开发人员
部署应用程序包。	1. 打开 Amazon S3 控制台。 2. 识别并选择之前由 CloudFormation 堆栈创建的 S3 存储桶。 3. 选择上传，然后选择添加文件。 4. 选择您的`out`文件夹的内容。 5. 选择 “添加文件夹”，然后选择`_next`目录。 **Important** 选择`_next`目录，而不是内容。 6. 选择上传，将文件和目录上传至您的 S3 存储桶。	AWS 应用程序开发人员 DevOps

测试应用程序

Task	描述	所需技能
访问和测试应用程序。	打开浏览器窗口，然后粘贴 CloudFront 分发域（您之前部署的 CloudFormation 堆栈的`SPADomain` 输出）以访问该应用程序。	AWS 应用程序开发人员 DevOps

清理资源

Task	描述	所需技能
删除 S3 存储桶内容。	1. 打开 Amazon S3 控制台，选择之前由堆栈创建的存储桶（第一个名称以开头的存储桶`react-cors-spa-`）。 2. 选择清空 可删除存储桶中的内容。 3. 选择之前由堆栈创建的第二个存储桶（名称以开头`react-cors-spa-` 和结尾的第二个存储桶`-logs`）。 4. 选择清空可删除存储桶中的内容。	AWS DevOps，应用程序开发者
删除 AWS CloudFormation 堆栈。	1. 打开AWS CloudFormation 控制台并选择您之前创建的堆栈。 2. 选择删除删除堆栈和所有相关资源。	AWS DevOps，应用程序开发者

相关资源

要部署和托管您的 Web 应用程序，您还可以使用 H AWS Amplify osting，它提供了基于 Git 的工作流程，用于托管持续部署的全栈无服务器 Web 应用程序。Amplify Hosting是其中的一员 AWS Amplify，它提供了一组专门构建的工具和功能，使前端网络和移动开发人员能够快速轻松地在其上构建全栈应用程序。 AWS

其他信息

为了处理用户提出的可能生成 403 错误的无效 URLs 请求，在 CloudFront 发行版中配置的自定义错误页面会捕获 403 错误并将其重定向到应用程序入口点 ()。`index.html`

为了简化 CORS 的管理，REST API 通过 CloudFront 发行版公开。

使用私有端点和应用程序负载均衡器在内部网站上部署 Amazon API Gateway API

由 Saurabh Kothari (AWS) 编写

摘要

此模式向您展示如何在可从本地网络访问的内部网站上部署 Amazon API Gateway API。您将学习使用专有终端节点、Application Load Balancer、AWS PrivateLink 和 Amazon Route 53 设计的架构，为私有 API 创建自定义域名。此架构可防止使用自定义域名和代理服务器帮助在 API 上进行基于域的路由带来的意外后果。例如，如果您在不可路由的子网中部署虚拟私有云（VPC）端点，则您的网络将无法访问 API Gateway。常见的解决方案是使用自定义域名，然后在可路由的子网中部署 API，但是当代理配置将流量 (`execute-api.{region}.vpce.amazonaws.com`) 传递到 AWS Direct Connect 时，这可能会破坏其他内部站点。最后，此模式可以帮助您满足使用无法通过 Internet 访问的私有 API 和自定义域名的组织要求。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 用于网站和 API 的 Server Name Indication (SNI) 证书
- 从本地环境到使用 AWS Direct Connect 或 AWS Site-to-Site VPN 设置的 AWS 账户的连接
- 带有相应域的私有托管区（例如 domain.com），该域从本地网络解析，并将 DNS 查询转发到 Route 53
- 可从本地网络访问的可路由私有子网

限制

有关负载均衡器、规则和其他资源的配额（以前称为限制）的更多信息，请参阅 Elastic Load Balancing 文档中的应用程序负载均衡器配额。

架构

技术堆栈

- Amazon API Gateway
- Amazon Route 53

- 应用程序负载均衡器
- AWS Certificate Manager
- AWS PrivateLink

目标架构

下图显示了如何在 VPC 中部署应用程序负载均衡器，该均衡器根据应用程序负载均衡器侦听器规则将 Web 流量引导到网站目标组或 API Gateway 目标组。API Gateway 目标组是 API Gateway 中 VPC 端点的 IP 地址列表。API Gateway 配置为通过其资源策略将 API 设为私有。该策略拒绝所有不是来自特定 VPC 端点的调用。API 网关中的自定义域名已更新为对 API 及其阶段使用 api.domain.com。添加了应用程序负载均衡器规则，以根据主机名路由流量。

图表显示了以下工作流：

1. 本地网络中的用户尝试访问内部网站。该请求已发送至 ui.domain.com 和 api.domain.com。然后，请求被解析至可路由私有子网的内部应用程序负载均衡器。SSL 在 ui.domain.com 和 api.domain.com 的应用程序负载均衡器处终止。
2. 在应用程序负载均衡器上配置的侦听器规则将检查主机标头。

   a. 如果主机标头为 api.domain.com，则请求将转发到 API Gateway 目标组。应用程序负载均衡器通过端口 443 启动与 API Gateway 的新连接。

   b. 如果主机标头为 ui.domain.com，则请求将转发到网站目标组。
3. 当请求到达 API Gateway 时，API Gateway 中配置的自定义域映射将确定主机名以及要运行的 API。

自动化和扩缩

可以使用 AWS CloudFormation 或 AWS Cloud Development Kit (AWS CDK) 自动执行此模式中的步骤。要配置 API Gateway 调用的目标组，您必须使用自定义资源来检索 VPC 端点的 IP 地址。API 调用describe-vpc-endpoints并describe-network-interfaces返回 IP 地址和安全组，这些地址和安全组可用于创建 API 目标 IP 地址组。

工具

- Amazon API Gateway 可帮助您创建、发布、维护、监控和保护任何规模的 RES WebSocket APIs T、HTTP。

- Amazon Route 53 是一种可用性高、可扩展性强的 DNS Web 服务。
- AWS Certificate Manager (ACM) 帮助您创建、存储和续订公有及私有 SSL/TLS X.509 证书和密钥，这些证书和密钥可保护您的 AWS 网站和应用程序。
- AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，可帮助您在代码中定义并预置 Amazon Web Services Cloud 基础设施。
- AWS PrivateLink 可帮助您创建从您 VPCs 到 VPC 之外的服务的单向私有连接。

操作说明

创建 SNI 证书

Task	描述	所需技能
创建 SNI 证书，并将该证书导入 ACM。	1. 为 ui.domain.com 和 api.domain.com 创建 SNI 证书。有关更多信息，请参阅 Amazon CloudFront 文档中的选择如何 CloudFront 处理 HTTPS 请求。 2. 导入 SNI 证书至 AWS Certificate Manager (ACM)。有关更多信息，请参阅 ACM 文档中的导入证书至 AWS Certificate Manager。	网络管理员

在不可路由私有子网中部署 VPC 端点

Task	描述	所需技能
在 API Gateway 中创建接口 VPC 端点。	要创建接口 VPC 端点，请按照 Amazon Virtual Private Cloud (Amazon VPC) 文档中的使用接口 VPC 端点访问 Amazon	云管理员

Task	描述	所需技能
	Web Services 中的说明进行操作。	

配置应用程序负载均衡器。

Task	描述	所需技能
为应用程序创建目标组。	为应用程序的 UI 资源创建目标组。	云管理员
为 API Gateway 端点创建目标组。	1. 创建具有 IP 地址类型的目标组，然后将 API Gateway 端点的 VPC 端点的 IP 地址添加至目标组。 2. 使用成功代码 403 为目标组@@ 配置运行状况检查。403 是必需的，因为目标组运行状况检查在调用 API Gateway 的 VPC 终端节点时会返回一个 403 代码，而目标组运行状况检查没有任何标头。	云管理员
创建应用程序负载均衡器。	1. 在可路由的私有子网中创建应用程序负载均衡器（内部）。 2. 将 443 侦听器添加至应用程序负载均衡器，然后从 ACM 中选择证书。	云管理员
创建侦听器规则。	创建侦听器规则，以执行以下操作： 1. 将主机 api.domain.com 转发到 API Gateway 目标组	云管理员

Task	描述	所需技能
	2. 将主机 ui.domain.com 转发到 UI 资源的目标组	

配置 Route 53

Task	描述	所需技能
创建私有托管区。	为 domain.com 创建私有托管区。	云管理员
创建域记录。	对于以下，创建 CNAME 记录： • 值设置为应用程序负载均衡器的 DNS 名称的 API • 值设置为应用程序负载均衡器的 DNS 名称的 UI	云管理员

在 API Gateway 中创建私有 API 端点

Task	描述	所需技能
创建和配置私有 API 端点。	1. 要创建私有 API 端点，请按照 API Gateway 文档在 Amazon API Gateway 中创建私有 API 的说明进行操作。 2. 将资源策略配置为仅允许从 VPC 端点调用 API。有关更多信息，请参阅 API Gateway 文档中的使用 API Gateway 资源策略控制 API 的访问权限。	应用程序开发人员、云管理员

Task	描述	所需技能
创建自定义域名。	1. 为 api.domain.com 创建自定义域名。有关更多信息，请参阅 API Gateway 文档 APIs中的为 REST 设置自定义域名。 2. 选择已创建的 API 和阶段。有关更多信息，请参阅 API Gateway 文档 APIs中的使用 REST 的 API 映射。	云管理员

相关资源

- Amazon API Gateway
- Amazon Route 53
- 应用程序负载均衡器
- AWS PrivateLink
- AWS Certificate Manager

在本地 Angular 应用程序中嵌入亚马逊 QuickSight 控制面板

创建者：Sean Griffin (AWS) 和 Milena Godau (AWS)

摘要

此模式为将 Amazon QuickSight 控制面板嵌入到本地托管的 Angular 应用程序中进行开发或测试提供了指导。中的嵌入式分析功能本身 QuickSight 不支持此功能。它需要一个拥有现有仪表板和Angular知识的 QuickSight 帐户。

使用嵌入式 QuickSight 仪表板时，通常必须将应用程序托管在 Web 服务器上才能查看仪表板。这使得开发变得更加困难，因为您必须不断地将更改推送到 Web 服务器以确保一切正常运行。此模式展示了如何运行本地托管的服务器，以及如何使用 QuickSight 嵌入式分析来简化开发流程。

先决条件和限制

先决条件

- 一个活动 Amazon Web Services (AWS) 账户
- 按会话容量定价的活跃 QuickSight 账户
- QuickSight 已安装嵌入 SDK
- 已安装 Angular CLI
- 熟悉 Angular
- 已安装 mkcert

限制

- 此模式为使用ANONYMOUS（可公开访问的）身份验证类型嵌入 QuickSight 仪表板提供了指导。如果您使用的是 AWS Identity and Access Management (IAM) 或使用嵌入式控制面板进行 QuickSight 身份验证，则提供的代码将不适用。但是，操作说明部分中托管 Angular 应用程序的步骤仍然有效。
- 将 GetDashboardEmbedUrlAPI 与ANONYMOUS身份类型一起使用需要 QuickSight 容量定价计划。

版本

- Angular CLI 版本 13.3.4
- QuickSight 嵌入 SDK 版本 2.3.1

架构

技术堆栈

- Angular 前端
- AWS Lambda 和 Amazon API Gateway 后端

架构

在这种架构中，API Gateway APIs 中的 HTTP 使本地 Angular 应用程序能够调用 Lambda 函数。Lambda 函数返回用于嵌入控制面板的网址。 QuickSight

自动化和扩缩

您可以使用 AWS CloudFormation 或 AWS 无服务器应用程序模型 (AWS SAM) Model 自动进行后端部署。

工具

工具

- Angular CLI 是命令行界面工具，用于直接从命令 Shell 初始化、开发、搭建和维护 Angular 应用程序。
- QuickSight 嵌入 SDK 用于将 QuickSight 仪表板嵌入到您的 HTML 中。
- mkcert 是用于创建本地受信任开发证书的简单工具。它不需要配置。mkcert 是必需的，因为只 QuickSight 允许 HTTPS 请求嵌入仪表板。

Amazon Web Services

- A@@ mazon API Gateway 是一项 AWS 服务，用于创建、发布、维护、监控和保护任何规模的 RES WebSocket APIs T、HTTP。
- AWS Lambda 是一项计算服务，支持无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。您只需为消耗的计算时间付费 - 代码未运行时不产生费用。
- Amazon QuickSight 是一项商业分析服务，用于构建可视化效果、执行临时分析以及从数据中获取业务见解。

操作说明

生成 EmbedURL

Task	描述	所需技能
创建 EmbedUrl 策略。	创建名为的 IAM 策略 QuicksightGetDashboardEmbedUrl，该策略具有以下属性。 `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "quicksight:GetDashboardEmbedUrl", "quickSight:GetAnonymousUserEmbedUrl" ], "Resource": "*" } ] }`	AWS 管理员
创建 Lambda 函数。	1. 在 Lambda 控制台上，打开函数页面。 2. 选择创建函数。 3. 选择从头开始创作。	应用程序开发人员

Task	描述	所需技能

	4. 对于函数名称，请输入 `get-qs-embed-url` 。
	
	5. 对于 Runtime（运行时），选择 Python 3.9。
	
	6. 选择创建函数。
	
	7. 在代码选项卡上，将以下代码复制到 Lambda 函数中。
	
	`import json`
	`import boto3`
	`from botocore.`
	`exceptions import`
	`ClientError`
	`import time`
	`from os import environ`
	
	`qs = boto3.client('quic`
	`ksight',region_nam`
	`e='us-east-1')`
	`sts = boto3.cli`
	`ent('sts')`
	
	`ACCOUNT_ID = boto3.cli`
	`ent('sts').get_cal`
	`ler_identity().get`
	`('Account')`
	`DASHBOARD_ID =`
	`environ['DASHBOARD`
	`_ID']`
	
	`def getDashboardURL(ac`
	`countId, dashboardId,`
	`quicksightNamespac`
	`e, resetDisabled,`
	`undoRedoDisabled):`
	`try:`	

Task	描述	所需技能

	<pre> response = qs.get_da
	shboard_embed_url(
	      AwsAccountId =
	 accountId,
	      DashboardId =
	 dashboardId,
	      Namespace =
	 quicksightNamespace,
	      IdentityType =
	 'ANONYMOUS',
	      SessionLi
	fetimeInMinutes = 600,
	      UndoRedoDisabled
	 = undoRedoDisabled,
	      ResetDisabled =
	 resetDisabled
	    )
	    return response
	
	  except ClientError as
	 e:
	    print(e)
	    return "Error
	 generating embeddedU
	RL: " + str(e)
	
	def lambda_handler(eve
	nt, context):
	  url = getDashbo
	ardURL(ACCOUNT_ID,
	 DASHBOARD_ID,
	 "default", True, True)
	['EmbedUrl']
	  return {
	      'statusCode':
	 200,
	      'url': url
	  }</pre>
	8. 选择部署。	

Task	描述	所需技能
将控制面板 ID 添加为环境变量。	将 DASHBOARD_ID 作为环境变量添加到您的 Lambda 函数中： 1. 在配置选项卡，选择环境变量、编辑、添加环境变量。 2. 添加含 DASHBOARD_ID 键的环境变量。 3. 要获取的值DASHBOARD_ID ，请导航到您的控制面板， QuickSight 然后在浏览器中复制网址末尾的 UUID。例如，如果 URL 是 `https://us-east-1.quicksight.aws.amazon.com/sn/dashboards/<dashboard-id>` ，则将 URL 的 `<dashboard-id>` 部分指定为密钥值。 4. 选择保存。	应用程序开发人员
添加 Lambda 函数的权限。	修改 Lambda 函数的执行角色并向其添加QuicksightGetDashboardEmbedUrl策略。 1. 在配置选项卡，选择权限，然后选择角色名称。 2. 选择附加策略，搜索 `QuicksightGetDashboardEmbedUrl` ，选中其复选框，然后选择附加策略。	应用程序开发人员

Task	描述	所需技能
测试 Lambda 函数。	创建和运行测试事件。您可以使用 Hello World 模板，因为该函数不会使用测试事件中的任何数据。 1. 选择测试选项卡。 2. 为您的测试事件命名，然后选择保存。 3. 要测试您的 Lambda 函数，请选择测试。响应结果应如下所示。 `{ "statusCode": 200, "url": "\"https://us-east-1.quicksight.aws.amazon.com/embed/f1acc0786687783b9a4543a05ba929b3a/dashboards/..." }` **Note** 如前提条件和限制部分所述，您的 QuickSight 账户必须使用会话容量定价计划。否则，此步骤会显示一条错误消息。	应用程序开发人员

Task	描述	所需技能
在 API Gateway 中创建一个 API。	1. 在 API Gateway 控制台上，选择创建 API，然后选择 REST API，构建。 • 对于 API 名称，请输入 `qs-embed-api` 。 • 选择创建 API。 2. 在操作中，选择创建方法。 • 选择 GET，并通过选中复选标记进行确认。 • 选择 Lambda 函数作为集成类型。 • 对于 Lambda 函数，请输入 `get-qs-embed-url`。 • 选择保存。 • 在向 Lambda 函数添加权限框中，选择确定。 3. 启用 CORS。 • 在操作中，选择启用 CORS。 • 对于 Access-Control-Allow-Origin，请输入 `'https://my-qs-app.net:4200'` 。 • 选择启用 CORS 并替换现有 CORS 标头，然后确认。 4. 部署 API。 • 对于操作，请选择部署 API。	应用程序开发人员

Task	描述	所需技能
	• 对于部署阶段，选择[新阶段]。 • 对于阶段名称，输入 `dev`。 • 选择部署。 • 复制调用 URL。 **Note** `my-qs-app.net` 可以是任何域。如果要使用其他域名，请务必更新步骤 3 中的 Access-Control-Allow-Origin 信息，并在后续步骤中更改 `my-qs-app.net` 。	

创建 Angular 应用程序

Task	描述	所需技能

通过 Angular CLI 创建应用程序。	1. 创建应用程序。
	`ng new quicksight-app --defaults`
	`cd quicksight-app/src/app`
	2. 创建控制面板组件。
	`ng g c dashboard`
		应用程序开发人员

Task	描述	所需技能
	3. 导航到您的 `src/environments/environment.ts` 文件并将 `apiUrl: '<Invoke URL from previous steps>'` 添加到环境对象。 `export const environment = { production: false, apiUrl: '<Invoke URL from previous steps>', };`	

添加 QuickSight 嵌入式 SDK。	1. 通过在 QuickSight 项目的根文件夹中运行以下命令来安装 Embedding SDK。
	`npm i amazon-quicksight-embedding-sdk`
	2. 在 `src` 文件夹中，创建一个使用以下内容的新 `decl.d.ts` 文件。
	`declare module 'amazon-quicksight-embedding-sdk';`
		应用程序开发人员

Task	描述	所需技能

将代码添加到您的 dashboard.component.ts 文件中。	`import { Component, OnInit } from '@angular/core';`
	`import { HttpClient } from '@angular/common/http';`
	`import * as QuicksightEmbedding from 'amazon-quicksight-embedding-sdk';`
	`import { environment } from "../../environments/environment";`
	`import { take } from 'rxjs';`
	`import { EmbeddingContext } from 'amazon-quicksight-embedding-sdk/dist/types';`
	`import { createEmbeddingContext } from 'amazon-quicksight-embedding-sdk';`
	
	`@Component({`
	`  selector: 'app-dashboard',`
	`  templateUrl: './dashboard.component.html',`
	`  styleUrls: ['./dashboard.component.scss']`
	`})`
	`export class DashboardComponent implements OnInit {`
	
	`  constructor(private http: HttpClient) { }`
		应用程序开发人员

Task	描述	所需技能

	loadingError = false;
	dashboard: any;
	
	ngOnInit() {
	this.GetDashboardU
	RL();
	}
	
	public GetDashbo
	ardURL() {
	this.http.get(envi
	ronment.apiUrl)
	.pipe(
	take(1),
	)
	.subscribe((data:
	any) => this.Dash
	board(data.url));
	}
	
	public async Dashboard
	(embeddedURL: any) {
	var containerDiv
	= document.getElemen
	tById("dashboardCo
	ntainer") || '';
	const frameOptions
	= {
	url: embeddedURL,
	container:
	containerDiv,
	height: "850px",
	width: "100%",
	resizeHei
	ghtOnSizeChangedEv
	ent: true,
	}
	const embedding
	Context: Embedding
	Context = await
	createEmbeddingCon
	text();	

Task	描述	所需技能
	`this.dashboard = embeddingContext.embedDashboard(frameOptions); } }`	
将代码添加到您的 dashboard.component.html 文件中。	将以下代码添加到您的 `src/app/dashboard/dashboard.component.html` 文件。 `<div id="dashboardContainer"></div>`	应用程序开发人员
修改您的 app.component.html 文件，以加载您的控制面板组件。	1. 删除 `src/app/app.component.html` 文件的内容。 2. 添加以下内容。 `<app-dashboard></app-dashboard>`	应用程序开发人员
导 HttpClientModule 入到你的 app.module.ts 文件中。	1. 在 `src/app/app.module.ts` 文件顶部，添加以下内容。 `import { HttpClientModule } from '@angular/common/http';` 2. 在 `imports` 数组中为您的 `AppModule` 添加 `HttpClientModule` 。	应用程序开发人员

托管 Angular 应用程序

Task	描述	所需技能

配置 mkcert。	**Note**
	以下命令适用于 Unix 或 macOS 计算机。如果您使用的是 Windows，请参阅其他信息部分了解等效的 echo 命令。
	
	1. 在您的计算机创建本地证书颁发机构 (CA)。
	`mkcert -install`
	2. 配置 `my-qs-app.net` ，以始终重定向到本地 PC。
	`echo "127.0.0.1 my-qs-app.net" | sudo tee -a /private/etc/hosts`
	3. 确保您位于 Angular 项目的 `src` 目录中。
	`mkcert my-qs-app.net 127.0.0.1`
		应用程序开发人员

配置 QuickSight 为允许您的域名。	1. 在中 QuickSight，在右上角选择您的姓名，然后选择“管理 Quicksight”。 2. 导航至域和嵌入。	AWS 管理员

Task	描述	所需技能
	3. 添加 `https://my-qs-app.net:4200` 作为允许的域。	
测试解决方案。	运行以下命令，启动本地 Angular 开发服务器。 `ng serve --host my-qs-app.net --port 4200 --ssl --ssl-key "./src/my-qs-app.net-key.pem" --ssl-cert "./src/my-qs-app.net.pem" -o` 这将启用您之前创建的自定义证书的 Secure Sockets Layer (SSL)。 构建完成后，它会打开一个浏览器窗口，你可以查看在 Angular 中本地托管的嵌入式 QuickSight 仪表板。	应用程序开发人员

相关资源

- Angular 网站
- 为匿名（未注册）用户嵌入 QuickSight 数据仪表板（QuickSight 文档）
- QuickSight 嵌入 SDK
- mkcert 工具

其他信息

如果您使用的是 Windows，请以管理员身份运行 Command Prompt 窗口，然后使用以下命令配置 `my-qs-app.net` 为始终重定向到本地 PC。

```
echo 127.0.0.1 my-qs-app.net >> %WINDIR%\System32\Drivers\Etc\Hosts
```

使用 Green Boost 探索全栈云原生 Web 应用程序开发

由 Ben Stickley (AWS) 和 Amiin Samatar (AWS) 编写

摘要

为了应对开发人员不断变化的需求，Amazon Web Services (AWS) 意识到开发云原生 Web 应用程序的高效方法的迫切需求。AWS 的重点是帮助您克服与在 AWS Cloud上部署 Web 应用程序相关的常见障碍。通过利用 TypeScript AWS Cloud Development Kit (AWS CDK)、React 和 Node.js 等现代技术的功能，这种模式旨在简化和加快开发流程。

在 Green Boost (GB) 工具包的支持下，该模式提供了构建充分利用 AWS 广泛功能的 Web 应用程序的实用指南。它是一份全面的路线图，引导您完成与 Amazon Aurora PostgreSQL 兼容版本集成的基本 CRUD（创建、读取、更新、删除）网络应用程序的部署过程。这是通过使用 Green Boost 命令行界面 (Green Boost CLI) 和建立本地开发环境来实现的。

成功部署应用程序后，该模式深入研究 Web 应用程序的关键组件，包括基础设施设计、后端和前端开发，以及用于可视化的 cdk-dia 等基本工具，从而促进高效的项目管理。

先决条件和限制

先决条件

- Git 已安装
- Visual Studio 代码（VS 代码）已安装
- AWS 命令行界面（AWS CLI）已安装
- 已安装 AWS CDK Toolkit
- Node.js 18 已安装，或者已激活带 pnpm 的 Node.js 18
- pnpm 已安装，前提是它不是 Node.js 安装的一部分
- 对 AWS CDK TypeScript、Node.js 和 React 有基本的熟悉程度
- 一个有效的 Amazon Web Services account
- 通过在 `us-east-1` 中使用 AWS CDK 引导的 Amazon Web Services account。`us-east-1`AWS 区域是支持 Amazon CloudFront Lambda @Edge 功能所必需的。
- 在终端环境中正确配置的 AWS 安全凭证，包括 `AWS_ACCESS_KEY_ID`
- 对于 Windows 用户，管理员模式下的终端（以适应 pnpm 处理节点模块的方式）

产品版本

- 适用于 JavaScript 版本 3 的 AWS 开发工具包
- AWS CDK 版本 2
- AWS CLI 版本 2.2
- Node.js 版本 18
- React 版本 18

架构

目标技术堆栈

- Amazon Aurora PostgreSQL 兼容版
- Amazon CloudFront
- Amazon CloudWatch
- 亚马逊弹性计算云（亚马逊 EC2）
- AWS Lambda
- AWS Secrets Manager
- Amazon Simple Notification Service(Amazon SNS)
- Amazon Simple Storage Service(Amazon S3)
- AWS WAF

目标架构

下图显示了用户请求在与 S3 存储桶、Aurora 数据库 CloudFront、实例进行交互并最终到达开发人员之前，先通过亚马逊、AWS WAF 和 AWS Lambda 传递。 EC2 另一方面，管理员使用亚马逊 SNS 和亚马逊 CloudWatch 进行通知和监控。

要在部署后更深入地了解应用程序，可以使用 cdk-dia 创建图表，如以下示例所示。

这些图表从两个不同的角度展示 Web 应用程序架构。cdk-dia 图表提供了 AWS CDK 基础设施的详细技术视图，重点介绍了特定的 Amazon Web Services，例如 Amazon Aurora PostgreSQL-Compatible 和 AWS Lambda。相比之下，另一个图采用了更广泛的视角，强调数据和用户交互的逻辑流。主要区别在于详细程度：cdk-dia 深入研究了技术的复杂性，而第一个图提供了更加以用户为中心的视图。

操作说明使用 AWS CDK 了解应用程序基础设施中介绍了 cdk-dia 图的创建。

工具

Amazon Web Services

- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩缩 PostgreSQL 部署。
- AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，可帮助您在代码中定义并预置 Amazon Web Services Cloud 基础设施。
- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- Amazon 通过全球数据中心网络交付您的网页内容，从而降低延迟并提高性能，从而 CloudFront加快网络内容的分发。
- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和在 AWS 上运行的应用程序的指标。
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Secrets Manager 帮助您将代码中的硬编码凭证（包括密码）替换为对 Secrets Manager 的 API 调用，以便以编程方式检索密钥。
- AWS Systems Manager 可帮助您管理在 Amazon Web Services Cloud 中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。此模式使用 AWS Systems Manager 会话管理器。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。
- AWS WAF 是一种 Web 应用程序防火墙，可帮助您监视转发至受保护 Web 应用程序资源的 HTTP 和 HTTPS 请求。

其他工具

- Git 是开源分布式版本控制系统。
- Green Boost 是用于在 AWS 上构建网络应用程序的工具包。

- Next.js 是用于添加功能和优化的 React 框架。
- Node.js 是一个事件驱动的 JavaScript 运行时环境，专为构建可扩展的网络应用程序而设计。
- pgAdmin 是一种适用于 PostgreSQL 的开源管理工具。它提供了一个图形界面，可帮助您创建、维护和使用数据库对象。
- pnpm 是 Node.js 项目依赖项的包管理器。

最佳实践

有关以下建议的更多信息，请参阅操作说明部分：

- 使用 Amazon CloudWatch 控制面板和警报监控基础设施。
- 使用 cdk-nag 运行静态基础设施即代码（IaC）分析，以执行 AWS 最佳实践。
- 使用 Systems Manager 会话管理器通过SSH (Secure Shell) 隧道建立数据库端口转发，这比公开的 IP 地址更安全。
- 通过运行 `pnpm audit` 来管理漏洞。
- 使用执行静态 TypeScript 代码分析 ESLint，使用 Pr ettier 来标准化代码格式，从而强制执行最佳实践。

操作说明

部署与 Aurora PostgreSQL 兼容的 CRUD Web 应用程序

Task	描述	所需技能
安装 Green Boost CLI。	要安装 Green Boost CLI，请运行以下命令。 `pnpm add -g gboost`	应用程序开发人员
创建 GB 应用程序。	1. 若要使用 Green Boost 创建应用程序，请运行命令 `gboost create`。 2. 选择 `CRUD App with Aurora PostgreSQL` 模板。	应用程序开发人员

Task	描述	所需技能
安装依赖项并部署应用程序。	1. 导航到项目目录：`cd <your directory>` 。 2. 要安装依赖项，请运行 `pnpm i` 命令。 3. 导航到 infra 目录：`cd infra`。 4. 要在本地部署应用程序，请运行命令 `pnpm deploy:local` 。 这是 `infra/package.json` 中定义的 `cdk deploy ...` 命令的别名。 等待部署完成(大约 20 分钟)。在等待期间，请在 CloudFormation 控制台中监控 AWS CloudFormation 堆栈。请注意代码中定义的构造如何映射至部署的资源。在 CloudFormation 控制台中查看 CDK 构造树视图。	应用程序开发人员

Task	描述	所需技能
访问该应用程序。	在本地部署 GB 应用程序后，您可以使用 CloudFront URL 对其进行访问。URL 打印终端输出中，但查找起来可能有点让人不知所措。若要更快地找到它，请使用以下步骤： 1. 打开运行 `pnpm deploy:local` 命令的终端。 2. 在终端输出中查找类似于以下文本部分。 `myapp5stickbui9C39A55A.CloudFrontDomainName = d1q16n5pof924c.cloudfront.net` 该网址将会是您部署的唯一名称。 或者，您可以通过访问 Amazon CloudFront 控制台来找到 CloudFront URL： 1. 登录 AWS 管理控制台并导航到该 CloudFront 服务。 2. 在列表中查找最新部署版本。 复制与分配关联的域名。这与 `your-unique-id.cloudfront.net` 类似。	应用程序开发人员

使用 Amazon 进行监控 CloudWatch

Task	描述	所需技能
查看 CloudWatch 控制面板。	1. 打开 CloudWatch 控制台并选择 “控制面板”。 2. 选择名为 <appId>-<stageName>-dashboard 的控制面板。 3. 查看控制面板。正在监控哪些资源？ 正在记录哪些指标？ 开源结构使这个仪表板成为可能cdk-monitoring-constructs。	应用程序开发人员
启用警报。	CloudWatch 控制面板可帮助您主动监控您的 Web 应用程序。若要被动监控您的 Web 应用程序，您可以启用警报。 1. 导航至 `/infra/src/app/stateless/monitor-stack.ts` ，它定义了监视器堆栈。 2. 取消对以下行的注释，并将`admin@example.com` 替换为您的电子邮件地址。 `onAlarmTopic.addSubscription(new EmailSubscription("admin@example.com"));` 3. 将以下导入信息添加到文件的顶部：	应用程序开发人员

Task	描述	所需技能
	``` import { EmailSubscription } from "aws-cdk-lib/aws-sns-subscriptions"; ``` 4. 在 `infra/` 中，运行以下命令。 ``` cdk deploy "*/monitor" --exclusively. ``` 5. 要确认您对启动监控警报时启动的 SNS 主题的订阅，请选择电子邮件中的链接。	

通过 AWS CDK 了解应用程序基础设施

Task	描述	所需技能
创建架构图。	使用 cdk-dia 生成 Web 应用程序的架构图。可视化架构有助于增进团队成员之间的理解与沟通。它清晰地概述系统的组件及其关系。 1. 安装 Graphviz。 2. 在 `infra/` 中，运行命令 `pnpm cdk-dia`。 3. 查看您的 `infra/diagram.png` 。	应用程序开发人员
使用 cdk-nag 强制执行最佳实践。	使用 cdk-nag 通过实施最佳实践，降低安全漏洞和配置错误的风险，以帮助您维护安全和合规的基础设施。	应用程序开发人员

Task	描述	所需技能
	1. 通过 cdk-nag 的规则部分探索 cdk-nag 的最佳实践执法情况，包括来自 AWS 解决方案库规则包的检查。 2. 要查看 cdk-nag 的规则执行方式，请修改代码。例如，在 `infra/src/app/stateful/data-stacks.ts` 中将 `storageEncrypted: true` 更改为 `storageEncrypted: false` 。 3. 在 `infra/` 中，运行命令 `cdk synth "*/data"`。在合成过程中，您会看到表示违反规则的编译错误。 `AwsSolutions-RDS2: The RDS instance or Aurora DB cluster does not have storage encryption enabled.` 此错误展示了 cdk-nag 如何成为一种用于强制实施基础设施最佳实践和防止安全配置错误的安全机制。 4. 如果需要，您还可以取消不同范围的规则。例如，要抑制 AwsSolutions-RDS2，请在的实例化下方添加以下代码。`DbIamCluster`	

Task	描述	所需技能

	``
	NagSuppressions.addResourceSuppressions(
	  cluster.node.findChild("Resource"),
	  [
	  {
	  id: "AwsSolutions-RDS2",
	  reason: "Customer requirement necessitates having unencrypted DB storage",
	  },
	  ],
	);
	``
	5. 抑制后，再次运行 cdk synth "*/data"。您的 AWS CDK 应用程序现在应该可成功合成了。您可以在。infra/cdk.out/assembly-<appId>-<stageName>/AwsSolutions-<appId>-<stageName>-${stackId}-NagReport.csv 中找到所有隐藏的规则	

评估数据库配置与架构

Task	描述	所需技能
获取环境变量。	若要获取所需环境变量，请使用以下步骤：	应用程序开发人员

Task	描述	所需技能
	1. 要查找DB_BASTION_ID ，请登录控制台，然后导航到 EC2 控制台。选择实例（正在运行），然后找到包含 -ssm-db-bastion 名称<stageName>的行。实例 ID 以 i- 开头。 2. 要查找 DB_ENDPOINT ，请在 Amazon Relational Database Service (Amazon RDS) 控制台，选择数据库实例，然后选择数据库标识符以 <appId>-<stageName>-data- 开头的区域集群。找到以 rds.amazonaws.com 结尾的写入器实例端点。	
创建端口转发。	若要创建端口转发，请使用以下步骤： 1. 安装 AWS Systems Manager 会话管理器插件。 2. 通过在 core/ 其中运行 pnpm db:connect 来启动端口转发，通过堡垒主机建立安全连接。 3. 在终端Waiting for connections..., 中看到文本后，您的本地计算机和 Aurora 服务器之间已成功通过 EC2 堡垒主机建立 SSH 隧道。	应用程序开发人员

Task	描述	所需技能
调整 Systems Manager 会话管理器超时。	（可选）如果默认 20 分钟的会话超时时间太短，则可以在 Systems Manager 控制台中选择 Session Manager、偏好、编辑、空闲会话超时，将其延长至 60 分钟。	应用程序开发人员

Task	描述	所需技能
可视化数据库。	pgadmin 是一款用户友好开源工具，用于管理 PostgreSQL 数据库。它简化了数据库任务，使您可高效地创建、管理和优化数据库。本节将指导您安装 pgadmin 并使用其功能进行 PostgreSQL 数据库管理。 1. 在对象资源管理器中，打开服务器 的上下文（右键单击）菜单，然后选择注册，服务器。 2. 在常规选项卡，在名称字段中输入 <appId>-<stageName>。 3. 要获取数据库密码，请打开 AWS Secrets Manager 控制台，选择描述为 CDK 为堆栈生成的密钥：-data<appId><stageName>，然后选择机密值卡。选择检索机密值，然后复制密钥值和密码密钥。 4. 在连接选项卡，在主机名/地址字段中输入 0.0.0，在用户名字段中输入 <appId>_admin。在密码字段，使用您之前获取的密码。选择是作为保存密码？ 字段。 5. 选择 Save(保存)。 6. 要查看表，请导航至 <appId>-<stageName>、数据库、<appId>_db、架构、<appId>、表格	应用程序开发人员

Task	描述	所需技能
	7. 打开项目表格的上下文（右键单击）菜单，然后选择查看/编辑数据，所有行。 8. 探索表格。	

使用 Node.js 调试

Task	描述	所需技能

调试创建项目用例。	若要调试创建项目用例，请执行以下步骤：
	1. 打开 `core/src/modules/item/create-item.use-case.ts` 文件 并插入以下代码。
	`import { fileURLToPath } from "node:url";`
	`// existing create-item.use-case.ts code here`
	`if (process.argv[1] === fileURLToPath(import.meta.url)) {`
	`  createItemUseCase({`
	`  description: "Item 1's Description",`
	`  name: "Item 1",`
	`  });`
	`}`
		应用程序开发人员

Task	描述	所需技能
	2. 上一步中添加的代码可确保在直接运行此模块时调用 `createItemUseCase` 函数。在此代码块中要启动 line-by-line调试的行上设置断点。 1. 打开 VS Code JavaScript 调试终端，然后运行`pnpm tsx core/src/modules/item/create-item.use-case.ts` 以运行带有 line-by-line调试功能的代码。或者，你可以使用`console.log` 语句，但是当你处理复杂的业务逻辑时，打印对账单可能不够用。 Line-by-line调试可以为你提供更多上下文。	

开发前端

Task	描述	所需技能
设置开发服务器。	1. 导航至 `ui/`，然后运行 `pnpm dev` 以启动 Next.js 开发服务器 2. 本地访问您的 Web 应用程序，网址为 `http://localhost:3000` 。Next.js 开发服务器设置了快速刷新增量，即时反馈对 React 组件所做的编辑。	应用程序开发人员

Task	描述	所需技能
	3. 尝试自定义应用栏颜色。打开 `ui/src/components/theme/theme.tsx` 文件，并找到定义应用栏主题的部分。在该 `colorSchemes.light.palette.primary` 部分，将主值从 `colors.lagoon` 更新为 `colors.carrot` 。进行该更改后，保存文件并在浏览器中观察更新。 4. 通过修改文本、组件和添加新页面进行实验。	

含绿色增强功能的工具

Task	描述	所需技能
设置 monorepo 与 pnpm 包管理器。	1. 在 GB 存储库的根目录中查看`pnpm-workspace.yaml` ，注意工作空间是如何定义的。有关工作空间的更多信息，请参阅 pnpm 文档。 2. 查看 `ui/package.json` ，并注意它如何使用包名称引 `core/` 用工作区 `"<appId>/core": "workspace:^",` 。 3. 观察内部定义的实用程序包中如何集中 ESLint 配置 TypeScript 和配置`packages/` 。然后，`core/`、`infra/` 和	应用程序开发人员

Task	描述	所需技能
	ui/ 等应用程序包将使用此配置。当您的应用程序扩缩并且定义更多应用程序包时，这非常有用，这些应用程序包可以引用实用程序包而无需重复配置代码。	
运行 pnpm 脚本。	在存储库的根目录中运行以下命令： 1. 运行 `pnpm lint`。此命令使用运行静态代码分析ESLint。 2. 运行 `pnpm typecheck` 。此命令运行编TypeScript 译器以检查代码的类型。 3. 运行 `pnpm test`。此命令运行 Vitest 以运行单元测试。 请注意这些命令如何在所有工作区中运行。这些命令是在每个工作空间的 `package.json#scripts` 字段中定义的。	应用程序开发人员

Task	描述	所需技能

ESLint 用于静态代码分析。	要测试的静态代码分析能力 ESLint，请执行以下操作：
	
	1. 首先，确保安装了 VS Code ESLint 扩展程序 (ID:dbaeumer.vscode-eslint)。我们还建议安装 VS Code Error Lens (ID：usernamehw.errorlens)以查看内联错误。
	2. 在您的代码中，有目的地包含一行使用 eval() 函数的代码，如以下示例中所示。
	
	``
	const userInput =
	"import("fs").then
	((fs) => console.l
	og(fs.readFileSync
	("/etc/passwd",
	{ encoding:
	"utf8" }))";
	eval(userInput);
	``
	
	Important
	这仅用于测试目的。使用 eval() 被认为具有潜在危险，由于存在安全风险，应避免使用。
	
	3. 添加该eval()行后，打开代码编辑器，使用红色波浪线确认 ESLint 表示代码有异味。
		应用程序开发人员

Task	描述	所需技能
	4. 查看 ESLint 插件和配置，网址为`packages/eslint-config-{node,next}/.eslintrc.cjs` 。	
管理依赖项和漏洞。	1. 要识别任何常见漏洞和漏洞 (CVEs)，请在存储库的根目录`pnpm audit`中运行。 您应该看到未发现已知漏洞。 2. 通过运行`pnpm add minimist@0.2.3` 在 `core/` 中安装一个故意存在漏洞的软件包，然后运行 `pnpm audit`。请注意所报告漏洞。 3. 通过运行 `pnpm remove minimist` 卸载`core/`中的易受攻击的软件包。	应用程序开发人员

Task	描述	所需技能
使用 Husky 预先提交挂钩。	1. 对整个存储库中的 TypeScript 文件进行几处小改动。这些更改可以像添加评论一样简单。 2. 然后可以通过 `git add -A` 和 `git commit -m "test husky"` 暂存和提交更改。 `.husky/pre-commit` 中定义的 Husky 预提交钩子触发器运行 `pnpm lint-staged` 命令。 3. 观察 lint-staged 是如何在 Git 暂存的文件上运行存储库中文件中 `*/.lintstagedrc.js` 指定的命令的。 这些工具是帮助防止不良代码进入您的应用程序的机制。	应用程序开发人员

拆除基础设施

Task	描述	所需技能
从您的账户中移除部署。	1. 要卸载您在第一个操作说明中预置的基础设施，请运行 `infra/` 中的 `pnpm destroy:local` 。 2. 完成 `pnpm destroy:local` 后等待 15 分钟，然后在 Lambda 控制台中搜	应用程序开发人员

Task	描述	所需技能
	索您的应用程序 ID，删除保留的 Lambda @Edge 函数。Lambda @Edge 函数已被复制，这使得它们难以删除。有关删除 Lambda @Edge 函数的更多信息，请参阅文档。CloudFront	

故障排除

事务	解决方案
无法建立端口转发	确保您的 AWS 凭证配置正确并具有必要的权限。 仔细检查堡垒主机 ID (`DB_BASTION_ID`) 与数据库端点 (`DB_ENDPOINT`) 环境变量是否设置正确。 如果您仍然遇到问题，请参阅 AWS 文档以排除 SSH 连接和会话管理器。
`localhost:3000` 网站未加载	确认终端输出显示端口转发成功，包含转发地址。 确保本地计算机上没有使用端口 3000 的冲突进程。 验证 Green Boost 应用程序是否已正确配置并在预期端口 (3000) 上运行。 检查您的 Web 浏览器是否有任何可能阻止本地连接的安全扩展或设置。
本地部署期间的错误消息 (`pnpm deploy:local`)	仔细查看错误消息，以确定问题的原因。

事务	解决方案
	验证必要的环境变量以及配置文件是否设置正确。

相关资源

- AWS CDK 文档
- Green Boost 文档
- Next.js 文档
- Node.js 文档
- React 文档
- TypeScript 文档

使用 AWS 对来自 GitHub 的 Node.js 应用程序运行单元测试 CodeBuild

创建者：Thomas Scott (AWS) 和 Jean-Baptiste Guillois (AWS)

摘要

此模式为 Node.js 游戏 API 提供了示例源代码和关键单元测试组件。它还包括使用 AWS 从 GitHub 存储库运行这些单元测试的说明 CodeBuild，这是持续集成和持续交付 (CI/CD) 工作流程的一部分。

单元测试是一个软件开发过程，在这个过程中，对应用程序的不同部分（称为单元）进行单独和独立的测试，以确保其正确运行。测试验证代码的质量并确认其功能是否符合预期。其他开发人员也可以通过查阅测试轻松熟悉您的代码库。单元测试可缩短未来的重构时间，帮助工程师更快地掌握代码库，并让他们对预期行为充满信心。

单元测试涉及测试单个函数，包括 AWS Lambda 函数。要创建单元测试，您需要一个测试框架和一种验证测试（断言）的方法。此模式中的代码示例使用 Mocha 测试框架和 Chai 断言库。

有关单元测试和测试组件示例的更多信息，请参阅其他信息部分。

先决条件和限制

- 具有正确 CodeBuild 权限的活跃 AWS 账户
- GitHub 账户（参见注册说明）
- Git（请参阅安装说明）
- 用于进行更改并将代码推送到的代码编辑器 GitHub

架构

此模式实施下图中所示的架构。

工具

工具

- Git 是一个版本控制系统，你可以用它来开发代码。
- AWS CodeBuild 是一项完全托管的持续集成服务，可编译源代码、运行测试并生成可随时部署的软件包。使用 CodeBuild，您无需预置、管理和扩展自己的构建服务器。 CodeBuild 持续扩展并同时

处理多个构建，因此您的构建不会在队列中等待。您可以使用预先打包的构建环境快速开始，也可以创建使用您自己的构建工具的自定义构建环境。使用 CodeBuild，按分钟计费所使用的计算资源。

代码

此模式的源代码可在 GitHub示例游戏单元测试应用程序存储库中找到。您可以根据此示例（选项 1）创建自己的 GitHub 存储库，也可以直接使用示例存储库（选项 2）来创建此模式。按照下一节中每个选项的说明操作。选项的选择因用例而定。

操作说明

选项 1-使用在您的个人 GitHub 存储库上运行单元测试 CodeBuild

Task	描述	所需技能
根据示例项目创建自己的 GitHub 存储库。	1. 登录到 GitHub。 2. 创建新存储库。有关说明，请参阅GitHub 文档。 3. 克隆示例存储库并将其推送到您账户中的新存储库中。	应用程序开发人员、AWS 管理员、AWS DevOps
创建新 CodeBuild 项目。	1. 登录 AWS 管理控制台，然后在 https://console.aws.amazon.com/codesuite/codebuild /h CodeBuild ome 中打开控制台。 2. 选择 Create build project（创建构建项目）。 3. 在“项目配置”部分，在“项目名称”中键入 aws-tests-sample-node-js。 4. 在“来源”部分中，对于源提供商，选择GitHub。 5. 对于“存储库”，在我的 GitHub 账户中选择“存储库”，然后将 URL 粘贴到新创建的 GitHub 存储库中。	应用程序开发人员、AWS 管理员、AWS DevOps

Task	描述	所需技能
	6. 在主要源 Webhook 事件中，选择每次将代码更改推送到此存储库时都会重新生成。 7. 对于事件类型，选择推送。 8. 在 “环境” 部分中，选择托管映像、Amazon Linux 和最新映像。 9. 所有其他选项保留默认设置，然后选择创建构建项目。	
开始构建。	在审核页面上，选择开始构建以运行构建。	应用程序开发人员、AWS 管理员、AWS DevOps

选项 2-使用以下命令在公共存储库上运行单元测试 CodeBuild

Task	描述	所需技能
创建新的 CodeBuild 构建项目。	1. 登录 AWS 管理控制台，然后在 https://console.aws.amazon.com/codesuite/codebuild /h CodeBuild ome 中打开控制台。 2. 选择 Create build project（创建构建项目）。 3. 在 “项目配置” 部分，在 “项目名称” 中键入 aws-tests-sample-node-js。 4. 在 “来源” 部分中，对于源提供商，选择GitHub。 5. 对于 “存储库”，选择 “公共存储库”，然后粘贴 URL:	应用程序开发人员、AWS 管理员、AWS DevOps

Task	描述	所需技能
	https://github.com/aws-samples/node-js-tests-sample。 6. 在“环境”部分中，选择托管映像、Amazon Linux 和最新映像。 7. 所有其他选项保留默认设置，然后选择创建构建项目。	
开始构建。	在审核页面上，选择开始构建以运行构建。	应用程序开发人员、AWS 管理员、AWS DevOps

分析单元测试

Task	描述	所需技能
查看测试结果。	在 CodeBuild 控制台中，查看 CodeBuild 作业的单元测试结果。这些结果应与其他信息部分中显示的结果相符。 这些结果验证了 GitHub 存储库与的集成 CodeBuild。	应用程序开发人员、AWS 管理员、AWS DevOps
应用 Webhook。	现在，您可以应用 Webhook，这样无论何时将代码变更推送到存储库的主分支，都可以自动启动构建。有关说明，请参阅CodeBuild 文档。	应用程序开发人员、AWS 管理员、AWS DevOps

相关资源

- 示例游戏单元测试应用程序（包含示例代码的GitHub 存储库）

- AWS CodeBuild 文档
- GitHub webhook 事件（CodeBuild 文档）
- 创建新存储库（GitHub 文档）

其他信息

单元测试结果

在 CodeBuild 控制台中，项目成功生成后，您应该会看到以下测试结果。

单元测试组件示例

本节介绍单元测试中使用的四种类型的测试组件：断言、间谍、存根和模拟。它包括每个组件的简要说明和代码示例。

断言

断言用于验证预期结果。这是一个重要的测试组件，因为它可以验证给定函数的预期响应。以下示例断言验证了初始化新游戏时返回的 ID 是否介于 0 和 1000 之间。

```
const { expect }  = require('chai');
const { Game } = require('../src/index');

describe('Game Function Group', () => {
 it('Check that the Game ID is between 0 and 1000', function() {
      const game = new Game();
      expect(game.id).is.above(0).but.below(1000)
 });
});
```

间谍

间谍用于观察函数运行时发生的情况。例如，您可能希望验证函数的调用是否正确。以下示例显示了在游戏类对象上调用启动和停止方法。

```
const { expect }  = require('chai');
const { spy } = require('sinon');

const { Game } = require('../src/index');
```

```
describe('Game Function Group', () => {
   it('should verify that the correct function is called', () => {
      const spyStart = spy(Game.prototype, "start");
      const spyStop = spy(Game.prototype, "stop");

      const game = new Game();
      game.start();
      game.stop();

      expect(spyStart.called).to.be.true
      expect(spyStop.called).to.be.true
    });
});
```

存根

存根用于覆盖函数的默认响应。当函数发出外部请求时，这特别有用，因为您想避免从单元测试中发出外部请求。（外部请求更适合集成测试，集成测试可以对不同组件之间的请求进行物理测试。）在以下示例中，存根强制从 getId 函数返回一个 ID。

```
const { expect }  = require('chai');
const {.stub } = require('sinon');

const { Game } = require('../src/index');

describe('Game Function Group', () =>  {
   it('Check that the Game ID is between 0 and 1000', function() {
      let generateIdStub = stub(Game.prototype, 'getId').returns(999999);

      const game = new Game();

      expect(game.getId).is.equal(999999);

      generateIdStub.restore();
    });
});
```

模拟

模拟是一种虚假的方法，具有用于测试不同场景的预编程行为。模拟可以被视为存根的扩展形式，可以同时执行多个任务。在以下示例中，使用模拟来验证三个场景：

- 调用函数
- 使用参数调用函数
- 函数返回整数 9

```
const { expect }  = require('chai');
const {.mock } = require('sinon');

const { Game } = require('../src/index');

describe('Game Function Group', () =>  {
   it('Check that the Game ID is between 0 and 1000', function() {
      let mock = mock(Game.prototype).expects('getId').withArgs().returns(9);

      const game = new Game();
      const id = get.getId();

      mock.verify();
      expect(id).is.equal(9);
    });
});
```

使用 AWS Lambda 以六边形架构构建 Python 项目

创建者：Furkan Oruc (AWS)、Dominik Goby (AWS)、Darius Kunce (AWS) 和 Michal Ploski (AWS)

摘要

此模式展示了如何使用 AWS Lambda 以六边形架构构建 Python 项目。该模式使用 AWS Cloud Development Kit (AWS CDK) 作为基础设施即代码（IaC）工具，使用 Amazon API Gateway 作为 REST API，使用 Amazon DynamoDB 作为持久层。六边形架构遵循域驱动设计原则。在六边形架构中，软件由三个组件组成：域、端口和适配器。有关六边形架构及其优势的详细信息，请参阅指南在 AWS 上构建六边形架构。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- Python 经验
- 熟悉 AWS Lambda、AWS CDK、Amazon API Gateway 和 DynamoDB
- GitHub 账户（参见注册说明）
- Git（请参阅安装说明）
- 一种代码编辑器，用于进行更改并将代码推送到 GitHub （例如，Visual Studio Code 或 JetBrains PyCharm）
- 安装了 Docker，Docker 进程守护程序启动并正在运行

产品版本

- Git 版本 2.24.3 或更高版本
- Python 版本 3.7 或更高版本
- AWS CDK v2
- EMR 版本 1.1.13 或更高版本
- AWS Lambda Powertools for Python 版本 1.25.6 或更高版本
- pytest 版本 7.1.1 或更高版本
- Moto 版本 3.1.9 或更高版本
- pydantic 版本 1.9.0 或更高版本

- Boto3 版本 1.22.4 或更高版本
- mypy-boto3-dynamodb 版本 1.24.0 或更高版本

架构

目标技术堆栈

目标技术堆栈由使用 API Gateway、Lambda 和 DynamoDB 的 Python 服务构成。该服务使用 DynamoDB 适配器保存数据。它提供了使用 Lambda 作为入口点的函数。该服务使用 Amazon API Gateway 公开 REST API。API 使用 AWS Identity and Access Management (IAM) 对客户端执行身份验证。

目标架构

为说明实现方式，此模式部署了无服务器目标架构。客户端可向 API Gateway 端点发送请求。API Gateway 将请求转发至实现六边形架构模式的目标 Lambda 函数。Lambda 函数可对 DynamoDB 表执行创建、读取、更新和删除 (CRUD) 操作。

> **Important**
>
> 此模式已在 PoC 环境中进行了测试。在将任何架构部署至生产环境之前，您必须进行安全审查以识别威胁模型并创建安全的代码库。

该 API 支持对产品实体的五种操作：

- `GET /products` 返回所有产品。
- `POST /products` 创建新产品。
- `GET /products/{id}` 返回特定产品。
- `PUT /products/{id}` 更新特定产品。
- `DELETE /products/{id}` 删除特定产品。

您可使用以下文件夹结构来组织项目，以遵循六边形架构模式：

```
app/  # application code
```

```
|--- adapters/  # implementation of the ports defined in the domain
     |--- tests/  # adapter unit tests
|--- entrypoints/  # primary adapters, entry points
     |--- api/  # api entry point
          |--- model/  # api model
          |--- tests/  # end to end api tests
|--- domain/  # domain to implement business logic using hexagonal architecture
     |--- command_handlers/  # handlers used to execute commands on the domain
     |--- commands/  # commands on the domain
     |--- events/  # events triggered via the domain
     |--- exceptions/  # exceptions defined on the domain
     |--- model/  # domain model
     |--- ports/  # abstractions used for external communication
     |--- tests/  # domain tests
|--- libraries/  # List of 3rd party libraries used by the Lambda function
infra/  # infrastructure code
simple-crud-app.py  # AWS CDK v2 app
```

工具

Amazon Web Services

- Amazon API Gateway 是一项完全托管的服务，可让开发人员轻松创建、发布、维护、监控和保护 APIs 任何规模。
- Amazon DynamoDB 是一个完全托管的无服务器键值的 NoSQL 数据库，专为运行任何规模的高性能应用程序而设计。
- AWS Lambda 是一项无服务器、事件驱动计算服务，让您能够为几乎任何类型的应用程序或后端服务运行代码，而无需预调配或管理服务器。您可以从 200 多种 Amazon Web Services 和软件即服务（SaaS）应用程序启动 Lambda 函数，并且只需为您使用的部分付费。

工具

- Git 用作此模式中代码开发的版本控制系统。
- Python 用作此模式的编程语言。Python 提供高级数据结构和面向对象编程方法。AWS Lambda 提供内置的 Python 运行时系统，可简化 Python 服务的操作。
- Visual Studio Code 用作开发和测试此模式的 IDE。您可以使用任何支持 Python 开发的 IDE（例如 PyCharm）。
- AWS Cloud Development Kit (AWS CDK) 是开源软件开发框架，它允许您使用熟悉的编程语言定义云应用程序资源。此模式使用 CDK 以代码形式编写和部署云基础设施。

- Poetry 用于管理模式中的依赖项。
- AWS CDK 使用 Docker 构建 Lambda 包和层。

代码

此模式的代码可在 GitHub Lambda 六边形架构示例存储库中找到。

最佳实践

要在生产环境使用此模式，请遵循以下最佳实践：

- 使用 AWS 密钥管理服务 (AWS KMS) 中的客户托管密钥对亚马逊 CloudWatch 日志组和亚马逊 DynamoDB 表进行加密。
- 为 Amazon API Gateway 配置 AWS WAF，以允许仅从贵组织的网络进行访问。
- 如果 IAM 不能满足您的需求，可考虑其他的 API Gateway 授权选项。例如，您可使用 Amazon Cognito 用户池或 API Gateway Lambda 授权者。
- 使用 DynamoDB 备份。
- 使用虚拟私有云（VPC）部署来配置 Lambda 函数，将网络流量保持在云内。
- 为跨源资源共享 (CORS) 预检更新允许的源配置，以限制仅访问请求的源域。
- 使用 cdk-nag 检查 AWS CDK 代码，了解安全最佳实践。
- 考虑使用代码扫描工具来查找代码中常见的安全问题。例如，Bandit 是旨在查找 Python 代码中常见安全问题的工具。Pip-audit 将扫描 Python 环境中是否存在已知漏洞的程序包。

此模式使用 AWS X-Ray 通过应用程序的入口点、域和适配器跟踪请求。AWS X-Ray 帮助开发人员识别瓶颈并确定高延迟以提高应用程序性能。

操作说明

初始化项目

Task	描述	所需技能
创建您自己的存储库。	1. 登录到 GitHub。 2. 创建新存储库。有关说明，请参阅GitHub 文档。	应用程序开发人员

Task	描述	所需技能
	3. 克隆此模式的示例存储库并将其推送到您账户中的新存储库中。	

安装依赖项。	1. 安装 Poetry。
	`pip install poetry`
	2. 从根目录安装程序包。以下命令安装应用程序和 AWS CDK 包。它还安装运行单元测试所需开发包。所有已安装的程序包都放置在新的虚拟环境中。
	`poetry install`
	3. 要查看已安装程序包的图形表示，请运行以下命令。
	`poetry show --tree`
	4. 更新所有依赖项。
	`poetry update`
	5. 在新创建的虚拟环境中打开新 Shell。它包含所有已安装的依赖项。
	`poetry shell`
		应用程序开发人员

Task	描述	所需技能
配置您的 IDE。	我们推荐 Visual Studio Code，但您可以使用您选择的任何支持 Python 的 IDE。以下步骤适用于 Visual Studio Code。 1. 更新 `.vscode/settings` 文件。 `{ "python.testing.pytestArgs": [ "app/adapters/tests", "app/entrypoints/api/tests", "app/domain/tests" ], "python.testing.unittestEnabled": false, "python.testing.pytestEnabled": true, "python.envFile": "${workspaceFolder}/.env", }` 2. 在项目的根目录中创建 `.env` 文件。这样可以确保项目的根目录包含在 `PYTHONPATH` 中，以便 `pytest` 可以找到它并正确发现所有程序包。	应用程序开发人员

Task	描述	所需技能
	`PYTHONPATH=.`	
运行单元测试，选项 1：使用 Visual Studio Code。	1. 选择由 Poetry 管理虚拟环境的 Python 解释器。 2. 从测试资源管理器中运行测试。	应用程序开发人员

运行单元测试，选项 2：使用 Shell 命令。	1. 在虚拟环境中启动一个新 Shell。
	`poetry shell`
	2. 从根目录运行 `pytest` 命令。
	`python -m pytest`
	或者，您可以直接从 Poetry 中运行该命令。
	`poetry run python -m pytest`
		应用程序开发人员

部署和测试应用程序

Task	描述	所需技能
请求临时凭证。	要在运行 `cdk deploy` 时在 Shell 中使用 AWS 凭证，请使用 AWS IAM Identity Center（AWS Single Sign-On 的后继任务）创建临时凭证。有关说明，请参阅博客文章如	AWS 应用程序开发人员 DevOps

Task	描述	所需技能
	何检索短期凭证，以便在 AWS IAM Identity Center 使用 CLI。	

部署 应用程序。	1. 安装 AWS CDK v2。
	`npm install -g aws-cdk`
	有关更多信息，请参阅 AWS CDK 文档。
	2. 将 AWS CDK 引导到您的账户和区域。
	`cdk bootstrap aws://12345678900/ us-east-1 --profile aws-profile-name`
	3. 使用 AWS 配置文件将应用程序部署为 AWS CloudFormation 堆栈。
	`cdk deploy --profile aws-profile-name`
		AWS 应用程序开发人员 DevOps

测试 API，选项 1：使用控制台。	使用 API Gateway 控制台测试 API。有关 API 操作和请求/响应消息的更多信息，请参阅存储库中自述文件的 API 用法部分。 GitHub	AWS 应用程序开发人员 DevOps

Task	描述	所需技能
测试 API，选项 2：使用 Postman。	如果您要使用 Postman 这样的工具，请执行以下操作： 1. 安装 Postman 作为独立应用程序或浏览器扩展程序。 2. 复制 API Gateway 的端点 URL。它将采用以下格式。 `https://{api-id}.execute-api.{region}.amazonaws.com/{stage}/{path}` 3. 在授权选项卡中配置 AWS 签名。有关说明，请参阅 AWS re: Post 中关于激活 API Gateway REST 的 IAM 身份验证的文章。 APIs 4. 使用 Postman 向 API 端点发送请求。	AWS 应用程序开发人员 DevOps

开发服务

Task	描述	所需技能
为业务域编写单元测试。	1. 通过使用 `test_` 文件名前缀在 `app/domain/tests` 文件夹中创建 Python 文件。 2. 通过使用以下示例创建新的测试方法，以测试新的业务逻辑。 `def test_create_product_should_`	应用程序开发人员

Task	描述	所需技能

	store_in_repository():
	    # Arrange
	    command = create_product_command.CreateProductCommand(
	        name="Test Product",
	        description="Test Description",
	    )
	    # Act
	    create_product_command_handler.handle_create_product_command(
	        command=command, unit_of_work=mock_unit_of_work
	    )
	# Assert
	3. 在 app/domain/commands 文件夹中创建命令类。
	4. 如果该功能是新增的，请在 app/domain/command_handlers 文件夹中为命令处理程序创建一个存根。
	5. 运行单元测试以查看其失败问题，因为仍然没有业务逻辑。
	python -m pytest	

Task	描述	所需技能

实施命令和命令处理程序。	1. 在新创建的命令处理程序文件中实施业务逻辑。
	2. 对于每个与外部系统交互的依赖项，请在 app/domain/ports 文件夹中声明一个抽象类。
	
	`class ProductsRepository(ABC):`
	`  @abstractmethod`
	`  def add(self, product: product.Product) -> None:`
	`  ...`
	
	`class UnitOfWork(ABC):`
	`  products: ProductsRepository`
	
	`  @abstractmethod`
	`  def commit(self) -> None:`
	`  ...`
	
	`  @abstractmethod`
	`  def __enter__(self) -> typing.Any:`
	`  ...`
	
	`  @abstractmethod`
	`  def __exit__(self, *args) -> None:`
	`  ...`
	
	3. 使用抽象端口类作为类型注释，更新命令处理程序签名以接受新声明的依赖项。
		应用程序开发人员

Task	描述	所需技能

	<pre>def handle_cr
	eate_product_comma
	nd(
	    command: create_pr
	oduct_command.Crea
	teProductCommand,
	    unit_of_work:
	 unit_of_work.UnitO
	fWork,
	) -> str:
	    ...</pre>
	4. 更新单元测试以模拟命令处理程序的所有声明的依赖项的行为。
	<pre> # Arrange
	    mock_unit_of_work
	 = unittest.mock.crea
	te_autospec(
	        spec=unit
	_of_work.UnitOfWor
	k, instance=True
	    )
	    mock_unit
	_of_work.products =
	 unittest.mock.crea
	te_autospec(
	        spec=unit
	_of_work.ProductsR
	epository, instance=
	True
	    )</pre>
	5. 更新测试中的断言逻辑以检查是否有预期的依赖项调用。
	<pre> # Assert</pre>	

Task	描述	所需技能

	<pre> mock_unit
	_of_work.commit.as
	sert_called_once()
	    product =
	 mock_unit_of_work.
	products.add.call_
	args.args[0]
	
	    assertpy.
	assert_that(produc
	t.name).is_equal_t
	o("Test Product")
	    assertpy.
	assert_that(produc
	t.description).is_
	equal_to("Test
	 Description")</pre>	

	6. 运行单元测试，以查看它是否成功。 <pre>python -m pytest</pre>	

Task	描述	所需技能
为辅助适配器编写集成测试。	1. 通过使用 `test_` 作为文件名前缀在 `app/adapters/tests` 文件夹中创建测试文件。 2. 使用 Moto 库模拟 Amazon Web Services。 3. 为适配器的集成测试创建新测试方法。	应用程序开发人员

```
@pytest.fixture
def mock_dynamodb():
   with moto.mock_dynamodb():
    yield boto3.resource("dynamodb", region_name="eu-central-1")
```

```
def test_add_and_commit_should_store_product(mock_dynamodb):
    # Arrange
    unit_of_work = dynamodb_unit_of_work.DynamoDBUnitOfWork(
        table_name=TEST_TABLE_NAME, dynamodb_client=mock_dynamodb.meta.client
    )
    current_time = datetime.datetime.now(datetime.timezone.utc).isoformat()
```

Task	描述	所需技能

	<pre> new_product_id = str(uuid.uuid4())
	    new_product = product.Product(
	        id=new_product_id,
	        name="test-name",
	        description="test-description",
	        createDate=current_time,
	        lastUpdateDate=current_time,
	    )
	
	    # Act
	    with unit_of_work:
	        unit_of_work.products.add(new_product)
	        unit_of_work.commit()
	
	    # Assert</pre> 4. 在 app/adapters 文件夹中创建适配器类。使用 ports 文件夹中的抽象类作为基础类。
	5. 运行单元测试，看看它是否失败，因为仍然没有逻辑。
	<pre>python -m pytest</pre>	

Task	描述	所需技能

实施辅助适配器。	1. 在新创建的适配器文件中实施逻辑。
	2. 更新测试断言。
	
	`# Assert`
	`    with unit_of_work_readonly:`
	`        product_from_db = unit_of_work_readonly.products.get(new_product_id)`
	
	`    assertpy.assert_that(product_from_db).is_not_none()`
	`    assertpy.assert_that(product_from_db.dict()).is_equal_to(`
	`        {`
	`            "id": new_product_id,`
	`            "name": "test-name",`
	`            "description": "test-description",`
	`            "createDate": current_time,`
	`            "lastUpdateDate": current_time,`
	`        }`
	`    )`
	
	3. 运行单元测试，以查看它是否成功。
		应用程序开发人员

Task	描述	所需技能
	`python -m pytest`	

Task	描述	所需技能
写 end-to-end测试。	1. 通过使用 `test_` 作为文件名前缀在 `app/entrypoints/api/tests` 文件夹中创建测试文件。 2. 创建 Lambda 上下文固定装置，测试将使用该装置来调用 Lambda。 3. 为 API 调用创建测试方法。	应用程序开发人员

```
@pytest.fixture
def lambda_co
ntext():
    @dataclass
    class LambdaCon
text:
        function_
name: str = "test"
        memory_li
mit_in_mb: int = 128
        invoked_f
unction_arn: str =
 "arn:aws:lambda:eu
-west-1:809313241:
function:test"
        aws_reque
st_id: str =
 "52fdfc07-2182-154
f-163f-5f0f9a621d7
2"

 return LambdaCon
text()
```

```
def test_crea
te_product(lambda_
context):
    # Arrange
    name = "TestName"
```

Task	描述	所需技能

	`description = "Test description" request = api_model.CreateProductRequest(name=name, description=description)`
	
	`minimal_event = api_gateway_proxy_event.APIGatewayProxyEvent( { "path": "/products", "httpMethod": "POST", "requestContext": { # correlation ID "requestId": "c6af9ac6-7b61-11e6-9a41-93e8deadbeef" }, "body": json.dumps(request.dict()), } )`
	
	`create_product_func_mock = unittest.mock.create_autospec( spec=create_product_command_handler.handle_create_product_command )`	

Task	描述	所需技能

	<pre> handler.c
	reate_product_comm
	and_handler.handle
	_create_product_co
	mmand = (
	        create_pr
	oduct_func_mock
	    )
	
	    # Act
	    handler.h
	andler(minimal_eve
	nt, lambda_context)</pre>
	4. 运行单元测试，看看它是否失败，因为仍然没有逻辑。
	<pre>python -m pytest</pre>	

Task	描述	所需技能

实施主适配器。	1. 为 API 业务逻辑创建函数，并将其声明为 API 资源。
	
	`@tracer.capture_method`
	`@app.post("/products")`
	`@utils.parse_event(model=api_model.CreateProductRequest, app_context=app)`
	`def create_product(`
	`request: api_model.CreateProductRequest,`
	`) -> api_model.CreateProductResponse:`
	`"""Creates a product."""`
	`...`
	
	Note
	你看到的所有装饰器都是 AWS Lambda Powertools for Python 库的功能。有关详细信息，请访问 AWS Lambda Powertools for Python 网站。
	
	2. 实施 API 逻辑。
	
	`id=create_product_command_handler.ha`
		应用程序开发人员

Task	描述	所需技能
	ndle_create_product_command(command=create_product_command.CreateProductCommand(name=request.name, description=request.description,), unit_of_work=unit_of_work,) response = api_model.CreateProductResponse(id=id) return response.dict() 3. 运行单元测试，以查看它是否成功。 python -m pytest	

相关资源

APG 指南

- 在 AWS 上构建六边形架构

AWS 参考

- AWS Lambda 文档
- AWS CDK 文档
  - 您的第一个 AWS CDK 应用程序

- API Gateway 文档
  - 使用 IAM 权限控制对 API 的访问
  - 使用 API Gateway 控制台测试 REST API 方法
- Amazon DynamoDB 文档

工具

- git-scm.com 网站
- 安装 Git
- 创建新 GitHub 存储库
- Python 网站
- AWS Lambda Powertools for Python
- Postman 网站
- Python 模拟对象库
- Poetry 网站

IDEs

- Visual Studio Code 网站
- PyCharm 网站

更多模式

- 使用 Amazon Cognito 身份池 AWS 服务 从 ASP.NET 核心应用程序进行访问
- 使用 AWS Fargate PrivateLink、AWS 和网络负载均衡器在 Amazon ECS 上私下访问容器应用程序
- 使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon ECS 上私密访问容器应用程序
- 将一个 AWS CodeCommit 存储库 AWS 账户 与另一个账户中的 Amazon A SageMaker I Studio Classic 关联起来
- 自动删除 AWS CloudFormation 堆栈和相关资源
- 使用，根据 IP 地址或地理位置限制访问 AWS WAF
- 使用 DevOps 实践和 AWS Cloud9 构建具有微服务的松散耦合架构
- 使用 AWS Amplify 构建无服务器 React Native 移动应用程序
- 使用 AWS CodeCommit、AWS 和 AWS Device Farm 构建和测试 iOS 应用程序 CodePipeline
- 使用在 Amazon Logs 中为.NET 应用程序配置 CloudWatch 日志记录 NLog
- 使用和 P HashiCorp acker 创建管道 CodePipeline 和 AMI
- 使用创建管道并将项目更新部署到本地 EC2 实例 CodePipeline
- 使用 Amazon EFS 创建 Amazon ECS 任务定义并在 EC2 实例上挂载文件系统
- 在 Amazon EKS 集群上部署基于 gRPC 的应用程序并使用应用程序负载均衡器访问它
- 部署 ChatOps 解决方案来管理 SAST 扫描结果，方法是在聊天应用程序中使用 Amazon Q Developer 自定义操作和 AWS CloudFormation
- 使用 Terraf CloudWatch orm 部署 Synthetics 加那利群岛
- 使用 AWS Fargate 在 Amazon ECS 上部署 Java 微服务
- 使用 Terraform 和 Amazon AWS Bedrock 部署 RAG 用例
- 使用 Terraform 在 AWS Wavelength 区域中部署资源
- 在 Amazon API Gateway 中使用自定义域名实现基于路径的 API 版本控制
- 将消息队列从 Microsoft Azure 服务总线迁移到 Amazon SQS
- 将 .NET 应用程序从 Microsoft Azure 应用服务迁移到 AWS Elastic Beanstalk
- 使用二进制方法将本地 Go Web 应用程序迁移至 AWS Elastic Beanstalk
- 将本地 SFTP 服务器迁移到使用 AWSAWS Transfer for SFTP
- 在 Amazon 上从 IBM WebSphere 应用程序服务器迁移到 Apache Tomcat EC2
- 使用 Auto Scaling 从 IBM WebSphere 应用程序服务器迁移到亚马逊 EC2 上的 Apache Tomcat
- 从 Oracle 迁移 GlassFish 到 AWS Elastic Beanstalk

- 使用 AWS App2Container 将本地 Java 应用程序迁移到 AWS
- 将 OpenText TeamSite 工作负载迁移到 AWS 云
- 使用 ACM 将 Windows SSL 凭证迁移到应用程序负载均衡器
- 在 AWS 上实现 ASP.NET Web 表单应用程序的现代化
- 在亚马 EC2 逊 Linux 实例上运行 ASP.NET Core Web API Docker 容器
- 使用亚马逊通过 VPC 在 Amazon S3 存储桶中提供静态内容 CloudFront
- 在 AWS 上设置高度可用的 PeopleSoft 架构
- 使用自动化工作流程简化 Amazon Lex 机器人的开发和部署
- 使用 Amazon Bedroc AWS Step Functions k 对中的状态进行故障排除
- 使用 Network Firewall 从服务器名称指示中捕获出站流量的 DNS 域名
- 使用 Flask 和 AWS Elastic Beanstalk 查看人工智能/机器学习 (AI/ML) 模型结果

IoT

主题

- 为 AWS IoT 环境中的安全事件配置日志记录和监控
- 提取和查询 AWS IoT SiteWise 数据湖中的元数据属性
- 使用客户端设备设置 Amazon IoT Greengrass 并对其故障排除
- 更多模式

为 AWS IoT 环境中的安全事件配置日志记录和监控

创建者：Prateek Prakash (AWS)

摘要

确保物联网（IoT）环境的安全是当务之急，尤其是因为组织正在将数十亿台设备连接到其 IT 环境。此模式提供了一个参考架构，您可以使用该架构对整个物联网环境中的安全事件进行日志记录和监控 AWS Cloud。通常，上的物联网环境包含以下三层： AWS Cloud

- 生成相关遥测数据的 IoT 设备。
- AWS IoT 将您的物联网设备连接到其他设备的服务（例如AWS IoT Device Management、或 AWS IoT Device Defender）以及 AWS 服务。AWS IoT Core
- 后端 AWS 服务 可帮助处理遥测数据，并为不同的业务用例提供有用的见解。

AWS IoT Lens-AWS WellArchitected Framework 白皮书提供的最佳实践可以帮助您审查和改进基于云的架构，并更好地了解您的设计决策对业务的影响。一项重要的建议是，您应分析设备上和中的应用程序日志和指标 AWS Cloud。您可以通过利用不同的方法和技术（例如威胁建模）来识别为检测潜在安全问题而必须监控的指标和事件，从而实现这一目标。

此模式描述了如何使用 AWS IoT 安全服务为上的 IoT 环境设计和实现安全日志和监控参考架构 AWS Cloud。该架构建立在现有 AWS 安全最佳实践的基础上，并将其应用于您的物联网环境。

先决条件和限制

先决条件

- 现有的登录区环境。有关这方面的更多信息，请参阅《 AWS 规范指南》网站上的 “设置安全且可扩展的多账户 AWS 环境” 指南。
- 以下账户在登录区中必须可用：
  - 日志存档帐户-此帐户适用于需要访问着陆区域组织单位中帐户的日志信息的用户（OUs）。有关这方面的更多信息，请参阅《 AWS 规范性指南》网站上《安全参考架构》指南中的 “AWS 安全 OU — 日志存档账户” 部分。
  - 安全账户 – 安全和合规团队使用此账户进行审计或执行紧急安全操作。此账户也被指定为 Amazon 的管理员账户 GuardDuty。管理员账户中的用户除了可以查看和管理自己的账户和所有成员账户的 GuardDuty 发现结果外，还可以进行配置 GuardDuty。有关这方面的更多信息，请参阅 GuardDuty 文档 GuardDuty中的管理多个账户。

- IoT 账户 – 此账户适用于 IoT 环境。

架构

这种模式扩展了解决方案库中的集中日志解决方案，AWS 以收集和处理与安全相关的物联网事件。集中日志解决方案部署在安全账户中，有助于在单个控制面板中收集、分析和显示 Amazon CloudWatch 日志。该解决方案整合、管理和分析来自多个来源的日志文件。最后，集中式日志解决方案还使用 Amazon OpenSearch 服务和OpenSearch 控制面板来显示所有日志事件的统一视图。

以下架构图显示了物联网安全日志和参考架构的关键组件 AWS Cloud。

图表显示了以下工作流：

1. IoT 是必须监控异常安全事件的设备。这些设备运行代理向 AWS IoT Core 和发布安全事件或指标 AWS IoT Device Defender。
2. 启用 AWS IoT 日志记录后，当每条消息从您的设备通过消息代理和规则引擎传递到 Amazon Logs 时，会将有关每条消息的进度事件 AWS IoT 发送到 Amazon CloudWatch Logs。您可以使用 CloudWatch 日志订阅将事件推送到集中式日志解决方案。有关这方面的更多信息，请参阅 AWS IoT Core 文档中的AWS IoT 指标和维度。
3. AWS IoT Device Defender 帮助监控物联网设备的不安全配置和安全指标。检测到异常情况时，警报会通知具有订阅功能的亚马逊简单通知服务 (Amazon SNS) Simple Notification Service AWS Lambda 。Lambda 函数将警报作为消息发送到 CloudWatch 日志。您可以使用 CloudWatch 日志订阅将事件推送到您的集中式日志记录解决方案。有关这方面的更多信息，请参阅文档中的审核检查、将设备端日志上传到 CloudWatch和配置 AWS IoT 日志记录。 AWS IoT Core
4. AWS CloudTrail 记录进行更改的 AWS IoT Core 控制平面操作（例如，创建、更新或附加 APIs）。当作为 CloudTrail landing zone 实现的一部分进行设置时，它会将事件发送到 CloudWatch Logs。您可以使用订阅将事件推送到您的集中式日志记录解决方案。
5. AWS Config 托管规则或自定义规则评估属于您的物联网环境的资源。使用带 CloudWatch 日志 CloudWatch 的事件作为目标，监控您的合规性变更通知。将合规性变更通知发送到 CloudWatch 日志后，您可以使用订阅将事件推送到您的集中日志记录解决方案。
6. Amazon GuardDuty 持续分析 CloudTrail 管理事件，并帮助识别从已知的恶意 IP 地址、异常地理位置或匿名代理向 AWS IoT Core 终端节点发出的 API 调用。使用以日志中的 CloudWatch 日志组为目标 CloudWatch 的事件来监控 GuardDuty 通知。当 GuardDuty 通知发送到 CloudWatch 日志时，您可以使用订阅将事件推送到您的集中式监控解决方案，或者使用安全账户中的 GuardDuty 控制台查看通知。

7. AWS Security Hub 使用安全最佳实践监控您的 IoT 帐户。使用带有日志中 CloudWatch 日志组 CloudWatch 的事件作为目标来监控 Security Hub 通知。当 Security Hub 通知发送到 CloudWatch 日志时，使用订阅将事件推送到您的集中式监控解决方案，或者使用安全帐户中的 Security Hub 控制台查看通知。
8. Amazon Detective 会评估和分析信息，找出根本原因，并根据针对物联网架构中 AWS IoT 终端节点或其他服务的异常调用的安全发现采取措施。
9. Amazon Athena 会查询存储在日志存档账户中的日志，以增强您对安全调查发现的理解，并识别趋势和恶意活动。

工具

- Amazon Athena 是一种交互式查询服务，让您能够轻松使用标准 SQL 直接分析 Amazon Simple Storage Service（Amazon S3）中的数据。
- AWS CloudTrail帮助您实现治理、合规以及运营和风险审计 AWS 账户。
- Amazon 会实时 CloudWatch监控您的 AWS 资源和您运行 AWS 的应用程序。 您可以使用 CloudWatch 来收集和跟踪指标，这些指标是您可以衡量资源和应用程序的变量。
- Amazon CloudWatch Logs 集中了您所有系统、应用程序和您 AWS 服务 使用的日志。您可以查看和监控日志，搜索特定的错误代码或模式，根据特定字段过滤日志，或将日志安全存档以备将来分析。
- AWS Config提供了中 AWS 资源配置的详细视图 AWS 账户。
- Amazon Detective 使您轻松分析、调查和快速识别安全调查发现或可疑活动的根本原因。
- AWS Glue是一项完全托管的提取、转换和加载 (ETL) 服务，它可以简单且经济高效地对数据进行分类、清理、丰富数据，并在各种数据存储和数据流之间可靠地移动数据。
- Amazon GuardDuty 是一项持续的安全监控服务。
- AWS IoT Core为连接互联网的设备（例如传感器、执行器、嵌入式设备、无线设备和智能设备）提供安全的双向通信，以通过 MQTT、HTTPS 和 W AWS Cloud AN 连接到网络。 LoRa
- AWS IoT Device Defender 是一项安全服务，您可以借助此服务审核设备的配置、监控互联设备以检测异常行为，并降低安全风险。
- Amazon OpenSearch Service 是一项托管服务，可让您在中轻松部署、操作和扩展 OpenSearch 集群 AWS Cloud。
- AWS Organizations是一项账户管理服务，可让您将多个账户整合 AWS 账户 到一个由您创建和集中管理的组织中。

- AWS Security Hub全面了解您的安全状态， AWS 并帮助您根据安全行业标准和最佳实践检查您的环境。
- Amazon Virtual Private Cloud（Amazon VPC）提供了一个逻辑上隔离的部分，您可以在 AWS Cloud 其中启动您定义的虚拟网络中的 AWS 资源。这个虚拟网络与您在数据中心中运行的传统网络极其相似，并会为您提供使用 AWS的可扩展基础设施的优势。

操作说明

在登录区环境中设置 IoT 账户

Task	描述	所需技能
验证 IoT 账户中的安全防护机制。	验证您的物联网账户中是否启用了 CloudTrail、 AWS Config GuardDuty、和 Security Hub 的防护栏。	AWS 管理员
验证 IoT 账户是否已配置为安全账户的成员账户。	验证您的物联网账户是否已配置并关联为安全账户中的成员账户 GuardDuty 和 Security Hub。 有关这方面的更多信息，请参阅 GuardDuty 文档 AWS Organizations中的使用管理 GuardDuty 账户和 Security Hub 文档中的管理管理员和成员帐户。	AWS 管理员
验证日志存档。	验证 CloudTrail AWS Config、和 VPC 流日志存储在日志存档账户中。	AWS 管理员

设置集中式日志解决方案

Task	描述	所需技能
在安全账户中设置集中式日志解决方案。	登录您的安全账户， AWS Management Console 然后从解决方案库中设置集中日志解决方案，以便在 Amazon AWS S OpenSearch ervice 和 Dashboard CloudWatch s 中收集、分析和显示日 OpenSearch 志。 有关这方面的更多信息，请参阅使用解决方案库中集中日志实施指南中的集中 CloudWatch 日志解决方案，在单个控制面板中收集、分析和显示 Amazon 日志。 AWS	AWS 管理员

在您的 IoT 账户中设置和配置 AWS 资源

Task	描述	所需技能
设置 AWS IoT 日志。	登录您的 AWS Management Console 物联网帐户。设置并配置为将日志发送 AWS IoT Core 到 CloudWatch 日志。 有关这方面的更多信息，请参阅 AWS IoT Core 文档中的配置 AWS IoT 日志和AWS IoT 使用 CloudWatch 日志进行监控。	AWS 管理员

Task	描述	所需技能
设置 AWS IoT Device Defender。	设置 AWS IoT Device Defender 以审核您的物联网资源并检测异常。 有关这方面的更多信息，请参阅 AWS IoT Core 文档 AWS IoT Device Defender中的入门。	AWS 管理员
设置 CloudTrail。	设置为将事件发送 CloudTrail 到 CloudWatch 日志。 有关这方面的更多信息，请参阅 CloudTrail 文档中的向 CloudWatch 日志发送事件。	AWS 管理员
设置 AWS Config 和 AWS Config 规则。	设置 AWS Config 和所需的 AWS Config 规则。 有关这方面的更多信息，请参阅 AWS Config 文档中的 AWS Config 使用控制台进行设置和添加 AWS Config 规则。	AWS 管理员
设置 GuardDuty。	设置和配置 GuardDuty 以将结果发送到 Amazon E CloudWatch vents，并将 CloudWatch 日志中的日志组作为目标。 有关这方面的更多信息，请参阅 GuardDuty 文档中的使用 Amazon Ev CloudWatch ents 创建对 GuardDuty 调查结果的自定义响应。	AWS 管理员

Task	描述	所需技能
设置 Security Hub。	设置 Security Hub 并启用 CIS AWS 基础基准测试和 AWS 基础安全防御最佳实践标准。 有关这方面的更多信息，请参阅 Security Hub 文档中的自动响应和补救。	AWS 管理员
设置 Amazon Detective。	设置 Detective 以促进对安全发现的分析。 有关这方面的更多信息，请参阅 Amazon Detective 文档中的 Amazon Detective 入门。	AWS 管理员
设置亚马逊 Athena 然后。AWS Glue	设置 Athena AWS Glue 并查询进行安全事件调查 AWS 服务的日志。 有关这方面的更多信息，请参阅 Amazon Athena 文档中的查询 AWS 服务 日志。	AWS 管理员

相关资源

- 什么是登录区？

提取和查询 AWS IoT SiteWise 数据湖中的元数据属性

创建者：Ambarish Dongaonkar (AWS)

摘要

AWS IoT SiteWise 使用资产模型和层次结构来表示您的工业设备、流程和设施。每个模型或资产可以具有特定于您的环境的多个属性。示例元数据属性包括资产的站点或物理位置、工厂详细信息和设备标识符。这些属性值与资产计量数据相辅相成，实现商业价值最大化。机器学习 (ML) 可以为这些元数据提供更多见解，并简化工程作业。

但是，不能直接从 AWS IoT SiteWise 服务中查询元数据属性。要使这些属性可查询，您必须将它们提取并摄取到数据湖中。此模式使用 Python 脚本提取所有 AWS IoT SiteWise 资产的属性，并将它们提取到亚马逊简单存储服务 (Amazon S3) Simple Service 存储桶中的数据湖中。完成此过程后，您可以在 Amazon Athena 中使用 SQL 查询来访问 AWS IoT SiteWise 元数据属性和其他数据集，例如测量数据集。使用 AWS IoT SiteWise 监视器或仪表板时，元数据属性信息也很有用。您也可以使用 Amazon S3 存储桶中提取的属性来构建亚马逊 QuickSight 控制面板。

该模式具有参考代码，您可以通过使用最适合您的用例的计算服务（例如 AWS Lambda 或）来实现代码 AWS Glue。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- 设置 AWS Lambda 职能或 AWS Glue 作业的权限。
- Amazon S3 存储桶。
- 资产模型和层次结构是在中 AWS IoT SiteWise设置的。有关更多信息，请参阅 AWS IoT SiteWise 文档中的创建资产模型。

架构

您可以使用 Lambda 函数或 AWS Glue 任务来完成此过程。如果您的模型少于 100 个，并且每个模型平均具有 15 个或更少的属性，我们建议使用 Lambda。对于所有其他用例，我们建议使用 AWS Glue。

下图演示了参考架构和工作流。

1. 计划 AWS Glue 任务或 Lambda 函数将运行。它从 AWS IoT SiteWise Amazon S3 存储桶中提取资产元数据属性并将其提取到 Amazon S3 存储桶中。
2. AWS Glue 爬虫会在 Amazon S3 存储桶中抓取的数据并在中创建表。 AWS Glue Data Catalog
3. Amazon Athena 使用标准 SQL 查询中的表。 AWS Glue Data Catalog

自动化和扩缩

您可以根据 AWS IoT SiteWise 资产配置的更新频率将 Lambda 函数或 AWS Glue 任务安排为每天或每周运行。

示例代码可以处理的 AWS IoT SiteWise 资源数量没有限制，但是大量的资源会增加完成该过程所需的时间。

工具

- Amazon Athena 是一种交互式查询服务，它可帮助您通过使用标准 SQL 直接在 Amazon S3 中分析数据。
- AWS Glue是一项完全托管的提取、转换和加载 (ETL) 服务。它可以帮助您在数据存储和数据流之间对数据进行可靠地分类、清理、扩充和移动。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS IoT SiteWise帮助您大规模收集、建模、分析和可视化来自工业设备的数据。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Storage Service (Amazon S3)是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- 适用于 Python (Boto3) 的 AWS SDK是一个软件开发套件，可帮助您将 Python 应用程序、库或脚本与集成 AWS 服务。

操作说明

设置作业或职能

Task	描述	所需技能
在 IAM 中配置权限。	在 IAM 控制台中，向 Lambda 函数或 AWS Glue 任务担任的 IAM 角色授予执行以下操作的权限： • 从 AWS IoT SiteWise 服务中读取 • 写入亚马逊 S3 存储桶 有关更多信息，请参阅 IAM 文档 AWS 服务中的为创建角色。	常规 AWS
创建 Lambda 函数或 AWS Glue 任务。	如果您使用 Lambda，请创建新的 Lambda 函数。对于运行时系统，请选择 Python。有关更多信息，请参阅 Lambda 文档中的使用 Python 构建 Lambda 函数。 如果您正在使用 AWS Glue，请在 AWS Glue 控制台中创建一个新的 Python 外壳任务。有关更多信息，请参阅 AWS Glue 文档中的添加 Python 外壳任务。	常规 AWS
更新 Lambda 函数或 AWS Glue 任务。	修改新的 Lambda 函数或 AWS Glue 任务，然后在 “其他信息” 部分输入代码示例。修改您的用例所需代码。有关更多信息，请参阅 Lambda 文	常规 AWS

Task	描述	所需技能
	档中的使用控制台编辑器编辑代码和文档中的AWS Glue 使用脚本。	

运行作业或函数

Task	描述	所需技能
运行 Lambda 函数或 AWS Glue 作业。	运行 Lambda 函数或 AWS Glue 作业。有关更多信息，请参阅 Lambda 文档中的调用 Lambda 函数或参阅文档中的使用触发器启动作业。AWS Glue 这会提取 AWS IoT SiteWise 层次结构中资产和模型的元数据属性，并将其存储在指定的 Amazon S3 存储桶中。	常规 AWS
设置 AWS Glue 爬虫。	为采用 CSV 格式文件设置必需的格式分类器的 C AWS Glue rawler。使用 Lambda 函数或 AWS Glue 任务中使用的 Amazon S3 存储桶和前缀详细信息。有关更多信息，请参阅 AWS Glue 文档中的定义抓取工具。	常规 AWS
运行 AWS Glue 爬行器。	运行爬网程序以处理由 Lambda 函数 AWS Glue 或任务创建的数据文件。Crawler 在指定的 AWS Glue Data Catalog中创建表。有关更多信息，请参阅 AWS Glue 文档中的使用触发器启动抓取工具。	常规 AWS

Task	描述	所需技能
查询元数据属性。	使用 Amazon Athena，根据您的用例使用需要，使用标准 SQL 进行查询 AWS Glue Data Catalog 。您可将元数据属性表与其他数据库和表联接。有关更多信息，请参阅 Amazon Athena 文档中的入门指南。	常规 AWS

相关资源

- Amazon Athena 文档
- AWS Glue 文档
- AWS IoT SiteWise API 参考
- AWS IoT SiteWise 用户指南
  - 入门
  - 工业资产建模
  - 定义资产模型（层次结构）之间的关系
  - 关联资产和取消资产关联
  - 创建演 AWS IoT SiteWise 示
- IOTSiteWise（适用于 Python 的 SDK 文档）
- Lambda 文档

其他信息

代码

提供的示例代码仅供参考，您可根据用例的需要自定义此代码。

```
# Following code can be used in an AWS Lambda function or in an AWS Glue Python shell
 job.
# IAM roles used for this job need read access to the AWS IoT SiteWise service and
 write access to the S3 bucket.
sw_client = boto3.client('iotsitewise')
```

```
s3_client = boto3.client('s3')
output = io.StringIO()

attribute_list=[]
bucket = '{3_bucket name}'
prefix = '{s3_bucket prefix}'
output.write("model_id,model_name,asset_id,asset_name,attribuet_id,attribute_name,attribute_val
\n")

m_resp = sw_client.list_asset_models()
for m_rec in m_resp['assetModelSummaries']:
     model_id = m_rec['id']
     model_name = m_rec['name']

     attribute_list.clear()
     dam_response = sw_client.describe_asset_model(assetModelId=model_id)
     for rec in dam_response['assetModelProperties']:
         if 'attribute' in rec['type']:
            attribute_list.append(rec['name'])

     response = sw_client.list_assets(assetModelId=model_id, filter='ALL')
     for asset in response['assetSummaries']:
         asset_id = asset['id']
         asset_name = asset['name']
         resp = sw_client.describe_asset(assetId=asset_id)
         for rec in resp['assetProperties']:
            if rec['name'] in attribute_list:
                p_resp = sw_client.get_asset_property_value(assetId=asset_id,
 propertyId=rec['id'])
                if 'propertyValue' in p_resp:
                    if p_resp['propertyValue']['value']:
                        if 'stringValue' in p_resp['propertyValue']['value']:
                             output.write(model_id + "," + model_name + ","
 + asset_id + "," + asset_name + "," + rec['id'] + "," + rec['name'] + "," +
 str(p_resp['propertyValue']['value']['stringValue']) + "\n")

                        if 'doubleValue' in p_resp['propertyValue']['value']:
                             output.write(model_id + "," + model_name + ","
 + asset_id + "," + asset_name + "," + rec['id'] + "," + rec['name'] + "," +
 str(p_resp['propertyValue']['value']['doubleValue']) + "\n")
                        if 'integerValue' in p_resp['propertyValue']['value']:
                             output.write(model_id + "," + model_name + ","
 + asset_id + "," + asset_name + "," + rec['id'] + "," + rec['name'] + "," +
 str(p_resp['propertyValue']['value']['integerValue']) + "\n")
```

```
                        if 'booleanValue' in p_resp['propertyValue']['value']:
                            output.write(model_id + "," + model_name + ","
 + asset_id + "," + asset_name + "," + rec['id'] + "," + rec['name'] + "," +
 str(p_resp['propertyValue']['value']['booleanValue']) + "\n")

output.seek(0)
s3_client.put_object(Bucket=bucket, Key= prefix + '/data.csv', Body=output.getvalue())
output.close()
```

使用客户端设备设置 Amazon IoT Greengrass 并对其故障排除

由 Marouane Sefiani 和 Akalanka De Silva (AWS) 编写

摘要

Amazon IoT Greengrass 是一种开源边缘运行时和云服务，用于在边缘设备上构建、部署和管理物联网 (IoT) 软件。Amazon IoT Greengrass 用例包括：

- 智能家居，其中 Amazon IoT Greengrass 网关用作家庭自动化中心
- 智能工厂，Amazon IoT Greengrass 可以促进车间数据的提取和本地处理

Amazon IoT Greengrass 可以充当其他边缘设备（也称为客户端设备）的安全、经过身份验证的 MQTT 连接端点，否则这些设备通常会直接连接到 AWS IoT Core。当客户端设备无法直接通过网络访问 AWS IoT Core 端点时，此功能非常有用。

您可将 Amazon IoT Greengrass 设置为与客户端设备一起使用，用于以下用例：

- 供客户端设备将数据发送到 Amazon IoT Greengrass
- 让 Amazon IoT Greengrass 将数据转发至 AWS IoT Core
- 利用高级 AWS IoT Core 规则引擎功能

这些功能需要在 Amazon IoT Greengrass 设备上安装与配置以下组件：

- MQTT 代理
- MQTT 网桥
- 客户端设备身份验证
- IP 探测器

此外，来自客户端设备的已发布消息必须采用 JSON 格式或协议缓冲区 (protobuf) 格式。

本示例介绍了如何安装和配置这些必需的组件，并提供了故障排除提示和最佳实践。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account

- AWS 命令行界面（AWS CLI）版本 2
- 两台运行 Python 3.7 或更高版本的客户端设备
- 一台运行 Java 运行时环境（JRE）版本 8 或更高版本以及 Amazon Corretto 11 或 OpenJDK 11的核心设备

限制

- 您必须选择一个提供 AWS IoT Core 的 AWS 区域。有关当前的 AWS IoT Core 区域列表，请参阅按区域划分的 Amazon Web Services。
- 核心设备必须至少有 172 MB 内存和 512 MB 磁盘空间。

架构

下图显示此模式的解决方案架构。

此架构包括：

- 两个客户端设备 每个设备都包含私钥、设备证书以及根证书颁发机构 (CA) 证书。每个客户端设备上还安装了包含 MQTT 客户端的 AWS IoT 设备开发工具包。
- 部署了 Amazon IoT Greengrass 的核心设备，包含以下组件：
  - MQTT 代理
  - MQTT 网桥
  - 客户端设备身份验证
  - IP 探测器

该架构支持以下场景：

- 客户端设备可以使用 MQTT 客户端通过核心设备的 MQTT 代理相互通信。
- 客户设备还可通过核心设备的 MQTT 代理和 MQTT 桥与云中的 AWS IoT Core 通信。
- 云端的 AWS IoT Core 可通过 MQTT 测试客户端以及核心设备的 MQTT 网桥器和 MQTT 代理向客户端设备发送消息。

有关客户端设备和核心设备之间通信的更多信息，请参阅其他信息部分。

工具

Amazon Web Services

- Amazon IoT Greengrass 是一项开源物联网 (IoT) 边缘运行时和云服务，可帮助您在设备上构建、部署和管理 IoT 应用程序。
- AWS IoT Core 为连接互联网的设备提供安全的双向通信，以连接到 Amazon Web Services Cloud。
- AWS IoT Device SDK 是一个软件开发工具包，包括开源库、开发人员指南（含示例）和移植指南，便于您在自己选择的硬件平台上构建创新的 IoT 产品或解决方案。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。

最佳实践

- 来自客户端设备的消息负载应采用 JSON 或 Protobuf 格式，以便利用 AWS IoT Core 规则引擎的高级功能，例如转换和条件操作。
- 配置 MQTT 巧劲儿以允许双向通信。
- 在 Amazon IoT Greengrass 中配置和部署 IP 检测器组件，以确保核心设备的 IP 地址包含在 MQTT 代理证书的使用者备用名称 (SAN) 字段中。

操作说明

设置核心设备

Task	描述	所需技能
在您的核心设备设置 Amazon IoT Greengrass。	按照开发人员指南中的说明安装 Amazon IoT Greengrass Core 软件。	Amazon IoT Greengrass
检查安装状态。	使用以下命令检查核心设备上 Amazon IoT Greengrass 服务的状态： `sudo systemctl status greengrass.service`	常规 AWS

Task	描述	所需技能
	命令预期输出是： `Launched Nucleus successfully`	

Task	描述	所需技能
设置 IAM policy 并将其附加至 Greengrass 服务角色。	1. 创建 IAM policy，以允许与 MQTT 网桥进行通信。以下为策略示例： `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "iot:*" ], "Resource": "*" }, { "Sid": "GreengrassActions", "Effect": "Allow", "Action": [ "greengrass:*" ], "Resource": "*" } ] }` 2. 将策略附加到 Greengrass 服务角色。要获取服务角色，请使用以下命令：	常规 AWS

Task	描述	所需技能
	`aws greengrassv2 get-service-role-for-account --region <region>` 其中，<region> 指的是您的 AWS 区域。	
在 Amazon IoT Greengrass 核心设备中配置和部署所需组件。	配置与部署以下组件： • greengrass.clientdevices.mqtt.Moquette （参阅配置详细信息） • greengrass.clientdevices.mqtt.Bridge （参阅配置详情和下一个任务） • greengrass.clientdevices.Auth （参阅配置详细信息以及下一个任务之后的任务） • aws.greengrass.clientdevices.IPDetector （参阅配置详细信息）	Amazon IoT Greengrass

Task	描述	所需技能
确认 MQTT 桥允许双向通信。	要在客户端设备和 AWS IoT Core 之间中继 MQTT 消息，请配置和部署 MQTT 桥组件并指定要中继的主题。示例如下： `{ "mqttTopicMapping": { "ClientDevicesToCloud": { "topic": "dt/#", "source": "LocalMqtt", "target": "IotCore" }, "CloudToClientDevices": { "topic": "cmd/#", "source": "IotCore", "target": "LocalMqtt" } } }`	Amazon IoT Greengrass

Task	描述	所需技能
确认身份验证组件允许客户端设备连接并发布或订阅主题。	以下 `aws.greengrass.clientdevices.Auth` 配置允许所有客户端设备连接、发布消息和订阅所有主题。 `{ "deviceGroups": { "formatVersion": "2021-03-05", "definitions": { "MyPermissiveDeviceGroup": { "selectionRule": "thingName: *", "policyName": "MyPermissivePolicy" } }, "policies": { "MyPermissivePolicy": { "AllowAll": { "statementDescription": "Allow client devices to perform all actions.", "operations": ["*"], "resources": ["*"] } } } } }`	Amazon IoT Greengrass

Task	描述	所需技能
	`}`	

设置客户端设备

Task	描述	所需技能

安装 AWS IoT Device SDK。	在客户端设备上安装 AWS IoT 设备软件开发工具包。有关支持的语言和相关语言的完整列表 SDKs，请参阅 AWS IoT Core 文档。
	
	例如，适用于 Python 的 AWS 物联网设备软件开发工具包位于上 GitHub。若要安装此 SDK，请执行以下操作
	
	1. 按照 GitHub 存储库的 “先决条件” 页面上的说明，确认已安装 Python 3.7 或更高版本。
	2. 使用 pip 命令安装 SDK。
	
	对于 MacOS 和 Linux：
	
	`python3 -m pip install awsiotsdk`
	
	对于 Windows：
	
	`python -m pip install awsiotsdk`
		常规 AWS IoT

Task	描述	所需技能

	或者，您可从源存储库安装 SDK：
	
	`# Create a workspace directory to hold all the SDK files`
	`mkdir sdk-workspace`
	`cd sdk-workspace`
	`# Clone the repository`
	`git clone https://github.com/aws/aws-iot-device-sdk-python-v2.git`
	`# Install using Pip (use 'python' instead of 'python3' on Windows)`
	`python3 -m pip install ./aws-iot-device-sdk-python-v2`	

Task	描述	所需技能
创建事物。	1. 在 AWS IoT 控制台中，如果显示开始使用按钮，请选择该按钮。否则，请在导航窗格中展开 Secure（安全），然后选择 Policies（策略）。 2. 如果显示您还没有任何策略对话框，请选择创建策略。否则，选择创建。 3. 输入 AWS IoT 策略的名称（例如，`ClientDevicePolicy`）。 4. 在添加语句部分中，将现有策略替换为以下 JSON 代码。使用您的 AWS 区域 和 AWS 账号替换 `<region>` 和 `<account>` 。 `{ "Version": "2012-10-17", "Statement": [{ "Effect": "Allow", "Action": "iot:Connect", "Resource": "arn:aws:iot:region:account:client/*" }, { "Effect": "Allow", "Action": "iot:Publish", "Resource": "*"`	AWS IoT Core

Task	描述	所需技能

	<pre> },
	  {
	  "Effect":
	"Allow",
	  "Action":
	"iot:Receive",
	  "Resource":
	"*"
	  },
	  {
	  "Effect":
	"Allow",
	  "Action":
	"iot:Subscribe",
	  "Resource":
	"*"
	  },
	  {
	  "Effect":
	"Allow",
	  "Action": [
	  "iot:GetT
	hingShadow",
	  "iot:Upda
	teThingShadow",
	  "iot:Dele
	teThingShadow"
	  ],
	  "Resource":
	"arn:aws:iot:regio
	n:account:thing/*"
	
	  }
	  ]
	}</pre>
	5. 选择创建。
	6. 在 AWS IoT 控制台导航窗格中，选择管理、事物。	

Task	描述	所需技能
	7. 如果显示您还没有任何事物对话框，请选择注册事物。否则，选择创建。 8. 在创建 AWS IoT 事物页面上，选择创建单个事物。 9. 在将您的设备添加到设备注册表 页面上，输入您的 IoT 事物的名称（例如 `ClientDevice1` ），然后选择下一步。 **Note** 您无法在创建事物后更改其名称。要更改名称，您必须创建一个新事物，为其指定新名称，然后删除旧事物。 10.在添加事物的证书页面上，选择创建证书。 11选择下载链接以下载证书、私有密钥和根 CA 证书。 **Important** 这是您下载证书和私钥的唯一机会。 12选择 Activate(激活)来激活您的证书。证书必须处于活动状态，设备才能连接到 AWS IoT。 13选择附加策略。	

Task	描述	所需技能
	14.在 “为你的事物添加政策” 中 ClientDevicePolicy，选择 “注册事物”。	
从 Greengrass 核心设备下载 CA 证书。	如果您希望 Greengrass 核心设备在离线环境中工作，则必须使 Greengrass 核心 CA 证书可供客户端设备使用，以便客户端设备可以验证 MQTT 代理的证书（由 Greengrass 核心 CA 颁发）。因此，获取此证书副本非常重要。通过以下方法之一下载 CA 证书： • 如果您可通过网络访问电脑上的 Amazon IoT Greengrass 设备，请在网络浏览器输入 `https://<device IP>:8883`，查看 MQTT 代理证书和 CA 证书。您也可将 CA 证书保存到客户端设备。 • 或者，您可使用 OpenSSL 命令行： `openssl s_client -showcerts -connect <device IP>:8883`	常规 AWS
在客户端设备复制凭证。	在客户端设备中复制 Greengrass 核心 CA 证书、设备证书以及私钥。	常规 AWS

Task	描述	所需技能
将客户端设备与核心设备相关联。	将客户端设备与核心设备关联，以便其可以发现核心设备。然后，客户端设备可以使用 Greengrass 发现 API 来检索其关联核心设备的连接信息和证书。有关更多信息，请参阅 Amazon IoT Greengrass 文档中的关联客户端设备。 1. 在 Amazon IoT Greengrass 控制台，选择核心设备。 2. 选择托管管理的核心设备。 3. 在核心设备的详细信息页面上，请选择客户端设备选项卡。 4. 在关联客户端设备部分，选择关联客户端设备。 5. 在将客户端设备与核心设备关联模式中，对要关联的每台客户端设备执行以下操作： a. 输入 AWS IoT 事物名称，以将其关联为客户端设备。 b. 选择 添加。 6. 选择关联 。 您关联的客户端设备现在可使用 Greengrass 发现 API 来发现此核心设备。	Amazon IoT Greengrass

发送与接收数据

Task	描述	所需技能
将数据从一个客户端设备发送到另一客户端设备。	使用设备中的 MQTT 客户端发布有关 `dt/client1/sensor` 主题的消息。	常规 AWS
将数据从客户端设备发送至 AWS IoT Core。	使用设备中的 MQTT 客户端发布有关 `dt/client1/sensor` 主题的消息。 在 MQTT 测试客户端中，订阅设备正在发送消息的主题，或订阅 # 以获取所有主题（查看详细信息）。	常规 AWS
将数据从 AWS IoT Core 发送至客户端设备。	在 MQTT 测试客户端页面上，在发布到主题选项卡上的主题名称字段中，输入您消息的主题名称。在此示例中，使用 `cmd/client1` 作为主题。	常规 AWS

故障排除

事务	解决方案
无法验证服务器证书错误	当 MQTT 客户端无法验证 MQTT 代理在 TLS 握手期间提供的证书时，会出现此错误。最常见的原因是 MQTT 客户端缺失 CA 证书。请按照以下步骤确保将 CA 证书提供给 MQTT 客户端。 1. 如果您可通过网络访问电脑上的 Amazon IoT Greengrass 设备，请在浏览器窗口中输入 `https://<device IP>:8883` 以查看 MQTT 代理证书和 CA 证书。您也可将 CA 证书保存到客户端设备。

事务	解决方案

	或使用 OpenSSL 命令行：
	
	`openssl s_client -showcerts -connect <device IP>:8883`
	
	2. 将 Moquette CA 和 Greengrass Core CA 证书内容保存到文件中，然后使用以下命令查看解码后的内容：
	
	`openssl x509 -in <Name of CA>.pem -text`
	
	Moquette CA 证书应显示 SAN 字段，如下所示：
	
	`X509v3 Subject Alternative Name: IP Address:XXX.XXX.XXX.XXX, IP Address:127.0.0.1, DNS:localhost`

无法验证服务器名称错误	当 MQTT 客户端无法验证是否连接到正确的服务器时，就会发生此错误。最常见的原因是 Greengrass 设备的 IP 地址未在证书的 SAN 字段中列出。 按照前一解决方案中的说明获取 MQTT 代理证书，并验证 SAN 字段是否包含 Amazon IoT Greengrass 设备的 IP 地址，如其他信息部分所述。如果没有，请确认 IP 检测器组件安装得当，然后重新启动核心设备。

事务	解决方案
仅在从嵌入式客户端设备连接时，无法验证服务器名称	Mbed TLS 是嵌入式设备中使用的流行 TLS 库，目前仅在证书的 SAN 字段中支持 DNS 名称验证，如 Mbed TLS 库代码所示。由于核心设备没有自己的域名并且依赖于 IP 地址，因此使用 Mbed TLS 的 TLS 客户端将在 TLS 握手期间无法通过服务器名称验证，从而导致连接失败。我们建议您通过 x509_crt_check_san 函数将 SAN IP 地址验证添加至 mbed TLS 库中。

相关资源

- Amazon IoT Greengrass 文档
- AWS IoT Core 文档
- MQTT 代理组件
- MQTT 网桥组件
- 客户端设备身份验证组件
- IP 探测器组件
- AWS IoT Device SDK
- 使用 Amazon IoT Greengrass 实现本地客户端设备(AWS Blog 文章)
- RFC 5280 — Internet X.509 公钥基础设施证书和证书吊销列表 (CRL) 配置文件

其他信息

本节提供有关客户端设备和核心设备之间通信的附加信息。

MQTT 代理在核心设备中的端口 8883 上侦听 TLS 客户端连接尝试。下图显示了 MQTT 代理的服务器证书示例。

示例证书显示以下详细信息：

- 该证书由 Amazon IoT Greengrass Core CA 颁发，该证书是本地的且特定于核心设备，也就是说，它充当本地 CA。

- 该证书由客户端身份验证组件每周自动轮换，如下图所示。您可在客户端身份验证组件配置中设置此间隔。

- 主题备用名称 (SAN) 在 TLS 客户端服务器名称验证中起着至关重要的作用。它可以帮助 TLS 客户端确保它连接到正确的服务器，并有助于避免在 TLS 会话设置期间 man-in-the-middle受到攻击。在示例证书中，SAN 字段表示此服务器正在本地主机 (本地 Unix 域套接字) 上侦听，并且网络接口的 IP 地址为 192.168.1.12。

TLS 客户端在服务器验证期间使用证书中的 SAN 字段来验证它是否正在连接到合法服务器。相比之下，在 HTTP 服务器和浏览器之间的典型 TLS 握手期间，通用名称 (CN) 字段或 SAN 字段中的域名用于在服务器验证过程中交叉检查浏览器实际连接到的域 。如果核心设备没有域名，SAN 字段中包含的 IP 地址也有同样的作用。有关更多信息，请参阅 RFC 5280 — Internet X.509 公钥基础设施证书和证书吊销列表 (CRL) 配置文件的主题备用名称部分。

Amazon IoT Greengrass 中的 IP 检测器组件可确保证书 SAN 字段中包含正确的 IP 地址。

示例中的证书由充当本地 CA 的 Amazon IoT Greengrass 设备进行签名。TLS 客户端（MQTT 客户端）不知道此 CA，因此我们必须提供如下所示的 CA 证书。

更多模式

- 使用 Amazon IoT Greengrass 将物联网数据直接摄取至 Amazon S3，经济实惠

迁移和现代化

主题

- 迁移
- 现代化
- 主机

迁移

主题

- 使用微软 Excel 和 Python 为 AWS DMS 任务创建 AWS CloudFormation 模板
- 开始使用自动发现产品组合
- 将本地 Cloudera 工作负载迁移到 Cloudera Data Platform on AWS
- 解决将 Microsoft SQL Server 迁移至 Amazon Web Services Cloud 后出现的连接错误
- 重启 RHEL 源服务器 SELinux 后，无需禁用 AWS 复制代理即可自动重启 AWS 复制代理
- 重构
- 更换主机
- 重新定位
- 更换平台
- 按照工作负载分类的迁移模式
- 更多模式

使用微软 Excel 和 Python 为 AWS DMS 任务创建 AWS CloudFormation 模板

由 Venkata Naveen Koppula (AWS)创建

摘要

此模式概述了使用微软 Excel 和 Python 自动为 AWS 数据库迁移服务 (AWS DMS) 创建 AWS CloudFormation 模板的步骤。

使用 AWS DMS 迁移数据库通常涉及创建 AWS CloudFormation 模板来配置 AWS DMS 任务。以前，创建 AWS CloudFormation 模板需要了解 JSON 或 YAML 编程语言。借助这一工具，您只需要了解 Excel 的基本知识以及如何使用终端或命令窗口运行 Python 脚本即可。

作为输入，该工具采用一个 Excel 工作簿，其中包括要迁移的表的名称、AWS DMS 终端节点的 Amazon 资源名称 (ARNs) 以及 AWS DMS 复制实例。然后，该工具会为所需的 AWS DMS 任务生成 AWS CloudFormation 模板。

有关详细步骤和背景信息，请参阅 AWS 数据库博客中的博客文章使用 Microsoft Excel 为 AWS DMS 任务创建 AWS CloudFormation 模板。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- Microsoft Excel 版本 2016 或更高版本。
- Python（2.7 或更高版本）
- xlrd Python 模块(在命令提示符下安装，命令为：pip install xlrd)
- AWS DMS 源端点和目标端点以及 AWS DMS 复制实例

限制

- 架构、表和关联列的名称将在目标端点处转换为小写字符。
- 此工具不涉及 AWS DMS 端点和复制实例的创建。
- 目前，该工具仅支持每个 AWS DMS 任务使用一个架构。

架构

源技术堆栈

- 本地数据库
- Microsoft Excel

目标技术堆栈

- AWS CloudFormation 模板
- Amazon Web Services Cloud 中的数据库

架构

工具

- Pycharm IDE 或任何支持 Python 版本 3.6 的集成式开发环境(IDE)
- Microsoft Office 2016（适用于 Microsoft Excel）

操作说明

配置网络、AWS DMS 复制实例和端点

Task	描述	所需技能
如有必要，可请求增加服务配额。	如有需要，可请求增加 AWS DMS 任务的服务限额。	常规 AWS
配置 AWS 区域、虚拟私有云(VPCs)、CIDR 范围、可用区和子网。		常规 AWS
配置 AWS DMS 复制实例。	AWS DMS 复制实例可以连接到本地数据库和 AWS 数据库。	常规 AWS

Task	描述	所需技能
配置 AWS DMS 端点。	为源数据库和目标数据库配置端点。	常规 AWS

为 AWS DMS 任务和标签准备工作表

Task	描述	所需技能
配置表列表。	列出迁移中涉及的所有表。	数据库
准备任务工作表。	使用您配置的表列表准备 Excel 工作表。	常规 AWS、Microsoft Excel
准备标签工作表。	详述附加到 AWS DMS 任务的 AWS 资源标签。	常规 AWS、Microsoft Excel

下载并运行工具

Task	描述	所需技能
从 GitHub 存储库下载并提取模板生成工具。	GitHub 存储库: https://github.com/aws-samples/dms-cloudformation-templates-generator/	
运行工具。	请按照“参考和帮助”下列出的博客文章中的详细说明进行操作。	

相关资源

- 使用微软 Excel 为 AWS DMS 任务创建 AWS CloudFormation 模板（博客文章）
- DMS CloudFormation 模板生成器（GitHub 存储库）
- Python 文档

- XLRD 说明和下载
- AWS DMS 文档
- AWS CloudFormation 文档

开始使用自动发现产品组合

创建者：Pratik Chunawala (AWS) 和 Rodolfo Jr. Cerrada (AWS)

摘要

在将应用程序和服务器迁移到 Amazon Web Services (AWS) 云时，评估产品组合和收集元数据是一项关键挑战，特别是对于服务器超过 300 台的大型迁移。使用自动产品组合发现工具可以帮助您收集有关应用程序的信息，例如用户数量、使用频率、依赖项以及有关应用程序基础设施的信息。在规划迁移浪潮时，这些信息是必不可少的，这样您就可以对具有相似特征的应用程序进行适当的优先级排序和分组。使用发现工具可以简化产品组合团队与应用程序所有者之间的沟通，因为产品组合团队可以验证发现工具的结果，而不必手动收集元数据。此模式讨论了选择自动发现工具的关键注意事项，以及有关如何在环境中部署和测试自动发现工具的信息。

此模式包括构建您自己的高级活动清单的起点模板。清单旁边是负责人、负责制者、协商参与者、知情人 (RACI) 矩阵的模板。您可以使用此 RACI 矩阵来确定谁负责清单中的每项任务。

操作说明

选择发现工具

Task	描述	所需技能
确定发现工具是否适合您的用例。	发现工具可能不是用例的最佳解决方案。考虑选择、采购、准备和部署发现工具所需时间。在环境中为无代理发现工具设置扫描设备或为所有范围内的工作负载安装代理可能需要 4–8 周的时间。部署完成后，发现工具必须等待 4–12 周才能通过扫描应用程序工作负载和执行应用程序堆栈分析来收集元数据。如果您迁移的服务器少于 100 台，则手动收集元数据和分析依赖项的速度可能快于使用自动发现工具部署和收集元数据所需时间。	迁移主管，迁移工程师

Task	描述	所需技能
选择发现工具。	在其他信息部分中查看选择自动发现工具的注意事项。确定为用例选择发现工具的适当标准，然后根据这些标准评估每个工具。有关自动发现工具的完整列表，请参阅发现、规划和建议迁移工具。	迁移主管，迁移工程师

准备安装

Task	描述	所需技能
准备部署前清单。	创建一份在部署工具之前必须完成的任务清单。有关示例，请参阅 Flexera 文档网站上的部署前清单。	构建主管，迁移工程师，迁移主管，网络管理员
准备网络要求。	预调配工具运行和访问目标服务器所需端口、协议、IP 地址和路由。有关更多信息，请参阅发现工具的安装指南。有关示例，请参阅 Flexera 文档网站上的部署要求。	迁移工程师、网络管理员、云架构师
准备账户和凭证要求。	确定访问目标服务器和安装该工具的所有组件所需凭证。	云管理员，常规 AWS，迁移工程师，迁移主管，网络管理员，AWS 管理员
准备好要安装该工具的设备。	确保要安装工具组件的设备符合该工具的规格和平台要求。	迁移工程师，迁移主管，网络管理员
准备变更单。	根据贵组织中的变更管理流程，准备所需所有变更单，并确保这些变更单获得批准。	构建主管，迁移主管

Task	描述	所需技能
向利益相关者发送要求。	将部署前清单和网络要求发送给利益相关者。利益相关者应在继续部署之前审查、评估和准备必要的要求。	构建主管，迁移主管

部署工具

Task	描述	所需技能
下载安装程序。	下载安装程序或虚拟机映像。虚拟机映像通常采用开放虚拟化格式 (OVF)。	构建主管，迁移主管
提取文件。	如果您使用的是安装程序，则必须在本地服务器上下载并运行该安装程序。	构建主管，迁移主管
在服务器上部署该工具。	在目标本地服务器上部署发现工具，如下所示： • 如果您的源文件是虚拟机映像，请将其部署到您的虚拟机环境中，例如 VMware。 • 如果源文件是安装程序，请运行安装程序来安装和设置该工具。	构建主管，迁移主管，网络管理员
登录发现工具。	按照屏幕上的提示进行操作，然后登录以开始使用该工具。	迁移主管，构建主管
激活产品。	输入许可证密钥。	构建主管，迁移主管
配置工具。	输入访问目标服务器所需的所有凭据，例如 Windows VMware、简单网络管理协	构建主管，迁移主管

Task	描述	所需技能
	议 (SNMP) 和安全外壳协议 (SSH) 或数据库的凭据。	

测试工具

Task	描述	所需技能
选择测试服务器。	识别可用于测试发现工具的非生产子网或 IP 地址的小子集。这可以帮助您快速验证扫描，快速识别和排除任何错误，并将测试与生产环境隔离开来。	构建主管，迁移主管，网络管理员
开始扫描选定的测试服务器。	对于无代理发现工具，请在发现工具控制台中输入所选测试服务器的子网或 IP 地址，然后开始扫描。 对于基于代理的发现工具，请在选定的测试服务器上安装代理。	构建主管，迁移主管，网络管理员
查看扫描结果。	查看测试服务器的扫描结果。如果发现任何错误，请进行故障排除并修复错误。记录错误和解决方案。您将来会参考这些信息，并且可以将这些信息添加到产品组合运行手册中。	构建主管，迁移主管，网络管理员
重新扫描测试服务器。	重新扫描完成后，重复扫描，直到没有错误。	构建主管，迁移主管，网络管理员

相关资源

AWS resources

- AWS Cloud 迁移应用程序组合评估指南
- 发现、规划和推荐迁移工具

常用发现工具的部署指南

- 部署 RN15 0 虚拟设备（Flexera 文档）
- Gatherer 安装（ModelizeIT 文档）
- 本地分析服务器安装（ModelizeIT 文档）

其他信息

选择自动发现工具的注意事项

每种发现工具都有其优点和局限性。在为用例选择合适的工具时，请注意以下方面：

- 选择一种发现工具，该工具可以收集实现产品组合评测目标所需大部分（如果不是全部）元数据。
- 找出由于该工具不支持所以您需要手动收集的所有元数据。
- 向利益相关者提供发现工具要求，以便他们可以根据其内部安全和合规性要求（例如服务器、网络和凭证要求）查看和评测该工具。
  - 该工具是否要求您在范围内的工作负载中安装代理？
  - 该工具是否要求您在自己的环境中设置虚拟设备？
- 确定数据驻留要求。有些组织不想将其数据存储在环境之外。要解决这个问题，您可能需要在本地环境中安装该工具的某些组件。
- 确保该工具支持范围内工作负载的操作系统 (OS) 和操作系统版本。
- 确定产品组合是否包括大型机、中端服务器和传统服务器。大多数发现工具可以将这些工作负载检测为依赖项，但有些工具可能无法获取设备详细信息，例如使用率和服务器依赖项。Device42 和 ModernizeIT 发现工具都支持大型机和中端服务器。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

将本地 Cloudera 工作负载迁移到 Cloudera Data Platform on AWS

由 Battulga Purevragchaa (AWS)、Nijjwol Lamsal（合伙人）和 Nidhi Gupta（AWS）创作

摘要

此模式描述了将本地 Cloudera 分布式 Hadoop (CDH)、Hortonworks 数据平台 (HDP) 和 Cloudera 数据平台 (CDP) 工作负载迁移到 CDP Public Cloud on AWS 的高级步骤。我们建议您与 Cloudera Professional Services 和系统集成商 (SI) 合作来实施这些步骤。

Cloudera 客户希望将其本地 CDH、HDP 和 CDP 工作负载迁移到云端的原因有很多。部分常见原因包括：

- 简化新数据平台范例（例如数据湖屋或数据网格）的采用
- 提高业务敏捷性，实现现有数据资产的访问和推理民主化
- 降低总拥有成本（TCO）
- 增强工作负载弹性
- 实现更大的可扩展性；与遗留的本地安装基础相比，大大减少了预调配数据服务的时间
- 停用遗留硬件；大大减少了硬件刷新周期
- 利用定价优势，该 pay-as-you-go定价已扩展到采用 Cloudera 许可模式 (CCU) 的 AWS 上的 Cloudera 工作负载
- 利用持续集成和持续交付 (CI/CD) 平台，实现更快的部署和改进的集成
- 使用单个统一平台 (CDP) 处理多工作负载

Cloudera 支持所有主要工作负载，包括机器学习、数据工程、数据仓库、操作数据库、流处理 (CSP) 及数据安全和治理。Cloudera 多年来一直在本地提供这些工作负载，您可以通过将 CDP 公共云与 Workload Manager和Replication Manager结合使用，将这些工作负载迁移到 Amazon Web Services Cloud。

Cloudera Shared Data Experience (SDX) 提供跨这些工作负载的共享元数据目录，以促进一致的数据管理和操作。SDX 还包括全面、精细的安全性，以防范威胁，以及审计和搜索功能的统一治理，以符合支付卡行业数据安全标准 (PCI DSS) 和 GDPR 等标准。

CDP 迁移一览

源工作负载	CDH、HDP 和 CDP Private Cloud

	源环境	• Windows、Linux • 本地、主机托管或任何非 AWS 环境
工作负载		
	目标工作负载	CDP Public Cloud on AWS
	目标环境	• 部署模式：客户账户 • 运营模式：客户/ClouderA 控制面板
	迁移策略 (7R)	更换主机、更换平台或重构
	这是工作负载版本的升级吗？	是
迁移	迁移持续时间	• 部署：创建客户账户、虚拟私有云（VPC）和 CDP 公有云客户管理的环境大约需要 1 周。 • 迁移时间：1-4 个月，具体取决于工作负载的复杂性与规模。

成本	在 AWS 上运行工作负载的成本	• 在较高层面上，CDH 工作负载迁移到 AWS 的成本的前提条件是您将在 AWS 上建立一个新环境。它包括对人员时间和精力的核算，以及为新环境预调配计算资源和许可软件。 • Cloudera 基于云消费定价模型使您能够灵活地利用突发和自动扩缩功能。有关更多信息，请参阅 Cloudera 网站上的 CDP 公有云服务费率。 • Cloudera Enterprise Data Hub 基于亚马逊弹性计算云 (Amazon EC2)，与传统集群非常接近。Data Hub 可自定义，但这会影响成本。 • CDP Public Cloud Data Warehouse、Cloudera Machine Learning 和 Cloudera Data Engineering (CDE) 基于容器，可以配置为自动扩缩。
基础设施协议与框架	系统要求	请参阅先决条件部分。
	SLA	请参阅 CDP 公共云的 Cloudera 服务级别协议。
	DR	请参阅 Cloudera 文档中的灾难恢复。
	（目标 Amazon Web Services account 的）许可和运营模式	自带许可（BYOL）模式
合规	安全要求	请参阅 Cloudera 文档中的 Cloudera 安全概述。

其他合规性认证	在 Cloudera 网站查看关于通用数据保护条例 (GDPR) 合规和 CDP Trust Center 的信息。

先决条件和限制

先决条件

- Amazon Web Services account 要求，包括账户、资源、服务和权限，如 AWS Identity and Access Management (IAM) 角色和策略设置
- 在 Cloudera 网站部署 CDP 的先决条件

迁移需要以下角色和专长：

角色	技能和责任
迁移主管	确保执行支持、团队协作、规划、实施和评测
Cloudera SME	CDH、HDP 以及 CDP 管理、系统管理和架构方面的专业技能
AWS 架构师	Amazon Web Services、联网、安全和架构方面的技能

架构

构建适当的架构是确保迁移和性能满足用户期望的关键步骤。为了使您的迁移工作满足本行动手册的假设，您在 Amazon Web Services Cloud 中的目标数据环境，无论是在虚拟私有云（VPC）托管的实例上还是在 CDP 上，都必须在操作系统和软件版本以及主要机器规范方面与您的源环境完全匹配。

下图（经许可转载自 Cloudera Shared Data Experience 数据表）显示 CDP 环境的基础设施组件以及各层或基础设施组件如何交互。

该架构包括以下 CDP 组件：

- Data Hub 是一项用于启动和管理由 Cloudera 运行时系统支持的工作负载集群的服务。您可使用 Data Hub 中的集群定义为自定义用例预调配和访问工作负载集群，并定义自定义集群配置。有关更多信息，请参阅 Cloudera 网站。
- 数据流和流处理解决了企业在动态数据方面面临的主要挑战。它将管理以下内容：
  - 处理大容量、高规模的实时数据流
  - 跟踪流数据的数据来源和沿袭
  - 管理和监控边缘应用程序与流媒体源

  欲了解更多信息，请参阅 Cloudera 网站上的 Cloudera DataFlow 和 CSP。
- 数据工程包括数据集成、数据质量和数据治理，帮助组织构建和维护数据管线和工作流。有关更多信息，请参阅 Cloudera 网站。了解对竞价型实例的支持，以便于在 AWS 上为 Cloudera Data Engineering 工作负载节省成本。
- Data Warehouse 使您能够创建独立的数据仓库和数据集市，这些数据仓库和数据集市可以自动扩缩以满足工作负载需求。该服务为每个数据仓库和数据集市提供隔离的计算实例和自动优化，可帮助您在开会时节省成本 SLAs。有关更多信息，请参阅 Cloudera 网站。了解如何对 Cloudera Data Warehouse on AWS 管理成本和自动扩缩。
- CDP 中的操作数据库为可扩展、高性能应用程序提供了可靠而灵活的基础。它提供实时、始终可用、可扩展的数据库，在统一的运营和仓储平台内提供传统的结构化数据以及新的非结构化数据。有关更多信息，请参阅 Cloudera 网站。
- 机器学习是一个云原生机器学习平台，它将自助数据科学和数据工程功能合并到企业数据云中的单一便携式服务中。它支持在任何地方的数据上可扩展地部署机器学习和人工智能（AI）。有关更多信息，请参阅 Cloudera 网站。

CDP on AWS

下图（经 Cloudera 网站许可改编）显示了 CDP on AWS 的高级架构。CDP 实施自有安全模型来管理账户和数据流。这些通过使用跨账户角色与 IAM 集成。

CDP 控制面板位于自己的 VPC 中的 Cloudera 主账户中。每个客户账户都有自己的子账户和唯一 VPC。跨账户 IAM 角色和 SSL 技术将控制面板之间的管理流量路由到位于每个客户 VPC 内可通过互联网路由的公有子网上的客户服务。在客户的 VPC 上，Cloudera Shared Data Experience (SDX) 通过统一的治理和合规性提供企业级安全性，因此您可以更快地从数据中获得见解。SDX 是融入所有 Cloudera 产品的设计理念。有关 SDX 和 AWS 的 CDP 公有云网络架构的更多信息，请参阅 Cloudera 文档。

工具

Amazon Web Services

- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可帮助您在 AWS 上运行 Kubernetes，而无需安装或维护您自己的 Kubernetes 控制面板或节点。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- Amazon Relational Database Service (Amazon RDS) 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展关系数据库。
- Amazon Simple Storage Service(Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

自动化和工具

- 要获得其他工具，您可以使用 Cloudera Backup Data Recovery (BDR)、AWS Snowball 和 AWS Snowmobile，帮助将数据从本地 CDH、HDP 和 CDP 迁移到 AWS 托管的 CDP。
- 对于新的部署，我们建议您使用适用于 CDP 的 AWS 合作伙伴解决方案。

操作说明

准备迁移

Task	描述	所需技能
与 Cloudera 团队合作。	Cloudera 追求与客户的标准化参与模式，并可以与您的系统集成商 (SI) 合作推广相同的方法。联系 Cloudera 客户团队，以便他们可以提供指导和必要的技术资源来启动项目。联系 Cloudera 团队，以确保所有必要的团队能够在迁移日期临近时为迁移做好准备。	迁移主管

Task	描述	所需技能
	您可联系 Cloudera 专业服务部门，以更低的成本快速将您的 Cloudera 部署从试点转移到生产环境，同时保持最佳性能。有关产品的完整列表，请参阅 Cloudera 网站。	
在 AWS 上为您的 VPC 创建 CDP 公有云环境。	与 Cloudera Professional Services 或您的 SI 合作，规划 CDP 公共云并将其部署到 AWS 上的 VPC 中。	云架构师、Cloudera SME
确定要迁移的工作负载的优先级，并对其进行评测。	评估所有本地工作负载，以确定最容易迁移的工作负载。非关键任务的应用程序最好首先移动，因为它们对客户的影响最小。成功迁移其他工作负载后，将任务关键型工作负载保存至最后。 **Note** 临时性（CDP 数据工程）工作负载比持久（CDP 数据仓库）工作负载更容易迁移。迁移时考虑数据量与位置也很重要。挑战可能包括将数据从本地环境连续复制到云，以及更改数据摄取管线以将数据直接导入到云。	迁移主管

Task	描述	所需技能
讨论 CDH、HDP、CDP 以及遗留应用程序迁移活动。	考虑和开始使用 Cloudera Workload Manager 计划以下活动： • 要复制到您的 AWS 环境的数据和工作负载 • 云就绪数据 • 吵闹的邻居，占用资源并给其他租户带来问题 • 弹性工作负载 • 高运营开销的小型集群	迁移主管

Task	描述	所需技能
完成 Cloudera Replication Manager 的要求和建议。	与 Cloudera Professional Services 和您的 SI 合作，准备将工作负载迁移到 AWS 上的 CDP Public Cloud 环境。了解以下要求和建议可以帮助您避免安装 Replication Manager 服务期间和之后的常见问题。 • 查看 Replication Manager 支持文档，确认您符合环境和系统要求。有关更多信息，请参阅 Cloudera 网站上的 CDP 公有云 Replication Manager 支持矩阵。 • 您不需要对将安装 Replication Manager 应用程序和数据生命周期管理器 (DLM) 引擎的节点进行根访问。 • 在初始安装 Replication Manager 期间安装 Apache Hive，除非您确定将来不会使用 Hive 复制。如果您决定在 Replication Manager 中创建 HDFS 复制策略后安装 Hive，则必须在添加 Hive 后删除所有 HDFS 复制策略，然后重新创建。 • Replication Manager 中使用的集群必须有对称配置。复制关系中的每个集群必须在安全性 (Kerberos)、用户管理 (LDAP/AD) 和 Knox 代理方面进行完全相同的配置。Hadoop Distributed File	迁移主管

Task	描述	所需技能
	System (HDFS)、Apache Hive、Apache Knox、Apache Ranger 和 Apache Atlas 等集群服务可以采用不同的配置来实现高可用性 (HA)。例如，源集群和目标集群可能具有单独的可用性和非可用性配置。	

将 CDP 迁移到 AWS

Task	描述	所需技能
使用 Cloudera Workload Manager 迁移开发/测试环境的第一个工作负载。	您的 SI 可以帮助您将第一个工作负载迁移到 AWS Cloud。这应该是一个不面向客户或任务关键型作业的应用程序。开发/测试迁移的理想候选者是具有云可以轻松摄取的数据的应用程序，例如 CDP 数据工程工作负载。这是一种瞬态工作负载，访问它的用户通常较少，而持久工作负载（例如 CDP 数据仓库工作负载）可能有许多需要不间断访问的用户。数据工程工作负载不是持久的，这可以最大限度地减少出现问题时对业务的影响。然而，这些工作对于生产报告可能至关重要，因此首先优先考虑影响较小的数据工程工作负载。	迁移主管
根据需要重复迁移的步骤。	Cloudera Workload Manager 可以帮助识别最适合云端的工作负载。它提供了云性能评	Cloudera SME

Task	描述	所需技能
	级、目标环境的大小/容量计划以及复制计划等指标。迁移的最佳选择是季节性工作负载、临时报告和不消耗大量资源的间歇性工作。 Cloudera Replication Manager 将数据从本地移动至云端，以及从云端移动到本地。 使用工作负载管理器主动优化数据仓库、数据工程和机器学习的工作负载、应用程序、性能和基础设施容量。有关如何实现数据仓库现代化的完整指南，请参阅 Cloudera 网站。	

相关资源

Cloudera 文档：

- 通过 CDP、Cloudera Manager 和 Replication Manager 注册经典集群：
  - 管理控制台
  - Replication Manager Hive 复制
- Sentry 复制
- Sentry 权限
- Data Hub 集群规划清单
- Workload Manager 架构
- Replication Manager 要求
- Cloudera 数据平台可观测性
- AWS 要求

AWS 文档：

- 云数据迁移

解决将 Microsoft SQL Server 迁移至 Amazon Web Services Cloud 后出现的连接错误

由 Premkumar Chelladurai (AWS) 编写

摘要

将在 Windows Server 2008 R2、2012 或 2012 R2 上运行的微软 SQL Server 迁移到亚马逊网络服务 (AWS EC2) 云上的亚马逊弹性计算云 (亚马逊) 实例后，与 SQL Server 的连接失败并出现以下错误：

- `[Microsoft][ODBC SQL Server Driver][DBNETLIB] General Network error`
- `ERROR [08S01] [Microsoft][SQL Native Client]Communication link failure. System.Data.SqlClient.SqlException: A transport-level error has occurred when sending the request to the server. (provider: TCP Provider, error: 0 - An existing connection was forcibly closed by the remote host.)`
- `TCP Provider: The semaphore timeout period has expired`

此模式描述了如何通过在 Windows Server 2008 R2、2012 或 2012 R2 上运行的 SQL Server 的操作系统 (OS) 和网络接口级别关闭 Windows 可扩展网络包 (SNP) 功能来解决这些错误。

先决条件和限制

先决条件

- Windows 服务器管理员权限。
- 如果您使用 AWS Application Migration Service 作为迁移工具，则需要以下 Windows Server 版本之一：
  - Windows Server 2008 R2 Service Pack 1、2012 或者 2012 R2
- 如果您使用 CloudEndure 迁移作为迁移工具，则需要以下 Windows 服务器版本之一：
  - Windows Server 2003 R2 Service Pack 3、2008、2008 R2 Service Pack 1、2012 或者 2012 R2

工具

- 亚马逊 EC2 — 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要使用 Amazon EC2 启动任意数量或数量的虚拟服务器，也可以进行横向扩展或扩展。
- Windows Server – Windows Server 是一个用于构建由互联应用程序、网络和 Web 服务组成的基础设施的平台。

操作说明

在操作系统和弹性网络接口级别关闭 SNP 功能

Task	描述	所需技能
在操作系统级关闭 SNP 功能。	1. 以管理员身份登录 Windows Server，并打开命令提示符。 2. 运行 `netsh int tcp show global`命令。 3. 在输出中，检查 `Receive-Side Scaling` 或 `Chimney Offload` 是否处于 `enabled` 模式中。如果其中的任何一个为 `enabled`，请运行以下命令： • `netsh int tcp set global chimney=disabled` • `netsh int tcp set global rss=disabled`	AWS 管理员、AWS 系统管理员、迁移工程师、云管理员
在弹性网络接口级别关闭 SNP 功能。	1. 选择开始，输入 `ncpa.cpl`，然后按 Enter。 2. 右键单击弹性网络适配器。 3. 在弹出式菜单中，选择属性。 4. 在以太网适配器属性窗口，选择配置。 5. 在 Amazon 弹性网络适配器属性弹出窗口中，选择 Advanced（高级）选项卡。	AWS 管理员、云管理员、AWS 系统管理员

Task	描述	所需技能
	6. 在属性部分，关闭所有卸载和 RSS。	

相关资源

- 如何对 RSS 和 NetDMA 等高级网络性能功能进行故障排除

重启 RHEL 源服务器 SELinux 后，无需禁用 AWS 复制代理即可自动重启 AWS 复制代理

由 Anil Kunapareddy (AWS)、Shanmugam Shanker (AWS) 和 Venkatramana Chintha (AWS) 编写

摘要

AWS Application Migration Service 可帮助简化、加快和自动将您的 Red Hat Enterprise Linux (RHEL) 工作负载迁移至 Amazon Web Services (AWS) Cloud。要将源服务器添加到 Application Migration Service，您可以在服务器上安装 AWS Replication Agent。

Application Migration Service 提供实时、异步、块级复制。这意味着您可以在整个复制过程中继续正常的 IT 操作。 这些 IT 操作可能需要您在迁移期间重新引导或重新启动 RHEL 源服务器。如果发生这种情况，AWS Replication Agent 将不会自动重启，您的数据复制也将停止。通常，您可以将安全增强型 Linux (SELinux) 设置为禁用或允许模式以自动重启 AWS Replication Agent。但是，贵组织的安全策略可能禁止禁用 SELinux，您可能还必须重新标记文件。

此模式描述了如何在迁移期间 RHEL 源服务器重启或重新启动 SELinux 时自动重启 AWS Replication Agent 而无需关闭。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 您想要迁移到 Amazon Web Services Cloud 的本地 RHEL 工作负载。
- Application Migration Service 已从 Application Migration Service 控制台初始化。仅首次使用此服务时才需要初始化。有关说明，请参阅 Application Migration Service 文档。
- 适用于 Application Migration Service 的现有 AWS Identity and Access Management (IAM) policy。有关更多信息，请参阅 Application Migration Service 文档。

版本

- RHEL 版本 7 或更高版本

工具

Amazon Web Services

- AWS 应用程序迁移服务是一种高度自动化 lift-and-shift（重新托管）的解决方案，可简化、加快应用程序迁移到 AWS 并降低其成本。

Linux 命令

下表列出了您将在 RHEL 源服务器上运行的 Linux 命令。在此模式的操作说明和故事中也有描述。

命令	描述
`#systemctl –version`	标识系统版本。
`#systemctl list-units --type=service`	列出 RHEL 服务器上所有可用的活动服务。
`#systemctl list-units --type=service | grep running`	列出当前在 RHEL 服务器上运行的所有服务。
`#systemctl list-units --type=service | grep failed`	列出 RHEL 服务器重新引导或重新启动后加载失败的所有服务。
`restorecon -Rv /etc/rc.d/init.d/aws-replication-service`	将上下文更改为 `aws-replication-service` 。
`yum install policycoreutils*`	安装 SELinux 系统运行所需的策略核心实用程序。
`ausearch -c "insmod" --raw | audit2allow -M my-modprobe`	搜索审核日志并创建策略模块。
`semodule -i my-modprobe.pp`	激活策略。
`cat my-modprobe.te`	显示 `my-modprobe.te` 文件的内容。
`semodule -l | grep my-modprobe`	检查策略是否已加载到 SELinux 模块。

操作说明

安装 AWS Replication Agent 并重启 RHEL 源服务器

Task	描述	所需技能
创建具有访问密钥与秘密访问密钥的 Application Service 用户。	要安装 AWS Replication Agent，您必须使用所需 AWS 凭证创建 Application Migration Service 用户。有关说明，请参阅 Application Migration Service 文档。	迁移工程师
安装 AWS Replication Agent。	1. 登录 AWS 管理控制台并在https://console.aws.amazon.com/mgn/家中打开 AWS Migration Service 控制台。 2. 按照 Application Migration Service 文档中的说明配置复制设置。 3. 按照 Application Migration Service 文档中的说明安装 AWS Replication Agent。 4. 在源服务器页面，选择 RHEL 源服务器，然后选择复制以开始初始复制。有关更多信息，请参阅 Application Migration Service 文档。	迁移工程师
重启 RHEL 源服务器。	当 RHEL 源服务器的数据复制状态在 “迁移” 仪表板上显示为 “已停止” 时，请重新启动或重新启动 RHEL 源服务器。	迁移工程师

Task	描述	所需技能
检查数据复制状态。	等待一小时，然后在迁移控制面板上再次检查数据复制状态。它应该处于 “健康” 状态。	迁移工程师

检查 RHEL 源服务器上的 AWS Replication Agent 状态

Task	描述	所需技能
确定系统版本。	打开 RHEL 源服务器命令行界面，然后运行以下命令来识别系统版本： `#systemctl –version`	迁移工程师
列出所有活跃服务。	要列出 RHEL 服务器上可用的所有活动服务，请运行以下命令： `#systemctl list-units --type=service`	迁移工程师
列出所有正在运行的服务。	要列出 RHEL 服务器上当前运行的所有服务，请使用以下命令： `#systemctl list-units --type=service | grep running`	迁移工程师
列出所有加载失败服务。	要列出 RHEL 服务器重新引导或重新启动后加载失败的所有服务，请运行以下命令： `#systemctl list-units --type=service | grep failed`	迁移工程师

创建并运行 SELinux 模块

Task	描述	所需技能
更改安全上下文。	在 RHEL 源服务器的命令行界面中，运行以下命令以将安全上下文更改为 AWS 复制服务： `restorecon -Rv /etc/rc.d/init.d/aws-replication-service`	迁移工程师
安装核心实用程序。	要安装 SELinux 系统及其策略运行所需的核心实用程序，请运行以下命令： `yum install policycoreutils*`	迁移工程师
搜索审核日志并创建策略模块。	运行命令： `ausearch -c "insmod" --raw | audit2allow -M my-modprobe`	迁移工程师
显示 my-modprobe-te文件内容。	该 `my-modprobe.te` 文件由 audit2allow 命令生成。它包括 SELinux 域、策略源目录和子目录，并指定与这些域相关的访问向量规则和过渡。运行以下命令以显示文件的内容： `cat my modprobe.te`	迁移工程师
激活策略。	若要插入模块并激活策略包，请运行以下命令：	迁移工程师

Task	描述	所需技能
	`semodule -i my-modprobe.pp`	
检查模块是否已加载。	运行命令： `semodule -l | grep my-modprobe` 加载 SELinux 模块后，迁移期间无需再设置 SELinux 为禁用模式或允许模式。	迁移工程师
重启 RHEL 源服务器并验证数据复制状态。	打开 AWS Migration Service 控制台，导航至数据复制进度，然后重启您的 RHEL 源服务器。现在，数据复制应在 RHEL 源服务器重新启动后自动恢复。	迁移工程师

相关资源

- Application Migration Service 文档
- 技术培训材料
- 对 AWS Replication Agent 问题进行故障排除
- Application Migration Service 策略

重构

主题

- 将 Oracle 的 VARCHAR2 (1) 种数据类型转换为亚马逊 Aurora PostgreSQL 的布尔数据类型
- 在 Aurora PostgreSQL 兼容中创建应用程序用户和角色
- 通过使用兼容 PostgreSQL 的 Aurora 全局数据库来模拟 Oracle 灾难恢复
- 从 SQL Server 迁移到 PostgreSQL 时，对 PII 数据实施 SHA1 哈希处理
- 使用 Oracle SQL Developer 和 AWS SCT 以增量方式从 Amazon RDS for Oracle 迁移至 Amazon RDS for PostgreSQL
- 使用兼容 Aurora PostgreSQL 的文件编码将 BLOB 文件加载至文本中
- 使用并使用和，将 Amazon RDS for Oracle 迁移到适用于 PostgreSQL AWS SCT 的 Amazon RDS AWS DMSAWS CLIAWS CloudFormation
- 使用 AWS DMS 在 SSL 模式下将 Amazon RDS for Oracle 迁移到 Amazon RDS for PostgreSQL
- 将 Oracle SERIALLY_REUSABLE pragma 包迁移至 PostgreSQL
- 将 Oracle 外部表迁移到 Amazon Aurora PostgreSQL-Compatible
- 将基于函数的索引从 Oracle 迁移到 PostgreSQL
- 使用扩展将 Oracle 原生函数迁移到 PostgreSQL
- 使用 AWS DMS 将 Db2 数据库从亚马逊迁移 EC2 到兼容 Aurora MySQL
- 使用 AWS DMS 将微软 SQL Server 数据库从亚马逊迁移 EC2 到亚马逊 DocumentDB
- 将本地 ThoughtSpot Falcon 数据库迁移到亚马逊 Redshift
- 使用 AWS DMS 将 Oracle 数据库迁移至 Amazon DynamoDB
- 使用 AWS DMS 将 Oracle 分区表迁移到 PostgreSQL
- 从 Amazon RDS for Oracle 迁移到 Amazon RDS for MySQL
- 使用 AWS DMS 和 AWS SCT 从亚马逊上的 IBM Db2 迁移 EC2 到兼容 Aurora PostgreSQL
- 使用和 AWS DMS 从 Oracle 8i 或 9i 迁移到适用于 SharePlex PostgreSQL 的亚马逊 RDS
- 使用实体化视图和 AWS DMS 从 Oracle 8i 或 9i 迁移至 Amazon RDS for PostgreSQL
- 使用 AWS DMS 和 AWS SCT 从亚马逊上的 Oracle 迁移 EC2 到适用于 MySQL 的亚马逊 RDS
- 使用 AWS DMS 从 Oracle 迁移至 Amazon DocumentDB
- 使用 AWS DMS 和 AWS SCT 将 Oracle 数据库从亚马逊迁移 EC2 到适用于 MariaDB 的亚马逊 RDS
- 使用 AWS DMS 和 AWS SCT 将本地 Oracle 数据库迁移至 Amazon RDS for MySQL

- 使用 Oracle Bystander 和 AWS DMS 将本地 Oracle 数据库迁移到 Amazon RDS for PostgreSQL
- 使用 Oracle 从 Oracle 数据库迁移到 Amazon RDS for PostgreSQL GoldenGate
- 使用 AWS DMS 和 AWS SCT 将 Oracle 数据库迁移至 Amazon Redshift
- 使用 AWS DMS 和 AWS SCT 将 Oracle 数据库迁移至 Aurora PostgreSQL
- 将数据从本地 Oracle 数据库迁移到 Aurora PostgreSQL
- 使用 AWS DMS 从 SAP ASE 迁移至 Amazon RDS for SQL Server
- 使用 AWS DMS 将本地 Microsoft SQL Server 数据库迁移至 Amazon Redshift
- 使用 AWS SCT 数据提取代理将本地 Microsoft SQL Server 数据库迁移至 Amazon Redshift
- 使用 AWS SCT 数据提取代理将 Teradata 数据库迁移到 Amazon Redshift
- 使用 AWS SCT 数据提取代理将本地 Vertica 数据库迁移至 Amazon Redshift
- 将遗留应用程序从 Oracle Pro*C 迁移到 ECPG
- 将虚拟生成的列从 Oracle 迁移至 PostgreSQL
- 在 Aurora PostgreSQL-Compatible 上设置 Oracle UTL_FILE 功能
- 从 Oracle 迁移至 Amazon Aurora PostgreSQL 后验证数据库对象

将 Oracle 的 VARCHAR2 (1) 种数据类型转换为亚马逊 Aurora PostgreSQL 的布尔数据类型

由 Naresh Damera (AWS) 编写

摘要

在从适用于 Oracle 的亚马逊关系数据库服务 (Amazon RDS) 迁移到兼容 Amazon Aurora PostgreSQL 的版本期间，在 () 中验证迁移时可能会遇到数据不匹配的问题。 AWS Database Migration Service AWS DMS为防止这种不匹配，可以将 VARCHAR2 (1) 数据类型转换为布尔数据类型。

VARCHAR2 数据类型存储可变长度的文本字符串， VARCHAR2(1) 表示该字符串的长度为 1 个字符或 1 个字节。有关更多信息 VARCHAR2，请参阅 Oracle 内置数据类型（Oracle 文档）。

在此模式中，在样本源数据表列中， VARCHAR2(1) 数据要么是 Y（表示 “是”），要么是 N（表示 “否”）。 此模式包括使用 AWS DMS 和 AWS Schema Conversion Tool (AWS SCT) 将此数据类型从 VARCHAR2 (1) 中的 Y 和 N 值转换为布尔值中的真值或假值的说明。

目标受众

建议那些有使用将 Oracle 数据库迁移到兼容 Aurora PostgreSQL 的经验的人使用这种模式。 AWS DMS完成迁移后，请遵守将 Oracle 转换为 Amazon RDS for PostgreSQL 或 Amazon Aurora PostgreSQL（文档）中的建议。AWS SCT

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- 确认您的环境已为 Aurora 做好准备，包括设置凭证、权限和安全组。有关更多信息，请参阅设置 Amazon Aurora 环境（Aurora 文档）。
- 一个源 Amazon RDS for Oracle 数据库，其中包含一个包含 VARCHAR2 (1) 个数据的表列。
- 目标 Amazon Aurora PostgreSQL-Compatible 数据库实例。有关更多信息，请参阅 创建数据库集群并连接到 Aurora PostgreSQL 数据库集群上的数据库（Aurora 文档）。

产品版本

- Amazon RDS for Oracle 版本 12.1.0.2 或更高版本。

- AWS DMS 版本 3.1.4 或更高版本。有关更多信息，请参阅使用 Oracle 数据库作为源 AWS DMS和使用 PostgreSQL 数据库作为目标 AWS DMS（文档）。AWS DMS 我们建议您使用最新版本的， AWS DMS 以获得最全面的版本和功能支持。
- AWS Schema Conversion Tool (AWS SCT) 版本 1.0.632 或更高版本。我们建议您使用最新版本的， AWS SCT 以获得最全面的版本和功能支持。
- Aurora 支持 PostgreSQL-Compatibl 的数据库引擎版本 列出的 PostgreSQL 版本（Aurora 文档）。

架构

源技术堆栈

Amazon RDS for Oracle 数据库实例

目标技术堆栈

Amazon Aurora PostgreSQL-Compatible 数据库实例

源架构和目标架构

工具

AWS 服务

- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩展 PostgreSQL 部署。
- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到云和本地设置的组合中，AWS Cloud 或者迁移到云端和本地设置的组合之间。
- 适用于 Oracle 的 Amazon Relational Database Service（Amazon RDS）可帮助您在中设置、操作和扩展 Oracle 关系数据库 AWS Cloud。
- AWS Schema Conversion Tool (AWS SCT) 通过自动将源数据库架构和大部分自定义代码转换为与目标数据库兼容的格式来支持异构数据库迁移。

其他服务

- Oracle SQL Developer 是一个集成的开发环境，可简化传统部署和基于云的部署中 Oracle 数据库的开发和管理。在这种模式中，您可以使用此工具连接至 Amazon RDS for Oracle 数据库实例并查询数据。

- pgAdmin 是一种适用于 PostgreSQL 的开源管理工具。它提供了一个图形界面，可帮助您创建、维护和使用数据库对象。在这种示例中，您可使用此工具连接到 Aurora 数据库实例并查询数据。

操作说明

准备迁移

Task	描述	所需技能
创建数据库迁移报告。	1. 在中 AWS SCT，创建数据库迁移评估报告。有关更多信息，请参阅创建迁移评测报告。 2. 查看和执行迁移评测报告中的操作项目。有关更多信息，请参阅评测报告操作项目。	数据库管理员、开发人员
在目标数据库上禁用外键约束。	在 PostgreSQL 中，外键通过使用触发器实现。在满载阶段，每次 AWS DMS 加载一个表。强烈建议您在完全加载期间使用以下方法之一禁用外键约束： • 从实例中临时禁用所有触发器并完成完全加载。 • 在 PostgreSQL 中使用 `session_replication_role` 参数。 如果禁用外键约束不可行，请为父表和子表特定的主数据创建 AWS DMS 迁移任务。	数据库管理员、开发人员

Task	描述	所需技能

禁用目标数据库的主键与唯一键。	使用以下命令禁用目标数据库的主键和约束。这有助于提高初始加载任务表现。
	`ALTER TABLE <table> DISABLE PRIMARY KEY;`
	`ALTER TABLE <table> DISABLE CONSTRAINT <constraint_name>;`
		数据库管理员、开发人员

创建初始加载任务。	在中 AWS DMS，为初始加载创建迁移任务。有关说明，请参阅创建任务。对于迁移方法，请选择迁移现有数据。此迁移方法在 API 中调用Full Load。暂时不要启动此任务。	数据库管理员、开发人员

Task	描述	所需技能
编辑初始加载任务设置。	编辑任务设置，以添加数据验证。这些验证设置必须在 JSON 文件内创建。有关说明和示例，请参见指定任务设置。添加以下验证： • 要验证目标数据库中的 VARCHAR2 (1) 数据是否准确地转换为布尔值，请在此模式的 “其他信息” 部分的 “数据验证脚本” 中添加代码。验证脚本将目标表中的布尔值 1 转换为 Y，将 0 转换为 N，然后将目标表值与源表中的值进行比较。 若要验证数据迁移的其余部分，请在任务中启用数据验证。有关更多信息，请参阅数据验证任务设置。	AWS 管理员，数据库管理员
创建持续复制任务。	在中 AWS DMS，创建迁移任务，使目标数据库与源数据库保持同步。有关说明，请参阅创建任务。对于迁移方法，请选择仅复制数据更改。暂时不要启动此任务。	数据库管理员

测试迁移任务

Task	描述	所需技能
创建用于测试的样本数据。	在源数据库中，创建包含用于测试目的的数据的示例表。	开发人员

Task	描述	所需技能
确认没有冲突活动。	使用 `pg_stat_activity` 检查服务器上是否存在任何可能影响迁移的活动。有关更多信息，请参阅统计数据收集器（PostgreSQL 文档）。	AWS 管理员
启动 AWS DMS 迁移任务。	在 AWS DMS 控制台的 “控制台” 页面上，启动您在上一篇长篇故事中创建的初始加载和正在进行的复制任务。	AWS 管理员
监控任务和表加载状态。	在迁移过程中，监控任务状态和表状态。初始加载任务完成后，在表统计选项卡上： • 加载状态应为表已完成。 • 验证状态应为已验证。	AWS 管理员
验证迁移结果。	使用 pgAdmin 在目标数据库上查询表。查询成功表示数据已成功迁移。	开发人员
在目标数据库添加主键和外键。	在目标数据库创建主键和外键。有关更多信息，请参阅更改表（PostgreSQL 网站）。	数据库管理员
清理测试数据。	在源数据库和目标数据库，清理为单元测试创建的数据。	开发人员

割接

Task	描述	所需技能
完成迁移。	重复前面的操作说明，使用真实的源数据测试迁移任务。将	开发人员

Task	描述	所需技能
	数据从源数据库迁移至目标数据库。	
验证源数据库和目标数据库是否同步。	验证源数据库和目标数据库是否同步。有关更多信息和说明，请参阅AWS DMS 数据验证。	开发人员
停止源数据库。	停止 Amazon RDS for Oracle 数据库。有关说明，请参阅暂时停止 Amazon RDS 数据库实例。停止源数据库后，中的 AWS DMS 初始加载和正在进行的复制任务将自动停止。无需执行其他操作即可停止上述任务。	开发人员

相关资源

AWS 参考文献

- 使用和（规范性指南）将 Oracle 数据库迁移到 Aurora PostgreSQ AWS DMS L AWS SCTAWS
- 将 Oracle 转换为适用于 PostgreSQL 的亚马逊 RDS 或亚马逊 Aurora PostgreSQL（文档）AWS SCT
- AWS DMS 工作原理（AWS DMS 文档）

其他参考资料

- 布尔数据类型（PostgreSQL 文档）
- Oracle 内置数据类型（Oracle文档）
- pgAdmin（pgAdmin 网站）
- SQL Developer（Oracle网站）

教程和视频

- 入门 AWS DMS
- Amazon RDS 入门
- AWS DMS（视频）简介
- 了解 Amazon RDS（视频）

其他信息

数据验证脚本

以下数据验证脚本将 1 转换为 Y，将 0 转换为 N。 这有助于 AWS DMS 任务成功完成并通过表验证。

```
{
"rule-type": "validation",
"rule-id": "5",
"rule-name": "5",
"rule-target": "column",
"object-locator": {
"schema-name": "ADMIN",
"table-name": "TEMP_CHRA_BOOL",
"column-name": "GRADE"
},
"rule-action": "override-validation-function",
"target-function": "case grade when '1' then 'Y' else 'N' end"
        }
```

脚本中的 `case` 语句执行验证。如果验证失败，则在目标数据库实例上的 public.awsdms_validation_failures_v1 表中 AWS DMS 插入一条记录。此记录包含表名、错误时间，以及源表和目标表中不匹配值的详细信息。

如果您没有将此数据验证脚本添加到 AWS DMS 任务中，并且数据已插入目标表中，则该 AWS DMS 任务会将验证状态显示为 “不匹配的记录”。

在 AWS SCT 转换过程中， AWS DMS 迁移任务将 VARCHAR2 (1) 数据类型的数据类型更改为布尔值，并在"NO"列上添加主键约束。

在 Aurora PostgreSQL 兼容中创建应用程序用户和角色

由 Abhishek Verma (AWS) 创建

摘要

迁移到 Amazon Aurora PostgreSQL 兼容版时，必须在 Aurora PostgreSQL 兼容数据库中创建源数据库上存在的数据库用户和角色。您可以使用两种不同的方法在 Aurora PostgreSQL 兼容中创建用户和角色：

- 在目标数据库中使用与源数据库中类似的用户和角色。在这种方法中，从源数据库中提取用户和角色的数据定义语言 (DDLs)。然后，它们将被转换并应用于目标 Aurora PostgreSQL 兼容数据库。例如，博客文章使用 SQL 将用户、角色和授权从 Oracle 映射到 PostgreSQL中介绍了如何使用从 Oracle 源数据库引擎中提取。
- 使用在开发、管理以及在数据库中执行其他相关操作时常用的标准化用户和角色。这包括由相应用户执行的只读、读/写、开发、管理和部署操作。

此模式包含在标准化用户和角色方法所需 Aurora PostgreSQL 兼容中创建用户和角色所需授权。用户和角色创建步骤与向数据库用户授予最低权限的安全策略保持一致。下表列出了用户、其对应的角色及其在数据库上的详细信息。

Users	角色	目的
APP_read	APP_RO	用于对架构 APP 的只读访问
APP_WRITE	APP_RW	用于对架构 APP 的写入和读取操作
APP_dev_user	APP_DEV	用于架构 APP_DEV 上的开发目的，对架构 APP 具有只读访问权限
Admin_User	rds_superuser	用于对数据库执行管理员操作
APP	APP_DEP	用于在 APP 架构下创建对象，以及在 APP 架构中部署对象

先决条件和限制

先决条件

- 活跃 Amazon Web Services (AWS) account
- PostgreSQL 数据库、Amazon Aurora PostgreSQL 兼容版数据库或适用于 PostgreSQL 的 Amazon Relational Database Service（Amazon RDS）数据库

产品版本

- 所有 PostgreSQL 版本

架构

源技术堆栈

- 任何数据库

目标技术堆栈

- 兼容 Amazon Aurora PostgreSQL

目标架构

下图显示了 Aurora PostgreSQL 兼容数据库中的用户角色和架构架构。

自动化和扩缩

此模式包含用户、角色和架构创建脚本，您可以多次运行这些脚本，而不会对源数据库或目标数据库的现有用户产生任何影响。

工具

Amazon Web Services

- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩展 PostgreSQL 部署。

其他服务

- psql 是一个基于终端的前端工具，随每个 PostgreSQL 数据库安装一起安装。它有一个命令行界面，用于运行 SQL、PL-PGSQL 和操作系统命令。
- pgAdmin 是一种适用于 PostgreSQL 的开源管理工具。它提供了一个图形界面，可帮助您创建、维护和使用数据库对象。

操作说明

创建用户和角色

Task	描述	所需技能

创建部署用户。	部署用户 APP 将用于在部署期间创建和修改数据库对象。使用以下脚本在架构 APP 中创建部署用户角色 APP_DEP。验证访问权限以确保此用户仅具有在所需架构 APP 中创建对象的权限。
	1. 连接到管理员用户，并创建架构。
	`CREATE SCHEMA APP;`
	2. 创建 用户。
	`CREATE USER APP WITH PASSWORD <password> ;`
	3. 创建角色。
	`CREATE ROLE APP_DEP ; GRANT all on schema APP to APP_DEP ; GRANT USAGE ON SCHEMA APP to APP_DEP ;`
		数据库管理员

Task	描述	所需技能
	GRANT connect on database <db_name> to APP_DEP ; GRANT APP_DEP to APP; 4. 要测试权限，请连接到 APP 并创建表。 set search_path to APP; SET CREATE TABLE test(id integer) ; CREATE TABLE 5. 检查权限。 select schemaname , tablename , tableowne r from pg_tables where tablename like 'test' ; schemaname | tablename | tableowner APP | test | APP	

Task	描述	所需技能

创建只读用户。	只读用户 APP_read 将用于在架构 APP 中执行只读操作。使用以下脚本创建只读用户。验证访问权限以确保此用户仅具有读取架构 APP 中的对象的权限，并自动授予对架构 APP 中创建的任何新对象的读取访问权限。
	
	1. 创建用户 APP_read。
	
	`create user APP_read ;`
	`alter user APP_read`
	`with password`
	`'your_password' ;`
	
	2. 创建角色。
	
	`CREATE ROLE APP_ro ;`
	`GRANT SELECT ON ALL`
	`TABLES IN SCHEMA APP`
	`TO APP_RO ;`
	`GRANT USAGE ON SCHEMA`
	`APP TO APP_RO`
	`GRANT CONNECT ON`
	`DATABASE testdb TO`
	`APP_RO ;`
	`GRANT APP_RO TO`
	`APP_read;`
	
	3. 要测试权限，请使用 APP_read 用户登录。
	
	`set search_path to`
	`APP ;`
	`create table test1( id`
	`integer) ;`
	`ERROR: permission`
	`denied for schema APP`
		数据库管理员

Task	描述	所需技能
	``` LINE 1: create table test1(id integer) ; insert into test values (34) ; ERROR: permission denied for table test SQL state: 42501 select from test no rows selected ```	

Task	描述	所需技能

创建读/写用户。	读/写用户 APP_WRITE 将用于对架构 APP 执行读写操作。使用以下脚本创建读/写用户并授予其 APP_RW 角色。验证访问权限以确保此用户仅对架构 APP 中的对象具有读写权限，并自动授予对架构APP中创建的任何新对象的读取和写入访问权限。
	
	1. 创建 用户。
	
	`CREATE USER APP_WRITE ;`
	`alter user APP_WRITE with password 'your_password' ;`
	
	2. 创建角色。
	
	`CREATE ROLE APP_RW;`
	`GRANT SELECT, INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA APP TO APP_RW ;`
	`GRANT CONNECT ON DATABASE postgres to APP_RW ;`
	`GRANT USAGE ON SCHEMA APP to APP_RW ;`
	`ALTER DEFAULT PRIVILEGES IN SCHEMA APP`
	`GRANT SELECT, INSERT, UPDATE, DELETE ON TABLES TO APP_RW ;`
	`GRANT APP_RW to APP_WRITE`	

Task	描述	所需技能

	3. 要测试权限，请使用 APP_WRITE 用户登录。
	
	`SET SEARCH_PATH to APP;`
	`CREATE TABLE test1( id integer) ;`
	`ERROR: permission denied for schema APP`
	`LINE 1: create table test1( id integer) ;`
	`SELECT * FROM test ;`
	`id`
	`----`
	`12`
	`INSERT INTO test values (31) ;`
	`INSERT 0 1`	

Task	描述	所需技能

创建管理员用户。	管理员用户 Admin_User 将用于对数据库执行管理操作。这些操作的示例包括 CREATE ROLE 和 CREATE DATABASE。Admin_User 使用内置角色 rds_superuser 对数据库执行管理操作。使用以下脚本在数据库中创建并测试管理员用户 Admin_User 的权限。
	
	1. 创建用户并授予角色。
	
	`create user Admin_Use r WITH PASSWORD ‘Your password’ ALTER user Admin_user CREATEDB; ALTER user Admin_user CREATEROLE;`
	
	2. 要测试权限，请从 Admin_User 用户登录。
	
	`SELECT * FROM APP.test ; id ---- 31 CREATE ROLE TEST ; CREATE DATABASE test123 ;`
		数据库管理员

Task	描述	所需技能

创建开发用户。	开发用户 APP_dev_user 将有权在其本地架构 APP_DEV 中创建对象，并有权在架构 APP 中读取访问权限。使用以下脚本在数据库中创建和测试用户 APP_dev_user 的权限。
	1. 创建 用户。
	`CREATE USER APP1_dev_user with password ‘your password’;`
	2. 为 App_dev_user 创建 APP_DEV 架构。
	`CREATE SCHEMA APP1_DEV ;`
	3. 创建 APP_DEV角色。
	`CREATE ROLE APP1_DEV ;`
	`GRANT APP1_RO to APP1_DEV ;`
	`GRANT SELECT ON ALL TABLES IN SCHEMA APP1_DEV to APP1_dev_user`
	`GRANT USAGE, CREATE ON SCHEMA APP1_DEV to APP1_DEV_USER`
	`GRANT APP1_DEV to APP1_DEV_USER ;`
	4. 要测试权限，请从 APP_dev_user 登录。
		数据库管理员

Task	描述	所需技能

	`CREATE TABLE APP1_dev.test1( id integer ) ;`
	`CREATE TABLE`
	`INSERT into APP1_dev.test1 ( select * from APP1.test );`
	`INSERT 0 1`
	`CREATE TABLE APP1.test4 ( id int) ;`
	`ERROR: permissio n denied for schema APP1`
	`LINE 1: create table APP1.test4 ( id int) ;`	

相关资源

PostgreSQL 文档

- CREATE ROLE
- CREATE USER
- 预定义路由

其他信息

PostgreSQL 14 增强功能

PostgreSQL 14 提供了一组预定义的角色，这些角色允许访问某些常用的特权功能和信息。管理员（包括具有 CREATE ROLE 权限的角色）可以将这些角色或其环境中的其他角色授予用户，从而为他们提供对指定功能和信息的访问权限。

管理员可以使用 GRANT 命令授予用户访问这些角色的权限。例如，要向 Admin_User 授予 pg_signal_backend 角色，可以运行以下命令。

```
GRANT pg_signal_backend TO Admin_User;
```

pg_signal_backend 角色旨在允许管理员启用受信任的非超级用户角色向其他后端发送信号。有关更多信息，请参阅 PostgreSQL 14 增强功能。

微调访问

在某些情况下，可能需要为用户提供更精细的访问（例如，基于表的访问或基于列的访问）。在这种情况下，可以创建其他角色来向用户授予这些权限。有关更多信息，请参阅 PostgreSQL 授予。

通过使用兼容 PostgreSQL 的 Aurora 全局数据库来模拟 Oracle 灾难恢复

由 HariKrishna Boorgadda (AWS) 创建

摘要

企业灾难恢复 (DR) 的最佳实践基本上包括设计和实施容错硬件和软件系统，这些系统可以在灾难中幸存下来（业务连续性）并恢复正常运营（业务恢复），干预最少，理想情况下不会丢失数据。构建容错环境以满足企业灾难恢复目标，这可能既昂贵又耗时，并且需要企业的坚定承诺。

Oracle Database 提供了三种不同的灾难恢复方法，与任何其他保护 Oracle 数据的方法相比，这些方法可提供最高级别的数据保护和可用性。

- Oracle 零数据丢失恢复设备
- Oracle Active Data Guard
- 甲骨文 GoldenGate

这种模式提供了一种使用 Amazon Aurora 全球数据库模拟 Oracle GoldenGate 灾难恢复的方法。参考架构使用 Oracle 在三个 AWS 区域 GoldenGate 进行灾难恢复。该模式将源架构重塑为基于 Amazon Aurora PostgreSQL–Compatible Edition 的云原生 Aurora 全局数据库。

Aurora 全局数据库专为遍布全球的应用程序而设计。一个 Aurora 数据库跨越多个 AWS 区域以及多达五个辅助区域。Aurora 全局数据库提供以下功能：

- 物理存储级复制
- 低延迟全局读取
- 从区域范围内的中断中快速灾难恢复
- 快速跨区域迁移
- 跨区域复制延迟低
- Little-to-no 对数据库性能的影响

有关 Aurora 全局数据库功能和优势的更多信息，请参阅使用 Amazon Aurora 全局数据库。有关计划外和托管失效转移的更多信息，请参阅在 Amazon Aurora Global Database 中使用失效转移。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account

- 用于应用程序连接 Java 数据库连接 (JDBC) PostgreSQL 驱动程序
- Aurora 全局数据库基于 Amazon Aurora PostgreSQL-Compatible Edition
- 基于 Aurora PostgreSQL-Compatible 的 Oracle Real Application Clusters (RAC) 数据库迁移到 Aurora 全局数据库

Aurora 全局数据库的限制

- Aurora 全局数据库不适用于所有 AWS 区域。有关支持的区域列表，请参阅 含 Aurora PostgreSQL 的 Aurora 全局数据库。
- 有关不支持的功能和 Aurora 全局数据库的其他限制的信息，请参阅 Amazon Aurora 全局数据库的限制。

产品版本

- Amazon Aurora PostgreSQL–Compatible Edition 10.14 或更高版本

架构

源技术堆栈

- Oracle RAC 四节点数据库
- 甲骨文 GoldenGate

源架构

下图显示了使用 Oracle 在不同的 AWS 区域中使用四节点 Oracle RAC 复制的三个集群。 GoldenGate

目标技术堆栈

- 基于 Aurora PostgreSQL–Compatible 的三集群 Amazon Aurora Global Database，其中一个集群位于主区域，两个集群位于不同的辅助区域

目标架构

工具

Amazon Web Services

- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩展 PostgreSQL 部署。
- Amazon Aurora 全局数据库跨越多个 AWS 区域，可实现低延迟的全局读取，并可从可能影响整个 AWS 区域 的罕见停机事件中快速恢复。

操作说明

通过读取器数据库实例添加区域

Task	描述	所需技能
连接一个或多个 Aurora 辅助集群。	在 AWS 管理控制台 上，选择 Amazon Aurora。选择主集群，选择操作，然后从下拉列表选择添加区域。	数据库管理员
选择实例类。	您可更改辅助集群的实例类。但是我们建议将其与主集群实例类保持相同。	数据库管理员
添加第三个区域。	重复此操作说明中的步骤，在第三个区域中添加集群。	数据库管理员

对 Aurora 全局数据库进行失效转移

Task	描述	所需技能
从 Aurora 全局数据库删除主集群。	1. 在“数据库”页面上，选择主集群。 2. 选择从全局数据库删除以失效转移到辅助集群。	数据库管理员

Task	描述	所需技能
重新配置应用程序，以使写入流量转向新提升的集群。	使用新升级的集群的端点修改应用程序中的端点。	数据库管理员
停止向不可用的集群发出任何写操作。	停止您删除的集群的应用程序和任何数据操作语言 (DML) 活动。	数据库管理员
创建一个新的 Aurora 全局数据库。	现在您可以创建一个 Aurora 全局数据库，并将新提升的集群用作主集群。	数据库管理员

启动主集群

Task	描述	所需技能
从全局数据库中选择要启动的主集群。	在 Amazon Aurora 控制台的全局数据库设置，选择主集群。	数据库管理员
启动集群。	在操作下拉列表，选择开始。此过程可能需要一些时间。操作完成后，刷新屏幕以查看状态，或者在状态列中查看集群的当前状态。	数据库管理员

清理资源

Task	描述	所需技能
删除剩余辅助集群。	失效转移试点完成后，从全局数据库中删除辅助集群。	数据库管理员
删除主集群。	删除集群。	数据库管理员

相关的资源

- 使用 Amazon Aurora 全局数据库
- 使用 Amazon Aurora Global Database 的 Aurora PostgreSQL 灾难恢复解决方案（博客文章）

从 SQL Server 迁移到 PostgreSQL 时，对 PII 数据实施 SHA1 哈希处理

由 Rajkumar Raghuwanshi (AWS) 和 Jagadish Kantubugata (AWS) 创作

摘要

此模式描述了在从 SQL Server 迁移到适用于 PostgreSQL 的亚马逊 RDS 或兼容 Amazon Aurora PostgreSQL 时，如何对电子邮件地址实施安全哈希算法 1 (SHA1) 哈希。电子邮件地址是个人身份信息 (PII) 的一个示例。PII 是直接查看或与其他相关数据配对时可用于合理推断个人身份的信息。

这种模式涵盖了在不同的数据库归类和字符编码中保持一致的哈希值所面临的挑战，并提供了使用 PostgreSQL 函数和触发器的解决方案。尽管这种模式侧重于 SHA1 哈希，但它可以适应PostgreSQL 模块支持的其他哈希算法。`pgcrypto`请务必考虑哈希策略的安全影响，如果要处理敏感数据，请咨询安全专家。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 源 SQL 服务器数据库
- 目标 PostgreSQL 数据库（适用于 PostgreSQL 的亚马逊 RDS 或兼容 Aurora PostgreSQL）
- PL/pgSQL 编码专业知识

限制

- 这种模式需要根据用例更改数据库级别的排序规则。
- 尚未评估对大型数据集的性能影响。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅按地区划分的AWS 服务。有关特定终端节点，请参阅服务终端节点和配额，然后选择服务的链接。

产品版本

- Microsoft SQL Server 2012 或更高版本

架构

源技术堆栈

- SQL Server
- NET Framework。

目标技术堆栈

- PostgreSQL
- `pgcrypto`延期

自动化和扩缩

- 为了便于维护，可以考虑将哈希函数作为存储过程来实现。
- 对于大型数据集，请评估性能并考虑批处理或索引策略。

工具

AWS 服务

- 兼容 Amazon Aurora PostgreSQL 是一款完全托管、符合 ACID 标准的关系数据库引擎，可帮助您设置、操作和扩展 PostgreSQL 部署。
- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到云和本地设置的组合中，AWS Cloud 或者迁移到云端和本地设置的组合之间。
- 亚马逊关系数据库服务 Amazon RDS f or PostgreSQL 可帮助你在中设置、操作和扩展 PostgreSQL 关系数据库。 AWS Cloud
- AWS Schema Conversion Tool (AWS SCT) 通过自动将源数据库架构和大部分自定义代码转换为与目标数据库兼容的格式来支持异构数据库迁移。

其他工具

- pgAdmin 是一种适用于 PostgreSQL 的开源管理工具。它提供了一个图形界面，可帮助您创建、维护和使用数据库对象。
- SQL Server 管理工作室 (SSMS) 是一个用于管理任何 SQL 基础架构的集成环境。

最佳实践

- 使用适当的归类设置来处理目标数据库端的特殊字符。

- 使用各种电子邮件地址（包括带有非 ASCII 字符的地址）进行彻底测试。
- 保持应用程序层和数据库层之间大写和小写处理的一致性。
- 使用哈希值对查询的性能进行基准测试。

操作说明

分析源代码哈希实现

Task	描述	所需技能
查看 SQL 服务器代码。	要查看生成 SHA1 哈希值的 SQL Server 代码，请执行以下操作： • 分析现有的 SQL Server SHA1 哈希实现。 • 确定用于生成哈希值的确切方法。 • 记录输入参数和输出格式。 • 查看所有数据类型转换或转换。 • 检查归类设置及其影响。	数据工程师、数据库管理员、应用程序开发人员
记录哈希算法和数据转换。	要记录确切的哈希算法和数据转换，请执行以下操作： • 创建哈希过程的详细技术文档。 • 记录 step-by-step转换逻辑。 • 指定输入和输出格式以及数据类型。 • 包括边缘案例和特殊字符处理。	应用程序开发人员、数据工程师、数据库管理员

创建 PostgreSQL 哈希函数

Task	描述	所需技能
创建pgcrypto扩展。	要创建pgcrypto扩展，pgAdmin/psql 请使用运行以下命令： `CREATE EXTENSION pgcrypto;`	数据库管理员、数据工程师
实现一个 PostgreSQL 函数。	实现以下 PostgreSQL 函数来复制 SQL Server 的哈希逻辑。简而言之，此函数使用以下步骤： 1. （可选）将输入转换为大写。 2. 创建输入的 SHA1 哈希值。 3. 取此哈希值的最后 10 个字节（80 位）。 4. 将这些字节转换为 64 位整数。 `CREATE OR REPLACE FUNCTION utility.hex_to_bigint ( par_val character varying, par_upper character varying DEFAULT 'lower'::character varying) RETURNS bigint LANGUAGE 'plpgsql' AS $BODY$ DECLARE retnumber bigint;`	数据工程师、数据库管理员、应用程序开发人员

Task	描述	所需技能

	<pre> digest_bytes bytea;
	BEGIN
	    if lower(par_upper)
	 = 'upper'
	    then
	        digest_bytes :=
	 digest(upper(par_v
	al), 'sha1');
	    else
	        digest_bytes :=
	 digest((par_val),
	 'sha1');
	    end if;
	    retnumber := ('x'
	 || encode(substring(d
	igest_bytes, length(di
	gest_bytes)-10+1),
	 'hex'))::bit(64)::
	bigint;
	    RETURN retnumber;
	END;
	$BODY$;</pre>	

Task	描述	所需技能

测试函数。	要测试该函数，请使用来自 SQL Server 的示例数据来验证匹配的哈希值。运行以下命令：
	
	`select 'alejandro_rosalez@example.com' as Email, utility.hex_to_bigint('alejandro_rosalez@example.com', 'upper') as HashValue;`
	
	`--OUTPUT`
	`/*`
	`email hashvalue`
	`"alejandro_rosalez@example.com" 451397011176045063`
	`*/`
		应用程序开发人员、数据库管理员、数据工程师

实现自动哈希的触发器

Task	描述	所需技能

在相关表上创建触发器。	要在相关表上创建触发器以在插入或更新时自动生成哈希值，请运行以下命令：
	
	`CREATE OR REPLACE FUNCTION update_email_hash()`
	`RETURNS TRIGGER`
	`AS $$`
	`BEGIN`
		应用程序开发人员、数据工程师、数据库管理员

Task	描述	所需技能

	<pre> NEW.email_hash =
	 utility.hex_to_big
	int(NEW.email,
	 'upper');
	    RETURN NEW;
	END;
	$$ LANGUAGE plpgsql;</pre>
	<pre>CREATE TRIGGER email_has
	h_trigger BEFORE INSERT
	 OR UPDATE ON users
	 FOR EACH ROW EXECUTE
	 FUNCTION update_em
	ail_hash();</pre>	

迁移现有数据

Task	描述	所需技能
开发迁移脚本或使用 AWS DMS。	开发迁移脚本或 AWS DMS 用于填充现有数据的哈希值（包括存储在源系统BIGINT中的哈希值）。完成以下任务： • 使用哈希值为数据传输创建迁移脚本。 • 使用适当的转换规则配置 AWS DMS 任务。 • 在中设置源端点和目标端点 AWS DMS。 • 实现错误处理和日志记录机制。 • 为大型数据集设计批处理策略。	数据工程师、应用程序开发人员、数据库管理员

Task	描述	所需技能
	• 创建用于数据验证的验证查询。	
使用新的 PostgreSQL 哈希函数。	要使用新的 PostgreSQL 哈希函数来确保一致性，请执行以下操作： • 实施验证程序以验证哈希一致性。 • 在源系统和目标系统之间创建比较脚本。 • 为哈希值验证设置自动测试。 • 记录任何差异和解决步骤。	应用程序开发人员、数据库管理员、 DevOps 工程师

更新应用程序查询

Task	描述	所需技能
识别应用程序查询。	要识别使用哈希值的应用程序查询，请执行以下操作： • 使用哈希值分析应用程序代码库中的查询。 • 查看引用哈希操作的存储过程和函数。 • 记录查询性能指标和执行计划。 • 确定对基于哈希的查找的依赖关系。 • 绘制受影响的应用程序组件。	应用程序开发人员、数据库管理员、数据工程师

Task	描述	所需技能
修改查询。	如有必要，可以修改查询以使用新的 PostgreSQL 哈希函数。执行以下操作： • 重构现有查询以使用 PostgreSQL 哈希函数。 • 更新存储过程和函数。 • 实施和测试新的查询模式。 • 优化修改后的查询以提高性能。	应用程序开发人员、数据库管理员、数据工程师

测试和验证

Task	描述	所需技能
执行测试。	要对生产数据的子集进行全面测试，请执行以下操作： • 为数据子集验证创建测试计划。 • 提取具有代表性的生产数据样本。 • 使用适当的配置设置测试环境。 • 执行数据加载和转换测试。 • 执行音量和压力测试。	应用程序开发人员、数据工程师、数据库管理员
验证哈希值是否匹配。	要验证 SQL Server 和 PostgreSQL 之间的哈希值是否匹配，请执行以下操作： • 为哈希值开发比较脚本。 • 为哈希匹配创建验证报告。	应用程序开发人员、数据工程师、数据库管理员

Task	描述	所需技能
	• 实施自动验证程序。 • 记录发现的任何差异。 • 分析并解决哈希不匹配的问题。	
验证应用程序的功能。	要使用迁移的数据和新的哈希实现来验证应用程序的功能，请执行以下操作： • 执行 end-to-end应用程序测试。 • 使用哈希数据验证所有应用程序功能。 • 使用新的实现来测试应用程序性能。 • 验证 API 集成和依赖关系。	应用程序开发人员、数据库管理员、数据工程师

故障排除

事务	解决方案
哈希值不匹配。	验证源和目标之间的字符编码和排序规则。有关更多信息，请参阅在 Amazon A urora 和 Amazon RDS 上管理 PostgreSQL 中的排序规则更改（博客）。AWS

相关资源

AWS 博客

- 在 Amazon Aurora 和 Amazon RDS 上的 PostgreSQL 中管理排序规则变更
- 使用最佳实践和从现场吸取的经验教训将 SQL Server 迁移到 Amazon Aurora PostgreSQL

其他资源

- P@@ ostgreSQL pgcrypto 模块（PostgreSQL 文档）
- Po@@ stgreSQL 触发器函数（PostgreSQL 文档）
- SQL Server HASHBYTES 函数（微软文档）

使用 Oracle SQL Developer 和 AWS SCT 以增量方式从 Amazon RDS for Oracle 迁移至 Amazon RDS for PostgreSQL

由 Pinesh Singal (AWS) 编写

摘要

许多迁移策略和方法都分为多个阶段，可能持续几周到几个月不等。在此期间，由于要迁移至 PostgreSQL 数据库实例的源 Oracle 数据库实例正在进行修补或升级，您可能会遇到延迟。为避免这种情况，我们建议您将剩余 Oracle 数据库代码增量地迁移到 PostgreSQL 数据库代码。

这种模式为在初始迁移后执行了大量事务且必须迁移至 PostgreSQL 数据库的多 TB 的 Oracle 数据库实例提供了一种不停机的增量迁移策略。您可以使用这种模式的 step-by-step方法将适用于 Oracle 数据库实例的亚马逊关系数据库服务 (Amazon RDS) 逐步迁移到适用于 PostgreSQL 的 Amazon RDS 数据库实例，而无需登录亚马逊网络服务 (AWS) 管理控制台。

该模式使用 Oracle SQL Developer 查找源 Oracle 数据库中两个架构之间的区别。然后，您可以使用 AWS Schema Conversion Tool (AWS SCT) 将 Amazon RDS for Oracle 数据库架构对象转换为 Amazon RDS for PostgreSQL 数据库架构对象。然后，您可在 Windows 命令提示符中运行 Python 脚本，为源数据库对象的增量更改创建 AWS SCT 对象。

> **Note**
>
> 在迁移生产工作负载之前，我们建议您在测试或非生产环境中针对此模式的方法运行概念验证 (PoC)。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 现有 Amazon RDS for Oracle DB 实例。
- 现有 Amazon RDS for PostgreSQL 数据库实例。
- AWS SCT，安装并配置了适用于 Oracle 和 PostgreSQL 数据库引擎的 JDBC 驱动程序。有关更多信息，请参阅 AWS SCT 文档中的安装 AWS SCT 和安装所需数据库驱动程序。
- Oracle SQL Developer，已安装并配置。有关更多信息，请参阅 Oracle SQL Developer 文档。
- 将（附件）`incremental-migration-sct-sql.zip` 文件下载到本地计算机中。

限制

- 您的源 Amazon RDS for Oracle DB 实例的最低要求是：
  - Oracle 10.2 和更高版本（对于版本 10.x）、11g（版本 11.2.0.3.v1 和更高版本）直至 12.2 以及 18c 版本（Enterprise、Standard、Standard One 和 Standard Two 版）。
- 您的目标 Amazon RDS for PostgreSQL DB 实例的最低要求是：
  - PostgreSQL 版本 9.4 和更高版本（对于版本 9.x）、10.x 和 11.x
- 此模式使用 Oracle SQL Developer。如果您使用其他工具查找和导出架构差异，结果可能会有所不同。
- Oracle SQL Developer 生成的 SQL 脚本可能会引发转换错误，这意味着您需要执行手动迁移。
- 如果 AWS SCT 源和目标测试连接失败，请确保为虚拟私有云（VPC）安全组配置 JDBC 驱动程序版本以及入站规则，以接受传入流量。

产品版本

- Amazon RDS for Oracle 数据库实例版本 12.1.0.2（版本 10.2 和更高版本）
- Amazon RDS for PostgreSQL 数据库实例版本 11.5（版本 9.4 和更高版本）
- Oracle SQL 开发人员版本 19.1 及更高版本
- AWS SCT 版本 1.0.632 及更高版本

架构

源技术堆栈

- Amazon RDS for Oracle 数据库实例

目标技术堆栈

- Amazon RDS for PostgreSQL 数据库实例

源架构和目标架构

下图显示了将 Amazon RDS for Oracle 数据库实例迁移至 Amazon RDS for PostgreSQL 数据库实例的情况。

图表显示了以下迁移工作流：

1. 打开 Oracle SQL Developer 并连接到源数据库和目标数据库。
2. 生成差异报告，然后为架构差异对象生成 SQL 脚本文件。有关差异报告的更多信息，请参阅 Oracle 文档中的详细差异报告。
3. 配置 AWS SCT 和运行 Python 代码。
4. SQL 脚本文件从 Oracle 转换至 PostgreSQL。
5. 在目标 PostgreSQL 数据库实例运行 SQL 脚本文件。

自动化和扩缩

您可以通过在 Python 脚本中为单个程序的多个功能添加其他参数，和与安全相关的更改，来自动执行此迁移。

工具

- AWS SCT – AWS Schema Conversion Tool (AWS SCT) 将现有数据库架构从一个数据库引擎转换为另一个数据库引擎。
- Oracle SQL Developer – Oracle SQL Developer 是一个集成式开发环境（IDE），可简化传统部署和基于云的部署中 Oracle 数据库的开发和管理。

代码

该 `incremental-migration-sct-sql.zip` 文件（附后）包含此模式的完整源代码。

操作说明

为源数据库架构差异创建 SQL 脚本文件

Task	描述	所需技能
在 Oracle SQL Developer 中运行 Database Diff。	1. 登录您的源 Oracle 数据库实例，选择工具，然后选择 Database Diff。 2. 在源连接中选择源数据库。 3. 在目标连接中选择已更新或已修补的源数据库。	数据库管理员

Task	描述	所需技能
	4. 根据您的要求配置其余选项，选择下一步，然后选择完成以生成差异报告。	
生成 SQL 脚本文件。	选择生成脚本以在 SQL 文件中生成差异。 这将生成 SQL 脚本文件，AWS SCT 使用此文件将您的数据库从 Oracle 转换为 PostgreSQL。	数据库管理员

s使用 Python 脚本在 AWS SCT 中创建目标数据库对象

Task	描述	所需技能
通过 Windows 命令提示符配置 AWS SCT。	1. 从您预安装的 AWS SCT 文件夹中复制`AWSSchemaConversionToolBatch.jar` 文件并将其粘贴到您的工作目录中。 2. 从 `run_aws_sct_sql.py` 文件夹 `incremental-migration-sct-sql.zip` 文件（附后）中部署 Python 代码。这将在包含源数据库和目标数据库环境配置详细信息的 `projects` 目录中创建 .xml 文件和 .sct 文件。它还会读取您在 Oracle SQL Developer 中生成的 SQL 脚本文件。最后，它会在 `output` 目录中创建 .sql 文件对象。	数据库管理员

Task	描述	所需技能

	3. 使用以下格式在 `database_migration.txt` 文件中配置源环境和目标环境配置的详细信息：
	
	`#source_vendor,source_hostname,source_dbname,source_user,source_pwd,source_schema,source_port,source_sid,target_vendor,target_hostname,target_user,target_pwd,target_dbname,target_port`
	
	`ORACLE,myoracledb.cokmvis0v46q.us-east-1.rds.amazonaws.com,ORCL,orcl,orcl1234,orcl,1521,ORCL,POSTGRESQL,mypgdbinstance.cokmvis0v46q.us-east-1.rds.amazonaws.com,pguser,pgpassword,pgdb,5432`
	
	4. 根据要求修改 AWS SCT 配置参数，然后将 SQL 脚本文件复制到工作目录的 `input` 子目录中。	

Task	描述	所需技能
运行 Python 脚本。	1. 使用以下命令运行 Python 脚本：`$ python run_aws_sct_sql.py database_migration .txt` 2. 这将会创建数据库对象 SQL 文件。存在转换错误的未转换代码可手动转换。	数据库管理员
在 Amazon RDS for PostgreSQL 中创建对象	运行 SQL 文件，并在您的 Amazon RDS for PostgreSQL DB 实例中创建对象。	数据库管理员

相关的资源

- Amazon RDS 上的 Oracle
- Amazon RDS 上的 PostgreSQL
- 使用 AWS SCT 用户界面
- 将 Oracle 作为 AWS SCT 的源

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用兼容 Aurora PostgreSQL 的文件编码将 BLOB 文件加载至文本中

由 Bhanu Ganesh Gudivada (AWS) 和 Jeevan Shetty (AWS) 创建

摘要

通常，在迁移过程中，您必须处理从本地文件系统上的文件加载的、非结构化和结构化数据。数据也可采用与数据库字符集不同的字符集。

这些文件包含以下类型数据：

- 元数据 - 此数据描述了文件结构。
- 半结构化数据 - 这些是特定格式的文本字符串，例如 JSON 或 XML。你可以对此类数据做出断言，例如将始终以“<”开头” 或不包含任何换行符”。
- 全文 - 此数据通常包含所有类型的字符，包括换行符和引号字符。它还可能由 UTF-8 格式的多字节字符构成。
- 二进制数据-此数据可能包含字节或字节组合，包括空值和 end-of-file标记。

混合加载这些类型的数据，可能是一项挑战。

该模式可用于本地 Oracle 数据库、亚马逊弹性计算云 (AWS) 云上的亚马逊弹性计算云 (Amazon EC2) 实例上的 Oracle 数据库，以及适用于 Oracle 数据库的亚马逊关系数据库服务 (Amazon RDS)。例如，这种模式使用的是与 Amazon Aurora PostgreSQL-Compatible Edition。

在 Oracle 数据库中，借助 `BFILE`（二进制文件）指针、`DBMS_LOB` 软件包和 Oracle 系统函数，您可以从文件加载并使用字符编码转换为 CLOB。由于 PostgreSQL 在迁移到 Amazon Aurora PostgreSQL-Compatible Edition 数据库时不支持 BLOB 数据类型，因此必须将这些函数转换为兼容 PostgreSQL 的脚本。

此模式提供了两种将文件加载至兼容 Amazon Aurora PostgreSQL 的数据库中的单个数据库列中的方法：

- 方法 1 — 您通过使用带有编码选项的扩展 `table_import_from_s3` `aws_s3` 函数，从 Amazon Simple Storage Service (Amazon S3) 存储桶导入数据。
- 方法 2 — 在数据库外部编码为十六进制，然后解码以在数据库内部查看 `TEXT`。

我们建议使用 Aurora PostgreSQL，因为PostgreSQL-Compatible 可以直接与 `aws_s3` 扩展集成。

本文介绍：将包含电子邮件模板的平面文件加载到 Amazon Aurora PostgreSQL-Compatible 数据库中的示例，该模板具有多字节字符和不同的格式。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 一个 Amazon RDS 实例或 Aurora PostgreSQL-Compatible 实例
- 对 SQL 和 Relational Database Management System (RDBMS) 有基本了解
- Amazon Simple Storage Service (Amazon S3) 存储桶。
- 了解 Oracle 和 PostgreSQL 中的系统函数
- RPM Packag HexDump e-XXD-0.1.1（包含在亚马逊 Linux 2 中）

> **Note**
>
> 亚马逊 Linux 2 的支持已接近终止。欲了解更多信息，请参阅亚马逊 Linux 2 FAQs。

限制

- 对于 TEXT 数据类型，可以存储的最长字符串约为 1 GB。

产品版本

- Aurora 支持 Amazon Aurora PostgreSQL 更新中列出的 PostgreSQL 版本。

架构

目标技术堆栈

- Aurora PostgreSQL-Compatible

目标架构

方法 1 — 使用 aws_s3.table_import_from_s3

将包含多字节字符和自定义格式的电子邮件模板文件从本地服务器传输至 Amazon S3。本文提供的自定义数据库函数使用带 `file_encoding` 的 `aws_s3.table_import_from_s3` 函数将文件加载至数据库，并将查询结果以 `TEXT` 数据类型的形式返回。

1. 文件将传输至 Staging S3 存储桶。
2. 文件将上传至 Amazon Aurora PostgreSQL-Compatible 数据库。
3. 使用 pgAdmin 客户端，将自定义 `load_file_into_clob` 函数部署至 Aurora 数据库。
4. 自定义函数内部 `table_import_from_s3` 与 file_encoding 一起使用。该函数的输出是通过使用 `array_to_string` 和 `array_agg` 作为 `TEXT` 输出获得。

方法 2 — 在数据库外部编码为十六进制，然后解码以查看数据库内文本

来自本地服务器或本地文件系统的文件将转换至十六进制转储。然后，该文件将作为 `TEXT` 字段导入 PostgreSQL。

1. 使用 `xxd -p` 选项在命令行中将文件转换为十六进制转储。
2. 使用 `\copy` 选项将十六进制转储文件上传至兼容 Aurora PostgreSQL 文件，然后将十六进制转储文件解码为二进制。
3. 编码二进制数据，以返回为 `TEXT`。

工具

Amazon Web Services

- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩展 PostgreSQL 部署。
- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。

其他工具

- pgadmin4 是 PostgreSQL 的开源管理和开发平台。pgadmin4 可以在 Linux、Unix、Mac OS 和 Windows 上使用 postgreSQL 管理 PostgreSQL。

操作说明

方法 1：将数据从 Amazon S3 导入至兼容 Amazon S3 的 Aurora

Task	描述	所需技能
启动实 EC2 例。	有关启动实例的说明，请参阅启动实例。	数据库管理员
安装 PostgreSQL 客户端 pgAdmin 工具。	下载并安装 pgAdmin。	数据库管理员
创建一个 IAM 策略。	创建名为 `aurora-s3-access-pol` 的 AWS Identity and Acess Management (IAM) policy，用于授予对存储文件的 S3 存储桶的访问权限。请使用以下代码，将 `<bucket-name>` 替换为您的 S3 存储桶名称。 `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "s3:GetObject", "s3:AbortMultipartUpload", "s3:DeleteObject", "s3:ListMultipartUploadParts", "s3:PutObject",`	数据库管理员

Task	描述	所需技能

	<pre>"s3:ListBucket"
	            ],
	            "Resource":
	[
	
	"arn:aws:s3:::<buc
	ket-name>/*",
	
	"arn:aws:s3:::<buc
	ket-name>"
	            ]
	        }
	    ]
	}</pre>	

创建 IAM 角色，将对象从 Amazon S3 导入至 Aurora PostgreSQL-Compatible。	使用以下代码创建名 `aurora-s3-import-role` 为AssumeRole信任关系的 IAM 角色。 `AssumeRole` 允许 Aurora 代表您访问其他 AWS 服务。
	<pre>{
	  "Version": "2012-10-
	17",
	  "Statement": [
	    {
	      "Effect":
	"Allow","Principal":
	{
	        "Service":
	"rds.amazonaws.com"
	      },"Action":
	"sts:AssumeRole"
	    }
	  ]
	}</pre>
		数据库管理员

Task	描述	所需技能
将 IAM 角色与集群关联。	要将 IAM 角色与 Aurora PostgreSQL-Compatible 数据库集群关联，请运行以下 AWS CLI 命令。将 `<Account-ID>` 更改为托管 Aurora PostgreSQL-Compatible 数据库的 Amazon Web Services account ID。这使与 Aurora PostgreSQL 兼容的数据库可访问 S3 存储桶。 `aws rds add-role-to-db-cluster --db-cluster-identifier aurora-postgres-cl --feature-name s3Import --role-arn arn:aws:iam::<Account-ID>:role/aurora-s3-import-role`	数据库管理员
将示例上传到 Amazon S3。	1. 在此模式的其他信息部分，将电子邮件模板代码复制到名为 `salary.event.notification.email.vm` 的文件中。 2. 将文件上传到 S3 存储桶。	数据库管理员、应用程序所有者

Task	描述	所需技能
部署自定义函数。	1. 在其他信息部分，将自定义函数 `load_file_into_clob` SQL 文件内容复制到临时表中。 2. 登录与 Aurora PostgreSQL 兼容的数据库，然后使用 pgAdmin 客户端将其部署至数据库架构。	应用程序所有者，数据库管理员

运行可将数据导入数据库的自定义函数。	运行以下 SQL 命令，将尖括号中的项目替换为相应值。
	`select load_file_into_clob('aws-s3-import-test'::text,'us-west-1'::text,'employee.salary.event.notification.email.vm'::text);`
	运行命令之前，请将尖括号中的项目替换为相应的值，如以下示例所示。
	`Select load_file_into_clob('aws-s3-import-test'::text,'us-west-1'::text,'employee.salary.event.notification.email.vm'::text);`
	该命令从 Amazon S3 加载文件并将输出返回为 TEXT。
		应用程序所有者，数据库管理员

方法 2：在本地 Linux 系统中将模板文件转换至十六进制转储

Task	描述	所需技能

将模板文件转换至十六进制转储。	**Note**
	Hexdump 实用程序以十六进制、十进制、八进制或 ASCII 格式显示二进制文件内容。该 `hexdump` 命令是 `util-linux` 软件包的一部分，已预先安装在 Linux 发行版中。Hexdump RPM 软件包也是 Amazon Linux 2 的一部分。(: 亚马逊 Linux 2 的支持已接近终止。 欲了解更多信息，请参阅亚马逊 Linux 2 FAQs.)
	
	要将文件内容转换至十六进制转储，请运行以下 Shell 命令。
	
	`xxd -p </path/file.vm> | tr -d '\n' > </path/file.hex>`
	
	将路径和文件替换为相应的值，如以下示例所示。
	
	`xxd -p employee.salary.event.notification.email.vm | tr -d '\n' > employee.`
		数据库管理员

Task	描述	所需技能
	`salary.event.notification.email.vm.hex`	

Task	描述	所需技能

将十六进制转储文件加载至数据库架构。	使用以下命令将十六进制转储文件加载至 Aurora PostgreSQL-Compatible 数据库。
	1. 登录 Aurora PostgreSQL DB，然后创建一个名为 `email_template_hex` 的新表。
	`CREATE TABLE email_template_hex(hex_data TEXT);`
	2. 使用以下命令，将文件从本地文件系统加载至数据库架构。
	`\copy email_template_hex FROM '/path/file.hex';`
	将路径替换为本地文件系统位置。
	`\copy email_template_hex FROM '/tmp/employee.salary.event.notification.email.vm.hex';`
	3. 再创建一个名为 `email_template_bytea` 的表。
	`CREATE TABLE email_template_bytea(hex_data bytea);`
		数据库管理员

Task	描述	所需技能

	4. 将数据从 email_template_hex 插入至 email_template_bytea 。
	
	`INSERT INTO email_template_bytea (hex_data)`
	
	`(SELECT decode(hex_data, 'hex') FROM email_template_hex limit 1);`
	
	5. 要将十六进制字节码作为 TEXT 数据返回，请运行以下命令。
	
	`SELECT encode(hex_data::bytea, 'escape') FROM email_template_bytea;`	

相关资源

参考

- 将 PostgreSQL 数据库用作 AWS Database Migration Service 的目标
- 参照 PostgreSQL Compatibility (12.4) 迁移手册将 Oracle Database 19c 迁移至 Amazon Aurora
- 创建 IAM policy
- 将 IAM 角色与 Amazon Aurora MySQL 数据库集群关联
- pgAdmin

教程

- Amazon RDS 入门
- 从 Oracle 迁移到 Amazon Aurora

其他信息

load_file_into_clob custom function

```
CREATE OR REPLACE FUNCTION load_file_into_clob(
    s3_bucket_name text,
    s3_bucket_region text,
    file_name text,
    file_delimiter character DEFAULT '&'::bpchar,
    file_encoding text DEFAULT 'UTF8'::text)
    RETURNS text
    LANGUAGE 'plpgsql'
    COST 100
    VOLATILE PARALLEL UNSAFE
AS $BODY$
DECLARE
    blob_data BYTEA;
    clob_data TEXT;
    l_table_name CHARACTER VARYING(50) := 'file_upload_hex';
    l_column_name CHARACTER VARYING(50) := 'template';
    l_return_text TEXT;
    l_option_text CHARACTER VARYING(150);
    l_sql_stmt CHARACTER VARYING(500);

BEGIN

    EXECUTE format ('CREATE TEMPORARY TABLE %I (%I text, id_serial serial)',
 l_table_name, l_column_name);

    l_sql_stmt := 'select ''(format text, delimiter ''''' || file_delimiter || ''''',
 encoding ''''' || file_encoding ||  ''''')'' ';

    EXECUTE FORMAT(l_sql_stmt)
    INTO l_option_text;

    EXECUTE FORMAT('SELECT aws_s3.table_import_from_s3($1,$2,$6,
 aws_commons.create_s3_uri($3,$4,$5))')
    INTO l_return_text
```

```
    USING l_table_name, l_column_name, s3_bucket_name,
 file_name,s3_bucket_region,l_option_text;

    EXECUTE format('select array_to_string(array_agg(%I order by id_serial),E''\n'')
 from %I', l_column_name, l_table_name)
    INTO clob_data;

    drop table file_upload_hex;

    RETURN clob_data;
END;
$BODY$;
```

电子邮件模板

```
##############################################################################################
##
 ##
##    johndoe Template Type: email
 ##
##    File: johndoe.salary.event.notification.email.vm
 ##
##    Author: Aimée Étienne    Date 1/10/2021
      ##
##  Purpose: Email template used by EmplmanagerEJB to inform a johndoe they         ##
##        have been given access to a salary event
 ##
##    Template Attributes:
 ##
##        invitedUser - PersonDetails object for the invited user
  ##
##        salaryEvent - OfferDetails object for the event the user was given access
 ##
##        buyercollege - CompDetails object for the college owning the salary event
 ##
##        salaryCoordinator - PersonDetails of the salary coordinator for the event
 ##
##        idp - Identity Provider of the email recipient
 ##
##        httpWebRoot - HTTP address of the server
 ##
```

```
##
 ##
################################################################################################

$!invitedUser.firstname $!invitedUser.lastname,

Ce courriel confirme que vous avez ete invite par $!salaryCoordinator.firstname $!
salaryCoordinator.lastname de $buyercollege.collegeName a participer a l'evenement
 "$salaryEvent.offeringtitle" sur johndoeMaster Sourcing Intelligence.

Votre nom d'utilisateur est $!invitedUser.username

Veuillez suivre le lien ci-dessous pour acceder a l'evenement.

${httpWebRoot}/myDashboard.do?idp=$!{idp}

Si vous avez oublie votre mot de passe, utilisez le lien "Mot de passe oublie" situe
 sur l'ecran de connexion et entrez votre nom d'utilisateur ci-dessus.

Si vous avez des questions ou des preoccupations, nous vous invitons a
 communiquer avec le coordonnateur de l'evenement $!salaryCoordinator.firstname $!
salaryCoordinator.lastname au ${salaryCoordinator.workphone}.

*******

johndoeMaster Sourcing Intelligence est une plateforme de soumission en ligne pour les
 equipements, les materiaux et les services.

Si vous avez des difficultes ou des questions, envoyez un courriel a
 support@johndoeMaster.com pour obtenir de l'aide.
```

使用并使用和，将 Amazon RDS for Oracle 迁移到适用于 PostgreSQL AWS SCT 的 Amazon RDS AWS DMSAWS CLIAWS CloudFormation

由 Pinesh Singal (AWS) 编写

摘要

此模式显示如何使用 () 将适用于 Oracle 数据库实例的多 TB 的亚马逊关系数据库服务 (Amazon RDS) 迁移到适用于 Postgre SQL 的 Amazon RDS 数据库实例。 AWS Command Line Interface AWS CLI 该方法可最大限度地减少停机时间，并且不需要登录 AWS Management Console。

这种模式有助于使用 () 和 AWS Schema Conversion Tool AWS Database Migration Service (AWS SCTAWS DMS) 控制台避免手动配置和单独迁移。该解决方案为多个数据库设置一次性配置，并在 AWS DMS 中使用 AWS SCT 和执行迁移。 AWS CLI

该模式用于 AWS SCT 将数据库架构对象从 Amazon RDS for Oracle 转换为适用于 PostgreSQL 的 Amazon RDS，然后 AWS DMS 使用来迁移数据。使用中的 Python 脚本 AWS CLI，您可以使用 AWS CloudFormation 模板创建 AWS SCT 对象和 AWS DMS 任务。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- 现有 Amazon RDS for Oracle DB 实例。
- 现有 Amazon RDS for PostgreSQL 数据库实例。
- 用于运行脚本的亚马逊弹性计算云 (Amazon EC2) 实例或装有 Windows 或 Linux 操作系统的本地计算机。
- 了解以下 AWS DMS 迁移任务类型：`full-load`、`cdc`、`full-load-and-cdc`。 有关更多信息，请参阅 AWS DMS 文档中的创建任务。
- AWS SCT，安装并配置了适用于 Oracle 和 PostgreSQL 数据库引擎的 Java 数据库连接 (JDBC) 驱动程序。有关更多信息，请参阅 AWS SCT 文档 AWS SCT中的安装和配置。
- 已安装 AWS SCT 文件夹中的`AWSSchemaConversionToolBatch.jar`文件已复制到您的工作目录。
- `cli-sct-dms-cft.zip` 文件（附后），已下载并解压缩到您的工作目录中。
- 最新的 AWS DMS 复制实例引擎版本。有关更多信息，请参阅 AWS 支持 文档和AWS DMS 发行说明中的如何创建 AWS DMS 复制实例。

- AWS CLI 版本 2，安装并配置了您的访问密钥 ID、私有访问密钥以及运行脚本的 EC2 实例或操作系统的默认 AWS 区域 名称。有关更多信息，请参阅 AWS CLI 文档 AWS CLI中的安装或更新到最新版本 AWS CLI和配置设置。
- 熟悉 AWS CloudFormation 模板。有关更多信息，请参阅 AWS CloudFormation 文档中的AWS CloudFormation 工作原理。
- Python 版本 3，在运行脚本的 EC2 实例或操作系统上安装和配置。有关更多信息，请参阅 Python 文档。

限制

- 您的源 Amazon RDS for Oracle 数据库实例的最低要求是：
  - 企业版、标准版、标准一版和标准二版的 Oracle 版本 12c（12.1.0.2、12.2.0.1）、18c（18.0.0.0）和 19c（19.0.0.0）。
  - 尽管 Amazon RDS 支持 Oracle 18c (18.0.0.0)，但此版本已处于弃用状态，因为在该日期之后，甲骨文不再提供 18c 的补丁。 end-of-support有关更多信息，请参阅 Amazon RDS 文档中的Amazon RDS for Oracle。
  - 不再支持 Amazon RDS for Oracle 11g。
- 您的目标 Amazon RDS for PostgreSQL 数据库实例的最低要求是：
  - PostgreSQL 版本 9（9.5 和 9.6）、10.x、11.x、12.x 和 13.x

产品版本

- Amazon RDS for Oracle 数据库实例版本 12.1.0.2 及更高版本
- Amazon RDS for PostgreSQL 数据库实例版本 11.5 及更高版本
- AWS CLI 第 2 版
- 最新版本的 AWS SCT
- Python 3 的最新版本。

架构

源技术堆栈

- Amazon RDS for Oracle

目标技术堆栈

- Amazon RDS for PostgreSQL

源架构和目标架构

下图显示了使用和 Python 脚本将 Amazon RDS for Oracle 数据库实例迁移到 Amazon RDS for PostgreSQL 数据库实例 AWS DMS 的情况。

图表显示了以下迁移工作流：

1. Python 脚本 AWS SCT 用于连接源数据库实例和目标数据库实例。
2. 用户从 Python 脚本 AWS SCT 开始，将 Oracle 代码转换为 PostgreSQL 代码，然后在目标数据库实例上运行该代码。
3. Python 脚本为源数据库实例和目标数据库实例创建 AWS DMS 复制任务。
4. 用户部署 Python 脚本来启动 AWS DMS 任务，然后在数据迁移完成后停止任务。

自动化和扩缩

您可以通过在 Python 脚本中添加参数和与安全相关的更改来自动执行迁移，以提供其他功能。

工具

- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您通过命令行外壳中的命令与 AWS 服务进行交互。
- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置这些资源，并在各个区域的整个生命周期中 AWS 账户 对其进行管理。此模式使用 Python 脚本将`.json`输入文件转换为输入文件。`.csv`该`.json`文件用于 AWS CLI 命令创建 AWS CloudFormation 堆栈，该堆栈使用 Amazon 资源名称 (ARNs)、迁移类型、任务设置和表映射创建多个 AWS DMS 复制任务。
- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到云和本地设置的组合之间 AWS Cloud 或迁移。此模式 AWS DMS 用于使用在命令行上运行的 Python 脚本创建、启动和停止任务，以及创建 AWS CloudFormation 模板。
- AWS Schema Conversion Tool (AWS SCT) 通过自动将源数据库架构和大部分自定义代码转换为与目标数据库兼容的格式来支持异构数据库迁移。此模式需要已安装 AWS SCT 目录中的`AWSSchemaConversionToolBatch.jar`文件。

代码

该 `cli-sct-dms-cft.zip` 文件（附后）包含此模式的完整源代码。

操作说明

在中配置 AWS SCT 和创建数据库对象 AWS CLI

Task	描述	所需技能

配置 AWS SCT 为从中运行 AWS CLI。	1. 使用以下格式在 `database_migration.txt` 文件中配置源环境和目标环境配置的详细信息：
	`#source_vendor,source_hostname,source_dbname,source_user,source_pwd,source_schema,source_port,source_sid,target_vendor,target_hostname,target_user,target_pwd,target_dbname,target_port`
	`ORACLE,myoracledb.cokmvis0v46q.us-east-1.rds.amazonaws.com,ORCL,orcl,orcl1234,orcl,1521,ORCL,POSTGRESQL,mypgdbinstance.cokmvis0v46q.us-east-1.rds.amazonaws.com,pguser,pgpassword,pgdb,5432`
	2. 根据您的要求在以下文件中修改 AWS SCT 配置参数：`project_settings.x`
		数据库管理员

Task	描述	所需技能
	ml Oracle_PG _Test_Batch.xml 、和ORACLE-orcl-to-POSTGRESQL.xml 。	
运行 run_aws_sct.py Python 脚本。	使用以下命令运行 run_aws_sct.py Python 脚本： $ python run_aws_sct.py database_migration.txt Python 脚本将数据库对象从 Oracle 转换为 PostgreSQL，并创建 PostgreSQL 格式 SQL 文件。该脚本还会创建 PDF 文件Database migration assessment report ，该文件为您提供数据库对象的详细建议和转换统计信息。	数据库管理员
在 Amazon RDS for PostgreSQL 中创建对象。	1. 如果需要，可以手动修改由生成的 AWS SCT SQL 文件。 2. 运行 SQL 文件，并在您的 Amazon RDS for PostgreSQL DB 实例中创建对象。	数据库管理员

使用和配置 AWS CLI 和创建 AWS DMS 任务 AWS CloudFormation

Task	描述	所需技能
创建 AWS DMS 复制实例。	登录 AWS Management Console，打开AWS DMS 控	数据库管理员

Task	描述	所需技能
	制台，然后创建根据您的要求配置的复制实例。 有关更多信息，请参阅 AWS DMS 文档中的创建复制实例和文档中的如何创建 AWS DMS 复制实例。 AWS 支持	
创建源端点。	在 AWS DMS 控制台上，选择终端节点，然后根据需要为 Oracle 数据库创建源端点。 **Note** 额外的连接属性必须`numberDataTypeScale` 带有`-2`值。 有关更多信息，请参阅 AWS DMS 文档中的创建源端点和目标端点。	数据库管理员
创建目标端点。	在 AWS DMS 控制台上，选择终端节点，然后根据需要为 PostgreSQL 数据库创建目标终端节点。 有关更多信息，请参阅 AWS DMS 文档中的创建源端点和目标端点。	DevOps 工程师

Task	描述	所需技能

将 AWS DMS 复制详细信息配置为从中运行 AWS CLI。	使用以下格式使用 AWS DMS 源终端节点 ARN、目标终端节点 ARN 和复制实例 ARN 在`dms-arn-list.txt` 文件中配置源和目标终端节点以及复制详细信息：
	
	`#sourceARN,targetARN,repARN`
	`arn:aws:dms:us-east-1:123456789012:endpoint:EH7AINRUDZ5GOYIY6HVMXECMCQ`
	`arn:aws:dms:us-east-1:123456789012:endpoint:HHJVUV57N7O3CQF4PJZKGIOYY5`
	`arn:aws:dms:us-east-1:123456789012:rep:LL57N77AQQAHHJF4PJFHNEDZ5G`
		数据库管理员

Task	描述	所需技能
运行 dms-create-task.py Python 脚本来创建 AWS DMS 任务。	1. 使用以下命令运行 dms-create-task.py Python 脚本： `$ python dms-create-task.py database_migration.txt dms-arn-list.txt <cft-stack-name> <migration-type>` 其中： • database_migration.txt 是数据库迁移文本文件。 • dms-arn-list.txt 是的 ARN 列表。 AWS DMS • <cft-stack-name> 是用户定义的 AWS CloudFormation 堆栈名称。 • <migration-type> 为 full-load 、cdc 或 full-load-and-cdc。 2. 根据您的迁移类型，您可以使用以下命令创建三种类型的 AWS DMS 任务： • $ python dms-create-task.py database_migration.txt dms-arn-list.txt dms-cli-c	数据库管理员

Task	描述	所需技能
	ft-stack full-load • $ python dms-create-task.py database_migration.txt dms-arn-list.txt dms-cli-cft-stack cdc • $ python dms-create-task.py database_migration.txt dms-arn-list.txt dms-cli-cft-stack full-load-and-cdc	
确认 AWS DMS 任务已准备就绪。	在 AWS DMS 控制台上，在 “Ready状态” 部分检查您的 AWS DMS 任务是否处于状态。	数据库管理员

使用启动和停止 AWS DMS 任务 AWS CLI

Task	描述	所需技能
开始 AWS DMS 任务。	使用以下命令运行 dms-start-task.py Python 脚本： $ python dms-start-task.py start '<cdc-start-datetime>'	数据库管理员

Task	描述	所需技能
	Note 开始日期和时间必须'YYYY-MM-DDTHH:MI:SS' 采用'DD-MON-YYYY' 或格式（例如，'01-Dec-2019' 或'2018-03-08T12:12:12' ）。 您可以在 AWS DMS 控制台 AWS DMS 任务页面的表格统计选项卡中查看任务状态。	

Task	描述	所需技能
验证数据。	1. 满载迁移完成后，CDC 的任务将持续运行。 2. 当 CDC 完成或无需迁移更多更改时，请查看并验证 Oracle 和 PostgreSQL DB 中的迁移任务结果和数据。 您可以通过在 AWS DMS 控制台任务页面的数据库迁移任务的表统计选项卡中检查状态和计数列（**Validation state** **Validation suspended** 、、、和**Validation details**）来验证数据。**Validation pending** **Validation failed** 有关更多信息，请参阅 AWS DMS 文档中的AWS DMS 数据验证。	数据库管理员

Task	描述	所需技能
停止 AWS DMS 任务。	使用以下命令运行 Python 脚本： `$ python dms-start-task.py stop` **Note** AWS DMS 任务可能会以`failed`状态停止，具体取决于验证状态。有关更多信息，请参阅下一部分。	数据库管理员

故障排除

事务	解决方案
AWS SCT 源和目标测试连接失败。	配置 JDBC 驱动程序版本与 VPC 安全组入站规则以接受传入流量。
源端点或目标端点测试运行失败。	检查终端节点设置和复制实例是否处于`Available` 状态。检查端点连接状态是否为`Successful` 。 有关更多信息，请参阅 AWS 支持 文档中的如何排除 AWS DMS 终端节点连接故障。
满载运行失败。	检查源数据库和目标数据库是否具有匹配的数据类型与大小。 有关更多信息，请参阅 AWS DMS 文档AWS DMS中的迁移任务疑难解答。
您会遇到验证运行错误。	检查该表是否有主键，因为非主键表未经验证。

事务	解决方案
	如果表有主键和错误，请检查源端点中的额外连接属性是否有 `numberDataTypeScale=-2` 。 有关更多信息，请参阅 AWS DMS 文档中的使用 Oracle 作为来源时的端点设置和疑难解答。AWS DMSOracleSettings

相关资源

- 安装和配置 AWS SCT
- 简介 AWS DMS（视频）
- 和的 CloudFormation 堆栈操作命令示例 AWS CLI PowerShell
- 浏览的用户界面 AWS SCT
- 使用 Oracle 数据库作为来源 AWS DMS
- 使用连接到 Oracle 数据库 AWS SCT
- 使用 PostgreSQL 数据库作为目标 AWS DMS
- 数据迁移的源
- 数据迁移的目标
- 云形成（AWS CLI 文档）
- 创建堆栈（文档）AWS CLI
- dms（AWS CLI 文档）

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS DMS 在 SSL 模式下将 Amazon RDS for Oracle 迁移到 Amazon RDS for PostgreSQL

由 Pinesh Singal (AWS) 编写

摘要

此模式提供了将 Amazon Relational Database Service （Amazon RDS）for Oracle 数据库实例迁移到 Amazon Web Services （AWS）云上的 Amazon RDS for PostgreSQL 数据库的指导。为了加密数据库之间的连接，该模式在 Amazon RDS 和 AWS Database Migration Service （AWS DMS）中使用证书颁发机构 (CA) 和 SSL 模式。

该模式描述了一种在线迁移策略，对于具有大量事务的多 TB Oracle 源数据库，停机时间很少或没有停机时间。为了数据安全，该模式在传输数据时使用 SSL。

此模式使用 AWS Schema Conversion Tool（AWS SCT）将 Amazon RDS for Oracle 数据库架构转换为 Amazon RDS for PostgreSQL 架构。然后，该模式使用 AWS DMS 将数据从 Amazon RDS for Oracle 数据库迁移到 Amazon RDS for PostgreSQL 数据库。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 仅使用 rds-ca-rsa2048-g1 配置的 Amazon RDS 数据库证书颁发机构 (CA)
  - rds-ca-2019 证书已于 2024 年 8 月到期。
  - rds-ca-2015 证书已于 2020 年 3 月 5 日到期。
- AWS SCT
- AWS DMS
- pgAdmin
- SQL 工具（例如 SQL Developer 或 SQL*Plus）

限制

- Amazon RDS for Oracle 数据库 – 企业版和标准版两个版本的最低要求是 Oracle 版本 19c。
- Amazon RDS for PostgreSQL 数据库 – 最低要求是 PostgreSQL 版本 12 及更高版本（适用于版本 9.x 及更高版本）。

产品版本

- Amazon RDS for Oracle 数据库版本 12.1.0.2 实例
- Amazon RDS for PostgreSQL 数据库版本 11.5 实例

架构

源技术堆栈

- 版本 12.1.0.2.v18 的 Amazon RDS for Oracle 数据库实例。

目标技术堆栈

- AWS DMS
- 版本 11.5 的 Amazon RDS for PostgreSQL 数据库实例。

目标架构

下图显示了 Oracle（源）和 PostgreSQL（目标）数据库之间的数据迁移体系结构。该架构包括以下内容：

- 虚拟私有云（VPC）
- 可用区
- 私有子网
- Amazon RDS for Oracle 数据库
- AWS DMS 复制实例
- RDS for PostgreSQL 数据库

要加密源数据库和目标数据库的连接，必须在 Amazon RDS 和 AWS DMS 中启用 CA 和 SSL 模式。

工具

Amazon Web Services

- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到 Amazon Web Services Cloud，或者在云和本地设置的组合之间迁移。

- Amazon Relational Database Service (Amazon RDS) for Oracle 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展 Oracle 关系数据库。
- Amazon Relational Database Service（Amazon RDS）for PostgreSQL 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展PostgreSQL 关系数据库。
- AWS Schema Conversion Tool（AWS SCT）通过自动将源数据库架构和大部分自定义代码转换为与目标数据库兼容的格式来支持异构数据库迁移。

其他服务

- pgAdmin 是一种适用于 PostgreSQL 的开源管理工具。它提供了一个图形界面，可帮助您创建、维护和使用数据库对象。

最佳实践

Amazon RDS 提供新的 CA 证书作为 AWS 安全最佳实践。有关新证书和支持的 AWS 区域的信息，请参阅使用 SSL/TLS 加密与数据库实例或集群的连接。

如果您的 RDS 实例当前使用的是 CA 证书`rds-ca-2019`，并且您想要升级到`rds-ca-rsa2048-g1`，请按照通过修改数据库实例或集群更新 CA 证书或通过应用维护来更新 CA 证书中的说明进行操作。

操作说明

配置 Amazon RDS for Oracle 实例

Task	描述	所需技能
创建 Oracle 数据库实例。	登录您的 Amazon Web Services account，打开 AWS 管理控制台，然后导航到 Amazon RDS 控制台。在控制台上，选择创建数据库，然后选择 Oracle。	常规 AWS、数据库管理员
配置安全组。	配置入站和出站安全组。	常规 AWS
创建选项组。	在与 Amazon RDS for Oracle 数据库相同的 VPC 和安全组	常规 AWS

Task	描述	所需技能
	中创建选项组。对于选项，选择 SSL。对于端口，选择 2484（对于 SSL 连接）。	
配置选项设置。	使用以下设置： • `SQLNET.CIPHER_SUITE`：`SSL_RSA_WITH_AES_256_CBC_SHA` • `SQLNET.SSL_VERSION`：`1.2 or 1.0`	常规 AWS
修改 Oracle 数据库实例的 RDS。	将 CA 证书设置为 rds-ca-rsa2048-g1。 在选项组下，附加之前创建的选项组。	数据库管理员、常规 AWS

Task	描述	所需技能

确认 RDS for Oracle 数据库实例可用。	确保 Amazon RDS for Oracle 数据库实例已启动并正在运行，并且数据库架构可访问。
	
	要连接到 RDS for Oracle DB，请从命令行使用 `sqlplus` 命令。
	
	`$ sqlplus orcl/**** @myoracledb.cokmvis0v46q.us-east-1.rds.amazonaws.com:1521/ORCL`
	`SQL*Plus: Release 12.1.0.2.0 Production on Tue Oct 15 18:11:07 2019`
	`Copyright (c) 1982, 2016, Oracle. All rights reserved.`
	`Last Successful login time: Mon Dec 16 2019 23:17:31 +05:30`
	`Connected to:`
	`Oracle Database 12c Enterprise Edition Release 12.1.0.2.0 - 64bit Production`
	`With the Partitioning, OLAP, Advanced Analytics and Real Application Testing options`
	`SQL>`
		数据库管理员

在 RDS for Oracle 数据库中创建对象和数据。	创建对象并在架构中插入数据。	数据库管理员

配置 Amazon RDS for PostgreSQL 实例

Task	描述	所需技能
创建 RDS for PostgreSQL 数据库。	在 Amazon RDS 控制台创建数据库页面上，选择 PostgreSQL 以创建 Amazon RDS for PostgreSQL 数据库实例。	数据库管理员、常规 AWS
配置安全组。	配置入站和出站安全组。	常规 AWS
创建参数组。	如果您使用的是 PostgreSQL 版本 11.x，请创建一个参数组来设置 SSL 参数。在 PostgreSQL 版本 12 中，默认情况下启用 SSL 参数组。	常规 AWS
编辑参数。	将 `rds.force_ssl` 参数更改为 1 (on)。 默认情况下，`ssl` 参数设置为 1 (on)。通过将 `rds.force_ssl` 参数设置为 1，可以强制所有连接仅通过 SSL 模式进行连接。	常规 AWS
修改 RDS for PostgreSQL 数据库实例。	将 CA 证书设置为 rds-ca-rsa2048-g1。 附加默认参数组或之前创建的参数组，具体取决于您的 PostgreSQL 版本。	数据库管理员、常规 AWS
确认 RDS for PostgreSQL 数据库实例可用。	确保 Amazon RDS for PostgreSQL 数据库已启动并正在运行。 `psql` 命令通过命令行设置了 `sslmode` 建立 SSL 连接。	数据库管理员

Task	描述	所需技能

	一种选择是在参数组中设置 sslmode=1 并使用 psql 连接，而不在命令中包含 sslmode 参数。
	
	以下输出显示 SSL 连接已建立。
	
	`$ psql -h mypgdbinstance.cokmvis0v46q.us-east-1.rds.amazonaws.com -p 5432 "dbname=pgdb user=pguser"`
	`Password for user pguser:`
	`psql (11.3, server 11.5)`
	`SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256, compression: off)`
	`Type "help" for help.`
	`pgdb=>`
	
	第二个选项是在参数组中设置 sslmode=1 ，并在 psql 命令中包含 sslmode 参数。
	
	以下输出显示 SSL 连接已建立。
	
	`$ psql -h mypgdbinstance.cokmvis0v46q.us-east-1.rds.amazonaws.com -p 5432`	

Task	描述	所需技能
	"dbname=pgdb user=pgus er sslmode=require" Password for user pguser: psql (11.3, server 11.5) SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA- AES256-GCM-SHA384, bits: 256, compressi on: off) Type "help" for help. pgdb=>	

配置和运行 AWS SCT

Task	描述	所需技能
安装 AWS SCT。	安装最新版本的 AWS SCT 应用程序。	常规 AWS
使用 JDBC 驱动程序配置 AWS SCT。	下载适用于 Oracle（ojdbc8.jar）和 PostgreSQL（postgresql-42.2.5.jar）的 Java 数据库连接（JDBC）驱动程序。 要在 AWS SCT 中配置驱动程序，请依次选择设置、全局设置和驱动程序。	常规 AWS
创建 AWS SCT 项目。	使用 Oracle 作为源数据库引擎，使用 Amazon RDS for PostgreSQL 作为目标数据库引擎，创建 AWS SCT 项目和报告：	常规 AWS

Task	描述	所需技能
	1. 通过提供连接详细信息来测试与源 Oracle 数据库和目标 Amazon RDS for PostgreSQL 数据库的连接。 对于源 Oracle 数据库，需要以下权限或特权： • `CONNECT` • `SELECT_CATALOG_ROLE` • `SELECT ANY DICTIONARY` • `SELECT on SYS.USER$ TO <sct_user>` 有关更多信息，请参阅将 Oracle 数据库作为 AWS SCT 的源。 源连接和目标连接必须成功，AWS SCT 才能启动迁移报告。 2. 报告后，输入要转换的架构，然后选择完成。	

Task	描述	所需技能
验证数据库对象。	1. 选择 加载架构。 AWS SCT 显示源对象和转换后的目标对象，包括有错误的对象。更新目标数据库上任何不正确的对象。 2. 查看错误，然后使用手动干预将其清除。 3. 清除所有错误后，再次选择加载架构。 4. 选择应用于数据库。 5. 连接到 pgAdmin 或任何支持 PostgreSQL 数据库连接的工具，并检查架构和对象。	数据库管理员、常规 AWS

配置和运行 AWS DMS

Task	描述	所需技能
创建复制实例。	1. 登录您的账户，打开 AWS 管理控制台，然后导航到 AWS DMS 控制台。 2. 使用 VPC、安全组、可用区和额外连接属性的有效设置创建复制实例。	常规 AWS
导入证书。	下载适用于您的 AWS 区域的证书包 (PEM)。 该捆绑包中包含 `rds-ca-2019` 中间证书和根证书。该捆绑包还包含 `rds-ca-rsa2048-g1` 、`rds-ca-rsa4096-g1` 和 `rds-ca-ec`	常规 AWS

Task	描述	所需技能
	c384-g1 根 CA 证书。您的应用程序信任存储仅需要注册根 CA 证书。	
创建源端点。	1. 通过选择选择 RDS 数据库实例，然后选择您创建的 RDS for Oracle 数据库实例，为 Amazon RDS for Oracle 创建源端点。端点配置详细信息将自动填充。 2. 选择手动提供访问信息。对于端口，请确保输入 2484。 3. 在安全套接字层 (SSL) 模式下verify-ca ，选择，然后选择您之前创建的 CA 证书。 4. 在端点设置下，添加额外的连接属性NumberDataTypeScale=-2 以支持无大小的 NUMBER数据类型。 有关更多信息，请参阅使用 Oracle 数据库作为 AWS Database Migration Service 的源。	常规 AWS

Task	描述	所需技能
创建目标端点。	1. 通过选择选择 RDS数据库实例，然后选择您的 RDS for PostgreSQL 数据库实例，为 Amazon RDS for PostgreSQL 创建目标端点。端点配置详细信息将自动填充。 2. 选择手动提供访问信息。对于端口，请确保输入 2484。 有关更多信息，请参阅使用 PostgreSQL 数据库作为 AWS Database Migration Service 的目标。	常规 AWS
测试端点。	1. 测试源端点和目标端点以确认两者均成功且可用。 2. 如果测试失败，请确保安全组入站规则有效。	常规 AWS

Task	描述	所需技能
创建迁移任务。	要创建用于完全加载和更改数据捕获 (CDC) 或数据验证的迁移任务，请执行以下操作： 1. 要创建数据库迁移任务，请选择复制实例、源数据库端点、目标数据库端点。将迁移类型指定为以下类型之一： • 迁移现有数据（满载） • 仅复制数据更改 (CDC) • 迁移现有数据并复制持续更改（满载和 CDC） 2. 在表映射下，您可以配置 GUI 或 JSON 格式的选择规则和转换规则： • 在选择规则下，选择架构，输入表名，选择要配置的操作（“包含”或“排除”）；例如，Schema ORCL、Table name %、Action Include。 • 在转换规则下，执行下列操作之一： • 选择架构并选择操作（大小写、前缀、后缀）；例如，Target Schema ORCL、Action Make 小写。 • 选择架构，输入表名，然后选择操作（大小写、前缀、后缀）；例如，Target Schema	常规 AWS

Task	描述	所需技能
	ORCL， Table %， Action Make 小写。 3. 开启 Amazon CloudWatch 日志监控。 4. 对于映射规则，请添加以下 JSON 代码。 `{ "rules": [ { "rule-type": "transformation", "rule-id": "1", "rule-name": "1", "rule-target": "table", "object-locator": { "schema-name": "%", "table-name": "%" }, "rule-action": "convert-lowercase", "value": null, "old-value": null }, { "rule-type": "transformation", "rule-id": "2",`	

Task	描述	所需技能

	<pre>"rule-nam
	e": "2",
	"rule-tar
	get": "schema",
	"object-l
	ocator": {
	
	"schema-name":
	"ORCL",
	
	"table-name": "%"
	},
	"rule-
	action": "convert-
	lowercase",
	"value":
	null,
	"old-valu
	e": null
	},
	{
	"rule-typ
	e": "selection",
	"rule-id"
	: "3",
	"rule-nam
	e": "3",
	"object-l
	ocator": {
	
	"schema-name":
	"ORCL",
	
	"table-name": "DEPT"
	},
	"rule-act
	ion": "include",
	"filters"
	: []
	}
	]</pre>	

Task	描述	所需技能
	<pre>}</pre>	
计划生产运行。	与应用程序所有者等利益相关者确认停机时间，以便在生产系统中运行 AWS DMS。	迁移主管

Task	描述	所需技能

运行迁移任务。	1. 启动状态为 “就绪” 的 AWS DMS 任务，并在 Amazon CloudWatch 中监控迁移任务日志中是否存在任何错误。
	
	如果选择迁移现有数据并复制正在进行的更改作为迁移类型，并且状态为加载完成正在进行的复制，则 CDC 数据迁移的完全加载已完成，并且验证正在进行中。
	2. 开始迁移后，可以在中获取其他 SSL 连接信息。CloudWatch对于 Oracle，CloudWatch 显示以下连接字符串。
	
	`2019-12-17T09:15:11 [SOURCE_UNLOAD ]I: Connecting to Oracle: Beginning session (oracle_endpoint_conn.c:834)`
	
	PostgreSQL 连接字符串将类似于以下示例。
	
	`2019-12-17T09:15:11 [TARGET_LOAD ]I: Going to connect to ODBC connection string: PROTOCOL=7.4-0;DRIVER={Post`
		常规 AWS

Task	描述	所需技能
	greSQL};SERVER=mypgdbinstance.cokmvis0v46q.us-east-1.rds.amazonaws.com;DATABASE=pgdb;PORT=5432;sslmode=require;UID=pguser;(odbc_endpoint_imp.c:2218)	
验证数据。	查看源 Oracle 和目标 PostgreSQL 数据库中的迁移任务结果和数据： 1. 连接到 pgAdmin 并使用架构 ORCL 检查 PostgreSQL 数据库中的数据。 2. 对于 CDC，通过在源 Oracle 数据库中插入或更新数据来检查正在进行的更改。	数据库管理员
停止迁移任务。	成功完成数据验证后，停止迁移任务。	常规 AWS

清除资源

Task	描述	所需技能
删除 AWS DMS 任务。	1. 在 AWS DMS 控制台上，导航到数据库迁移任务，然后停止任何正在进行或正在运行的 AWS DMS 任务。	常规 AWS

Task	描述	所需技能
	2. 选择一个或多个任务，选择操作，然后选择删除。	
删除 AWS DMS 端点。	选择您创建的源端点和目标端点，选择操作，然后选择删除。	常规 AWS
删除 AWS DMS 复制实例。	选择复制实例，选择操作，然后选择删除。	常规 AWS
删除 PostgreSQL 数据库。	1. 在 Amazon RDS 控制台中，选择数据库。 2. 选择您创建的 PostgreSQL 数据库实例，选择操作，然后选择删除。	常规 AWS
删除 Oracle 数据库。	在 Amazon RDS 控制台上，选择 Oracle 数据库实例，选择操作，然后选择删除。	常规 AWS

故障排除

事务	解决方案
AWS SCT 源和目标测试连接失败。	配置 JDBC 驱动程序版本和 VPC 安全组入站规则以接受传入流量。
Oracle 源端点测试运行失败。	检查端点设置以及复制实例是否可用。
AWS DMS 任务满载运行失败。	检查源数据库和目标数据库的数据类型和大小是否匹配。
AWS DMS 验证迁移任务返回错误。	1. 检查表是否有主键。没有主键的表不会被验证。 2. 如果表具有主键但返回错误，请检查源端点中的额外连接属性。额外的连接属性必须具

事务	解决方案
	有 numberDataTypeScale=-2 才能支持 NUMBER 数据类型，并且不会根据表中可用的数据动态调整大小。

相关资源

数据库

- Amazon RDS for Oracle
- Amazon RDS for PostgreSQL

SSL DB 连接

- 使用 SSL/TLS 加密与数据库实例的连接
  - 对 RDS for Oracle 数据库实例使用 SSL
  - 使用 SSL/TLS 保护与 RDS for PostgreSQL 的连接
  - 下载特定 AWS 区域的证书捆绑包
    - 下载 CA-2019 根证书（已于 2024 年 8 月过期）
- 使用选项组
  - 向 Oracle 数据库实例添加选项
  - Oracle 安全套接字层
- 使用参数组
- PostgreSQL sslmode 连接参数
- 从 JDBC 使用 SSL
- 轮换您的 SSL/TLS 证书
  - 通过修改数据库实例或集群来更新 CA 证书
  - 通过应用维护来更新您的 CA 证书

AWS SCT

- AWS Schema Conversion Tool
- AWS Schema Conversion Tool 用户指南

- 使用 AWS SCT 用户界面
- 将 Oracle 数据库作为 AWS SCT 的源

AWS DMS

- AWS Database Migration Service
- AWS Database Migration Service 用户指南
  - 将 Oracle 数据库作为 AWS DMS 的源
  - 将 PostgreSQL 数据库作为 AWS DMS 的目标
- 将 SSL 与 AWS Database Migration Service 结合使用
- 将运行关系数据库的应用程序迁移到 AWS

其他信息

亚马逊 RDS 证书颁发机构证书`rds-ca-2019`已于 2024 年 8 月过期。如果您使用或计划使用带有证书验证的 SSL 或 TLS 来连接您的 RDS 数据库实例或多可用区数据库集群，请考虑使用新的 CA 证书之一：`rds-ca-rsa2048-g1``rds-ca-rsa4096-g1`、或`rds-ca-ecc384-g1`。

将 Oracle SERIALLY_REUSABLE pragma 包迁移至 PostgreSQL

由 Vinay Paladi (AWS) 编写

摘要

这种模式提供了 step-by-step一种将定义为 SERIALLY_REPLAGMA 的 Oracle 包迁移到亚马逊网络服务 (AWS) 上的 PostgreSQL 的方法。此方法保留了 SERIALLY_REUSABLE 编译指示的功能。

PostgreSQL 不支持包的概念和 SERIALLY_REUSABLE pragma。要在 PostgreSQL 中获得类似的功能，您可为包创建架构，并在架构中部署所有相关对象（例如函数、过程和类型）。为了实现 SERIALLY_REFLEY 编译指示的功能，此模式中提供的示例包装函数脚本使用了 AWS Schema Conversion Tool (AWS SCT) 扩展包。

有关更多信息，请参阅 Oracle 文档中的 SERIALLY_REUSABLE Pragma。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 最新版本的 AWS SCT 和所需驱动程序
- Amazon Aurora PostgreSQL 兼容版数据库或 Amazon Relational Database Service (Amazon RDS) for PostgreSQL 数据库

产品版本

- Oracle 数据库版本 10g 及更高版本

架构

源技术堆栈

- 本地 Oracle 数据库

目标技术堆栈

- Aurora PostgreSQL 兼容版或 Amazon RDS for PostgreSQL
- AWS SCT

迁移架构

工具

Amazon Web Services

- AWS Schema Conversion Tool（AWS SCT）通过自动将源数据库架构和大部分自定义代码转换为与目标数据库兼容的格式来支持异构数据库迁移。
- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩展 PostgreSQL 部署。
- Amazon Relational Database Service（Amazon RDS）for PostgreSQL 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展PostgreSQL 关系数据库。

其他工具

- pgAdmin 是一种适用于 PostgreSQL 的开源管理工具。它提供了一个图形界面，可帮助您创建、维护和使用数据库对象。

操作说明

使用 AWS SCT 迁移 Oracle 软件包

Task	描述	所需技能
设置 AWS SCT。	配置与源数据库的 AWS SCT 连接。有关更多信息，请参阅使用 Oracle 数据库作为 AWS SCT 的源。	数据库管理员、开发人员
转换脚本。	通过选择与 Aurora PostgreSQL 兼容的目标数据库，通过 AWS SCT 转换 Oracle 软件包。	数据库管理员、开发人员
保存 .sql 文件。	在保存.sql 文件之前，请将 AWS SCT 中的项目设置选	数据库管理员、开发人员

Task	描述	所需技能
	项修改为每个阶段单个文件。AWS SCT 会根据对象类型将 .sql 文件分成多个.sql 文件。	
更改代码。	打开 AWS SCT 生成的 `init` 函数，然后按其他信息部分的示例所示对其进行更改。它将添加一个变量来实现 `pg_serialize = 0` 功能。	数据库管理员、开发人员
测试转换。	将 `init` 函数部署到与 Aurora PostgreSQL 兼容数据库，然后测试结果。	数据库管理员、开发人员

相关资源

- AWS Schema Conversion Tool
- Amazon RDS
- Amazon Aurora 功能
- SERIALLY_REUSABLE Pragma

其他信息

```
Source Oracle Code:

CREATE OR REPLACE PACKAGE test_pkg_var
IS
PRAGMA SERIALLY_REUSABLE;
PROCEDURE function_1
 (test_id number);
PROCEDURE function_2
 (test_id number
 );
END;
```

```
CREATE OR REPLACE PACKAGE BODY test_pkg_var
IS
PRAGMA SERIALLY_REUSABLE;
v_char VARCHAR2(20) := 'shared.airline';
v_num number := 123;

PROCEDURE function_1(test_id number)
IS
begin
dbms_output.put_line( 'v_char-'|| v_char);
dbms_output.put_line( 'v_num-'||v_num);
v_char:='test1';
function_2(0);
END;

PROCEDURE function_2(test_id number)
is
begin
dbms_output.put_line( 'v_char-'|| v_char);
dbms_output.put_line( 'v_num-'||v_num);
END;
END test_pkg_var;

Calling the above functions

set serveroutput on


EXEC test_pkg_var.function_1(1);


EXEC test_pkg_var.function_2(1);


Target Postgresql Code:


CREATE SCHEMA test_pkg_var;

CREATE OR REPLACE FUNCTION test_pkg_var.init(pg_serialize IN INTEGER DEFAULT 0)

RETURNS void
AS
$BODY$
```

```
DECLARE

BEGIN

if aws_oracle_ext.is_package_initialized( 'test_pkg_var' ) AND pg_serialize = 0

then

return;

end if;

PERFORM aws_oracle_ext.set_package_initialized( 'test_pkg_var' );

PERFORM aws_oracle_ext.set_package_variable( 'test_pkg_var', 'v_char',
 'shared.airline.basecurrency'::CHARACTER

VARYING(100));

PERFORM aws_oracle_ext.set_package_variable('test_pkg_var', 'v_num', 123::integer);

END;

$BODY$

LANGUAGE plpgsql;


CREATE OR REPLACE FUNCTION test_pkg_var.function_1(pg_serialize int default 1)

RETURNS void
AS

$BODY$
DECLARE

BEGIN

PERFORM test_pkg_var.init(pg_serialize);

raise notice 'v_char%',aws_oracle_ext.get_package_variable( 'test_pkg_var', 'v_char');

raise notice 'v_num%',aws_oracle_ext.get_package_variable( 'test_pkg_var', 'v_num');
```

```
PERFORM aws_oracle_ext.set_package_variable( 'test_pkg_var', 'v_char',
 'test1'::varchar);

PERFORM test_pkg_var.function_2(0);
END;

$BODY$
LANGUAGE plpgsql;


CREATE OR REPLACE FUNCTION test_pkg_var.function_2(IN pg_serialize integer default 1)

RETURNS void

AS

$BODY$

DECLARE

BEGIN

PERFORM test_pkg_var.init(pg_serialize);

raise notice 'v_char%',aws_oracle_ext.get_package_variable( 'test_pkg_var', 'v_char');

raise notice 'v_num%',aws_oracle_ext.get_package_variable( 'test_pkg_var', 'v_num');

END;
$BODY$
LANGUAGE plpgsql;


Calling the above functions

select test_pkg_var.function_1()

 select test_pkg_var.function_2()
```

将 Oracle 外部表迁移到 Amazon Aurora PostgreSQL-Compatible

创建者：anuradha chintha (AWS) 和 Rakesh Raghav (AWS)

摘要

外部表使 Oracle 能够查询存储在数据库外部的平面文件中的数据。您可以使用 ORACLE_LOADER 驱动程序访问以任何格式存储的、可由 SQL*Loader 实用程序加载的任何数据。不能对外部表使用数据操作语言 (DML)，但可以使用外部表进行查询、联接和排序操作。

Amazon Aurora PostgreSQL-Compatible 不提供与 Oracle 中外部表相似的功能。相反，您必须使用现代化来开发符合功能要求且节俭的可扩展解决方案。

此模式提供了使用 aws_s3 扩展程序将不同类型的 Oracle 外部表迁移到 Amazon Web Services（AWS）Cloud 上 Aurora PostgreSQL-Compatible Edition 的步骤。

建议在生产环境中实施该解决方案之前，对方案进行全面测试。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- AWS 命令行界面（AWS CLI）
- 可用的 Aurora PostgreSQL-Compatible 数据库实例。
- 带有外部表的本地 Oracle 数据库
- pg.Client API
- 数据文件

限制

- 这种模式不提供可替代 Oracle 外部表的功能。但是，可以进一步增强步骤和示例代码，以实现数据库现代化目标。
- 文件不应包含在 aws_s3 导出和导入函数中作为分隔符传递的字符。

产品版本

- 要从 Amazon S3 导入到 RDS for PostgreSQL，数据库必须运行 PostgreSQL 版本 10.7 或更高版本。

架构

源技术堆栈

- Oracle

源架构

目标技术堆栈

- 兼容 Amazon Aurora PostgreSQL
- Amazon CloudWatch
- AWS Lambda
- AWS Secrets Manager
- Amazon Simple Notification Service(Amazon SNS)
- Amazon Simple Storage Service（Amazon S3）

目标架构

下图高度概括此解决方案。

1. 文件上传到 S3 存储桶。
2. Lambda 函数已启动。
3. Lambda 函数启动数据库函数调用。
4. Secrets Manager 提供访问数据库的凭证。
5. 根据数据库功能，创建 SNS 警报。

自动化和扩缩

对外部表的任何添加或更改都可以通过元数据维护来处理。

工具

- Amazon Aurora PostgreSQL-Compatible – Amazon Aurora PostgreSQL-Compatible Edition 是一个完全托管式、兼容 PostgreSQL 和 ACID 的关系数据库引擎，结合了高端商用数据库的速度和可靠性，同时还具有开源数据库的成本效益。
- AWS CLI – AWS 命令行界面（AWS CLI）是用于管理 Amazon Web Services 的统一工具。只通过一个工具进行下载和配置，您就可以使用命令行控制多个 Amazon Web Services 并利用脚本来自动执行这些服务。
- 亚马逊 CloudWatch — 亚马逊 CloudWatch 监控亚马逊 S3 的资源和利用率。
- AWS Lambda – AWS Lambda 是一种无服务器计算服务，支持在不预置或管理服务器的情况下运行代码、创建工作负载感知型集群扩展逻辑、维护事件集成或管理运行时。在这种模式下，每当文件上传到 Amazon S3 时，Lambda 都会运行数据库函数。
- AWS Secrets Manager – AWS Secrets Manager 是一项用于凭证存储和检索的服务。使用 Secrets Manager，您可以将代码中的硬编码凭证（包括密码）替换为对 Secrets Manager 的 API 调用，以便以编程方式检索密钥。
- Amazon S3 – Amazon Simple Storage Service（Amazon S3）提供了一个存储层，用于接收和存储文件，以便在 Aurora PostgreSQL-Compatible 集群之间使用和传输这些文件。
- aws_s3 – 该 `aws_s3` 扩展程序集成了 Amazon S3 和 Aurora PostgreSQL-Compatible。
- Amazon SNS – Amazon Simple Notification Service（Amazon SNS）可协调和管理发布者和客户端之间消息的传送或发送。在这种模式中，Amazon SNS 用于发送通知。

代码

每当在 S3 存储桶中放置文件时，都必须创建数据库函数并从处理应用程序或 Lambda 函数中调用。有关详细信息，请参阅代码（附件）。

操作说明

创建外部文件

Task	描述	所需技能
向源数据库添加外部文件。	创建外部文件，并将其移至 `oracle` 目录。	数据库管理员

配置目标 (Aurora PostgreSQL-Compatible)

Task	描述	所需技能
创建 Aurora PostgreSQL 数据库。	在您的 Amazon Aurora PostgreSQL-Compatible 集群中创建数据库实例。	数据库管理员
创建架构、aws_s3 扩展程序和表。	使用其他信息部分 `ext_tbl_scripts` 下方的代码。这些表包括实际表、暂存表、错误和日志表以及元表。	数据库管理员、开发人员
创建数据库函数。	要创建数据库函数，请使用附加信息部分 `load_external_table_latest` 函数下的代码。	数据库管理员、开发人员

创建并配置 Lambda 函数

Task	描述	所需技能
创建角色。	创建有权访问 Amazon S3 和 Amazon Relational Database Service（Amazon RDS）的角色。此角色将分配给 Lambda 以运行该模式。	数据库管理员
创建 Lambda 函数。	创建一个 Lambda 函数，该函数从 Amazon S3 读取文件名（例如 `file_key = info.get('object', {}).get('key')` ），并使用文件名作为输入参数调用数据库函数（例如 `curs.callproc("load_externa`	数据库管理员

Task	描述	所需技能
	l_tables", [file_key])) 。 根据函数调用结果，启动 SNS 通知（例如 client.publish(TopicArn='arn:',Message='fileloadsuccess',Subject='fileloadsuccess')) 。 根据您的业务需求，可以使用额外代码创建 Lambda 函数。有关更多信息，请参阅 Lambda 文档。	
配置 S3 存储桶事件触发器。	配置一种机制，以便为 S3 存储桶中的所有对象创建事件调用 Lambda 函数。	数据库管理员
创建密钥。	通过使用 Secrets Manager 为数据库凭证创建密钥名称。在 Lambda 函数中传递密钥。	数据库管理员
上传 Lambda 支持文件。	上传一个 .zip 文件，其中包含 Lambda 支持包和所附的用于连接到 Aurora PostgreSQL-Compatible 的 Python 脚本。Python 代码调用您在数据库中创建的函数。	数据库管理员
创建 SNS 主题。	创建 SNS 主题以发送有关数据加载成功或失败的邮件。	数据库管理员

添加与 Amazon S3 的集成

Task	描述	所需技能
创建 S3 存储桶。	在 Amazon S3 控制台上，创建一个 S3 存储桶。该存储桶名称具有唯一性，且不包含前导斜杠。S3 存储桶名称是全局唯一的，并且命名空间由所有 Amazon Web Services account 共享。	数据库管理员
创建 IAM policy。	要创建 AWS Identity and Access Management (IAM) 策略，请使用其他信息部分 `s3bucketpolicy_for_import` 下方的代码。	数据库管理员
创建角色。	为 Aurora PostgreSQL-Compatible 创建两个角色：一个用于导入的角色和一个用于导出的角色。为角色分配相应的策略。	数据库管理员
将角色附加到 Aurora PostgreSQL-Compatible 集群。	在管理角色下方，将导入和导出角色附加到 Aurora PostgreSQL 集群。	数据库管理员
为 Aurora PostgreSQL-Compatible 创建支持对象。	对于表格脚本，使用其他信息部分 `ext_tbl_scripts` 下方的代码。 对于定制脚本，使用其他信息部分 `load_external_Table_latest` 下方的代码。	数据库管理员

处理测试文件

Task	描述	所需技能
将文件上传到 S3 存储桶。	要将测试文件上传到 S3 存储桶，请使用控制台或在 AWS CLI 中使用以下命令。 `aws s3 cp /Users/Desktop/ukpost/exttbl/"testing files"/aps s3://s3importtest/inputext/aps` 文件上传后，存储桶事件就会启动 Lambda 函数，该函数运行 Aurora PostgreSQL-Compatible 函数。	数据库管理员
检查数据、日志和错误文件。	Aurora PostgreSQL-Compatible 函数将文件加载到主表中，然后在 S3 存储桶中创建 `.log` 和 `.bad` 文件。	数据库管理员
监控解决方案。	在亚马逊 CloudWatch 控制台中，监控 Lambda 函数。	数据库管理员

相关的资源

- Amazon S3 集成
- Amazon S3
- 使用 Amazon Aurora PostgreSQL-Compatible Edition
- AWS Lambda
- Amazon CloudWatch
- AWS Secrets Manager
- 设置 Amazon SNS 通知

其他信息

ext_table_scripts

```
CREATE EXTENSION aws_s3 CASCADE;
CREATE TABLE IF NOT EXISTS meta_EXTERNAL_TABLE
(
    table_name_stg character varying(100) ,
    table_name character varying(100)  ,
    col_list character varying(1000)  ,
    data_type character varying(100)  ,
    col_order numeric,
    start_pos numeric,
    end_pos numeric,
    no_position character varying(100)  ,
    date_mask character varying(100)  ,
    delimeter character(1)  ,
    directory character varying(100)  ,
    file_name character varying(100)  ,
    header_exist character varying(5)
);
CREATE TABLE IF NOT EXISTS ext_tbl_stg
(
    col1 text
);
CREATE TABLE IF NOT EXISTS error_table
(
    error_details text,
    file_name character varying(100),
    processed_time timestamp without time zone
);
CREATE TABLE IF NOT EXISTS log_table
(
    file_name character varying(50) COLLATE pg_catalog."default",
    processed_date timestamp without time zone,
    tot_rec_count numeric,
    proc_rec_count numeric,
    error_rec_count numeric
);
sample insert scripts of meta data:
INSERT INTO meta_EXTERNAL_TABLE (table_name_stg, table_name, col_list, data_type,
 col_order, start_pos, end_pos, no_position, date_mask, delimeter, directory,
 file_name, header_exist) VALUES ('F_EX_APS_TRANSACTIONS_STG', 'F_EX_APS_TRANSACTIONS',
```

```
 'source_filename', 'character varying', 2, 8, 27, NULL, NULL, NULL, 'databasedev',
 'externalinterface/loaddir/APS', 'NO');
INSERT INTO meta_EXTERNAL_TABLE (table_name_stg, table_name, col_list, data_type,
 col_order, start_pos, end_pos, no_position, date_mask, delimeter, directory,
 file_name, header_exist) VALUES ('F_EX_APS_TRANSACTIONS_STG', 'F_EX_APS_TRANSACTIONS',
 'record_type_identifier', 'character varying', 3, 28, 30, NULL, NULL, NULL,
 'databasedev', 'externalinterface/loaddir/APS', 'NO');
INSERT INTO meta_EXTERNAL_TABLE (table_name_stg, table_name, col_list, data_type,
 col_order, start_pos, end_pos, no_position, date_mask, delimeter, directory,
 file_name, header_exist) VALUES ('F_EX_APS_TRANSACTIONS_STG', 'F_EX_APS_TRANSACTIONS',
 'fad_code', 'numeric', 4, 31, 36, NULL, NULL, NULL, 'databasedev', 'externalinterface/
loaddir/APS', 'NO');
INSERT INTO meta_EXTERNAL_TABLE (table_name_stg, table_name, col_list, data_type,
 col_order, start_pos, end_pos, no_position, date_mask, delimeter, directory,
 file_name, header_exist) VALUES ('F_EX_APS_TRANSACTIONS_STG', 'F_EX_APS_TRANSACTIONS',
 'session_sequence_number', 'numeric', 5, 37, 42, NULL, NULL, NULL, 'databasedev',
 'externalinterface/loaddir/APS', 'NO');
INSERT INTO meta_EXTERNAL_TABLE (table_name_stg, table_name, col_list, data_type,
 col_order, start_pos, end_pos, no_position, date_mask, delimeter, directory,
 file_name, header_exist) VALUES ('F_EX_APS_TRANSACTIONS_STG', 'F_EX_APS_TRANSACTIONS',
 'transaction_sequence_number', 'numeric', 6, 43, 48, NULL, NULL, NULL, 'databasedev',
 'externalinterface/loaddir/APS', 'NO');
```

s3bucketpolicy_for import

```
---Import role policy
--Create an IAM policy to allow, Get,  and list actions on S3 bucket
 {
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "s3import",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::s3importtest",
                "arn:aws:s3:::s3importtest/*"
            ]
        }
    ]
```

```
}
--Export Role policy
--Create an IAM policy to allow, put,  and list actions on S3 bucket
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "s3export",
            "Action": [
                "S3:PutObject",
                "s3:ListBucket"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::s3importtest/*"
            ]
        }
    ]
}
```

示例数据库函数 load_external_tables_latest

```
CREATE OR REPLACE FUNCTION public.load_external_tables(pi_filename text)
 RETURNS character varying
 LANGUAGE plpgsql
AS $function$
/* Loading data from S3 bucket into a APG table */
DECLARE
 v_final_sql TEXT;
 pi_ext_table TEXT;
 r refCURSOR;
 v_sqlerrm text;
 v_chunk numeric;
 i integer;
 v_col_list TEXT;
 v_postion_list CHARACTER VARYING(1000);
 v_len  integer;
 v_delim varchar;
 v_file_name CHARACTER VARYING(1000);
 v_directory CHARACTER VARYING(1000);
 v_table_name_stg CHARACTER VARYING(1000);
 v_sql_col TEXT;
 v_sql TEXT;
```

```
 v_sql1 TEXT;
 v_sql2 TEXT;
 v_sql3 TEXT;
 v_cnt integer;
 v_sql_dynamic TEXT;
 v_sql_ins TEXT;
 proc_rec_COUNT integer;
 error_rec_COUNT integer;
 tot_rec_COUNT integer;
 v_rec_val integer;
 rec record;
 v_col_cnt integer;
 kv record;
 v_val text;
 v_header text;
 j integer;
 ERCODE VARCHAR(5);
 v_region text;
 cr CURSOR FOR
 SELECT distinct DELIMETER,
   FILE_NAME,
   DIRECTORY
 FROM  meta_EXTERNAL_TABLE
 WHERE table_name = pi_ext_table
   AND DELIMETER IS NOT NULL;


 cr1 CURSOR FOR
   SELECT   col_list,
   data_type,
   start_pos,
   END_pos,
   concat_ws('',' ',TABLE_NAME_STG) as TABLE_NAME_STG,
   no_position,date_mask
 FROM  meta_EXTERNAL_TABLE
 WHERE table_name = pi_ext_table
 order by col_order asc;
cr2 cursor FOR
SELECT  distinct table_name,table_name_stg
   FROM  meta_EXTERNAL_TABLE
   WHERE upper(file_name) = upper(pi_filename);


BEGIN
```

```
 -- PERFORM utl_file_utility.init();
   v_region := 'us-east-1';
   /* find tab details from file name */


   --DELETE FROM  ERROR_TABLE WHERE file_name= pi_filename;
  -- DELETE FROM  log_table WHERE file_name= pi_filename;


 BEGIN


   SELECT distinct table_name,table_name_stg INTO strict pi_ext_table,v_table_name_stg
   FROM  meta_EXTERNAL_TABLE
   WHERE upper(file_name) = upper(pi_filename);
 EXCEPTION
   WHEN NO_DATA_FOUND THEN
    raise notice 'error 1,%',sqlerrm;
    pi_ext_table := null;
    v_table_name_stg := null;
      RAISE USING errcode = 'NTFIP' ;
    when others then
        raise notice 'error others,%',sqlerrm;
 END;
 j :=1 ;

for rec in  cr2
 LOOP




  pi_ext_table     := rec.table_name;
  v_table_name_stg := rec.table_name_stg;
  v_col_list := null;


 IF pi_ext_table IS NOT NULL
  THEN
    --EXECUTE concat_ws('','truncate table  ' ,pi_ext_table) ;
   EXECUTE concat_ws('','truncate table  ' ,v_table_name_stg) ;
```

```
      SELECT distinct DELIMETER INTO STRICT v_delim
      FROM  meta_EXTERNAL_TABLE
      WHERE table_name = pi_ext_table;


      IF v_delim IS NOT NULL THEN
    SELECT distinct DELIMETER,
      FILE_NAME,
      DIRECTORY ,
      concat_ws('',' ',table_name_stg),
      case  header_exist when 'YES' then 'CSV HEADER' else 'CSV' end as header_exist
    INTO STRICT v_delim,v_file_name,v_directory,v_table_name_stg,v_header
    FROM  meta_EXTERNAL_TABLE
    WHERE table_name = pi_ext_table
      AND DELIMETER IS NOT NULL;


    IF    upper(v_delim) = 'CSV'
    THEN
      v_sql := concat_ws('','SELECT aws_s3.table_import_FROM_s3 ( ''',
      v_table_name_stg,''','''',
      ''DELIMITER '''','''' CSV HEADER QUOTE ''''"''''''', aws_commons.create_s3_uri
( ''',
      v_directory,''','''',v_file_name,''', ''',v_region,'''))');
      ELSE
      v_sql := concat_ws('','SELECT aws_s3.table_import_FROM_s3(''',
          v_table_name_stg, ''','''', ''DELIMITER AS ''''^''''',''',','
         aws_commons.create_s3_uri
          ( ''',v_directory, ''','''',
          v_file_name, ''',',
           '''',v_region,''')
         )');
         raise notice 'v_sql , %',v_sql;
      begin
       EXECUTE  v_sql;
      EXCEPTION
        WHEN OTHERS THEN
          raise notice 'error 1';
        RAISE USING errcode = 'S3IMP' ;
      END;


      select count(col_list) INTO v_col_cnt
```

```
      from  meta_EXTERNAL_TABLE where table_name = pi_ext_table;









       -- raise notice 'v_sql 2, %',concat_ws('','update ',v_table_name_stg, ' set
col1 = col1||''',v_delim,''''');


      execute concat_ws('','update ',v_table_name_stg, ' set col1 =
col1||''',v_delim,''''');




      i :=1;
      FOR rec in cr1
      loop
      v_sql1 := concat_ws('',v_sql1,'split_part(col1,''',v_delim,''',', i,')',' as
',rec.col_list,',');
      v_sql2 := concat_ws('',v_sql2,rec.col_list,',');
  --    v_sql3 := concat_ws('',v_sql3,'rec.',rec.col_list,'::',rec.data_type,',');


      case
        WHEN upper(rec.data_type) = 'NUMERIC'
        THEN v_sql3 := concat_ws('',v_sql3,' case WHEN
length(trim(split_part(col1,''',v_delim,''',', i,'))) =0
               THEN null
                ELSE
                coalesce((trim(split_part(col1,''',v_delim,''',',
i,')))::NUMERIC,0)::',rec.data_type,' END as ',rec.col_list,',') ;
        WHEN UPPER(rec.data_type) = 'TIMESTAMP WITHOUT TIME ZONE' AND rec.date_mask =
'YYYYMMDD'
        THEN v_sql3 := concat_ws('',v_sql3,' case WHEN
length(trim(split_part(col1,''',v_delim,''',', i,'))) =0
               THEN null
                ELSE
                to_date(coalesce((trim(split_part(col1,''',v_delim,''',',
i,'))),''99990101''),''YYYYMMDD'')::',rec.data_type,' END as ',rec.col_list,',');
        WHEN UPPER(rec.data_type) = 'TIMESTAMP WITHOUT TIME ZONE' AND rec.date_mask =
 'MM/DD/YYYY hh24:mi:ss'
```

```
        THEN v_sql3 := concat_ws('',v_sql3,' case WHEN
length(trim(split_part(col1,''',v_delim,''',', i,'))) =0
               THEN null
                ELSE
                to_date(coalesce((trim(split_part(col1,''',v_delim,''',',
i,'))),''01/01/9999 0024:00:00''),''MM/DD/YYYY hh24:mi:ss'')::',rec.data_type,' END as
',rec.col_list,',');
         ELSE
       v_sql3 := concat_ws('',v_sql3,' case WHEN
length(trim(split_part(col1,''',v_delim,''',', i,'))) =0
               THEN null
                ELSE
                 coalesce((trim(split_part(col1,''',v_delim,''',',
i,'))),''''):',rec.data_type,' END as ',rec.col_list,',') ;
      END case;


      i :=i+1;
      end loop;


        -- raise notice 'v_sql 3, %',v_sql3;


      SELECT trim(trailing ' ' FROM v_sql1) INTO v_sql1;
      SELECT trim(trailing ',' FROM v_sql1) INTO v_sql1;


      SELECT trim(trailing ' ' FROM v_sql2) INTO v_sql2;
      SELECT trim(trailing ',' FROM v_sql2) INTO v_sql2;


      SELECT trim(trailing ' ' FROM v_sql3) INTO v_sql3;
      SELECT trim(trailing ',' FROM v_sql3) INTO v_sql3;


      END IF;
     raise notice 'v_delim , %',v_delim;


    EXECUTE concat_ws('','SELECT COUNT(*) FROM ',v_table_name_stg)  INTO v_cnt;


   raise notice 'stg cnt , %',v_cnt;
```

```
    /* if upper(v_delim) = 'CSV' then
       v_sql_ins := concat_ws('', ' SELECT * from ' ,v_table_name_stg );
     else
      -- v_sql_ins := concat_ws('',' SELECT ',v_sql1,'  from (select col1 from
 ' ,v_table_name_stg , ')sub ');
       v_sql_ins := concat_ws('',' SELECT ',v_sql3,'  from (select col1 from
 ' ,v_table_name_stg , ')sub ');
       END IF;*/


v_chunk := v_cnt/100;




for i in 1..101
loop
     BEGIN
    -- raise notice 'v_sql , %',v_sql;
       -- raise notice 'Chunk number , %',i;
       v_sql_ins := concat_ws('',' SELECT ',v_sql3,'  from (select col1 from
 ' ,v_table_name_stg , ' offset ',v_chunk*(i-1), ' limit ',v_chunk,') sub ');


     v_sql := concat_ws('','insert into  ', pi_ext_table ,' ', v_sql_ins);
     -- raise notice 'select statement , %',v_sql_ins;
          -- v_sql := null;
     -- EXECUTE concat_ws('','insert into  ', pi_ext_table ,' ', v_sql_ins, 'offset
 ',v_chunk*(i-1), ' limit ',v_chunk );
     --v_sql := concat_ws('','insert into  ', pi_ext_table ,' ', v_sql_ins );


     -- raise notice 'insert statement , %',v_sql;


    raise NOTICE 'CHUNK START %',v_chunk*(i-1);
   raise NOTICE 'CHUNK END %',v_chunk;


     EXECUTE v_sql;
```

```
 EXCEPTION
      WHEN OTHERS THEN
      -- v_sql_ins := concat_ws('',' SELECT ',v_sql1, '  from (select col1 from
' ,v_table_name_stg , ' )sub ');
        -- raise notice 'Chunk number for cursor , %',i;


   raise NOTICE 'Cursor - CHUNK START %',v_chunk*(i-1);
  raise NOTICE 'Cursor -  CHUNK END %',v_chunk;
        v_sql_ins := concat_ws('',' SELECT ',v_sql3, '  from (select col1 from
' ,v_table_name_stg , ' )sub ');


        v_final_sql := REPLACE (v_sql_ins, ''''::text, ''''''::text);
       -- raise notice 'v_final_sql %',v_final_sql;
        v_sql :=concat_ws('','do $a$ declare  r refcursor;v_sql text; i
numeric;v_conname text;  v_typ  ',pi_ext_table,'[]; v_rec  ','record',';
          begin









          open r for execute ''select col1 from ',v_table_name_stg ,'  offset
',v_chunk*(i-1), ' limit ',v_chunk,''';
          loop
          begin
          fetch r into v_rec;
          EXIT WHEN NOT FOUND;




          v_sql := concat_ws('''','insert into  ',pi_ext_table,' SELECT ',REPLACE
(v_sql3, ''''::text, ''''''::text) , '  from ( select ''''''',v_rec.col1,''''''' as
col1) v'');
           execute v_sql;


          exception
           when others then
         v_sql := ''INSERT INTO  ERROR_TABLE VALUES (concat_ws(''''''''','''Error
Name: ''',$$''||SQLERRM||''$$,'''Error State: ''',''''''||
```

```
SQLSTATE||''''',''''record : '''',$$''||v_rec.col1||''$$),'''''||
pi_filename||''''',now())'';


              execute v_sql;
            continue;
          end ;
          end loop;
          close r;
          exception
          when others then
        raise;
          end ; $a$');
     -- raise notice ' inside excp v_sql %',v_sql;
         execute v_sql;
     --  raise notice 'v_sql %',v_sql;
      END;
  END LOOP;
     ELSE


     SELECT distinct DELIMETER,FILE_NAME,DIRECTORY ,concat_ws('',' ',table_name_stg),
       case  header_exist when 'YES' then 'CSV HEADER' else 'CSV' end as header_exist
       INTO STRICT v_delim,v_file_name,v_directory,v_table_name_stg,v_header
     FROM  meta_EXTERNAL_TABLE
     WHERE table_name = pi_ext_table                  ;
     v_sql := concat_ws('','SELECT aws_s3.table_import_FROM_s3(''',
       v_table_name_stg, ''','''', ''DELIMITER AS ''''#'''' ',v_header,' '','','
      aws_commons.create_s3_uri
       ( ''',v_directory, ''','''',
       v_file_name, ''',',
        '''',v_region,''')
      )');
         EXECUTE  v_sql;


     FOR rec in cr1
     LOOP


      IF rec.start_pos IS NULL AND rec.END_pos IS NULL AND rec.no_position = 'recnum'
      THEN
        v_rec_val := 1;
      ELSE
```

```
      case
        WHEN upper(rec.data_type) = 'NUMERIC'
        THEN v_sql1 := concat_ws('',' case WHEN length(trim(substring(COL1,
',rec.start_pos ,',', rec.END_pos,'-',rec.start_pos ,'+1))) =0
               THEN null
                ELSE
                coalesce((trim(substring(COL1, ',rec.start_pos ,',',
rec.END_pos,'-',rec.start_pos ,'+1')))::NUMERIC,0)::',rec.data_type,' END as
',rec.col_list,',') ;
        WHEN UPPER(rec.data_type) = 'TIMESTAMP WITHOUT TIME ZONE' AND rec.date_mask =
'YYYYMMDD'
        THEN v_sql1 := concat_ws('','case WHEN length(trim(substring(COL1,
',rec.start_pos ,',', rec.END_pos,'-',rec.start_pos ,'+1))) =0
               THEN null
                ELSE
                to_date(coalesce((trim(substring(COL1, ',rec.start_pos ,',',
rec.END_pos,'-',rec.start_pos ,'+1'))),''99990101''),''YYYYMMDD''):',rec.data_type,'
END as ',rec.col_list,',');
        WHEN UPPER(rec.data_type) = 'TIMESTAMP WITHOUT TIME ZONE' AND rec.date_mask =
'YYYYMMDDHH24MISS'
        THEN v_sql1 := concat_ws('','case WHEN length(trim(substring(COL1,
',rec.start_pos ,',', rec.END_pos,'-',rec.start_pos ,'+1))) =0
               THEN null
                ELSE
                to_date(coalesce((trim(substring(COL1, ',rec.start_pos ,',',
rec.END_pos,'-',rec.start_pos ,'+1'))),''9999010100240000''),''YYYYMMDDHH24MISS''):',rec.data_
END as ',rec.col_list,',');
         ELSE
       v_sql1 := concat_ws('',' case WHEN length(trim(substring(COL1,
',rec.start_pos ,',', rec.END_pos,'-',rec.start_pos ,'+1))) =0
               THEN null
                ELSE
                 coalesce((trim(substring(COL1, ',rec.start_pos ,',',
rec.END_pos,'-',rec.start_pos ,'+1'))),''''):',rec.data_type,' END as
',rec.col_list,',') ;
      END case;


     END IF;
     v_col_list := concat_ws('',v_col_list ,v_sql1);
    END LOOP;
```

```
          SELECT trim(trailing ' ' FROM v_col_list) INTO v_col_list;
          SELECT trim(trailing ',' FROM v_col_list) INTO v_col_list;


          v_sql_col   :=  concat_ws('',trim(trailing ',' FROM v_col_list) , ' FROM
 ',v_table_name_stg,' WHERE col1 IS NOT NULL AND length(col1)>0 ');




          v_sql_dynamic := v_sql_col;


          EXECUTE  concat_ws('','SELECT COUNT(*) FROM ',v_table_name_stg) INTO v_cnt;




        IF v_rec_val = 1 THEN
            v_sql_ins := concat_ws('',' select row_number() over(order by ctid) as
line_number ,' ,v_sql_dynamic) ;


        ELSE
              v_sql_ins := concat_ws('',' SELECT' ,v_sql_dynamic) ;
          END IF;


    BEGIN
      EXECUTE concat_ws('','insert into  ', pi_ext_table ,' ', v_sql_ins);
          EXCEPTION
             WHEN OTHERS THEN
         IF v_rec_val = 1 THEN
                v_final_sql := ' select row_number() over(order by ctid) as
line_number ,col1 from ';
               ELSE
                v_final_sql := ' SELECT col1 from';
              END IF;
      v_sql :=concat_ws('','do $a$ declare  r refcursor;v_rec_val numeric :=
',coalesce(v_rec_val,0),';line_number numeric; col1 text; v_typ  ',pi_ext_table,'[];
v_rec  ',pi_ext_table,';
```

```
            begin
            open r for execute ''',v_final_sql, ' ',v_table_name_stg,' WHERE col1 IS
NOT NULL AND length(col1)>0 '' ;
            loop
            begin
            if   v_rec_val = 1 then
            fetch r into line_number,col1;
            else
            fetch r into col1;
            end if;


            EXIT WHEN NOT FOUND;
             if v_rec_val = 1 then
             select line_number,',trim(trailing ',' FROM v_col_list) ,' into v_rec;
             else
               select ',trim(trailing ',' FROM v_col_list) ,' into v_rec;
             end if;


            insert into  ',pi_ext_table,' select v_rec.*;
             exception
             when others then
              INSERT INTO  ERROR_TABLE VALUES (concat_ws(''''',''Error Name:
'',SQLERRM,''Error State: '',SQLSTATE,''record : '',v_rec),''',pi_filename,''',now());
              continue;
             end ;
              end loop;
            close r;
             exception
             when others then
             raise;
             end ; $a$');
        execute v_sql;


    END;


        END IF;


  EXECUTE concat_ws('','SELECT COUNT(*) FROM  ' ,pi_ext_table)   INTO proc_rec_COUNT;
```

```
   EXECUTE concat_ws('','SELECT COUNT(*) FROM  error_table WHERE file_name
 ='''',pi_filename,''' and processed_time::date = clock_timestamp()::date')  INTO
 error_rec_COUNT;


   EXECUTE concat_ws('','SELECT COUNT(*) FROM ',v_table_name_stg)   INTO tot_rec_COUNT;


   INSERT INTO  log_table values(pi_filename,now(),tot_rec_COUNT,proc_rec_COUNT,
 error_rec_COUNT);


   raise notice 'v_directory, %',v_directory;


   raise notice 'pi_filename, %',pi_filename;


   raise notice 'v_region, %',v_region;


  perform aws_s3.query_export_to_s3('SELECT
 replace(trim(substring(error_details,position(''('' in
 error_details)+1),'')''),'','','';''),file_name,processed_time FROM  error_table WHERE
 file_name = '''||pi_filename||'''',
   aws_commons.create_s3_uri(v_directory, pi_filename||'.bad', v_region),
   options :='FORmat csv, header, delimiter $$,$$'
   );


raise notice 'v_directory, %',v_directory;


   raise notice 'pi_filename, %',pi_filename;


   raise notice 'v_region, %',v_region;


  perform aws_s3.query_export_to_s3('SELECT * FROM  log_table WHERE file_name = '''||
pi_filename||'''',
   aws_commons.create_s3_uri(v_directory, pi_filename||'.log', v_region),
   options :='FORmat csv, header, delimiter $$,$$'
```

```
   );




   END IF;
 j := j+1;
 END LOOP;


       RETURN 'OK';
EXCEPTION
    WHEN  OTHERS THEN
  raise notice 'error %',sqlerrm;
   ERCODE=SQLSTATE;
   IF ERCODE = 'NTFIP' THEN
     v_sqlerrm := concat_Ws('',sqlerrm,'No data for the filename');
   ELSIF ERCODE = 'S3IMP' THEN
    v_sqlerrm := concat_Ws('',sqlerrm,'Error While exporting the file from S3');
   ELSE
      v_sqlerrm := sqlerrm;
   END IF;


 select distinct directory into v_directory from  meta_EXTERNAL_TABLE;




 raise notice 'exc v_directory, %',v_directory;


   raise notice 'exc pi_filename, %',pi_filename;


   raise notice 'exc v_region, %',v_region;


  perform aws_s3.query_export_to_s3('SELECT * FROM  error_table WHERE file_name = '''||
pi_filename||'''',
   aws_commons.create_s3_uri(v_directory, pi_filename||'.bad', v_region),
   options :='FORmat csv, header, delimiter $$,$$'
   );
    RETURN null;
```

```
END;
$function$
```

将基于函数的索引从 Oracle 迁移到 PostgreSQL

创建者：Veeranjaneyulu Grandhi (AWS) 和 Navakanth Talluri (AWS)

摘要

索引是增强数据库性能的常用方法。索引允许数据库服务器比无索引时更快地查找和检索特定行。但索引也会增加整个数据库系统的开销，因此应该明智地使用它们。基于函数的索引基于函数或表达式，可以涉及多个列和数学表达式。基于函数的索引可提高使用索引表达式的查询的性能。

本质上，PostgreSQL 不支持使用将波动性定义为稳定的函数创建基于函数的索引。但是，您可创建波动性为 `IMMUTABLE` 的类似函数，并在创建指数时使用它们。

`IMMUTABLE` 函数无法修改数据库，并且在给定相同参数的情况下，可以保证永远返回相同的结果。当查询使用常量参数调用函数时，此类别允许优化程序预先对函数求值。

当与 `to_char`、`to_date` 和 `to_number` 等函数一起使用时，这种模式有助于将基于 Oracle 函数的索引迁移到 PostgreSQL 等效版本。

先决条件和限制

先决条件

- 一个活动 Amazon Web Services (AWS) 账户
- 已设置并运行侦听器服务的源 Oracle 数据库实例
- 熟悉 PostgreSQL 数据库

限制

- 数据库大小限制为 64 TB。
- 创建索引时使用的函数必须是 IMMUTABLE。

产品版本

- 版本 11g（版本 11.2.0.3.v1 及更高版本）以及最高 12.2 和 18c 的所有 Oracle 数据库版本
- PostgreSQL 版本 9.6 及更高版本

架构

源技术堆栈

- 本地或亚马逊弹性计算云 (Amazon EC2) 实例上的 Oracle 数据库，或者适用于 Oracle 的 Amazon RDS 数据库实例

目标技术堆栈

- 任何 PostgreSQL 引擎

工具

- pgAdmin 4 是一种适用于 Postgres 的开源管理工具。pgAdmin 4 工具提供了用于创建、维护和使用数据库对象的图形界面。
- Oracle SQL Developer 是一个集成式开发环境（IDE），用于在传统部署和云部署中开发和管理 Oracle 数据库。

操作说明

使用默认函数创建基于函数索引

Task	描述	所需技能

使用 to_char 函数在列上创建基于函数的索引。	使用以下代码创建基于函数的索引。
	`postgres=# create table funcindex( col1 timestamp without time zone);`
	`CREATE TABLE`
	`postgres=# insert into funcindex values (now());`
	`INSERT 0 1`
	`postgres=# select * from funcindex;`
	`col1`
	`----------`
	`2022-08-09 16:00:57.77414`
		数据库管理员，应用程序开发人员

Task	描述	所需技能
	(1 rows) postgres=# create index funcindex_idx on funcindex(to_char(col1,'DD-MM-YYYY HH24:MI:SS')); ERROR: functions in index expression must be marked IMMUTABLE **Note** PostgreSQL 不允许在没有子句的情况下创建基于函数的索引。IMMUTABLE	
检查函数的波动性。	要检查函数的波动性，请使用其他信息部分中的代码。	数据库管理员

使用包装函数创建基于函数索引

Task	描述	所需技能
创建包装函数。	要创建包装函数，请使用其他信息部分中的代码。	PostgreSQL 开发人员
使用包装函数创建索引。	使用其他信息部分中的代码创建用户定义的函数，其关键字 IMMUTABLE 与应用程序处于相同的架构中，并在索引创建脚本中引用该函数。	数据库管理员、PostgreSQL 开发人员

Task	描述	所需技能
	如果用户定义的函数是在通用架构中创建的(来自前面的示例)，请按所示更新 `search_path` 。 `ALTER ROLE <ROLENAME> set search_path=$user, COMMON;`	

验证索引创建

Task	描述	所需技能
验证索引创建。	根据查询访问模式验证是否需要创建索引。	数据库管理员
验证索引是否可以使用。	要检查基于函数的索引是否由 PostgreSQL 优化器获取，请使用解释或解释分析运行 SQL 语句。使用其他信息部分中的代码。如有可能，还要收集表格统计信息。 **Note** 如果您注意到解释计划，PostgreSQL 优化器会因为谓词条件而选择了基于函数的索引。	数据库管理员

相关的资源

- 基于函数的索引（Oracle 文档）
- 表达式索引（PostgreSQL 文档）

- PostgreSQL 波动性（PostgreS QL 文档）
- PostgreSQL 搜索路径（PostgreSQL 文档）
- Oracle Database 19c 至 Amazon Aurora PostgreSQL 迁移行动手册

其他信息

创建包装函数

```
CREATE OR REPLACE FUNCTION myschema.to_char(var1 timestamp without time zone, var2
 varchar) RETURNS varchar AS $BODY$ select to_char(var1, 'YYYYMMDD'); $BODY$ LANGUAGE
 sql IMMUTABLE;
```

使用包装函数创建索引

```
postgres=# create function common.to_char(var1 timestamp without time zone, var2
 varchar) RETURNS varchar AS $BODY$ select to_char(var1, 'YYYYMMDD'); $BODY$ LANGUAGE
 sql IMMUTABLE;
CREATE FUNCTION
postgres=# create index funcindex_idx on funcindex(common.to_char(col1,'DD-MM-YYYY
 HH24:MI:SS'));
CREATE INDEX
```

检查函数的波动性

```
SELECT DISTINCT p.proname as "Name",p.provolatile as "volatility" FROM
 pg_catalog.pg_proc p
 LEFT JOIN pg_catalog.pg_namespace n ON n.oid = p.pronamespace
 LEFT JOIN pg_catalog.pg_language l ON l.oid = p.prolang
 WHERE n.nspname OPERATOR(pg_catalog.~) '^(pg_catalog)$' COLLATE pg_catalog.default AND
 p.proname='to_char'GROUP BY p.proname,p.provolatile
ORDER BY 1;
```

验证索引是否可以使用

```
explain analyze <SQL>


postgres=# explain select col1 from funcindex where common.to_char(col1,'DD-MM-YYYY
 HH24:MI:SS') = '09-08-2022 16:00:57';
                                                    QUERY PLAN
```

```
-------------------------------------------------------------------------------------------------------
 Index Scan using funcindex_idx on funcindex  (cost=0.42..8.44 rows=1 width=8)
   Index Cond: ((common.to_char(col1, 'DD-MM-YYYY HH24:MI:SS'::character
 varying))::text = '09-08-2022 16:00:57'::text)
(2 rows)
```

使用扩展将 Oracle 原生函数迁移到 PostgreSQL

由 Pinesh Singal (AWS) 编写

摘要

此迁移模式通过修改 PostgreSQL () 原生内置代码和orafce扩展，为将适用于 Oracle 的亚马逊关系数据库服务 (Amazon RDS) 数据库实例迁移到亚马逊 RDS for PostgreSQL 或兼容 Amazon Aurora PostgreSQL 的aws_oracle_ext版本数据库提供了 step-by-step指导。psql这将会节省处理时间。

该模式描述了一种离线手动迁移策略，对于具有大量事务的多 TB Oracle 源数据库，无需停机。

迁移过程使用具有 aws_oracle_ext 和 orafce 扩展的 AWS Schema Conversion Tool (AWS SCT) 将 Amazon RDS for Oracle 数据库架构转换为 Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 兼容数据库架构。然后，将代码手动更改为 PostgreSQL 支持的原生 psql 内置代码。这是因为扩展调用会影响 PostgreSQL 数据库服务器上的代码处理，而且并非所有扩展代码都完全符合 PostgreSQL 代码或与 PostgreSQL 代码兼容。

这种模式主要侧重于使用 AWS SCT 以及扩展 aws_oracle_ext 和 orafce 手动迁移 SQL 代码。您可以将已使用的扩展转换为原生 PostgreSQL (psql) 内置插件。然后，删除对扩展的所有引用，并相应地转换代码。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 操作系统（Windows 或 Mac）或亚马逊 EC2 实例（已启动并正在运行）
- Orafce

限制

并非所有使用 aws_oracle_ext 或 orafce 扩展的 Oracle 函数都可以转换为原生 PostgreSQL 函数。它可能需要手动返工才能使用 PostgreSQL 库进行编译。

使用 AWS SCT 扩展的一个缺点是它在运行和获取结果方面的性能较慢。其成本可以从简单的 PostgreSQL EXPLAIN plan 计划（语句的执行计划）中理解，该计划涉及所有三个代码（aws_oracle_ext、orafce 和 psql 默认）之间的 Oracle SYSDATE 函数迁移到 PostgreSQL NOW() 函数，如所附文档中的性能比较检查部分所述。

产品版本

- 源：Amazon RDS for Oracle 数据库 10.2 及更高版本（适用于 10.x）、11g（11.2.0.3.v1 及更高版本）以及适用于企业版、标准版、标准版 1 的最高 12.2、18c 和 19c（及更高版本），和标准版 2
- 目标：Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 兼容数据库 9.4 及更高版本（适用于 9.x）、10.x、11.x、12.x、13.x 和 14.x（及更高版本）
- AWS SCT：最新版本（此模式已使用 1.0.632 进行了测试）
- Orafce：最新版本（此模式已使用 3.9.0 进行了测试）

架构

源技术堆栈

- 版本 12.1.0.2.v18 的 Amazon RDS for Oracle 数据库实例

目标技术堆栈

- 版本 11.5 的“Amazon RDS for PostgreSQL”或 Aurora PostgreSQL 兼容数据库实例

数据库迁移架构

下图展示了源 Oracle 数据库和目标 PostgreSQL 数据库之间的数据库迁移架构。该架构涉及 AWS Cloud、虚拟私有云（VPC）、可用区、私有子网、Amazon RDS for Oracle 数据库、AWS SCT、Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 兼容数据库、Oracle 扩展（`aws_oracle_ext` 和 `orafce`）以及结构化查询语言（SQL）文件。

1. 启动 Amazon RDS for Oracle 数据库实例（源数据库）。
2. 使用 AWS SCT 以及 `aws_oracle_ext` 和 `orafce` 扩展包将源代码从 Oracle 转换为 PostreSQL。
3. 转换生成 PostgreSQL 支持的已迁移的 .sql 文件。
4. 手动将未转换的 Oracle 扩展代码转换为 PostgreSQL (`psql`) 代码。
5. 手动转换生成 PostgreSQL 支持的已转换的 .sql 文件。
6. 在 Amazon RDS for PostgreSQL 数据库实例（目标数据库）上运行这些.sql 文件。

工具

工具

Amazon Web Services

- AWS SCT - AWS Schema Conversion Tool (AWS SCT) 可将您的现有数据库架构从一个数据库引擎转换为另一个数据库引擎。您可以转换关系型联机事务处理（OLTP）模式或数据仓库模式。转换后的架构适用于 Amazon RDS for MySQL 数据库实例、Amazon Aurora 数据库集群、 Amazon RDS for PostgreSQL 数据库实例或 Amazon Redshift 集群。转换后的架构也可以与 Amazon EC2 实例上的数据库一起使用，或者作为数据存储在 Amazon S3 存储桶中。

  AWS SCT 提供基于项目的用户界面，以便将您的源数据库的数据库架构自动转换为兼容目标 Amazon RDS 实例的格式。

  您可以使用 AWS SCT 从 Oracle 源数据库迁移到前面列出的任何目标。使用 AWS SCT，您可以导出源数据库对象定义，例如架构、视图、存储过程和函数。

  您可以使用 AWS SCT 将数据从 Oracle 转换到 Amazon RDS for PostgreSQL 或 Amazon Aurora PostgreSQL 兼容版。

  在此模式中，您使用 AWS SCT 使用扩展 `aws_oracle_ext` 和 `orafce` 将 Oracle 代码转换并迁移到 PostgreSQL，并手动将扩展代码迁移到 `psql` 默认代码或原生内置代码。
- AWS SCT 扩展包是一个附加模块，用于模拟源数据库中存在的特定函数（将对象转换为目标数据库时需要）。您需要先转换数据库架构，然后才能安装 AWS SCT 扩展包。

  转换数据库或数据仓库架构时，AWS SCT 会向您的目标数据库添加一个额外的架构。该架构用于实现将转换后的架构写入到目标数据库时必需的源数据库的 SQL 系统功能。这个额外的架构称为扩展包架构。

  OLTP 数据库的扩展包架构按照源数据库命名。对于 Oracle 数据库，扩展包架构为 `AWS_ORACLE_EXT`。

其他工具

- Orafce — Oracce 是一个实现 Oracle 兼容函数、数据类型和包的模块。它是一个具有伯克利源代码分发 (BSD) 许可证的开源工具，因此任何人都可以使用它。`orafce` 模块对于从 Oracle 迁移到 PostgreSQL 很有用，因为它在 PostgreSQL 中实现了许多 Oracle 函数。

代码

有关为避免使用 AWS SCT 扩展代码而从 Oracle 迁移到 PostgreSQL 的所有常用代码和迁移代码的列表，请参阅随附的文档。

操作说明

配置 Amazon RDS for Oracle 源数据库

Task	描述	所需技能
创建 Oracle 数据库实例。	从 Amazon RDS 控制台创建 Amazon RDS for Oracle 或 Aurora PostgreSQL 兼容的数据库实例。	常规 AWS、数据库管理员
配置安全组。	配置入站和出站安全组。	常规 AWS
创建数据库。	创建包含所需用户和架构的 Oracle 数据库。	常规 AWS、数据库管理员
创建对象。	创建对象并在架构中插入数据。	数据库管理员

配置 Amazon RDS for PostgreSQL 目标数据库

Task	描述	所需技能
创建 PostgreSQL 数据库实例。	从 Amazon RDS 控制台创建 Amazon RDS for PostgreSQL 或 Amazon Aurora PostgreSQL 数据库实例。	常规 AWS、数据库管理员
配置安全组。	配置入站和出站安全组。	常规 AWS
创建数据库。	创建包含所需用户和架构的 PostgreSQL 数据库。	常规 AWS、数据库管理员

Task	描述	所需技能
验证扩展。	确保 `aws_oracle_ext` 和 `orafce` 已在 PostgreSQL 数据库中正确安装和配置。	数据库管理员
验证 PostgreSQL 数据库是否可用。	确保 PostgreSQL 数据库已启动并正在运行。	数据库管理员

使用 AWS SCT 和扩展将 Oracle 架构迁移到 PostgreSQL

Task	描述	所需技能
安装 AWS SCT。	安装 AWS SCT 的最新版本。	数据库管理员
配置 AWS SCT。	使用适用于 Oracle (`ojdbc8.jar`) 和 PostgreSQL (`postgresql-42.2.5.jar`) 的 Java 数据库连接 (JDBC) 驱动程序配置 AWS SCT。	数据库管理员
启用 AWS SCT 扩展包或模板。	在 AWS SCT 项目设置下，使用 Oracle 数据库架构的 `aws_oracle_ext` 和 `orafce` 扩展启用内置函数实现。	数据库管理员
转换架构。	在 AWS SCT 中，选择转换架构将架构从 Oracle 转换为 PostgreSQL 并生成 .sql 文件。	数据库管理员

将 AWS SCT 扩展代码转换为 psql 代码

Task	描述	所需技能
手动转换代码。	手动将每一行支持扩展的代码转换为 `psql` 默认内置代码，详见所附文档。例如，将 `AWS_ORACLE_EXT.SYSDATE()` 或 `ORACLE.SYSDATE()` 更改为 `NOW()`。	数据库管理员
验证代码	（可选）通过在 PostgreSQL 数据库中临时运行每行代码来验证它。	数据库管理员
在 PostgreSQL 数据库中创建对象。	要在 PostgreSQL 数据库中创建对象，请运行由 AWS SCT 生成并在前两个步骤中修改过的 .sql 文件。	数据库管理员

相关的资源

- 数据库
  - Amazon RDS 上的 Oracle
  - Amazon RDS 上的 PostgreSQL
  - 使用 Amazon Aurora PostgreSQL
  - PostgreSQL EXPLAIN 计划
- AWS SCT
  - AWS Schema Conversion Tool 概述
  - AWS SCT 用户指南
  - 使用 AWS SCT 用户界面
  - 将 Oracle 数据库作为 AWS SCT 的源
- AWS SCT 的扩展程序
  - 使用 AWS SCT 扩展包

- Oracle 功能（英文）
- PGXN orace
- GitHub orafce

其他信息

有关更多信息，请按照随附文档中的详细命令（包含语法和示例）手动转换代码。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS DMS 将 Db2 数据库从亚马逊迁移 EC2 到兼容 Aurora MySQL

由 Pinesh Singal (AWS) 编写

摘要

将 IBM Db2 for LUW 数据库迁移到亚马逊弹性计算云 (Amazon EC2) 后，可以考虑通过迁移到亚马逊网络服务 (AWS) 云原生数据库来重新架构数据库。此模式包括将运行在亚马逊 EC2 实例上的 IBM Db2 f or LUW 数据库迁移到 AWS 上与 Amazon Aurora MySQL 兼容的版本数据库。

该模式描述了一种联机迁移策略，该策略对具有大量事务的多 TB Db2 源数据库的停机时间最短。

此模式使用 AWS Schema Conversion Tool (AWS SCT) 将 Db2 数据库架构转换为与 Aurora MySQL 兼容的架构。然后，该模式使用 AWS Database Migration Service (AWS DMS) 将数据从 Db2 数据库迁移到 Aurora MySQL 兼容数据库。AWS SCT 未转换的代码将需要手动转换。

先决条件和限制

先决条件

- 具有虚拟私有云（VPC）的有效 Amazon Web Services account
- AWS SCT
- AWS DMS

产品版本

- AWS SCT 最新版本
- Db2 for Linux V11.1.4.4 及更高版本

架构

源技术堆栈

- DB2/Linux x86-64 位挂载在实例上 EC2

目标技术堆栈

- Amazon Aurora MySQL-兼容版数据库实例

源架构和目标架构

下图显示了源 Db2 和目标 Aurora MySQL 兼容数据库之间的数据迁移架构。Amazon Web Services Cloud 上的架构包括虚拟私有云（VPC）、可用区、Db2 实例和 AWS DMS 复制实例的公有子网以及 Aurora MySQL 兼容数据库的私有子网。

工具

Amazon Web Services

- Amazon Aurora 是与 MySQL 和 PostgreSQL 兼容的完全托管式的云端关系数据库引擎。
- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到 Amazon Web Services Cloud，或者在云和本地设置的组合之间迁移。
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- AWS Schema Conversion Tool（AWS SCT）通过自动将源数据库架构和大部分自定义代码转换为与目标数据库兼容的格式来支持异构数据库迁移。AWS SCT 支持 IBM Db2 for LUW 版本 9.1、9.5、9.7、10.1、10.5、11.1 和 11.5。

最佳实践

有关最佳实践的信息，请参阅 AWS Database Migration Service 最佳实践。

操作说明

配置源 IBM Db2 数据库

Task	描述	所需技能
在亚马逊 EC2上创建 IBM Db2 数据库。	您可以使用来自 AWS Marketplace 的亚马逊系统映像 (AMI) 或在 EC2 实例上安装 Db2 软件，在实例 EC2 上创建 IBM Db2 数据库。 通过选择适用于 IBM Db2 的 AMI（例如，IBM D b2 v11.5.7	数据库管理员、常规 AWS

Task	描述	所需技能
	RHEL 7.9）来启动 EC2 实例，该AMI 与本地数据库类似。	
配置安全组。	分别配置端口为 22 和 50000 的 SSH（Secure Shell）和 TCP 的 VPC 安全组入方向规则。	常规 AWS
创建数据库实例。	创建新的实例（用户）和数据库（架构），或使用默认的 `db2inst1` 实例和示例数据库。 1. 使用终端连接到 Db2 数据库，从而连接到 EC2 实例。或者，您可以安装任何将连接到 Db2 数据库的数据库客户端软件。 2. 要设置 db2inst1 用户的密码，请运行命令 `sudo passwd db2inst1`。 3. 要连接到 db2inst1 实例，请运行命令 `sudo su - db2inst1`。 4. 要连接到 Db2 数据库，请运行命令 db2。 5. 若要连接到示例数据库，请使用命令 `connect to sample`。或者，连接到您创建的数据库。 6. 连接到数据库实例后，使用 Db2 SQL 语句创建对象并将数据插入到这些对象中。	数据库管理员

Task	描述	所需技能
确认 Db2 数据库实例可用。	要确认 Db2 数据库实例已启动并正在运行，请使用 `Db2pd -` 命令。	数据库管理员

配置目标 Aurora MySQL 兼容数据库

Task	描述	所需技能
创建 Aurora MySQL 兼容数据库。	从 AWS RDS 服务创建兼容 MySQL 的 Amazon Aurora 数据库 • 在 Amazon Aurora 上创建具有 MySQL 兼容性和您选择的版本的数据库，例如 Aurora（MySQL）–5.6.10a • 安装 MySQL Workbench 应用程序或您首选的数据库客户端软件，它允许您连接到 MySQL 数据库	数据库管理员、常规 AWS
配置安全组。	配置 SSH 和 TCP 连接的 VPC 安全组入方向规则。	常规 AWS
确认 Aurora 数据库可用。	要确保 Aurora MySQL 兼容数据库已启动并运行，请执行以下操作： 1. 通过 SSH 连接到 EC2 实例。 2. 从 MySQL Workbench 配置并连接到 Aurora MySQL 兼容实例。使用端点作为主机名，如以下示例所示。	数据库管理员

Task	描述	所需技能
	``` mysql-cluster-instance-1.cokmvis0v46q.us-east-1.rds.amazonaws.com ``` 3. 创建并连接到新架构（例如，`mysql-sample-db2`）。 4. 执行 MySQL 语句，检查数据库中的 Schema 和对象。	

配置和运行 AWS SCT

Task	描述	所需技能
安装 AWS SCT。	下载并安装最新版本的 AWS SCT（当前最新版本 1.0.628）。	常规 AWS
配置 AWS SCT。	1. 下载适用于 IBM Db2（4.22.X 版本）和 MySQL（8.x）的 Java 数据库连接（JDBC）驱动程序。 2. 要在 AWS SCT 中配置驱动程序，请依次选择设置、全局设置和驱动程序。	常规 AWS
创建 AWS SCT 项目。	创建一个 AWS SCT 项目和报告，该项目和报告使用 Db2 for LUW 作为源数据库引擎，并使用 Aurora MySQL 兼容作为目标数据库引擎。	常规 AWS

Task	描述	所需技能
	要确定连接到 Db2 for LUW 数据库所需特权，请参阅将 Db2 LUW 用作 AWS SCT 的源。	
验证对象。	选择 加载架构，验证对象。更新目标数据库上任何不正确的对象： 1. 通过提供连接详细信息连接到 Amazon Aurora MySQL 兼容服务器，然后选择 测试连接。 源连接和目标连接必须成功，AWS SCT 才能启动迁移报告。 2. 报告完成后，输入要转换的架构，然后选择 完成。 AWS SCT 列出了已转换且存在错误的任何源对象和目标对象。 3. 查看错误，并手动清除它们。 4. 清除所有错误后，打开架构的上下文（右键单击）菜单，然后选择 加载架构。 5. 选择应用于数据库。 6. 在 MySQL Workbench 中，连接到 Aurora MySQL 兼容数据库，然后检查架构和对象。	数据库管理员、常规 AWS

配置和运行 AWS DMS

Task	描述	所需技能
创建复制实例。	登录 AWS 管理控制台，导航到 AWS DMS 服务，然后使用您为源数据库和目标数据库配置的 VPC 安全组的有效设置创建复制实例。	常规 AWS
创建端点。	为 Db2 数据库创建源端点，并为 Aurora MySQL 兼容数据库创建目标端点： 1. 通过选择选择 RDS 数据库实例，然后选择您创建的 Db2 实例，为 IBM Db2 创建端点作为源。端点配置详细信息将自动填充。 2. 在特定于端点的设置中，添加以下额外连接属性。 `CurrentLSN=<scan>; MaxKBytesPerRead=64; SetDataCaptureChanges=true` 如果不提及这些属性，源端点测试连接将不会成功。有关更多信息，请参阅将 IBM Db2 LUW 作为 AWS DMS 的源。 3. 通过选择选择 RDS 数据库实例，然后选择您创建的 Aurora MySQL 兼容实例，为 Aurora MySQL 兼容创建端点作为目标。端点配	常规 AWS

Task	描述	所需技能
	置详细信息将自动填充。有关更多信息，请参阅使用 MySQL-兼容数据库作为 AWS Database Migration Service 的目标。 4. 测试源和目标数据库端点。确认两者均成功且可用 5. 如果测试失败，请检查安全组入站规则是否有效。	

Task	描述	所需技能
创建迁移任务。	创建单个迁移任务或多个迁移任务，以实现完全加载和 CDC 或数据验证： 1. 要创建数据库迁移任务，请选择复制实例、源数据库端点、目标数据库端点。将迁移类型指定为迁移现有数据（完全加载）、仅复制数据更改 （CDC）或迁移现有数据并复制正在进行的更改（完全加载和 CDC）。 2. 在表映射下，您可以配置 GUI 或 JSON 格式的选择规则和转换规则。 3. 在选择规则下，选择架构，输入表名，选择要配置的“操作（包含/排除）”（例如，“架构：SAMPLE;表名：%，操作：包含）。 4. 在转换规则下，选择目标（“架构”、“表”或“列”）。选择架构名称，然后选择操作（大小写、前缀、后缀）;例如，目标：架构；`mysql-sample-db` ；动作：小写。 5. 开启 Amazon CloudWatch 日志监控。	常规 AWS
计划生产运行。	与应用程序所有者等利益相关者确认停机时间，以便在生产系统中运行 AWS DMS。	迁移主管

Task	描述	所需技能
运行迁移任务。	1. 启动状态为 就绪 的 AWS DMS 任务。 2. 在 Amazon Logs 中监控迁移任务 CloudWatch 日志中是否存在任何错误。	常规 AWS
验证数据。	查看源 Db2 和目标 MySQL 数据库中的迁移任务结果和数据： 1. 如果状态为加载完成正在进行的复制，则表示 CDC 数据迁移的完全加载已完成，并且验证正在进行中。 2. 连接到 Aurora MySQL 兼容数据库，然后检查数据。 3. 通过在 Db2 数据库中插入或更新数据来检查正在进行的更改。	数据库管理员
停止迁移任务。	数据验证成功完成后，停止验证迁移任务。	常规 AWS

故障排除

事务	解决方案
AWS SCT 源和目标测试连接失败。	配置 JDBC 驱动程序版本和 VPC 安全组入站规则以接受传入流量。
Db2 源端点测试运行失败。	配置额外连接设置 `CurrentLSN=<scan>;` 。
该 AWSDMS 任务无法连接到 Db2 源，并返回以下错误。	若要避免此错误，请运行以下命令：

事务	解决方案

database is recoverable if either or both of the database configuration parameters LOGARCHMETH1 and LOGARCHMETH2 are set to ON	1. $ db2 update db cfg for sample using LOGARCHMETH1 DISK:/home/db2inst1/logs
	2. $ db2stop
	3. $ db2start
	4. $ db2 connect to sample
	`SQL1116N A connection to or activation of database "SAMPLE" cannot be made because of BACKUP PENDING. SQLSTATE=57019`
	5. $ db2 backup database sample to ../logs
	`SQL2036N The path for the file or device "../logs" is not valid`
	6. $ cd
	7. $ pwd
	`/home/db2inst1`
	8. $ mkdir /tmp/backup
	9. $ db2 backup database sample to /tmp/backup
	`Backup successful. The timestamp for this backup image is : 20190530084921`
	10$ db2 connect to sample
	`Database Connection Information`
	`Database server = DB2/LINUX 9.7.1`
	`SQL authorization ID = DB2INST1`

事务	解决方案
	`Local database alias = SAMPLE`

相关资源

Amazon EC2

- Amazon EC2
- Amazon EC2 用户指南

数据库

- IBM Db2 数据库
- Amazon Aurora
- 使用 Amazon Aurora MySQL

AWS SCT

- AWS DMS 架构转换
- AWS Schema Conversion Tool 用户指南
- 使用 AWS SCT 用户界面
- 将 IBM Db2 LUW 用作 AWS SCT 的源

AWS DMS

- AWS Database Migration Service
- AWS Database Migration Service 用户指南
- 数据迁移的源
- 数据迁移的目标
- AWS Database Migration Service 和 AWS Schema Conversion Tool 现在支持将 IBM Db2 LUW 作为源（博客文章）
- 将运行关系数据库的应用程序迁移到 AWS

使用 AWS DMS 将微软 SQL Server 数据库从亚马逊迁移 EC2 到亚马逊 DocumentDB

由 Umamaheswara Nooka (AWS) 创建

摘要

此模式描述了如何使用 AWS 数据库迁移服务 (AWS DMS) 将托管在亚马逊弹性计算云 (亚马逊 EC2) 实例上的微软 SQL Server 数据库迁移到亚马逊文档数据库（兼容 MongoDB）数据库。

AWS DMS 复制任务读取 SQL Server 数据库的表结构，在 Amazon DocumentDB 中创建相应的集合，并执行完全加载迁移。

您还可以使用此模式将本地 SQL Server 或 Amazon Relational Database Service（Amazon RDS）for SQL Server 数据库实例迁移到 Amazon DocumentDB。有关更多信息，请参阅 AWS Prescriptive Guidance 网站上的指南将 Microsoft SQL Server 数据库迁移到 Amazon Web Services Cloud。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- EC2 实例上的现有 SQL Server 数据库。
- 在 SQL Server 数据库中分配给 AWS DMS 的固定数据库（db_owner）角色。有关更多信息，请参阅 SQL Server 文档中的数据库级别角色。
- 熟悉如何使用 `mongodump`、`mongorestore`、`mongoexport` 和 `mongoimport` 实用程序将数据移入和移出 Amazon DocumentDB 集群。
- Microsoft SQL Server Management Studio，已安装并配置。

限制

- Amazon DocumentDB 中的集群大小限制为 64 TB。有关更多信息，请参阅 Amazon DocumentDB 文档中的集群限制。
- AWS DMS 不支持将多个源表合并到单个 Amazon DocumentDB 集合中。
- 如果 AWS DMS 在没有主键的情况下处理来自源表的任何更改，它将忽略源表中的大型对象（LOB）列。

架构

源技术堆栈

- Amazon EC2

目标架构

目标技术堆栈

- Amazon DocumentDB

工具

- AWS DMS – AWS Database Migration Service（AWS DMS）可帮助您轻松安全地迁移数据库。
- Amazon DocumentDB – Amazon DocumentDB（与 MongoDB 兼容）是一种快速、可靠且完全托管的数据库服务。
- 亚马逊 EC2 — 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。
- Microsoft SQL Server – SQL Server 是一个关系数据库管理系统。
- SQL Server Management Studio (SSMS) – SSMS 是一种用于管理 SQL Server 的工具，包括访问、配置和管理 SQL Server 组件。

操作说明

创建和配置 VPC

Task	描述	所需技能
创建 VPC。	登录 AWS 管理控制台，打开 Amazon VPC 控制台。创建具有 IPv4 CIDR 区块范围的虚拟私有云 (VPC)。	系统管理员
创建安全组和网络 ACLs。	在 Amazon VPC 控制台上，根据您的要求为您的 VPC 创建安全组和网络访问控制列表（网	系统管理员

Task	描述	所需技能
	络 ACLs）。您还可以对这些配置使用默认设置。有关此故事和其他故事的详细信息，请参阅“相关资源”部分。	

创建和配置 Amazon DocumentDB 集群

Task	描述	所需技能
创建 Amazon DocumentDB 集群。	打开 Amazon DocumentDB 控制台，然后选择“集群”。选择“创建”，然后创建一个具有一个实例的 Amazon DocumentDB 集群。重要提示：请确保使用 VPC 的安全组配置此集群。	系统管理员
安装 mongo shell。	Mongo Shell 是一个命令行实用程序，用于连接和查询 Amazon DocumentDB 集群。要安装它，请运行 “/etc/yum.repos.d/mongodb-org-3.6.repo” 命令来创建存储库文件。运行 “sudo yum install-y mongodb-org-shell” 命令来安装 mongo 外壳。要加密传输中数据，请下载 Amazon DocumentDB 的公有密钥，然后连接到您的 Amazon DocumentDB 实例。有关这些步骤的更多信息，请参阅“相关资源”部分。	系统管理员
在 Amazon DocumentDB 集群中创建数据库。	使用数据库名称运行“use”命令，以在 Amazon DocumentDB 集群中创建数据库。	系统管理员

创建和配置 AWS DMS 复制实例

Task	描述	所需技能
创建 AWS DMS 复制实例。	打开 AWS DMS 控制台，然后选择“创建复制实例”。输入复制任务的名称和描述。选择实例类、引擎版本、存储、VPC、多可用区，并使其可公开访问。选择“高级”选项卡以设置网络和加密设置。指定维护设置，然后选择“创建复制实例”。	系统管理员
配置 SQL Server 数据库。	登录到 Microsoft SQL Server 并添加用于源端点和 AWS DMS 复制实例之间通信的入站规则。使用复制实例的私有 IP 地址作为源。重要提示：复制实例和目标端点应位于同一 VPC 上。如果源实例和复制实例 VPCs 不同，请使用安全组中的备用源。	系统管理员

在 AWS DMS 中创建和测试源端点和目标端点

Task	描述	所需技能
创建源数据库和目标数据库端点。	打开 AWS DMS 控制台，然后选择“连接源和目标数据库端点”。指定源数据库和目标数据库的连接信息。如果需要，请选择“高级”选项卡以设置“额外连接属性”的值。在端点配置中下载并使用证书捆绑包。	系统管理员
测试端点连接。	选择“运行测试”以测试连接。通过验证安全组设置以及从源数	系统管理员

Task	描述	所需技能
	据库实例和目标数据库实例到 AWS DMS 复制实例的连接来排查任何错误消息。	

迁移数据

Task	描述	所需技能
创建 AWS DMS 迁移任务。	在 AWS DMS 控制台上，依次选择“任务”、“创建任务”。指定任务选项，包括源和目标端点名称以及复制实例名称。在“迁移类型”下，选择“迁移现有数据”和“仅复制数据更改”。选择“启动任务”。	系统管理员
运行 AWS DMS 迁移任务。	在“任务设置”下，指定表准备模式的设置，例如“不执行任何操作”、“删除目标中的表”、“截断”和“在复制中包含 LOB 列”。设置 AWS DMS 将接受的最大 LOB 大小，然后选择“启用日志记录”。将“高级设置”保留为默认值，然后选择“创建任务”。	系统管理员
监控迁移。	在 AWS DMS 控制台上，选择“任务”，然后选择您的迁移任务。选择“任务监控”以监控您的任务。当完成满载迁移并应用缓存更改后，任务停止。	系统管理员

测试和验证迁移

Task	描述	所需技能
使用 mongo shell 连接到 Amazon DocumentDB 集群。	打开 Amazon DocumentDB 控制台，在“集群”下选择您的集群。在“连接和安全性”选项卡中，选择“使用 mongo shell 连接到此集群”。	系统管理员
验证迁移结果。	使用数据库名称运行“use”命令，然后运行“show collections”命令。运行“db.count ( ) ”；命令替换为数据库的名称。如果结果与源数据库匹配，则表示迁移成功。	系统管理员

相关资源

创建和配置 VPC

- 为 VPC 创建安全组
- 创建网络 ACL

创建和配置 Amazon DocumentDB 集群

- 创建 Amazon DocumentDB 集群
- 安装适用于 Amazon DocumentDB 的 mongo shell
- 连接到 Amazon DocumentDB 集群

创建和配置 AWS DMS 复制实例

- 公有和私有复制实例

在 AWS DMS 中创建和测试源端点和目标端点

- 将 Amazon DocumentDB 作为 AWS DMS 的目标
- 使用 SQL Server 数据库作为 AWS DMS 的源
- 使用 AWS DMS 端点

迁移数据

- 迁移到 Amazon DocumentDB

其他资源

- 使用 SQL Server 作为 AWS DMS 源的限制
- 如何使用 Amazon DocumentDB 大规模构建和管理应用程序

将本地 ThoughtSpot Falcon 数据库迁移到亚马逊 Redshift

由 Battulga Purevragchaa (AWS) 和 Antony Prasad Thevaraj (AWS) 创建

摘要

本地数据仓库需要大量时间和资源进行管理，大型数据集更是如此。此外，构建、维护和扩建这些仓库的财务成本很高。若要帮助管理成本，降低提取、转换、加载（ETL）的复杂性，并随着数据的增长提供性能，您必须不断选择要加载的数据和要存档的数据。

通过将您的本地 ThoughtSpot Falcon 数据库迁移到 Amazon Web Services (AWS) 云，您可以访问基于云的数据湖和数据仓库，从而提高业务灵活性、安全性和应用程序可靠性，同时降低总体基础设施成本。Amazon Redshift 有助于显著降低数据仓库的成本与运营开销。您还可以使用 Amazon Redshift Spectrum 分析大量原生格式的数据，而无需加载数据。

此模式描述了将 ThoughtSpot Falcon 数据库从本地数据中心迁移到 AWS 云上的 Amazon Redshift 数据库的步骤和过程。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 托管在本地数据中心的 ThoughtSpot Falcon 数据库

产品版本

- ThoughtSpot 版本 7.0.1

架构

图表显示了以下工作流：

1. 将数据托管至本地关系数据库。
2. AWS Schema Conversion Tool (AWS SCT) 转换与 Amazon Redshift 兼容的数据定义语言 (DDL)。
3. 创建表后，您可以使用 AWS Database Migration Service (AWS DMS)迁移数据。
4. 数据已加载至 Amazon Redshift。

5. 如果您使用 Redshift Spectrum 或已经在 Amazon S3 中托管数据，则数据将存储至 Amazon Simple Storage Service (Amazon S3)。

工具

- AWS DMS – AWS Data Migration Service (AWS DMS) 可帮助您快速安全地将数据库迁移到 AWS。
- Amazon Redshift – Amazon Redshift 是一种快速且完全托管的 PB 级数据仓库，可让您使用现有的商业智能工具轻松且经济高效地分析所有数据。
- AWS SCT – AWS Schema Conversion Tool (AWS SCT) 将现有数据库架构从一个数据库引擎转换为另一个数据库引擎。

操作说明

准备迁移

Task	描述	所需技能
确定适当的 Amazon Redshift 配置。	根据您的要求和数据量确定适当的 Amazon Redshift 集群配置。 有关更多信息，请参阅 Amazon Redshift 文档中的 Amazon Redshift 集群。	数据库管理员
研究 Amazon Redshift，以评估其是否符合您的要求。	使用亚马逊 Redshift FAQs 来了解和评估亚马逊 Redshift 是否符合你的要求。	数据库管理员

准备目标 Amazon Redshift 集群

Task	描述	所需技能
创建一个 Amazon Redshift 集群。	登录 AWS 管理控制台，打开 Amazon Redshift 控制台，然	数据库管理员

Task	描述	所需技能
	后在虚拟私有云（VPC）中创建 Amazon Redshift 集群。 有关更多信息，请参阅 Amazon Redshift 文档中的在 VPC 中创建集群。	
为您的 Amazon Redshift 数据库设计进行 PoC。	通过对数据库设计进行概念验证 (PoC) 遵循最佳 Amazon Redshift 实践。 有关更多信息，请参阅 Amazon Redshift 文档中的对 Amazon Redshift 执行概念验证。	数据库管理员
创建数据库用户。	在您的 Amazon Redshift 数据库中创建用户，并授予其相应的角色，以访问架构和表。 有关更多信息，请参阅 Amazon Redshift 文档中的向用户或用户组授予访问权限。	数据库管理员
将配置设置应用至目标数据库。	根据您的要求将配置设置应用至 Amazon Redshift 数据库。 有关启用数据库、会话和服务器级参数的更多信息，请参阅 Amazon Redshift 文档中的配置参考。	数据库管理员

在 Amazon Redshift 集群中创建对象

Task	描述	所需技能
在 Amazon Redshift 中使用 DDL 手动创建表。	（可选）如果您使用 AWS SCT，则会自动创建表。但是，如果复制失败 DDLs，则必须手动创建表	数据库管理员
为 Redshift Spectrum 创建外部表。	为 Amazon Redshift Spectrum 创建带外部架构的外部表。若要创建外部表，您必须是外部架构的所有者或数据库超级用户。 有关更多信息，请参阅 Amazon Redshift 文档中的为 Amazon Redshift Spectrum 创建外部表。	数据库管理员

使用 AWS DMS 迁移数据

Task	描述	所需技能
使用 AWS DMS 迁移数据。	在 Amazon Redshift 数据库中创建表 DDL 后，使用 AWS DMS 将数据迁移至 Amazon Redshift。 有关详细步骤和说明，请参阅 AWS DMS 文档中的使用 Amazon Redshift 数据库作为 AWS DMS 的目标。	数据库管理员
使用 COPY 命令加载数据。	使用 Amazon Redshift COPY 命令将数据从 Amazon S3 加载至 Amazon Redshift。	数据库管理员

Task	描述	所需技能
	有关更多信息，请参阅 Amazon Redshift 文档中的使用 COPY 命令从 Amazon S3 加载。	

验证 Amazon Redshift 集群

Task	描述	所需技能
验证源记录和目标记录。	验证从源系统加载的源记录和目标记录表数。	数据库管理员
实施 Amazon Redshift 最佳实践，以进行性能调整。	实施表和数据库设计的 Amazon Redshift 最佳实践。 有关更多信息，请参阅博客文章 Amazon Redshift 的十大性能优化技术。	数据库管理员
优化查询性能。	Amazon Redshift 使用基于 SQL 的查询与系统中的数据和对象进行交互。数据操作语言 (DML) 是用于查看、添加、更改和删除数据的 SQL 子集。DDL 是用于添加、更改和删除数据库对象(如表和视图)的 SQL 子集。 有关更多信息，请参阅 Amazon Redshift 文档中的优化查询性能。	数据库管理员
实施 WLM。	您可以使用工作负载管理 (WLM) 定义多个查询队列并在运行时将查询路由到适当的队列。	数据库管理员

Task	描述	所需技能
	有关更多信息，请参阅 Amazon Redshift 文档中的实施工作负载管理。	
使用并发扩展。	使用并发扩展功能，您可以支持几乎无限的并发用户和并发查询，同时提供始终如一的快速查询性能。 有关更多信息，请参阅 Amazon Redshift 文档中的使用并发扩展。	数据库管理员
使用设计表的 Amazon Redshift 最佳实践。	在规划数据库时，某些重要表设计决策对整体查询性能影响很大。 有关选择最合适的表设计选项的更多信息，请参阅 Amazon Redshift 文档中的设计表的 Amazon Redshift 最佳实践。	数据库管理员
在 Amazon Redshift 中创建实体化视图。	实体化视图 包含一个预计算的结果集，该结果集基于对一个或多个基表进行的 SQL 查询。您可以发出 SELECT 语句来查询实体化视图，这与查询数据库中的其他表或视图的方式相同。 有关更多信息，请参阅 Amazon Redshift 文档中的在 Amazon Redshift 中创建实体化视图。	数据库管理员

Task	描述	所需技能
定义表与表之间的连接。	要在中同时搜索多个表 ThoughtSpot，必须通过指定包含跨两个表的匹配数据的列来定义表之间的联接。这些列表示联接的 `primary key` 和 `foreign key`。 你可以使用 Amazon Redshif `ALTER TABLE` t 中的命令来定义它们，或者。 ThoughtSpot有关更多信息，请参阅 Amazon Redshift 文档中的 ALTER TABLE：	数据库管理员

设置 ThoughtSpot 与亚马逊 Redshift 的连接

Task	描述	所需技能
添加 Amazon Redshift 连接。	向您的本地 F ThoughtSpot alcon 数据库添加 Amazon Redshift 连接。 有关更多信息，请参阅文档中的添加亚马逊 Redshift 连接。 ThoughtSpot	数据库管理员
编辑 Amazon Redshift 连接。	您可以编辑 Amazon Redshift 连接以添加表和列。 有关更多信息，请参阅文档中的编辑 Amazon Redshift 连接。 ThoughtSpot	数据库管理员

Task	描述	所需技能
重新映射 Amazon Redshift 连接。	通过编辑您在添加 Amazon Redshift 连接时创建的源映射 .yaml 文件修改连接参数。 例如，您可以将现有表或列重新映射到现有数据库连接中的其他表或列。 ThoughtSpot 建议您在重新映射连接中的表或列之前和之后检查依赖关系，以确保它们按需要显示。 有关更多信息，请参阅文档中的重新映射 Amazon Redshift 连接。 ThoughtSpot	数据库管理员
从 Amazon Redshift 连接中删除表。	（可选）如果您尝试删除 Amazon Redshift 连接中的表，则 ThoughtSpot 会检查依赖关系并显示依赖对象列表。您可以选择列出的对象，以将其删除或移除依赖项。然后，您可以删除该表。 有关更多信息，请参阅文档中的从 Amazon Redshift 连接中 ThoughtSpot 删除表。	数据库管理员

Task	描述	所需技能
从 Amazon Redshift 连接中删除含依赖对象的表。	（可选）如果您尝试删除包含依赖对象表，则阻止该操作。将显示 `Cannot delete` 窗口，其中包含指向依赖对象的链接列表。移除所有依赖项后即可删除该表 有关更多信息，请参阅文档中的从 Amazon Redshift 连接中删除包含依赖对象的 ThoughtSpot 表。	数据库管理员
删除 Amazon Redshift 连接。	（可选）由于一个连接可用于多个数据来源或可视化效果，因此必须先删除使用该连接的所有来源和任务，然后才能删除 Amazon Redshift 连接。 有关更多信息，请参阅文档中的删除亚马逊 Redshift 连接。ThoughtSpot	数据库管理员
请查看 Amazon Redshift 连接参考。	请务必使用文档中的连接参考提供您的 Amazon Redshift 连接所需的信息。 ThoughtSpot	数据库管理员

其他信息

- 使用 Amazon Redshift 进行 ThoughtSpot 任何规模的人工智能驱动型分析
- Amazon Redshift 定价
- AWS SCT 入门
- Amazon Redshift 入门
- 使用数据提取代理
- Chick-fil-A 借助 ThoughtSpot 和 AWS 提高获取洞察的速度

使用 AWS DMS 将 Oracle 数据库迁移至 Amazon DynamoDB

由 Rambabu Karnena (AWS) 创建

摘要

此模式将指导您完成以下步骤：使用 AWS Database Migration Service (AWS DMS) 将 Oracle 数据库迁移至Amazon DynamoDB。其涵盖了三类源数据库：

- 本地 Oracle 数据库
- 亚马逊弹性计算云 (亚马逊 EC2) 上的 Oracle 数据库
- 适用于 Oracle 数据库实例的 Amazon Relational Database Service (Amazon RDS)

在此概念验证中，此模式侧重于从 Amazon RDS for Oracle 数据库实例迁移。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 连接至 Amazon RDS for Oracle 数据库的应用程序
- 在源 Amazon RDS for Oracle 数据库中创建的、包含主键和示例数据的表

限制

- 不考虑迁移 Oracle 数据库对象，例如过程、函数、包和触发器，因为 Amazon DynamoDB 不支持上述数据库对象。

产品版本

- 此模式适用于 AWS DMS 支持的所有版本的 Oracle 数据库。有关更多信息，请参阅 Oracle 数据库作为 AWS DMS 的源以及 Amazon DynamoDB 数据库作为 AWS DMS 的目标。建议使用最新版本的 AWS DMS，以获得最全面的版本和功能支持。

架构

源技术堆栈

- 适用于 Oracle 数据库实例的 Amazon RDS EC2、亚马逊上的 Oracle 或本地 Oracle 数据库

目标技术堆栈

- Amazon DynamoDB

AWS 数据迁移架构

工具

- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到 Amazon Web Services Cloud，或者在云和本地设置的组合之间迁移。
- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- Amazon Relational Database Service (Amazon RDS) 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展关系数据库。此模式使用了 Amazon RDS for Oracle。

操作说明

计划迁移

Task	描述	所需技能
创建 VPC。	在 Amazon Web Services account 中，创建虚拟私有云（VPC）和私有子网。	系统管理员
创建安全组和网络访问控制列表。	有关更多信息，请参阅 AWS 文档。	系统管理员
配置并启动 Amazon RDS for Oracle 数据库实例。	有关更多信息，请参阅 AWS 文档。	数据库管理员、系统管理员

迁移数据

Task	描述	所需技能
为访问 DynamoDB 创建 IAM 角色。	在 AWS Identity and Access Management (IAM) 控制台中，创建角色，附加策略 `AmazonDynamoDBFullAccess to it` ，然后选择 AWS DMS 作为服务。	系统管理员
为迁移创建 AWS DMS 复制实例。	复制实例应与源数据库位于同一可用区和同一 VPC 。	系统管理员
在 AWS DMS 中创建源端点和目标端点	若要创建源数据库端点，有两个选项供您选择： • 在 Amazon RDS 控制台上，选择数据库、DB 标识符、连接和安全，然后选择端点。 • 在 AWS DMS 控制台上，选择选择 RDS 数据库实例。 若要创建目标数据库端点，请从先前 DynamoDB 访问任务中选择 Amazon 资源名称（ARN）的角色。	系统管理员
创建将源 Oracle 数据库表加载至 DynamoDB 的 AWS DMS 任务。	从前述步骤中选择源和目标端点名称以及复制实例。该类型可完全加载。选择 Oracle 架构并指定 %，以选择所有表。	系统管理员
验证 DynamoDB 中的表格。	若要查看迁移结果，请从 DynamoDB 控制台的左侧导航窗格中选择表格。	数据库管理员

迁移应用程序

Task	描述	所需技能
修改应用程序代码。	若要连接至 DynamoDB 并从中检索数据，请更新应用程序代码。	应用程序所有者，数据库管理员，系统管理员

割接

Task	描述	所需技能
将应用程序客户端切换至使用 DynamoDB。		数据库管理员、应用程序所有者、系统管理员

关闭项目

Task	描述	所需技能
关闭 AWS 资源。	例如，关闭 Amazon RDS for Oracle 实例、DynamoDB 和 AWS DMS 复制实例。	数据库管理员、系统管理员
收集指标。	指标包括迁移时间、手动工作和工具执行工作的百分比以及成本节约。	数据库管理员、应用程序所有者、系统管理员

相关资源

- AWS Database Migration Service 和 Amazon DynamoDB：您需要了解的内容（博客文章）
- 将 Oracle 数据库作为 AWS DMS 的源
- 将 Amazon DynamoDB 数据库作为 AWS Database Migration Service 的目标
- 从 RDBMS 迁移至 Amazon DynamoDB 的最佳实践（白皮书）

使用 AWS DMS 将 Oracle 分区表迁移到 PostgreSQL

创建者：Saurav Mishra (AWS) 和 Eduardo Valentim (AWS)

摘要

此模式描述如何使用不支持本机分区的 AWS Database Migration Service (AWS DMS) 加速将分区表从 Oracle 加载到 PostgreSQL。目标 PostgreSQL 数据库可以安装在亚马逊弹性计算云（亚马逊）上，也可以是适用于 PostgreSQL 的 EC2亚马逊关系数据库服务 (Amazon RDS) 或兼容亚马逊 Aurora PostgreSQL 的版本数据库实例。

分区表上传包括以下步骤：

1. 创建类似于 Oracle 分区表的父表，但不包含任何分区。
2. 创建将从步骤 1 中创建的父表继承的子表。
3. 创建过程函数和触发器以处理父表中的插入。

但是，由于每次插入都会触发触发器，因此使用 AWS DMS 的初始加载可能会非常慢。

为了加快从 Oracle 到 PostgreSQL 9.0 的初始加载速度，此模式为每个分区创建一个单独的 AWS DMS 任务并加载相应的子表。然后，您可以在割接期间创建触发器。

PostgreSQL 版本 10 支持本机分区。但是，在某些情况下，您可能会决定使用继承分区。有关更多信息，请参阅其他信息部分。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 具有分区表的源 Oracle 数据库
- AWS 上的 PostgreSQL 数据库

产品版本

- PostgreSQL 9.0

架构

源技术堆栈

- Oracle 中的分区表

目标技术堆栈

- PostgreSQL 中的分区表（在 EC2亚马逊上、适用于 PostgreSQL 的亚马逊 RDS 或 Aurora PostgreSQL 上）

目标架构

工具

- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到 Amazon Web Services Cloud，或者在云和本地设置的组合之间迁移。

操作说明

设置 AWS DMS

Task	描述	所需技能
在 PostgreSQL 中创建表。	使用所需分区检查条件在 PostgreSQL 中创建父表和相应子表。	数据库管理员
为每个分区创建 AWS DMS 任务。	在 AWS DMS 任务中包括分区的筛选条件。将分区映射至相应的 PostgreSQL 子表。	数据库管理员
使用满载操作和更改数据捕获 (CDC) 运行 AWS DMS 任务。	确保将 `StopTaskCachedChangesApplied` 参数设置为 `true`，将 `StopTaskCachedChangesNotApplied` 参数设置为 `false`。	数据库管理员

割接

Task	描述	所需技能
停止复制任务。	在停止任务之前，请确认源和目标是否同步。	数据库管理员
在父表创建触发器。	由于父表将接收所有插入和更新命令，因此创建一个触发器，根据分区条件将这些命令路由到相应的子表。	数据库管理员

相关的资源

- AWS DMS
- 表分区（PostgreSQL 文档）

其他信息

尽管 PostgreSQL 版本 10 支持原生分区，但您可能决定在以下用例中使用继承分区：

- 分区强制执行一条规则，即所有分区必须具有与父分区相同的列集，但表继承支持子分区具有额外的列。
- 表继承支持多重继承。
- 声明式分区仅支持列表与范围分区。通过表继承，您可按需要划分数据。但是，如果约束排除不能有效地修剪分区，查询性能将会受到影响。
- 使用声明性分区时，某些运算需要比使用表继承时更强的锁。例如，在分区表中添加分区或从分区表中删除分区需要 `ACCESS EXCLUSIVE` 锁定父表，而 `SHARE UPDATE EXCLUSIVE` 锁足以进行常规继承。

当您使用单独作业分区时，如果存在任何 AWS DMS 验证问题，您还可以重新加载分区。为了获得更好的性能和复制控制，请在单独的复制实例上运行任务。

从 Amazon RDS for Oracle 迁移到 Amazon RDS for MySQL

由 Jitender Kumar (AWS)、Neha Sharma (AWS) 和 Srini Ramaswamy (AWS) 创作

摘要

此模式为在亚马逊网络服务 (AWS) 上将适用于 Oracle 数据库实例的亚马逊关系数据库服务 (Amazon RDS) 迁移到亚马逊 RDS for MySQL 数据库实例提供了指导。该模式使用 AWS 数据库迁移服务 (AWS DMS) 和 AWS 架构转换工具 (AWS SCT)。

该模式提供了处理存储过程迁移的最佳实践。它还涵盖了为支持应用程序层而进行的代码更改。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- Amazon RDS for Oracle 源数据库
- Amazon RDS for MySQL 目标数据库。源数据库和目标数据库应位于同一个虚拟私有云 (VPC) 中。如果您使用多个 VPCs，或者您必须拥有所需的访问权限。
- 允许在源数据库和目标数据库、AWS SCT、Application Server和 AWS DMS 之间建立连接的安全组。
- 具有源数据库运行 AWS SCT 所需权限的用户账户。
- 为在源数据库上运行 AWS DMS 启用了补充日志。

限制

- 源和目标 Amazon RDS 数据库大小限制为 64 TB。有关 Amazon RDS 的大小信息，请参阅 AWS 文档。
- Oracle 对于数据库对象不区分大小写，但 MySQL 则不然。AWS SCT 可以在创建对象时处理此问题。但是，要支持完全不区分大小写，则需要一些手动操作。
- 此迁移不使用 MySQL 扩展来启用 Oracle 原生功能。AWS SCT 处理大部分转换，但需要进行一些工作来手动更改代码。
- 应用程序中需要更改 Java Database Connectivity (JDBC) 驱动程序。

产品版本

- 适用于 Oracle 的 Amazon RDS 12.2.0.1 及更高版本。有关当前支持的 RDS for Oracle 版本，请参阅 AWS 文档。
- 适用于 MySQL 的 Amazon RDS 8.0.15 及更高版本。有关当前支持的 RDS for MySQL 版本，请参阅 AWS 文档。
- AWS DMS 版本 3.3.0 及更高版本。有关 AWS DMS 支持的源终端节点和目标终端节点的更多信息，请参阅 AWS 文档。
- AWS SCT 版本 1.0.628 及更高版本。 请参阅 AWS 文档中的 AWS SCT 源和目标终端节点支持矩阵。

架构

源技术堆栈

- Amazon RDS for Oracle。有关更多信息，请参阅使用 Oracle 数据库作为 AWS DMS 的来源。

目标技术堆栈

- Amazon RDS for MySQL。有关更多信息，请参阅使用与 MySQL 兼容的数据库作为 AWS DMS 的目标。

迁移架构

在下图中，AWS SCT 从 Amazon RDS for Oracle 源数据库复制和转换架构对象，并将这些对象发送到 Amazon RDS for MySQL 目标数据库。AWS DMS 复制源数据库中的数据并将其发送到 Amazon RDS for MySQL 实例。

工具

- AWS 数据迁移服务可帮助您将数据存储迁移到 AWS 云中，或者在云和本地设置的组合之间迁移。
- Amazon Relational Database Service (Amazon RDS) 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展关系数据库。这种模式使用适用于甲骨文的亚马逊 RDS 和适用于 MySQL 的亚马逊 RDS。
- AWS Schema Conversion Tool（AWS SCT）通过自动将源数据库架构和大部分自定义代码转换为与目标数据库兼容的格式来支持异构数据库迁移。

操作说明

准备迁移

Task	描述	所需技能
验证源数据库和目标数据库的版本和引擎。		数据库管理员
确定目标服务器实例的硬件要求。		数据库管理员，SysAdmin
识别存储需求（存储类型和容量）。		数据库管理员，SysAdmin
选择适当的实例类型（容量、存储功能、网络功能）。		数据库管理员，SysAdmin
确定源数据库和目标数据库的网络访问安全要求。		数据库管理员，SysAdmin
选择应用程序迁移策略。	考虑割接活动是要完全停机还是部分停机。	DBA、SysAdmin、应用程序所有者

配置基础设施

Task	描述	所需技能
创建 VPC 和子网。		SysAdmin
创建安全组和网络访问控制列表 (ACLs)。		SysAdmin
配置和启动 Amazon RDS for Oracle 实例。		数据库管理员，SysAdmin
配置和启动 Amazon RDS for MySQL 实例。		数据库管理员，SysAdmin

Task	描述	所需技能
准备一个测试用例来验证代码转换。	这将有助于对转换后的代码执行单元测试。	数据库管理员、开发人员
配置 AWS DMS 实例。		
在 AWS DMS 中配置源和目标端点。		

迁移数据

Task	描述	所需技能
通过 AWS SCT 生成目标数据库脚本。	检查 AWS SCT 转换代码的准确性。将需要一些手动操作。	数据库管理员、开发人员
在 AWS SCT 中，选择不区分大小写设置。	在 AWS SCT 中选择项目设置、目标区分大小写、不区分大小写。	数据库管理员、开发人员
在 AWS SCT 中选择不使用 Oracle 原生函数。	在 “项目设置” 中，选中 TO_CHAR/TO_NUMBER/TO_DATE 的功能。	数据库管理员、开发人员
对“sql%notfound”代码进行更改。	您可能需要手动转换代码。	
在存储过程中查询表和对象(使用小写查询)。		数据库管理员、开发人员
完成所有更改后创建主脚本，然后将主脚本部署到目标数据库上。		数据库管理员、开发人员
使用示例数据对存储过程和应用程序调用执行单元测试。		

Task	描述	所需技能
清理单元测试期间所创建的数据。		数据库管理员、开发人员
删除目标数据库上的外键约束。	需要执行此步骤来加载初始数据。如果您不想删除外键约束，则必须为特定于主表和辅助表的数据创建迁移任务。	数据库管理员、开发人员
删除目标数据库的主键与唯一键。	此步骤可提高初始加载性能。	数据库管理员、开发人员
在源数据库上启用补充日志记录。		数据库管理员
在 AWS DMS 中为初始加载创建迁移任务，然后运行该任务。	选择该选项以迁移现有数据。	数据库管理员
向目标数据库添加主键和外键。	初始加载后需要添加约束。	数据库管理员、开发人员
创建用于持续复制的迁移任务。	持续复制使目标数据库与源数据库保持同步。	数据库管理员

迁移应用程序

Task	描述	所需技能
将 Oracle 原生函数替换为 MySQL 原生函数。		应用程序所有者
确保 SQL 查询中的数据库对象仅使用小写名称。		DBA、 SysAdmin、应用程序所有者

割接至目标数据库

Task	描述	所需技能
关闭Application Server。		应用程序所有者
验证源数据库和目标数据库是否同步。		数据库管理员、应用程序所有者
停止 Amazon RDS for Oracle 数据库实例。		数据库管理员
停止迁移任务。	完成上一步后，该进程将自动停止。	数据库管理员
将 JDBC 连接从 Oracle 更改至 MySQL。		应用程序所有者，数据库管理员
启动应用程序。		DBA、 SysAdmin、应用程序所有者

关闭项目

Task	描述	所需技能
查看和验证项目文档。		数据库管理员， SysAdmin
收集与迁移时间、手动任务与工具任务的百分比、成本节约等相关的指标。		数据库管理员， SysAdmin
停止和删除 AWS DMS 实例。		数据库管理员
移除源和目标数据库端点		数据库管理员
移除迁移任务。		数据库管理员
拍摄 Amazon RDS for Oracle 数据库实例的快照。		数据库管理员

Task	描述	所需技能
删除 Amazon RDS for Oracle 数据库实例。		数据库管理员
关闭并删除您所用的任何其他临时 AWS 资源。		数据库管理员，SysAdmin
关闭项目并提供任何反馈。		数据库管理员

相关的资源

- AWS DMS
- AWS SCT
- Amazon RDS 定价
- AWS DMS 入门
- Amazon RDS 入门

使用 AWS DMS 和 AWS SCT 从亚马逊上的 IBM Db2 迁移 EC2 到兼容 Aurora PostgreSQL

由 Sirsendu Halder (AWS) 和 Abhimanyu Chhabra (AWS) 创作

摘要

此模式为将亚马逊弹性计算云（亚马逊 EC2）实例上的 IBM Db2 数据库迁移到兼容亚马逊 Aurora PostgreSQL 的版本数据库实例提供了指导。该模式使用 AWS Database Migration Service (AWS DMS) 和 AWS Schema Conversion Tool (AWS SCT) 进行数据迁移和架构转换。

该模式针对的是具有大量事务的多 TB IBM Db2 数据库的在线迁移策略，停机时间很少或没有停机时间。`NUMERIC`为了提高性能，我们建议您将数据类型的主键 (PKs) 和外键 (FKs) 中的列转换为 PostgreSQL `INT` 或`BIGINT`在 PostgreSQL 中。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 实例上的源 IBM Db2 数据库 EC2

产品版本

- DB2/LINUXX8664 版本 11.1.4.4 及更高版本

架构

源技术堆栈

- 实例上的 Db2 数据库 EC2

目标技术堆栈

- 兼容 Aurora PostgreSQL 10.18 或更高版本的数据库实例

数据库迁移架构

工具

- AWS Database Migration Service（AWS DMS）可帮助您将数据库迁移到 Amazon Web Services Cloud，或者在云和本地设置的组合之间迁移。源数据库可在迁移过程中保持全面运行，从而最大程度地为依赖该数据库的应用程序减少停机时间。您可以使用 AWS DMS 在最广泛使用的商用和开源数据库之间迁移数据。AWS DMS 支持不同数据库平台之间的异构迁移，例如 IBM Db2 到 Aurora PostgreSQL 兼容版本 10.18 或更高版本。有关详细信息，请参阅 AWS DMS 文档中的数据迁移源和数据迁移目标。
- AWS Schema Conversion Tool (AWS SCT) 通过自动将源数据库架构和大部分数据库代码对象（包括视图、存储过程和函数）转换为与目标数据库兼容的格式，支持异构数据库迁移。任何未自动转换的对象都会被明确标记，以便可以手动转换它们以完成迁移。AWS SCT 还可以扫描应用程序源代码中的嵌入式 SQL 语句并对其进行转换。

操作说明

设置环境

Task	描述	所需技能
创建 Aurora PostgreSQL-Compatible 数据库实例。	要创建数据库实例，请按照 AWS 文档中的说明进行操作。对于引擎类型，选择 Amazon Aurora。对于版本，选择 Amazon Aurora PostgreSQL 兼容版。 Aurora PostgreSQL-Compatible 10.18 或更高版本数据库实例应与源 IBM Db2 数据库位于同一虚拟私有云（VPC）中。	Amazon RDS

转换数据库架构

Task	描述	所需技能
安装和验证 AWS SCT。	1. 按照 AWS SCT 文档中的步骤安装 AWS SCT。	AWS 管理员，数据库管理员，迁移工程师

Task	描述	所需技能
	2. 按照 AWS SCT 文档 中的过程验证安装。	
启动 AWS SCT 并创建项目。	要启动 AWS SCT 工具并创建一个新项目来运行数据库迁移评测报告，请按 AWS SCT 文档中的说明进行操作。	迁移工程师
添加数据库服务器，并创建映射规则。	1. 按 AWS SCT 文档中的说明添加源数据库服务器和目标数据库服务器。 2. 创建映射规则，定义源数据库的目标数据库平台。有关说明，请参阅 AWS SCT 文档。	迁移工程师
创建数据库迁移评测报告。	按照 AWS SCT 文档 中的步骤创建数据库迁移评测报告。	迁移工程师
查看评测报告。	使用数据库迁移评测报告的摘要选项卡，查看报告并分析数据。此分析将帮助您确定迁移复杂性。有关更多信息，请参阅 AWS SCT 文档。	迁移工程师
转换架构。	要转换源数据库架构： 1. 在 AWS SCT 控制台，选择视图，然后选择主视图。 2. 从源架构中选择对象或父节点，打开上下文（右键单击）菜单，然后选择转换架构。 有关更多信息，请参阅 AWS SCT 文档。	迁移工程师

Task	描述	所需技能
将转换后的数据库架构应用于目标数据库实例。	1. 在显示目标数据库实例的计划架构的项目右侧面板中选择架构元素。 2. 打开架构元素的上下文 (右键单击) 菜单，然后选择 Apply to database。 有关更多信息，请参阅 AWS SCT 文档。	迁移工程师

迁移数据

Task	描述	所需技能
设置 VPC 与数据库参数组。	设置 VPC 和数据库参数组，并配置迁移所需入站规则和参数。有关说明，请参阅 AWS DMS 文档。 对于 VPC 安全组，请同时选择 Db2 的 EC2 实例和兼容 Aurora PostgreSQL 的数据库实例。此复制实例必须与源数据库实例和目标数据库实例位于同一 VPC。	迁移工程师
准备源数据库实例与目标数据库实例。	准备要迁移的源数据库实例和目标数据库实例。在生产环境中，源数据库已经存在。 对于源数据库，服务器名称必须是运行 Db2 的 EC2 实例的公共域名系统 (DNS)。对于用户名，您可使用 `db2inst1` 后	迁移工程师

Task	描述	所需技能
	跟端口，对于 IBM Db2，该端口将是 5000。	
创建 Amazon EC2 客户端和终端节点。	1. 创建亚马逊 EC2 客户端。您可使用此客户端在源数据库中填充要复制的数据。您还可以使用此客户端通过在目标数据库上运行查询来验证复制。 2. 为源数据库和目标数据库实例创建端点，以用于后续步骤。有关说明，请参阅 AWS DMS 文档。为源数据库和目标数据库创建单独端点。对于兼容 Aurora PostgreSQL 10.18 或更高版本，端口将为 5432，您可从数据库实例的端点获取服务器名称。	迁移工程师
创建复制实例。	使用 AWS DMS 控制台创建复制实例，并指定源和目标端点。复制实例执行端点之间的数据迁移。有关更多信息，请参阅 AWS DMS 文档。	迁移工程师

Task	描述	所需技能
创建 AWS DMS 任务以迁移数据。	按照 AWS DMS 文档中的步骤创建任务，将源 IBM Db2 表加载到目标 PostgreSQL 数据库实例。 • 对于源和目标，请使用源端点名称和目标端点名称。 • 该迁移类型可完全加载。 • 对于架构规则，您可使用 Db2 数据库中的 `inst1` 架构。 • 在表名中，指定 `%` 以迁移所有表格。加载完成后，您将看到 `inst1` 架构的 Db2 表格出现在 Aurora PostgreSQL 兼容数据库中。	迁移工程师

相关资源

参考

- Amazon Aurora 文档
- PostgreSQL 外部数据包装器 (FDW) 文档
- PostgreSQL IMPORT FOREIGN SCHEMA 文档
- AWS DMS 文档
- AWS SCT 文档

教程和视频

- AWS DMS 入门（演练）
- Amazon 简介 EC2 -使用 AWS 实现弹性云服务器和托管（视频）

使用和 AWS DMS 从 Oracle 8i 或 9i 迁移到适用于 SharePlex PostgreSQL 的亚马逊 RDS

由 Kumar Babu P G (AWS) 编写

摘要

此模式描述了如何将本地的 Oracle 8i 或 9i 数据库迁移到适用于 PostgreSQL 的Amazon Relational Database Service (Amazon RDS) 或 Amazon Aurora PostgreSQL。AWS Database Migration Service (AWS DMS) 不支持 Oracle 8i 或 9i 作为来源，因此 Quest SharePlex 会将数据从本地 8i 或 9i 数据库复制到与 AWS DMS 兼容的中间 Oracle 数据库（Oracle 10g 或 11g）。

使用 AWS Schema Conversion Tool (AWS SCT) 和 AWS DMS 将架构和数据从中间 Oracle 实例迁移到 AWS 上的 PostgreSQL 数据库。此方法有助于以最小的复制延迟实现从源 Oracle 数据库到目标 PostgreSQL 数据库实例的数据连续流式传输。实施此示例时，停机时间将限于创建或验证目标 PostgreSQL 数据库上的所有外键、触发器和序列所需时间长度。

迁移使用安装了 Oracle 10g 或 11g 的亚马逊弹性计算云 (Amazon EC2) 实例来托管来自源 Oracle 数据库的更改。AWS DMS 使用此中间 Oracle 实例作为源将数据流式传输到 Amazon RDS for PostgreSQL 或 Aurora PostgreSQL。可以暂停和恢复从本地 Oracle 数据库到中间 Oracle 实例的数据复制。它还可以从中间 Oracle 实例到目标 PostgreSQL 数据库暂停和恢复，以便您可以使用 AWS DMS 数据验证或自定义数据验证工具来验证数据。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 本地数据中心中的源 Oracle 8i 或 9i 数据库
- AWS Direct Connect 配置至本地数据中心和 AWS 之间
- 安装在本地计算机或安装了 AWS SCT 的实例上的 AWS SCT 连接器的 Java 数据库连接 (JDBC) 驱动程序 EC2
- 熟悉使用 Oracle 数据库作为 AWS DMS 的源
- 熟悉使用 PostgreSQL 数据库作为 AWS DMS 的目标
- 熟悉 Quest SharePlex 数据复制

限制

- 数据库大小限制为 64 TB
- 本地 Oracle 数据库必须为 Enterprise Edition

产品版本

- 作为源数据库的 Oracle 8i 或 9i
- 作为中间数据库的 Oracle 10g 或 11g
- PostgreSQL 9.6 或更高版本

架构

源技术堆栈

- Oracle 8i 或 9i 数据库
- 任务 SharePlex

目标技术堆栈

- MySQL 到 Amazon RDS for PostgreSQL 或 Amazon Aurora (PostgreSQL)。

源架构和目标架构

工具

- AWS DMS — AWS Database Migration Service (AWS DMS) 可帮助您快速安全地迁移数据库。源数据库可在迁移过程中保持全面运行，从而最大程度地为依赖该数据库的应用程序减少停机时间。AWS DMS 可以在最广泛使用的商用和开源数据库之间迁移数据。
- AWS SCT - AWS Schema Conversion Tool (AWS SCT) 通过以下方法来简化异构数据库的迁移工作：将源数据库架构和大部分的自定义代码 (包括视图、存储过程和函数等) 自动转换成与目标数据库兼容的格式。任何无法自动转换的对象都会被明确标记，以便可以手动转换它们以完成迁移。AWS SCT 还可以扫描您的应用程序源代码中的嵌入式 SQL 语句，并将其作为数据库架构转换项目的一部分进行转换。在此过程中，AWS SCT 通过将旧的 Oracle 和 SQL Server 功能转换为其 AWS 等效功能来执行云原生代码优化，以帮助您在迁移数据库的同时实现应用程序现代化。架构转

换完成后，AWS SCT 可以使用内置的数据迁移代理帮助将数据从一系列数据仓库迁移至 Amazon Redshift。

- Ques t SharePlex — Quest SharePlex 是一款 Oracle-to-Oracle数据复制工具，用于在最短的停机时间内移动数据，并且不会丢失任何数据。

操作说明

创建 EC2 实例并安装 Oracle

Task	描述	所需技能
为 Amazon 设置网络 EC2。	创建新虚拟私有云（VPC）、子网、互联网网关、路由表和安全组。	AWS SysAdmin
创建新 EC2 实例。	为 EC2 实例选择亚马逊系统映像 (AMI)。选择实例大小并配置实例详细信息：实例数量（1）、您在上一个任务中创建的 VPC 和子网、自动分配公有 IP 地址以及其他选项。添加存储、配置安全组并启动实例。出现提示时，创建并保存密钥对，以供下一步使用。	AWS SysAdmin
在 EC2 实例上安装 Oracle。	获取许可证和所需的 Oracle 二进制文件，然后在实例上安装 Oracle 10g 或 11g。 EC2	数据库管理员

SharePlex 在 EC2 实例上设置并配置数据复制

Task	描述	所需技能
设置 SharePlex。	创建亚马逊 EC2 实例并安装与 Oracle 8i 或 9i 兼容的 SharePlex 二进制文件。	AWS SysAdmin、DBA

Task	描述	所需技能
配置数据复制。	按照 SharePlex 最佳实践配置从本地 Oracle 8i/9i 数据库到 Oracle 10g/11g 实例的数据复制。	数据库管理员

将 Oracle 数据库架构转换为 PostgreSQL

Task	描述	所需技能
设置 AWS SCT。	创建新报告，然后连接到 Oracle 作为源，将 PostgreSQL 作为目标。在项目设置中，打开SQL 脚本选项卡，将目标 SQL 脚本更改为多个文件。	数据库管理员
转换 Oracle 数据库架构。	在操作选项卡中，选择生成报告、转换架构，然后选择另存为 SQL。	数据库管理员
修改 AWS SCT 生成的 SQL 脚本。		数据库管理员

创建和配置 Amazon RDS 数据库实例

Task	描述	所需技能
创建 Amazon RDS 数据库实例。	在 Amazon RDS 控制台，创建新的 PostgreSQL 数据库实例。	AWS SysAdmin、DBA
配置数据库实例。	指定数据库引擎版本、数据库实例类、多可用区部署、存储类型和分配的存储空间。输入	AWS SysAdmin、DBA

Task	描述	所需技能
	数据库实例标识符、主用户名和主密码。	
配置网络和安全。	指定 VPC、子网组、公共可访问性、可用区首选项和安全组。	AWS SysAdmin、DBA
配置数据库选项。	指定数据库名称、端口、参数组、加密和主密钥。	AWS SysAdmin、DBA
配置备份。	指定备份保留期、备份窗口、开始时间、持续时间以及是否将标签复制到快照。	AWS SysAdmin、DBA
配置监控选项。	启用或禁用增强的监控和性能洞察。	AWS SysAdmin、DBA
配置维护选项。	指定次要版本自动升级、维护窗口以及开始日期、时间和持续时间。	AWS SysAdmin、DBA
运行 AWS SCT 中的预迁移脚本。	在 Amazon RDS 实例，运行以下脚本：create_database.sql、create_sequence.sql、create_table.sql、create_view.sql 和 create_function.sql。	AWS SysAdmin、DBA

使用 AWS DMS 迁移数据

Task	描述	所需技能
在 AWS DMS 中创建 AWS DMS 复制实例。	填写名称、实例类别、VPC（与 EC2 实例相同）、多可用区和公共可访问性字段。在高级下，指定分配的存	AWS SysAdmin、DBA

Task	描述	所需技能
	储、子网组、可用区、VPC 安全组和 AWS Key Management Service（AWS KMS）密钥。	
创建源数据库端点。	指定终端节点名称、类型、源引擎 (Oracle)、服务器名称（Amazon EC2 私有 DNS 名称）、端口、SSL 模式、用户名、密码、SID、VPC（指定具有复制实例的 VPC）和复制实例。要测试连接，请选择运行测试，然后创建端点。您还可以配置以下高级设置：maxFileSize 和 Sc numberDataType ale。	AWS SysAdmin、DBA
创建 AWS DMS 复制任务。	指定任务名称、复制实例、源端点和目标端点以及复制实例。对于迁移类型，选择迁移现有数据并复制持续更改。清除创建时启动任务复选框。	AWS SysAdmin、DBA
配置 AWS DMS 复制任务设置。	对于目标表格准备模式，请选择什么都不做。完全加载完成后停止任务（创建主键）。指定受限或完整 LOB 模式，然后启用控制表。或者，您可以配置 CommitRate 高级设置。	数据库管理员
配置表映射。	在表映射部分，为迁移中包含的所有架构中的所有表创建包含规则，然后创建排除规则。添加三个转换规则，将架构、表和列名转换为小写，并添加此特定迁移所需任何其他规则。	数据库管理员

Task	描述	所需技能
启动任务。	启动复制任务。确保全负载正在运行。在 Oracle 主数据库上运行 ALTER SYSTEM SWITCH LOGFILE 以启动任务。	数据库管理员
运行 AWS SCT 中的迁移中脚本。	在 Amazon RDS for PostgreSQL 中，运行以下脚本：create_index.sql 和 create_constraint.sql。	数据库管理员
重新启动任务以继续更改数据捕获 (CDC)。	在 Amazon RDS for PostgreSQL 数据库实例上运行 VACUUM，然后重启 AWS DMS 任务以应用缓存的 CDC 更改。	数据库管理员

割接至 PostgreSQL 数据库

Task	描述	所需技能
查看 AWS DMS 日志和验证表。	验证所有错误，并在需要时进行修复。	数据库管理员
停止所有 Oracle 依赖项。	停止所有 Oracle 依赖项，关闭 Oracle 数据库上的侦听器，然后运行 ALTER SYSTEM SWITCH LOGFILE。当 AWS DMS 任务没有显示任何活动时，将其停止。	数据库管理员
运行 AWS SCT 中的迁移后脚本。	在 Amazon RDS for PostgreSQL 中，运行以下脚本：create_foreign_key	数据库管理员

Task	描述	所需技能
	_constraint.sql 和 create_triggers.sql。	
完成 Amazon RDS for PostgreSQL 的其他步骤。	如果需要，增量序列以匹配 Oracle，运行 VACUUM 和 ANALYZE，然后拍摄合规性快照。	数据库管理员
打开 Amazon RDS for PostgreSQL 的连接。	从 Amazon RDS for PostgreSQL 中移除 AWS DMS 安全组，添加生产安全组，并将应用程序指向新数据库。	数据库管理员
清除 AWS 资源。	移除终端节点、复制任务、复制实例和 EC2 实例。	SysAdmin，DBA

相关资源

- AWS DMS 文档
- AWS SCT 文档
- Amazon RDS for PostgreSQL 定价
- 将 Oracle 数据库作为 AWS DMS 的源
- 将 PostgreSQL 数据库作为 AWS DMS 的目标
- 任务 SharePlex 文档

使用实体化视图和 AWS DMS 从 Oracle 8i 或 9i 迁移至 Amazon RDS for PostgreSQL

由 Kumar Babu PG (AWS) 和 Pragnesh Patel (AWS) 编写

摘要

此模式介绍如何将本地旧版 Oracle 8i 或 9i 数据库迁移到 Amazon Relational Database Service（Amazon RDS）for PostgreSQL 或 Amazon Aurora PostgreSQL 兼容版。

AWS Database Migration Service (AWS DMS) 不支持 Oracle 8i 或 9i 作为源，因此此模式使用与 AWS DMS 兼容的中间 Oracle 数据库实例，例如 Oracle 10g 或 11g。它还使用实例化视图功能将数据从源 Oracle 8i/9i 实例迁移至中间 Oracle 10g/11g 实例。

AWS Schema Conversion Tool (AWS SCT) 转换数据库架构，AWS DMS 会将数据迁移至目标 PostgreSQL 数据库。

此模式可以帮助希望以最短的数据库停机时间从旧版 Oracle 数据库迁移的用户。实施此示例时，停机时间将限于创建或验证目标数据库上的所有外键、触发器和序列所需时间长度。

该模式使用安装了 Oracle 10g/11g 数据库的亚马逊弹性计算云 (Amazon EC2) 实例来帮助 AWS DMS 流式传输数据。您可以暂时暂停从本地 Oracle 数据库到中间 Oracle 实例的流式复制，以使 AWS DMS 能够赶上数据验证或使用其他数据验证工具。当 AWS DMS 完成迁移当前更改后，PostgreSQL 数据库实例和中间 Oracle 数据库将具有相同的数据。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 本地数据中心中的源 Oracle 8i 或 9i 数据库
- AWS Direct Connect 配置至本地数据中心和 AWS 之间
- 安装在本地计算机或安装了 AWS SCT 的实例上的 AWS SCT 连接器的 Java 数据库连接 (JDBC) 驱动程序 EC2
- 熟悉使用 Oracle 数据库作为 AWS DMS 的源
- 熟悉使用 PostgreSQL 数据库作为 AWS DMS 的目标

限制

- 数据库大小限制为 64 TB

产品版本

- 作为源数据库的 Oracle 8i 或 9i
- 作为中间数据库的 Oracle 10g 或 11g
- PostgreSQL 10.17 或更高版本

架构

源技术堆栈

- Oracle 8i 或 9i 数据库

目标技术堆栈

- Amazon RDS for PostgreSQL 或 Aurora PostgreSQL-Compatible

目标架构

工具

- AWS DMS 可帮助您快速安全地迁移数据库。源数据库可在迁移过程中保持全面运行，从而最大程度地为依赖该数据库的应用程序减少停机时间。AWS DMS 可以在最广泛使用的商用和开源数据库之间迁移数据。
- AWS SCT 通过自动将源数据库架构和大部分数据库代码对象转换为与目标数据库兼容的格式，帮助您预测异构数据库迁移。任何无法自动转换的对象都会被明确标记，以便可以手动转换它们以完成迁移。AWS SCT 还可以扫描您的应用程序源代码中的嵌入式 SQL 语句，并将其作为数据库架构转换项目的一部分进行转换。在此过程中，AWS SCT 通过将旧的 Oracle 和 SQL Server 功能转换为其 AWS 等效功能来执行云原生代码优化，以帮助您在迁移数据库的同时实现应用程序现代化。架构转换完成后，AWS SCT 可以使用内置的数据迁移代理帮助将数据从一系列数据仓库迁移至 Amazon Redshift。

最佳实践

有关刷新实例化视图的最佳实践，请参见以下 Oracle 文档：

- 刷新实体化视图

- 实体化视图快速刷新

操作说明

在 EC2 实例上安装 Oracle 并创建物化视图

Task	描述	所需技能
为 EC2 实例设置网络。	创建新虚拟私有云（VPC）、子网、互联网网关、路由表和安全组。	AWS SysAdmin
创建实 EC2 例。	为 EC2 实例选择亚马逊系统映像 (AMI)。选择实例大小并配置实例详细信息：实例数量（1）、您在上一个任务中创建的 VPC 和子网、自动分配公有 IP 地址以及其他选项。添加存储、配置安全组并启动实例。出现提示时，创建并保存密钥对，以供下一步使用。	AWS SysAdmin
在 EC2 实例上安装 Oracle。	获取许可证和所需的 Oracle 二进制文件，然后在实例上安装 Oracle 10g 或 11g。 EC2	数据库管理员
配置 Oracle 联网。	在中修改或添加 `listener.ora` 条目，以连接到本地源 Oracle 8i/9i 数据库，然后创建数据库链接。	数据库管理员
创建实体化视图。	确定要在源 Oracle 8i/9i 数据库中复制的数据库对象，然后使用数据库链接为所有对象创建实体化视图。	数据库管理员
部署脚本，以按所需间隔刷新实例化视图。	开发和部署脚本，以便在 Amazon EC2 Oracle 10g/11g	数据库管理员

Task	描述	所需技能
	实例上按所需间隔刷新实体化视图。使用增量刷新选项来刷新实体化视图。	

将 Oracle 数据库架构转换为 PostgreSQL

Task	描述	所需技能
设置 AWS SCT。	创建新报告，然后连接到 Oracle 作为源，将 PostgreSQL 作为目标。在项目设置中，打开 SQL 脚本选项卡。将目标 SQL 脚本更改为多个文件。（AWS SCT 不支持 Oracle 8i/9i 数据库，因此您必须在中间 Oracle 10g/11g 实例上恢复仅限架构的转储，并将其用作 AWS SCT 的来源。）	数据库管理员
转换 Oracle 数据库架构。	在操作选项卡上，选择生成报告、转换架构，然后选择另存为 SQL。	数据库管理员
修改 SQL 脚本。	根据最佳实践标准进行修改。例如，切换到合适的数据类型，为特定 Oracle 函数开发 PostgreSQL 等效函数。	数据库管理员、Dev数据库管理员

创建并配置 Amazon RDS 数据库实例，以托管转换后的数据库

Task	描述	所需技能
创建 Amazon RDS 数据库实例。	在 Amazon RDS 控制台，创建新的 PostgreSQL 数据库实例。	AWS SysAdmin、DBA
配置数据库实例。	指定数据库引擎版本、数据库实例类、多可用区部署、存储类型和分配的存储空间。输入数据库实例标识符、主用户名和主密码。	AWS SysAdmin、DBA
配置网络和安全。	指定 VPC、子网组、公共可访问性、可用区首选项和安全组。	数据库管理员，SysAdmin
配置数据库选项。	指定数据库名称、端口、参数组、加密和主密钥。	DBA、AWS SysAdmin
配置备份。	指定备份保留期、备份窗口、开始时间、持续时间以及是否将标签复制到快照。	AWS SysAdmin、DBA
配置监控选项。	启用或禁用增强的监控和性能洞察。	AWS SysAdmin、DBA
配置维护选项。	指定次要版本自动升级、维护窗口以及开始日期、时间和持续时间。	AWS SysAdmin、DBA
运行 AWS SCT 中的预迁移脚本。	在目标 Amazon RDS for PostgreSQL 实例，使用来自 AWS SCT 的 SQL 脚本，并进行其他修改来创建数据库架构。这些可能包括运行多个脚本，并包括用户创建、数据库	AWS SysAdmin、DBA

Task	描述	所需技能
	创建、模式创建、表、视图、函数和其他代码对象。	

使用 AWS DMS 迁移数据

Task	描述	所需技能
在 AWS DMS 中创建 AWS DMS 复制实例。	填写名称、实例类别、VPC（与 EC2 实例相同）、多可用区和公共可访问性字段。在高级配置选项下，指定分配的存储、子网组、可用区、VPC 安全组和 AWS Key Management Service（AWS KMS）密钥。	AWS SysAdmin、DBA
创建源数据库端点。	指定终端节点名称、类型、源引擎 (Oracle)、服务器名称（ EC2 实例的私有 DNS 名称）、端口、SSL 模式、用户名、密码、SID、VPC（指定具有复制实例的 VPC）和复制实例。要测试连接，请选择运行测试，然后创建端点。您还可以配置以下高级设置：maxFileSize和 Sc numberDataTypeal e。	AWS SysAdmin、DBA
将 AWS DMS 连接到 Amazon RDS for PostgreSQL。	如果您的 PostgreSQL 数据库位于其他 VPC 中 VPCs，请为跨连接创建迁移安全组。	AWS SysAdmin、DBA
创建目标数据库端点。	指定端点名称、类型、源引擎 (PostgreSQL)、服务器名称（Amazon RDS 端点）、	AWS SysAdmin、DBA

Task	描述	所需技能
	端口、SSL 模式、用户名、密码、数据库名称、VPC（指定具有复制实例的 VPC）和复制实例。要测试连接，请选择运行测试，然后创建端点。您还可以配置以下高级设置：maxFileSize和 Sc numberDataTypeal e。	
创建 AWS DMS 复制任务。	指定任务名称、复制实例、源端点和目标端点以及复制实例。对于迁移类型，选择迁移现有数据并复制持续更改。清除创建时启动任务复选框。	AWS SysAdmin、DBA
配置 AWS DMS 复制任务设置。	对于目标表格准备模式，请选择什么都不做。完全加载完成后停止任务（创建主键）。指定受限或完整 LOB 模式，然后启用控制表。或者，您可以配置CommitRate高级设置。	数据库管理员
配置表映射。	在表映射部分，为迁移中包含的所有架构中的所有表创建包含规则，然后创建排除规则。添加三个转换规则，将架构、表和列名转换为小写，并添加此特定迁移所需任何其他规则。	数据库管理员
启动任务。	启动复制任务。确保全负载正在运行。在 Oracle 主数据库上运行 `ALTER SYSTEM SWITCH LOGFILE` 以启动任务。	数据库管理员

Task	描述	所需技能
运行 AWS SCT 中的迁移中脚本。	在 Amazon RDS for PostgreSQL 中，运行以下脚本：`create_index.sql` 和 `create_constraint.sql` （如果最初没有创建完整架构）。	数据库管理员
重新启动任务以继续更改数据捕获（CDC）。	在 Amazon RDS for PostgreSQL 数据库实例上运行 `VACUUM`，然后重启 AWS DMS 任务以应用缓存的 CDC 更改。	数据库管理员

割接至 PostgreSQL 数据库

Task	描述	所需技能
查看 AWS DMS 日志和验证表。	检查并修复所有复制或验证错误。	数据库管理员
停止使用本地 Oracle 数据库及其依赖项。	停止所有 Oracle 依赖项，关闭 Oracle 数据库上的侦听器，然后运行 `ALTER SYSTEM SWITCH LOGFILE`。当 AWS DMS 任务没有显示任何活动时，将其停止。	数据库管理员
运行 AWS SCT 中的迁移后脚本。	在 Amazon RDS for PostgreSQL 中，运行以下脚本：`create_foreign_key_constraint.sql` and `create_triggers.sql` 。确保序列为最新。	数据库管理员

Task	描述	所需技能
完成 Amazon RDS for PostgreSQL 的其他步骤。	如果需要，增量序列以匹配 Oracle，运行 `VACUUM` 和 `ANALYZE`，然后拍摄合规性快照。	数据库管理员
打开 Amazon RDS for PostgreSQL 的连接。	从 Amazon RDS for PostgreSQL 中移除 AWS DMS 安全组，添加生产安全组，并将应用程序指向新数据库。	数据库管理员
清理 AWS DMS 对象。	移除终端节点、复制任务、复制实例和 EC2 实例。	SysAdmin，DBA

相关资源

- AWS DMS 文档
- AWS SCT 文档
- Amazon RDS for PostgreSQL 定价
- 将 Oracle 数据库作为 AWS DMS 的源
- 将 PostgreSQL 数据库作为 AWS DMS 的目标

使用 AWS DMS 和 AWS SCT 从亚马逊上的 Oracle 迁移 EC2 到适用于 MySQL 的亚马逊 RDS

由 Anil Kunapareddy (AWS) 和 Harshad Gohil 编写

摘要

在亚马逊弹性计算云 (Amazon EC2) 实例上管理 Oracle 数据库需要资源，而且成本可能很高。将这些数据库迁移至适用于 MySQL 数据库实例的 Amazon Relational Database Service (Amazon RDS)，可以优化整体 IT 预算，从而简化您的工作。Amazon RDS for MySQL 还提供多可用区、可扩展性和自动备份等功能。

此模式将引导您完成将亚马逊上的源 Oracle 数据库迁移 EC2 到目标 Amazon RDS for MySQL 数据库实例的过程。它使用 AWS Database Migration Service (AWS DMS) 迁移数据，并使用 AWS Schema Conversion Tool (AWS SCT) 将源数据库架构和对象转换为与 Amazon RDS for MySQL 兼容的格式。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 在 ARCHIVELOG 模式下运行实例和侦听器服务的源数据库
- 目标 Amazon RDS for MySQL 数据库，拥有足够的存储空间用于数据迁移

限制

- AWS DMS 不会在目标数据库上创建架构；您必须这样做。Oracle 目标中很可能已存在该架构名称。来自源架构的表导入到用户或架构，使用它连接到目标实例。如果要迁移多个架构，您必须创建多个复制任务。

产品版本

- 10.2 及更高版本、11g 直至 12.2、18c 的所有 Oracle 数据库版本。有关支持的 Oracle 数据库版本的信息，请参阅 AWS 文档中的使用 Oracle 数据库作为 AWS DMS 源和使用 Oracle 数据库作为 AWS DMS 目标。建议使用最新版本的 AWS DMS，以获得最全面的版本和功能支持。关于 AWS SCT 支持的 Oracle 数据库版本的信息，请参阅 AWS SCT 文档。
- AWS DMS 支持版本 5.5、5.6 以及版本 5.7 的 MySQL。

架构

源技术堆栈

- EC2实例上的 Oracle 数据库

目标技术堆栈

- Amazon RDS for MySQL 数据库实例

数据迁移架构

源架构和目标架构

工具

- AWS DMS-AWS Datab ase Migration Service (AWS DMS) 是一项网络服务，可用于将数据从本地、Amazon RDS 数据库实例或实例上的数据库迁移到 AWS 服务上的数据库， EC2 例如 Amazon RDS for MySQL 或实例 EC2 。您还可以将数据库从 Amazon Web Services 迁移到本地数据库。您可以在异构或同构数据库引擎间迁移数据。
- AWS SCT - AWS Schema Conversion Tool (AWS SCT) 通过以下方法来简化异构数据库的迁移工作：将源数据库架构和大部分的自定义代码 (包括视图、存储过程和函数等) 自动转换成与目标数据库兼容的格式。使用 AWS SCT 转换数据库架构与代码对象后，您可以使用 AWS DMS 将数据从源数据库迁移至目标数据库，以完成迁移项目。

操作说明

计划迁移

Task	描述	所需技能
确定源数据库和目标数据库版本和引擎。		数据库管理员/开发人员
识别 AWS DMS 复制实例。		数据库管理员/开发人员

Task	描述	所需技能
确定存储需求（存储类型和容量）。		数据库管理员/开发人员
确定网络要求，包括延迟与带宽。		数据库管理员/开发人员
根据 Microsoft SQL Server 兼容性列表和容量要求，确定目标服务器实例的硬件要求。		数据库管理员/开发人员
确定源数据库和目标数据库的网络访问安全要求。		数据库管理员/开发人员
安装 AWS SCT 和 Oracle 驱动程序。		数据库管理员/开发人员
确定备份策略。		数据库管理员/开发人员
确定可用性要求。		数据库管理员/开发人员
确定应用程序迁移/切换策略。		数据库管理员/开发人员
选择正确的实例类型（容量、存储功能、网络功能）。		数据库管理员/开发人员

配置环境

Task	描述	所需技能
创建虚拟私有云（VPC）。源、目标和复制实例应位于同一 VPC 中。将它们放在同一可用区也很好。		开发人员
为访问数据库创建必要的安全组。		开发人员

Task	描述	所需技能
生成并配置密钥对。		开发人员
配置子网、可用区域和 CIDR 块。		开发人员

配置源： EC2 实例上的 Oracle 数据库

Task	描述	所需技能
使用所需的用户和角色在亚马逊 EC2 上安装 Oracle 数据库。		数据库管理员
执行下一列中的三个步骤，从 EC2 实例外部访问 Oracle。	1. 将本地主机更改`tnsnames`为 Amazon EC2 公有 DNS。 2. 将本地主机更改`listener`为 Amazon EC2 公有 DNS。 3. 停止并重新启动侦听器。	数据库管理员
Amazon EC2 重启后，公有 DNS 会发生变化。请务必在 “tnsnames” 和 “监听器” 中更新亚马逊 EC2 公有 DNS，或者使用弹性 IP 地址。		数据库管理员/开发人员
配置 EC2 实例安全组，以便复制实例和所需的客户端可以访问源数据库。		数据库管理员/开发人员

配置目标：Amazon RDS for MySQL

Task	描述	所需技能
配置并运行 Amazon RDS for MySQL 数据库实例。		开发人员
在 Amazon RDS for MySQL 数据库实例中创建必要表空间。		数据库管理员
配置安全组以复制实例和所需客户端可以访问目标数据库。		开发人员

配置 AWS SCT 并在目标数据库创建架构

Task	描述	所需技能
安装 AWS SCT 和 Oracle 驱动程序。		开发人员
输入相应参数，然后连接至源和目标。		开发人员
生成架构转换报告。		开发人员
根据需要更正代码和架构，尤其是表空间和引号，并在目标数据库上运行。		开发人员
在迁移数据之前验证源与目标上的架构。		开发人员

使用 AWS DMS 迁移数据

Task	描述	所需技能
为完全加载和更改数据捕获 (CDC) 或仅 CDC，您必须设置一个额外的连接属性。		开发人员
必须向 AWS DMS 源 Oracle 数据库定义中指定的用户授予所有必需的权限。有关完整列表，请参阅 https://docs.aws.amazon.com/dms/ latest/userguide/CHAP _Source.oracle.html #CHAP_Source .Oracle.Self-Managed。		数据库管理员/开发人员
在源数据库上启用补充日志记录		数据库管理员/开发人员
对于满载和 CDC（或仅适用于 CDC），请在源数据库上启用存档日志模式。		数据库管理员
创建源和目标数据库，并测试端点。		开发人员
成功连接端点后，创建复制任务。		开发人员
在任务中选择仅 CDC 或满载加 CDC，以捕获仅用于连续复制的更改，或者分别为满载和持续更改捕获更改。		开发人员
运行复制任务并监控 Amazon CloudWatch 日志。		开发人员

Task	描述	所需技能
验证源数据库和目标数据库中的数据。		开发人员

迁移您的应用程序并进行割接

Task	描述	所需技能
遵照应用程序迁移策略的步骤。		数据库管理员、开发人员、应用程序所有者
遵循所选的应用程序割接/切换策略。		数据库管理员、开发人员、应用程序所有者

关闭项目

Task	描述	所需技能
验证源和目标数据库中架构和数据。		数据库管理员/开发人员
收集与迁移时间、手动与工具各自的百分比、成本节约等相关的指标。		DBA/Developer/AppOwner
查看项目文档和构件。		DBA/Developer/AppOwner
关闭临时 AWS 资源。		数据库管理员/开发人员
关闭项目并提供反馈。		DBA/Developer/AppOwner

相关资源

- AWS DMS 文档
- AWS DMS 网站

- AWS DMS 博客文章
- Strategies for Migrating Oracle Database to AWS
- Amazon RDS for Oracle FAQs
- Oracle常见问题
- Amazon EC2
- Amazon EC2 FAQs
- 在云计算环境内许可 Oracle 软件

使用 AWS DMS 从 Oracle 迁移至 Amazon DocumentDB

由 Sashikanta Pattanayak (AWS) 和 Munesh Siddappa (AWS) 创作

摘要

此模式提供有关使用 AWS Database Migration Service (AWS DMS) 将 Oracle 数据库迁移至 Amazon DocumentDB（与 MongoDB 兼容）数据库的指导。此方法可应用于本地 Oracle 源数据库以及适用于 Oracle 数据库实例的 Amazon Relational Database Service (Amazon RDS)。此模式以 Amazon RDS Oracle 数据库源实例为例。

Amazon DocumentDB（与 MongoDB 兼容）是一种完全托管、与 MongoDB 兼容的文档数据库服务，可以轻松存储、查询和索引 JSON 数据。

这种模式的用例是将 Oracle 数据库表 one-to-one复制到 Amazon DocumentDB 集合。该模式使用 AWS DMS 复制任务读取 Oracle 数据库的表结构，在 Amazon DocumentDB 中创建相应的集合，并执行完整负载迁移。您可在 Amazon DocumentDB 中查看和查询您的数据，就像在 MongoDB 中一样。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 熟悉 Oracle 数据库的使用
- 熟悉使用 Amazon DocumentDB
- 对于 Oracle 用户，`SELECT ANY TABLE`权限
- 若要使用 Amazon DocumentDB，则需要转储数据所需权限

限制

将 Amazon DocumentDB 作为 AWS DMS 的目标时存在以下限制：

- 在 Amazon DocumentDB 中，集合名称不能包含美元符号（$）。此外，数据库名称不能包含任何 Unicode 字符。
- AWS DMS 不支持将多个源表合并到单个 Amazon DocumentDB 集合中。
- 当 AWS DMS 处理没有主键的源表中的更改时，将忽略该表中的任何 大型二进制对象 (LOB) 列。

- 如果更改表选项处于启用状态，并且发现名为“_id”的源列，则该列将在更改表中显示为“__id”（两条下划线）。
- 如果选择 Oracle 作为源端点，则 Oracle 源必须启用完整补充日志记录。否则，如果源位置有未更改的列，则数据将作为空值加载到 Amazon DocumentDB。

产品版本

- Amazon RDS for Oracle 版本 11.2.0.3 或更高版本
- AWS DMS 版本 3.1.3 或更高版本（有关最新版本信息，请参阅 AWS DMS 文档中的使用 Amazon DocumentDB 作为 AWS DMS 目标）

架构

源技术堆栈

- Amazon RDS for Oracle 数据库实例

目标技术堆栈

- Amazon DocumentDB

源架构和目标架构

工具

- AWS DMS — AWS Database Migration Service (AWS DMS)是一种 Web 服务，可用于将数据从源数据存储迁移到目标数据存储。AWS DMS 用户指南指定了 AWS DMS 支持使用的 Oracle 源数据库版本和版本。有关此模式的其他信息，请参阅 使用 Amazon DocumentDB 作为 AWS DMS 的目标。
- 亚马逊 EC2 — 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您的 Amazon DocumentDB 集群应在默认的虚拟私有云（VPC）中运行。要与您的 Amazon DocumentDB 集群进行交互，您必须在默认 VPC 中启动一个 EC2 实例，该实例位于您创建亚马逊文档数据库集群的同一 AWS 区域。有关详情，请参阅亚马逊 DocumentDB 文档中的启动亚马逊 EC2 实例。

操作说明

计划迁移

Task	描述	所需技能
验证源数据库和目标数据库的版本和引擎。		AWS 管理员
选择适当的实例类型（容量、存储功能、网络功能）。		AWS 管理员
确定源数据库和目标数据库的网络/主机访问安全要求。		AWS 管理员
为源数据库和目标数据库创建出站安全组。		AWS 管理员
为亚马逊文档数据库创建和配置 EC2 实例。		AWS 管理员

配置基础设施

Task	描述	所需技能
创建 VPC 和子网。		AWS 管理员
创建安全组和网络访问控制列表 (ACLs)。		AWS 管理员
配置和启动源 Amazon RDS for Oracle 实例。		AWS 管理员
配置并启动 Amazon DocumentDB 实例。		AWS 管理员

准备源数据库

Task	描述	所需技能
使用连接详细信息验证 Oracle 数据库是否可连接。		AWS 管理员
验证 Oracle 用户是否具有选择任意表权限。		AWS 管理员

准备目标数据库

Task	描述	所需技能
通过选择正确的实例类和实例数量创建 Amazon DocumentDB 集群。		AWS 管理员

配置 Amazon EC2

Task	描述	所需技能
配置实 EC2 例。	要与您的 Amazon DocumentDB 集群进行交互，您必须在默认 VPC 中启动一个 EC2 实例，该实例位于您创建亚马逊文档数据库集群的同一 AWS 区域。为 EC2 实例配置 AWS 区域 VPCs、可用区和子网。	AWS 管理员
配置密钥对。	公钥/私钥对（public/Private key pair）允许您在 EC2 实例启动后安全地连接到该实例。	AWS 管理员

Task	描述	所需技能
设置堡垒主机 CIDR 范围（可选）。	设置允许外部 Secure Shell (SSH) 访问堡垒主机实例的 CIDR IP 范围。	AWS 管理员

迁移数据 - 满载

任务	描述	所需技能
创建 AWS DMS 复制实例。		AWS 管理员
创建源和目标端点。		AWS 管理员
为满负荷创建 AWS DMS 复制任务。		AWS 管理员

测试迁移

Task	描述	所需技能
通过该实例连接到 Amazon DocumentDB 集群。 EC2		AWS 管理员
使用 mongo shell 连接到集群。	有关说明，请参阅参考和帮助部分中的 Amazon DocumentDB 链接。	AWS 管理员
验证迁移结果。		AWS 管理员

相关资源

- AWS DMS 的工作原理
- 迁移至 Amazon DocumentDB
- 使用 Amazon DocumentDB 作为 AWS DMS 的目标
- Amazon DocumentDB 概述

- 使用 mongo Shell 访问和使用您的 Amazon DocumentDB 集群
- 使用离线方法从 MongoDB 迁移至 Amazon DocumentDB（博客文章）
- 如何使用 Amazon DocumentDB（与 MongoDB 兼容）大规模构建和管理应用程序（博客文章）

使用 AWS DMS 和 AWS SCT 将 Oracle 数据库从亚马逊迁移 EC2 到适用于 MariaDB 的亚马逊 RDS

由 Veeranjaneyulu Grandhi (AWS) 和 vinod kumar (AWS) 创建

摘要

此模式将引导您完成将亚马逊弹性计算云 (Amazon EC2) 实例上的 Oracle 数据库迁移到适用于 MariaDB 数据库实例的亚马逊关系数据库服务 (Amazon RDS) 的步骤。此模式使用 AWS Data Migration Service (AWS DMS) 进行数据迁移，使用 AWS Schema Conversion Tool (AWS SCT) 进行架构转换。

与在 Amazon RDS 上使用数据库相比，在 EC2 实例上管理 Oracle 数据库需要更多资源，而且成本更高。 Amazon RDS 可以轻松地在云中设置、操作和扩展关系数据库。Amazon RDS 提供经济实惠、且可调整的容量，同时自动执行耗时管理任务，例如硬件预置、数据库设置、修补和备份。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 启动并运行实例和侦听器服务的源 Oracle 数据库。此数据库应处于 ARCHIVELOG 模式。
- 熟悉使用 Oracle 数据库作为 AWS DMS 的源。
- 熟悉使用 Oracle 作为 AWS SCT 的源。

限制

- 数据库大小限制：64 TB

产品版本

- 10.2 及更高版本、11g 直至 12.2、18c 的所有 Oracle 数据库版本。有关支持版本的最新列表，请参阅 AWS 文档中的使用 Oracle 数据库作为 AWS DMS 的源和 AWS SCT 版本表。
- Amazon RDS 支持 MariaDB Server Community Server 版本 10.3、10.4、10.5 和 10.6。有关支持版本的最新列表，请参阅 Amazon RDS 文档。

架构

源技术堆栈

- EC2 实例上的 Oracle 数据库

目标技术堆栈

- Amazon RDS for MariaDB

数据迁移架构

目标架构

工具

- AWS Schema Conversion Tool（AWS SCT）通过自动将源数据库架构和大部分数据库代码对象（包括视图、存储过程和功能）转换为与目标数据库兼容的格式，预测异构数据库迁移。使用 AWS SCT 转换数据库架构与代码对象后，您可以使用 AWS DMS 将数据从源数据库迁移至目标数据库，以完成迁移项目。有关更多信息，请参阅 AWS SCT 文档中的使用 Oracle 作为 AWS SCT 的目标。
- AWS Database Migration Service (AWS DMS) 可帮助您快速安全地将数据库迁移到 AWS。源数据库可在迁移过程中保持全面运行，从而最大程度地为依赖该数据库的应用程序减少停机时间。AWS DMS 可以在最广泛使用的商用和开源数据库之间迁移数据。AWS DMS 支持同构迁移 (如 Oracle 到 Oracle)，也支持不同数据库平台之间的异构迁移（如 Oracle 到 Amazon Aurora，或 Microsoft SQL Server 到 Amazon Aurora）。若要了解有关迁移 Oracle 数据库的更多信息，请参阅 AWS DMS 文档中的使用 Oracle 数据库作为 AWS DMS 的源。

操作说明

计划迁移

Task	描述	所需技能
确定版本和数据库引擎。	确定源数据库和目标数据库版本和引擎。	数据库管理员、开发人员
确定复制实例。	确定 AWS DMS 复制实例。	数据库管理员、开发人员
确定存储要求。	确定存储类型与容量。	数据库管理员、开发人员

Task	描述	所需技能
识别网络要求。	确定网络延迟和带宽。	数据库管理员、开发人员
确定硬件要求。	（根据 Oracle 兼容性列表和容量要求）确定源服务器实例和目标服务器实例的硬件要求。	数据库管理员、开发人员
确定安全要求。	确定源数据库和目标数据库的网络访问安全要求。	数据库管理员、开发人员
安装驱动程序。	安装最新 AWS SCT 和 Oracle 驱动程序。	数据库管理员、开发人员
确定备份策略。		数据库管理员、开发人员
确定可用性要求。		数据库管理员、开发人员
选择应用程序迁移/切换策略。		数据库管理员、开发人员
选择实例类型	根据容量、存储和网络功能选择正确的实例类型。	数据库管理员、开发人员

配置环境

Task	描述	所需技能
创建虚拟私有云（VPC）。	源实例、目标实例和复制实例应位于同一 VPC 和同一可用区（推荐)。	开发人员
创建安全组。	为访问数据库创建必要的安全组。	开发人员
生成密钥对。	生成并配置密钥对。	开发人员
配置其他资源。	配置子网、可用区域和 CIDR 块。	开发人员

配置源

Task	描述	所需技能
启动实 EC2 例。	有关说明，请参阅 Amazon EC2 文档。	开发人员
安装 Oracle 数据库。	使用所需的用户和角色在 EC2 实例上安装 Oracle 数据库。	数据库管理员
按照任务描述中的步骤从 EC2 实例外部访问 Oracle。	1. 将本地主机更改`tnsnames`为 Amazon EC2 公有 DNS。 2. 将本地主机更改`listener`为 Amazon EC2 公有 DNS。 3. 停止并重新启动侦听器。	数据库管理员
更新 Amazon EC2 公有 DNS。	EC2 实例重启后，公有 DNS 会发生变化。请务必更新`tnsnames`和中的 Amazon EC2 公有 DNS`listener`，或者使用弹性 IP 地址。	数据库管理员、开发人员
配置 EC2 实例安全组。	配置 EC2 实例安全组，以便复制实例和所需的客户端可以访问源数据库。	数据库管理员、开发人员

配置 目标 Amazon RDS for MariaDB 环境

Task	描述	所需技能
启动 RDS 数据库实例。	配置并启动 Amazon RDS for MariaDB 数据库实例。	开发人员
创建表空间。	在 Amazon RDS MariaDB 数据库中创建任何必要表空间。	数据库管理员

Task	描述	所需技能
配置安全组。	配置实例安全组，以便复制实例和所需客户端访问目标数据库。	开发人员

配置 AWS SCT

Task	描述	所需技能
安装驱动程序。	安装最新 AWS SCT 和 Oracle 驱动程序。	开发人员
连接。	输入相应参数，然后连接至源和目标。	开发人员
生成架构转换报告。	生成 AWS SCT 架构转换报告。	开发人员
根据需要更正代码与架构。	对代码和架构（尤其是表空间和引号）进行必要的更正。	数据库管理员、开发人员
验证架构。	加载数据之前验证源和目标上的架构。	开发人员

使用 AWS DMS 迁移数据

Task	描述	所需技能
设置连接属性。	对于完全加载和更改数据捕获 (CDC) 或仅作为 CDC，请设置一个额外的连接属性。有关更多信息，请参阅 Amazon RDS 文档。	开发人员
设置补充日志记录。	在源数据库上启用补充日志记录。	数据库管理员、开发人员

Task	描述	所需技能
启用存档日志模式。	对于完全加载和 CDC（或仅适用于 CDC），请在源数据库上启用存档日志模式。	数据库管理员
创建并测试端点。	创建源端点和目标端点并测试连接。有关更多信息，请参阅 Amazon DMS 文档。	开发人员
创建复制任务。	成功连接端点后，创建复制任务。有关更多信息，请参阅 Amazon DMS 文档。	开发人员
选择 复制类型。	在任务中选择仅 CDC 或完全加载外加 CDC，以捕获仅用于连续复制的更改或完全加载和持续更改。	开发人员
启动并监视任务。	启动复制任务并监控 Amazon CloudWatch 日志。有关更多信息，请参阅 Amazon DMS 文档。	开发人员
验证数据。	验证源数据库和目标数据库中的数据。	开发人员

迁移应用程序并割接至目标数据库

Task	描述	所需技能
遵循选定的应用程序迁移策略。		数据库管理员、应用程序所有者、开发人员
遵循选定的应用程序割接/切换策略。		数据库管理员、应用程序所有者、开发人员

关闭项目

Task	描述	所需技能
验证架构与数据。	确保在项目结束之前在源和目标中成功验证架构与数据。	数据库管理员、开发人员
收集指标。	收集与迁移时间、手动任务与工具任务的百分比、成本节约等类似标准。	数据库管理员、应用程序所有者、开发人员
查看文档。	查看项目文档和构件。	数据库管理员、应用程序所有者、开发人员
关闭资源。	关闭临时 AWS 资源。	数据库管理员、开发人员
关闭项目。	关闭迁移项目并提供任何反馈。	数据库管理员、应用程序所有者、开发人员

相关资源

- MariaDB Amazon RDS 概述
- Amazon RDS for MariaDB 产品详细信息
- 将 Oracle 数据库作为 AWS DMS 的源
- 将 Oracle 数据库迁移至 AWS 的策略
- 在云计算环境内许可 Oracle 软件
- Amazon RDS for Oracle FAQs
- AWS DMS 概述
- AWS DMS 博客文章
- 亚马逊 EC2 概述
- Amazon EC2 FAQs
- AWS SCT 文档

使用 AWS DMS 和 AWS SCT 将本地 Oracle 数据库迁移至 Amazon RDS for MySQL

由谢尔盖·德米特里耶夫（AWS）和纳雷什·达梅拉（AWS）创作

摘要

此模式将引导您完成将本地 Oracle 数据库迁移至 Amazon Relational Database Service (Amazon RDS) for MySQL 数据库实例的过程。它使用 AWS Database Migration Service (AWS DMS) 迁移数据，并使用 AWS Schema Conversion Tool (AWS SCT) 将源数据库架构和对象转换为与 Amazon RDS for MySQL 兼容的格式。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 本地数据中心中的 Oracle 源数据库

限制

- 数据库大小限制：64 TB

产品版本

- 11g 版本（包括版本 11.2.0.3.v1 及更高版本）以及最高 12.2 和 18c 的所有 Oracle 数据库版本。有关支持版本的最新列表，请参阅使用 Oracle 数据库作为 AWS DMS 的源。建议使用最新版本的 AWS DMS，以获得最全面的版本和功能支持。关于 AWS SCT 支持的 Oracle 数据库版本的信息，请参阅 AWS SCT 文档。
- AWS DMS 当前支持 MySQL 5.5、5.6 和 5.7 版。有关受支持版本的最新列表，请参阅 AWS 文档中的将 MySQL 兼容数据库作为 AWS DMS 的目标。

架构

源技术堆栈

- 本地 Oracle 数据库

目标技术堆栈

- Amazon RDS for MySQL 数据库实例

数据迁移架构

工具

- AWS DMS — AWS Database Migration Services (AWS DMS) 可帮助您迁移关系数据库、数据仓库、NoSQL 数据库和其他类型的数据存储。您可以使用 AWS DMS 将数据迁移到 Amazon Web Services Cloud、本地实例之间（通过 Amazon Web Services Cloud 设置）或云和本地设置的组合之间。
- AWS SCT — AWS Schema Conversion Tool (AWS SCT) 用于将数据库架构从一个数据库引擎转换为另一个数据库引擎。该工具转换的自定义代码包括视图、存储过程和函数。该工具无法自动转换的任意代码会被清楚地标记，以便您自己进行转换。

操作说明

计划迁移

Task	描述	所需技能
验证源数据库和目标数据库版本和引擎。		数据库管理员
确定目标服务器实例的硬件要求。		数据库管理员，SysAdmin
确定存储需求（存储类型和容量）。		数据库管理员，SysAdmin
根据容量、存储功能和网络功能选择正确的实例类型。		数据库管理员，SysAdmin
确定源数据库和目标数据库的网络访问安全要求。		数据库管理员，SysAdmin
确定应用程序迁移策略。		DBA、SysAdmin、应用程序所有者

配置基础设施

Task	描述	所需技能
创建虚拟私有云（VPC）和子网。		SysAdmin
创建安全组和网络访问控制列表 (ACLs)。		SysAdmin
配置和启动运行 Amazon RDS 数据库实例。		数据库管理员，SysAdmin

迁移数据

Task	描述	所需技能
使用 AWS SCT 迁移数据库架构		数据库管理员
使用 AWS DMS 迁移数据。		数据库管理员

迁移应用程序

Task	描述	所需技能
使用 AWS SCT 分析并转换应用程序代码中的 SQL 代码。	有关更多信息，请参阅 https://docs.aws.amazon.com/SchemaConversionTool/latest/userguide/CHAP_converting.app.html。	应用程序所有者
遵循应用程序迁移策略。		DBA、SysAdmin、应用程序所有者

割接

Task	描述	所需技能
将应用程序客户端切换至新基础设施。		DBA、 SysAdmin、应用程序所有者

关闭项目

Task	描述	所需技能
关闭临时 AWS 资源。		数据库管理员 ， SysAdmin
查看和验证项目文档。		数据库管理员 ， SysAdmin
收集与迁移时间、手动与工具各自的百分比、成本节约等相关的指标。		数据库管理员 ， SysAdmin
关闭项目并提供反馈。		

相关资源

参考

- AWS DMS 文档
- AWS SCT 文档
- Amazon RDS 定价

教程和视频

- AWS DMS 入门
- Amazon RDS 入门
- AWS DMS（视频）
- Amazon RDS（视频）

使用 Oracle Bystander 和 AWS DMS 将本地 Oracle 数据库迁移到 Amazon RDS for PostgreSQL

创建者：Cady Motyka（AWS）

摘要

此模式描述了如何在最短的停机时间内将本地 Oracle 数据库迁移到以下任一与 PostgreSQL 兼容的 AWS 数据库服务：

- Amazon Relational Database Service (Amazon RDS) for PostgreSQL
- Amazon Aurora PostgreSQL 兼容版

该解决方案使用 AWS Database Migration Service（AWS DMS）迁移数据，使用 AWS Schema Conversion Tool（AWS SCT）来转换数据库架构，使用 Oracle Bystander 数据库来帮助管理迁移。在此实施中，停机时间仅限于在数据库上创建或验证所有外键所需时间。

该解决方案还使用带有 Oracle 旁观者数据库的亚马逊弹性计算云 (Amazon EC2) 实例，帮助控制通过 AWS DMS 的数据流。您可以暂时暂停从本地 Oracle 数据库到 Oracle Bystander 的流式复制，以激活 AWS DMS 以赶上数据验证或使用其他数据验证工具。当 AWS DMS 完成当前更改的迁移后，Amazon RDS for PostgreSQL 数据库实例或 Aurora PostgreSQL-Compatible 数据库实例和 Bystander 数据库将具有相同的数据。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 本地数据中心中的 Oracle 源数据库，配置了 Active Data Guard（ADG）备用数据库
- AWS Direct Connect 配置在本地数据中心和 AWS Secrets Manager 之间，用于存储数据库机密
- AWS SCT 连接器的 Java 数据库连接 (JDBC) 驱动程序，安装在本地计算机上或安装了 AWS SCT 的 EC2 实例上
- 熟悉使用 Oracle 数据库作为 AWS DMS 的源数据库
- 熟悉使用 PostgreSQL 数据库作为 AWS DMS 的目标数据库

限制

- 数据库大小限制：64 TB

产品版本

- AWS DMS 支持版本为 10.2 及更高版本（对于版本 10.x）、11g 直至 12.2、18c 以及 19c 的所有 Oracle 数据库版本。有关支持版本的最新列表，请参阅使用 Oracle 数据库作为 AWS DMS 的源。建议使用最新版本的 AWS DMS，以获得最全面的版本和功能支持。关于 AWS SCT 支持的 Oracle 数据库版本的信息，请参阅 AWS SCT 文档。
- AWS DMS 支持版本为 9.4 和更高版本（对于版本 9.x）、10.x、11.x、12.x 和 13.x 的 PostgreSQL。有关最新信息，请参阅 AWS 文档中的使用 PostgreSQL 数据库作为 AWS DMS 的目标。

架构

源技术堆栈

- 本地 Oracle 数据库
- 一个EC2 包含 Oracle 数据库旁观者的实例

目标技术堆栈

- Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 实例，PostgreSQL 9.3 及更高版本

目标架构

下图显示了使用 AWS DMS 和 Oracle Bystander 将 Oracle 数据库迁移至与 PostgreSQL 兼容的 AWS 数据库的示例工作流程：

工具

- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到 Amazon Web Services Cloud，或者在云和本地设置的组合之间迁移。
- AWS Schema Conversion Tool（AWS SCT） 通过以下方法来简化异构数据库的迁移工作：将源数据库架构和大部分的自定义代码自动转换成与目标数据库兼容的格式。
- Amazon Relational Database Service (Amazon RDS) 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展关系数据库。

操作说明

将 Oracle 数据库架构转换为 PostgreSQL

Task	描述	所需技能
设置 AWS SCT。	创建新报告，然后连接到 Oracle 作为源，将 PostgreSQL 作为目标。在项目设置中，转到 SQL 脚本选项卡。将目标 SQL 脚本更改为多个文件。这些文件将在以后使用，命名如下： • create_database.sql • create_sequence.sql • create_table.sql • create_view.sql • create_function.sql	数据库管理员
转换 Oracle 数据库架构。	在操作选项卡中，选择生成报告。然后，选择转换架构，然后选择另存为 SQL。	数据库管理员
修改脚本。	例如，如果源架构中的数字已转换为 PostgreSQL 中的数字格式，则可能需要修改脚本，但要改用 BIGINT 以提高性能。	数据库管理员

创建和配置 Amazon RDS 数据库实例

Task	描述	所需技能
创建 Amazon RDS 数据库实例。	在正确的 AWS 区域中，创建一个新 PostgreSQL 数据库实例。有关更多信息，请参阅	AWS SysAdmin、DBA

Task	描述	所需技能
	Amazon RDS 文档中的创建 PostgreSQL 数据库实例并连接到 PostgreSQL 数据库实例上的数据库。	
配置数据库实例规格。	指定数据库引擎版本、数据库实例类、多可用区部署、存储类型和分配的存储空间。输入数据库实例标识符、主用户名和主密码。	AWS SysAdmin、DBA
配置网络和安全。	指定虚拟私有云（VPC）、子网组、公共可访问性、可用区首选项和安全组。	数据库管理员，SysAdmin
配置数据库选项。	指定数据库名称、端口、参数组、加密和 KMS 密钥。	AWS SysAdmin、DBA
配置备份。	指定备份保留期、备份窗口、开始时间、持续时间以及是否将标签复制到快照。	AWS SysAdmin、DBA
配置监控选项。	激活或停用增强的监控和性能见解。	AWS SysAdmin、DBA
配置维护选项。	指定次要版本自动升级、维护窗口以及开始日期、时间和持续时间。	AWS SysAdmin、DBA

Task	描述	所需技能
运行 AWS SCT 中的预迁移脚本。	在 Amazon RDS 实例上，运行以下由 AWS SCT 生成的脚本： • create_database.sql • create_sequence.sql • create_table.sql • create_view.sql • create_function.sql	AWS SysAdmin、DBA

在 Amazon 中配置 Oracle 旁观者 EC2

Task	描述	所需技能
为 Amazon 设置网络 EC2。	创建新 VPC、子网、互联网网关、路由表和安全组。	AWS SysAdmin
创建实 EC2 例。	在相应的 AWS 区域中，创建一个新 EC2 实例。选择亚马逊机器映像（AMI），选择实例大小并配置实例详细信息：实例数量 (1)、您在上一个任务中创建的 VPC 和子网、自动分配公有 IP 地址以及其他选项。添加存储、配置安全组并启动。出现提示时，创建并保存密钥对，以供下一步使用。	AWS SysAdmin
将 Oracle 源数据库连接到实 EC2 例。	将 IPv4 公有 IP 地址和 DNS 复制到文本文件中，然后使用 SSH 进行连接，如下所示：ssh -i “your_file.pem” ec2-user@<your-ip--DNS>。address-or-public	AWS SysAdmin

Task	描述	所需技能
在 Amazon 中为旁观者设置初始主机。 EC2	设置 SSH 密钥、bash 配置文件、ORATAB 和符号链接。创建 Oracle 目录。	AWS SysAdmin，Linux 管理员
在 Amazon 中为旁观者设置数据库副本 EC2	使用 RMAN 创建数据库副本、启用补充日志记录和创建备用控制文件。复制完成后，将数据库置于恢复模式。	AWS SysAdmin、DBA
设置 Oracle Data Guard。	修改 listener.ora 文件并启动侦听器。设置新的存档目标。将旁观者置于恢复模式，替换临时文件以避免将来损坏，必要时安装 crontab 以防止存档目录空间不足，并编辑源和备用目录的 manage-trclog-files-oracle.cfg 文件。	AWS SysAdmin、DBA
准备 Oracle 数据库以同步传送。	添加备用日志文件并更改恢复模式。在源主服务器和源备用服务器上都将日志传送更改为 SYNC AFFIRM。将日志切换到主日志上，通过 Amazon EC2 旁观者警报日志确认您正在使用备用日志文件，并确认重做流正在同步流动。	AWS SysAdmin、DBA

使用 AWS DMS 迁移数据

Task	描述	所需技能
在 AWS DMS 中创建 AWS DMS 复制实例。	填写名称、实例类别、VPC（与 Amazon EC2 实例相同）、多可用区和公共可访问性字段。在高级下方，指	AWS SysAdmin、DBA

Task	描述	所需技能
	定分配的存储空间、子网组、可用区、VPC 安全组和 AWS Key Management Service（AWS KMS）密钥。	
创建源数据库端点。	指定终端节点名称、类型、源引擎 (Oracle)、服务器名称（Amazon EC2 私有 DNS 名称）、端口、SSL 模式、用户名、密码、SID、VPC（指定具有复制实例的 VPC）和复制实例。要测试连接，请选择运行测试，然后创建端点。您还可以配置以下高级设置：maxFileSize和 ScnumberDataTypeal e。	AWS SysAdmin、DBA
将 AWS DMS 连接到 Amazon RDS for PostgreSQL。	为跨连接创建迁移安全组 VPCs。	AWS SysAdmin、DBA
创建目标数据库端点。	指定端点名称、类型、源引擎 (PostgreSQL)、服务器名称（Amazon RDS 端点）、端口、SSL 模式、用户名、密码、数据库名称、VPC（指定具有复制实例的 VPC）和复制实例。要测试连接，请选择运行测试，然后创建端点。您还可以配置以下高级设置：maxFileSize 和 ScnumberDataTypeal e。	AWS SysAdmin、DBA

Task	描述	所需技能
创建 AWS DMS 复制任务。	指定任务名称、复制实例、源端点和目标端点以及复制实例。对于迁移类型，选择迁移现有数据并复制持续更改。取消勾选创建时启动任务复选框。	AWS SysAdmin、DBA
配置 AWS DMS 复制任务设置。	对于目标表格准备模式，请选择什么都不做。完全加载完成后停止任务（创建主键）。指定受限或完整 LOB 模式，然后激活控制表。或者，您可以配置CommitRate高级设置。	数据库管理员
配置表映射。	在表映射部分，为迁移中包含的所有架构中的所有表创建包含规则，然后创建排除规则。添加三个转换规则，将架构、表和列名转换为小写，并添加此特定迁移所需任何其他规则。	数据库管理员
启动任务。	启动复制任务。确保全负载正在运行。在 Oracle 主数据库上运行 ALTER SYSTEM SWITCH LOGFILE 以启动任务。	数据库管理员
运行 AWS SCT 中的迁移中脚本。	在 Amazon RDS for PostgreSQL 上，运行由 AWS SCT 生成的以下脚本： • create_index.sql • create_constraint.sql	数据库管理员

Task	描述	所需技能
重新启动任务以继续更改数据捕获 (CDC)。	在 Amazon RDS for PostgreSQL 数据库实例上运行 VACUUM，然后重启 AWS DMS 任务以应用缓存的 CDC 更改。	数据库管理员

割接至 PostgreSQL 数据库

Task	描述	所需技能
查看 AWS DMS 日志和验证表中是否存在任何错误。	检查并修复所有复制或验证错误。	数据库管理员
停止所有 Oracle 依赖项。	停止所有 Oracle 依赖项，关闭 Oracle 数据库上的侦听器，然后运行 ALTER SYSTEM SWITCH LOGFILE。当 AWS DMS 任务没有显示任何活动时，将其停止。	数据库管理员
运行 AWS SCT 中的迁移后脚本。	在 Amazon RDS for PostgreSQL 上，运行由 AWS SCT 生成的以下脚本： • create_foreign_key_constraint.sql • create_triggers.sql	数据库管理员
完成 Amazon RDS for PostgreSQL 的其他步骤。	如果需要，增量序列以匹配 Oracle，运行 VACUUM 和 ANALYZE，然后拍摄合规性快照。	数据库管理员
打开 Amazon RDS for PostgreSQL 的连接。	从 Amazon RDS for PostgreSQL 中移除 AWS	数据库管理员

Task	描述	所需技能
	DMS 安全组，添加生产安全组，并将应用程序指向新数据库。	
清理 AWS DMS 对象。	移除终端节点、复制任务、复制实例和 EC2 实例。	SysAdmin，DBA

相关资源

- AWS DMS 文档
- AWS SCT 文档
- Amazon RDS for PostgreSQL 定价

使用 Oracle 从 Oracle 数据库迁移到 Amazon RDS for PostgreSQL GoldenGate

由 Dhairya Jindani (AWS)、Rajeshkumar Sabankar (AWS) 和 Sindhusha Paturu (AWS) 编写

摘要

此模式展示了如何使用甲骨文云基础设施 (OCI) 将甲骨文数据库迁移到适用于 PostgreSQL 的亚马逊关系数据库服务 (Amazon RDS)。 GoldenGate

通过使用 Oracle GoldenGate，您可以在源数据库和一个或多个目标数据库之间复制数据，最大限度地减少停机时间。

> **Note**
>
> 源 Oracle 数据库可以位于本地，也可以位于亚马逊弹性计算云 (Amazon EC2) 实例上。使用本地复制工具时，您可使用类似的过程。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 甲骨文 GoldenGate 许可证
- 连接至 PostgreSQL 数据库的 Java Database Connectivity (JDBC) 驱动程序
- 在目标 Amazon RDS for PostgreSQL 数据库上使用 AWS Schema Conversion Tool (AWS SCT) 创建的架构和表

限制

- Oracle GoldenGate 只能复制现有表数据（初始加载）和正在进行的更改（更改数据捕获）

产品版本

- Oracle Database Enterprise Edition 10g 或更高版本
- 适用于 Oracle GoldenGate 或更高版本的 Oracle 12.2.0.1.1
- GoldenGate适用于 PostgreSQL 或更高版本的 Oracle 12.2.0.1.1

架构

下图显示了使用 Oracle 将 Oracle 数据库迁移到 Amazon RDS for PostgreSQL 的示例工作流程：GoldenGate

图表显示了以下工作流：

1. Oracle 数据 GoldenGate 提取进程对源数据库运行以提取数据。
2. Oracle GoldenGate Replicat 流程将提取的数据传送到目标 Amazon RDS for PostgreSQL 数据库。

工具

- Oracle GoldenGate 可帮助您在 Oracle 云基础设施中设计、运行、编排和监控数据复制和流数据处理解决方案。
- Amazon Relational Database Service（Amazon RDS）for PostgreSQL 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展 PostgreSQL 关系数据库。

操作说明

下载并安装 Oracle GoldenGate

Task	描述	所需技能
下载甲骨文 GoldenGate。	下载以下版本的 Oracle GoldenGate： • 适用于 Oracle 的 Oracle GoldenGate 12.2.0.1.1 或更高版本 • 适用于 PostgreSQL 或更 GoldenGate高版本的 Oracle 12.2.0.1.1 要下载该软件，请参阅 Oracle 网站上的 Oracle GoldenGate 下载。	数据库管理员

Task	描述	所需技能
在源 Oracle GoldenGate 数据库服务器上安装 Oracle for Oracle。	有关说明，请参阅 Oracle GoldenGate 文档。	数据库管理员
在亚马逊实例上安装 Oracle GoldenGate for PostgreSQL 数据库。 EC2	有关说明，请参阅 Oracle GoldenGate 文档。	数据库管理员

在源数据库和目标数据库 GoldenGate 上配置 Oracle

Task	描述	所需技能
在源数据库上设置 GoldenGate Oracle for Oracle 数据库。	有关说明，请参阅 Oracle GoldenGate 文档。 务必配置以下： • 补充日志记录 • 甲骨文 GoldenGate 用户 • 任何必要授权和权限 • 参数文件 • 管理器流程 • 目录 • GLOBALS 文件 • Oracle Wallet	数据库管理员
在目标数据库上设置 Oracle GoldenGate for PostgreSQL。	有关说明，请参阅 Oracle 网站上的第六部分 “使用 Oracle GoldenGate for PostgreSQL”。 务必配置以下： • 管理器流程	数据库管理员

Task	描述	所需技能
	• GLOBALS 文件 • Oracle Wallet	

配置数据捕获

Task	描述	所需技能
在源数据库设置提取进程。	在源 Oracle 数据库，创建用于提取数据的提取文件。 有关说明，请参阅 Oracle 文档中的 ADD EXTRACT。 **Note** 提取文件包括创建提取参数文件和跟踪文件目录。	数据库管理员
设置一个数据泵，以将跟踪文件从源传输到目标数据库。	通过按照 Oracle 网站上数据库实用程序中的 PARFILE 中的说明，创建 EXTRACT 参数文件和跟踪文件目录。 有关更多信息，请参阅什么是跟踪？ 在 Oracle GoldenGate 网站上的 Fusion 中间件了解甲骨文中。	数据库管理员
在 Amazon EC2 实例上设置复制。	创建复制参数文件和跟踪文件目录。 有关创建复制参数文件的更多信息，请参阅 Oracle 数据库文档中的第3.5 节验证参数文件。	数据库管理员

Task	描述	所需技能
	有关创建跟踪文件目录的更多信息，请参阅 Oracle Cloud 文档中的创建跟踪。 **Important** 确保在目标位置的 GLOBALS 文件中添加检查点表条目。 有关更多信息，请参阅什么是副本？ 在 Oracle GoldenGate 网站上的 Fusion 中间件了解甲骨文中。	

配置数据复制

Task	描述	所需技能
在源数据库中，创建一个参数文件，以提取初始加载的数据。	按照 Oracle Cloud 文档中的在 GGSCI 中创建参数文件说明进行操作。 **Important** 确保管理器已在目标上运行。	数据库管理员
在目标数据库中，创建一个参数文件，以复制初始加载的数据。	按照 Oracle Cloud 文档中的在 GGSCI 中创建参数文件说明进行操作。	数据库管理员

Task	描述	所需技能
	Important 请务必添加并启动复制进程。	

割接到 Amazon RDS for PostgreSQL 数据库

Task	描述	所需技能
停止复制进程，并确保源和目标数据库是同步的。	比较源数据库和目标数据库之间的行计数，以确保数据复制成功。	数据库管理员
配置数据定义语言 (DDL) 支持。	运行 DDL 脚本以在 PostgreSQL 上创建触发器、序列、同义词以及引用键。 **Note** 您可以使用任何标准 SQL 客户端应用程序连接到数据库集群中的数据库。例如，您可使用 pgadmin 连接至您的数据库实例。	数据库管理员

相关的资源

- Amazon RDS for PostgreSQL（Amazon RDS 用户指南）
- 亚马逊 EC2 文档
- Oracle GoldenGate 支持的处理方法和数据库（Oracle 文档）

使用 AWS DMS 和 AWS SCT 将 Oracle 数据库迁移至 Amazon Redshift

由 Piyush Goyal (AWS) 和 Brian motzer (AWS) 创作

摘要

此模式为使用 AWS Database Migration Service (AWS DMS) 和 AWS Schema Conversion Tool (AWS SCT)将 Oracle 数据库迁移至 Amazon Web Services (AWS) Cloud 中的 Amazon Redshift 云数据仓库提供了指导。该模式涵盖本地或安装在亚马逊弹性计算云 (Amazon EC2) 实例上的源 Oracle 数据库。其亦涵盖了 Amazon Relational Database Service (Amazon RDS) for Oracle 数据库。

先决条件和限制

先决条件

- 在本地数据中心或 Amazon Web Services 中运行的 Oracle 数据库。
- 一个有效的 Amazon Web Services account
- 熟悉使用 Oracle 数据库作为 AWS DMS 的源
- 熟悉适用 Amazon Redshift 数据库作为 AWS DMS 目标
- 了解 Amazon RDS、Amazon Redshift、适用的数据库技术以及 SQL
- 适用于 AWS SCT 连接器的 Java 数据库（安装了 AWS SCT）连接（JDBC）驱动程序。

产品版本

- 对于自管理 Oracle 数据库，AWS DMS 支持 10.2 及更高版本（版本 10.x）、11g 直至 12.2、18c 以及 19c 版本的所有 Oracle 数据库版本。对于由 AWS 托管的 Amazon RDS for Oracle 数据库，AWS DMS 支持用于版本 11g（版本 11.2.0.4 及更高版本）直至 12.2、18c 以及 19c 版本的所有 Oracle 数据库版本。建议使用最新版本的 AWS DMS，以获得最全面的版本和功能支持。

架构

源技术堆栈

下列情况之一：

- 本地 Oracle 数据库
- EC2 实例上的 Oracle 数据库
- Amazon RDS for Oracle 数据库实例

目标技术堆栈

- Amazon Redshift

目标架构

从 Amazon Web Services Cloud 内运行的 Oracle 数据库至 Amazon Redshift：

从本地数据中心内运行的 Oracle 数据库至 Amazon Redshift：

工具

- AWS DMS - AWS Data Migration Service (AWS DMS) 可帮助您快速安全地将数据库迁移到 AWS。源数据库可在迁移过程中保持全面运行，从而最大程度地为依赖该数据库的应用程序减少停机时间。AWS DMS 可以在最广泛使用的商用和开源数据库之间迁移数据。
- AWS SCT – AWS Schema Conversion Tool (AWS SCT) 可用于将现有数据库架构从一个数据库引擎转换为另一个数据库引擎。其支持各种数据库引擎（包括 Oracle、SQL Server 和 PostgresSQL）作为源。

操作说明

准备迁移

Task	描述	所需技能
验证数据库版本。	验证源数据库和目标数据库版本并确保其受 AWS DMS 支持。有关支持的 Oracle 数据库版本的信息，请参阅使用 Oracle 数据库作为 AWS DMS 的源。有关使用 Amazon Redshift 作为目标的信息，请参阅使用 Amazon Redshift 数据库作为 AWS DMS 的目标。	数据库管理员

Task	描述	所需技能
创建 VPC 和安全组。	在 Amazon Web Services account 中，创建虚拟私有云（VPC）（如果没有虚拟私有云（VPC））。为源数据库与目标数据库的出站流量创建安全组。有关更多信息，请参阅 Amazon Virtual Private Cloud (Amazon VPC)文档。	系统管理员
安装 AWS SCT。	下载并安装最新版本的 AWS SCT 及相应的驱动程序。有关更多信息，请参阅安装、验证和更新 AWS SCT。	数据库管理员
为 AWS DMS 任务创建用户。	在源数据库中创建 AWS DMS 用户，并授予其读取权限。此用户将被 AWS SCT 和 AWS DMS 使用。	数据库管理员
测试数据库连接。	测试至 Oracle 数据库实例的连接。	数据库管理员
在 AWS SCT 中创建新项目。	打开 AWS SCT 工具并创建新项目。	数据库管理员
分析待迁移的 Oracle 架构。	使用 AWS SCT 分析待迁移架构，并生成数据库迁移评测报告。有关更多信息，请参阅 AWS SCT 文档中的创建数据库迁移评测报告。	数据库管理员
查看评测报告。	查看报告，以了解迁移的可行性。某些数据库对象可能需要手动转换。有关报告的更多信息，请参阅 AWS SCT 文档中的查看评测报告。	数据库管理员

准备目标数据库

Task	描述	所需技能
创建一个 Amazon Redshift 集群。	在您之前创建的 VPC 中创建 Amazon Redshift 集群。有关更多信息，请参阅 Amazon Redshift 文档中的 Amazon Redshift 集群。	数据库管理员
创建数据库用户。	从 Oracle 源数据库中提取用户、角色和权限列表。在目标 Amazon Redshift 数据库中创建用户，并应用上一步骤中的角色。	数据库管理员
评估数据库参数。	查看 Oracle 源数据库中的数据库选项、参数、网络文件和数据库链接，然后评估其对目标的适用性。	数据库管理员
将所有相关设置应用至目标。	有关该步骤的更多信息，请参阅 Amazon Redshift API 文档中的配置参考。	数据库管理员

在目标数据库中创建对象

Task	描述	所需技能
在目标数据库中创建 AWS DMS 用户。	在目标数据库中创建 AWS DMS 用户，并授予其读写权限。验证来自 AWS SCT 的连接。	数据库管理员
转换架构，查看 SQL 报告，并保存所有错误或警告。	有关更多信息，请参阅 AWS SCT 文档中的使用 AWS SCT 转换数据库架构。	数据库管理员

Task	描述	所需技能
将架构更改应用至目标数据库或将其另存为 .sql 文件。	有关说明，请参阅 AWS SCT 文档中的在 AWS SCT 中保存和应用转换后的架构。	数据库管理员
验证目标数据库中的对象。	在目标数据库中验证上一步骤创建的对象。重写或重新设计所有未成功转换的对象。	数据库管理员
禁用外键和触发器。	禁用任何外键和触发器。在运行 AWS DMS 时，这可能会导致完全加载过程中出现数据加载问题。	数据库管理员

使用 AWS DMS 迁移数据

Task	描述	所需技能
创建 AWS DMS 复制实例。	登录 AWS 管理控制台，并打开 AWS DMS 控制台。在导航窗格中，选择复制实例、创建复制实例。有关详细说明，请参阅 AWS DMS 文档中 AWS DMS 入门中的步骤 1。	数据库管理员
创建源和目标端点。	创建源端点和目标端点，测试从复制实例至源端点和目标端点的连接。有关详细说明，请参阅 AWS DMS 文档中 AWS DMS 入门中的步骤 2。	数据库管理员
创建复制任务。	创建复制任务，并选择适当的迁移方法。有关详细说明，请参阅 AWS DMS 文档中 AWS DMS 入门中的步骤 3。	数据库管理员

Task	描述	所需技能
启动数据复制。	启动复制任务并监控日志中是否存在错误。	数据库管理员

迁移应用程序

Task	描述	所需技能
创建应用程序服务器。	在 AWS 上创建新应用程序服务器。	应用程序所有者
迁移应用程序代码。	将应用程序代码迁移至新服务器。	应用程序所有者
配置应用程序服务器。	为目标数据库和驱动程序配置应用程序服务器。	应用程序所有者
优化应用程序代码。	优化目标引擎的应用程序代码。	应用程序所有者

割接至目标数据库

Task	描述	所需技能
验证用户。	在目标 Amazon Redshift 数据库中，验证用户并向其授予角色和权限。	数据库管理员
验证应用程序是否已锁定。	确保应用程序已锁定，以避免进一步更改。	应用程序所有者
验证数据。	验证目标 Amazon Redshift 数据库中的数据。	数据库管理员
启用外键与触发器。	在目标 Amazon Redshift 数据库中启用外键与触发器。	数据库管理员

Task	描述	所需技能
连接至该数据库。	将应用程序配置为连接新 Amazon Redshift 数据库。	应用程序所有者
执行最终检查。	在上线前执行最终全面系统检查。	数据库管理员，应用程序所有者
上线。	使用目标 Amazon Redshift 数据库上线。	数据库管理员

关闭迁移项目

Task	描述	所需技能
关闭临时 AWS 资源。	关闭临时的 AWS 资源，例如 AWS DMS 复制实例和用于 AWS SCT 的 EC2 实例。	数据库管理员、系统管理员
查看文件。	查看和验证迁移项目文档。	数据库管理员、系统管理员
收集指标。	收集有关迁移项目的信息，例如迁移时间、手动任务与工具任务百分比以及总节省成本。	数据库管理员、系统管理员
关闭项目。	关闭项目并提供反馈。	数据库管理员、系统管理员

相关资源

参考

- AWS DMS 用户指南
- AWS SCT 用户指南
- Amazon Redshift 入门指南

教程和视频

- 深入了解 AWS SCT 和 AWS DMS（来自 AWS re:Invent 2019 的演讲）
- AWS Database Migration Service 入门

使用 AWS DMS 和 AWS SCT 将 Oracle 数据库迁移至 Aurora PostgreSQL

由 Senthil Ramasamy (AWS) 创建

摘要

此模式描述了如何使用 AWS Data Migration Service (AWS DMS) 和 AWS Schema Conversion Tool (AWS SCT) 将 Oracle 数据库迁移至 Amazon Aurora PostgreSQL-Compatible Edition。

该模式涵盖本地的 Oracle 源数据库、安装在亚马逊弹性计算云 (Amazon EC2) 实例上的 Oracle 数据库以及适用于 Oracle 数据库的亚马逊关系数据库服务 (Amazon RDS)。此模式将这些数据库转换为 Aurora PostgreSQL-Compatible。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 在本地数据中心或 Amazon Web Services Cloud 中的 Oracle 数据库。
- 安装在本地计算机或 EC2 实例上的 SQL 客户端。
- AWS SCT 连接器的 Java 数据库连接 (JDBC) 驱动程序，安装在本地计算机或安装了 AWS SCT EC2 的实例上。

限制

- 数据库大小限制：128 TB
- 如果源数据库支持商用 off-the-shelf (COTS) 应用程序或特定于供应商，则可能无法将其转换为其他数据库引擎。在使用此模式前，请确认该应用程序支持 Aurora PostgreSQL-Compatible。

产品版本

- 对于自管理 Oracle 数据库，AWS DMS 支持 10.2 及更高版本（版本 10.x）、11g 直至 12.2、18c 以及 19c 版本的所有 Oracle 数据库版本。有关支持的 Oracle 数据库版本（包括自行管理版本和 Amazon RDS for Oracle）的最新列表，请参阅使用 Oracle 数据库作为 AWS DMS 的源和使用 PostgreSQL 数据库作为 AWS DMS 的目标。
- 建议使用最新版本的 AWS DMS，以获得最全面的版本和功能支持。关于 AWS SCT 支持的 Oracle 数据库版本的信息，请参阅 AWS SCT 文档。

- Aurora 支持 Amazon Aurora PostgreSQL 发行版本和引擎版本 中列出的 PostgreSQL 版本。

架构

源技术堆栈

下列情况之一：

- 本地 Oracle 数据库
- EC2 实例上的 Oracle 数据库
- Amazon RDS for Oracle 数据库实例

目标技术堆栈

- Aurora PostgreSQL-Compatible

目标架构

数据迁移架构

- 从 Amazon Web Services Cloud 内运行的 Oracle 数据库

- 从本地数据中心内运行的 Oracle 数据库

工具

- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到 Amazon Web Services Cloud，或者在云和本地设置的组合之间迁移。
- AWS Schema Conversion Tool（AWS SCT）通过以下方法来简化异构数据库的迁移工作：将源数据库架构和大部分的自定义代码自动转换成与目标数据库兼容的格式。

操作说明

准备迁移

Task	描述	所需技能
准备源数据库。	若要准备源数据库，请参阅 AWS SCT 文档中的使用 Oracle 数据库作为 AWS SCT 的源。	数据库管理员
为 AWS SCT 创建一个 EC2 实例。	如果需要，为 AWS SCT 创建和配置 EC2 实例。	数据库管理员
下载 AWS SCT。	下载最新版本的 AWS SCT 和相关驱动程序。有关更多信息，请参阅 AWS SCT 文档中的安装、验证和更新 AWS SCT。	数据库管理员
添加用户和权限。	在源数据库中添加并验证必备用户和权限。	数据库管理员
创建 AWS SCT 项目。	为工作负载创建 AWS SCT 项目，然后连接至源数据库。有关说明，请参阅 AWS SCT 文档中的创建 AWS SCT 项目和添加数据库服务器。	数据库管理员
评估可行性。	生成评测报告，其中汇总了无法自动转换架构的操作项目，并提供了手动转换的估算值。有关更多信息，请参阅 AWS SCT 文档中的创建和查看数据库迁移评测报告。	数据库管理员

准备目标数据库

Task	描述	所需技能
创建目标 Amazon RDS 数据库实例。	使用 Amazon Aurora 作为数据库引擎创建目标 Amazon RDS 数据库实例。有关说明，请参阅 Amazon RDS 文档中的创建 Amazon RDS 数据库实例。	数据库管理员
提取用户、角色与权限。	从源数据库中提取用户、角色和权限列表。	数据库管理员
映射用户。	将现有数据库用户映射到新的数据库用户。	应用程序所有者
创建用户。	在目标数据库中创建用户。	数据库管理员、应用程序所有者
应用角色。	将上一步的角色应用至目标数据库。	数据库管理员
检查选项、参数、网络文件和数据库链接。	查看源数据库选项、参数、网络文件和数据库链接，然后评估其对目标数据库的适用性。	数据库管理员
应用设置。	将所有相关设置应用至目标数据库。	数据库管理员

传输对象

Task	描述	所需技能
配置 AWS SCT 连接。	为目标数据库配置 AWS SCT 连接。	数据库管理员
使用 AWS SCT 转换架构。	AWS SCT 会自动将源数据库架构和大多数自定义代码转换	数据库管理员

Task	描述	所需技能
	为与目标数据库兼容的格式。该工具无法自动转换的任意代码会被清楚地标记，以便您进行手动转换。	
查看报告。	查看生成的 SQL 报告并保存所有错误和警告。	数据库管理员
应用自动架构更改。	将自动架构更改应用至目标数据库或将其另存为 .sql 文件。	数据库管理员
验证对象。	验证 AWS SCT 是否在目标创建了对象。	数据库管理员
处理未转换的对象。	手动重写、拒绝或重新设计任何无法自动转换的项目。	数据库管理员、应用程序所有者
应用角色与用户权限。	应用生成的角色和用户权限，并查看所有例外情况。	数据库管理员

迁移数据

Task	描述	所需技能
确定方法。	确定数据迁移方法。	数据库管理员
创建复制实例。	从 AWS DMS 控制台创建复制实例。有关更多信息，请参阅 AWS DMS 文档中的使用 AWS DMS 复制实例。	数据库管理员
创建源端点和目标端点。	若要创建端点，请按照在 AWS DMS 中创建源端点和目标端点中的说明操作。	数据库管理员

Task	描述	所需技能
创建复制任务。	若要创建任务，请参阅 AWS DMS 文档中的处理 AWS DMS 任务。	数据库管理员
启动复制任务并监控日志。	有关此步骤的更多信息，请参阅 AWS DMS 文档中的监控 AWS DMS 任务。	数据库管理员

迁移应用程序

Task	描述	所需技能
分析并转换应用程序代码中的 SQL 项目。	使用 AWS SCT 分析和转换应用程序代码中的 SQL 项。当您将数据库架构从一个引擎转换到另一个引擎时，还需要更新应用程序中的 SQL 代码，以便与新数据库引擎 (而非旧引擎) 进行交互。您可以查看、分析、编辑和保存转换后的 SQL 代码。	应用程序所有者
创建应用程序服务器。	在 AWS 上创建新应用程序服务器。	应用程序所有者
迁移应用程序代码。	将应用程序代码迁移至新服务器。	应用程序所有者
配置应用程序服务器。	为目标数据库和驱动程序配置应用程序服务器。	应用程序所有者
修复代码。	修复应用程序中所有特定的源数据库引擎代码。	应用程序所有者

Task	描述	所需技能
优化代码。	针对目标数据库引擎优化应用程序代码。	应用程序所有者

割接

Task	描述	所需技能
割接至目标数据库。	执行至新数据库的割接。	数据库管理员
锁定应用程序。	锁定应用程序，避免任何进一步的更改。	应用程序所有者
验证更改。	验证所有更改是否都已传播到目标数据库。	数据库管理员
重定向至目标数据库。	将新的应用程序服务器指向目标数据库。	应用程序所有者
检查所有内容。	执行最终全面系统检查。	应用程序所有者
上线。	完成最终割接任务。	应用程序所有者

关闭项目

Task	描述	所需技能
关闭临时资源。	关闭临时的 AWS 资源，例如 AWS DMS 复制实例和用于 AWS SCT 的 EC2 实例。	数据库管理员、应用程序所有者
更新反馈。	更新内部团队对于 AWS DMS 流程的反馈。	数据库管理员、应用程序所有者
修改过程与模板。	如有必要，请修改 AWS DMS 流程并改进模板。	数据库管理员、应用程序所有者

Task	描述	所需技能
验证文档。	查看和验证项目文档。	数据库管理员、应用程序所有者
收集指标。	收集指标以评估迁移时间、手动与工具成本节约比等。	数据库管理员、应用程序所有者
关闭项目。	关闭迁移项目并向利益相关者提供反馈。	数据库管理员、应用程序所有者

相关资源

参考

- 将 Oracle 数据库作为 AWS DMS 的源
- 将 PostgreSQL 数据库用作 AWS Database Migration Service 的目标
- 参照 PostgreSQL Compatibility (9.6.x) 迁移手册将 Oracle Database 11g/12c 迁移至 Amazon Aurora
- 参照 PostgreSQL Compatibility (12.4) 迁移手册将 Oracle Database 19c 迁移至 Amazon Aurora
- 将 Amazon RDS for Oracle 数据库迁移至 Amazon Aurora PostgreSQL-Compatible Edition
- AWS Data Migration Service
- AWS Schema Conversion Tool
- 从 Oracle 迁移到 Amazon Aurora
- Amazon RDS 定价

教程和视频

- 数据库迁移 Step-by-Step演练
- AWS DMS 入门
- Amazon RDS 入门
- AWS Data Migration Service（视频）
- 将 Oracle 数据库迁移至 PostgreSQL（视频）

其他信息

.

将数据从本地 Oracle 数据库迁移到 Aurora PostgreSQL

创建者：Michelle Deng (AWS) 和 Shunan Xiang (AWS)

摘要

此模式为数据从本地 Oracle 数据库迁移到 Amazon Aurora PostgreSQL-Compatible Edition 提供了指导。它的目标是为包含具有大量数据操作语言 (DML) 活动的大型表的多 TB Oracle 数据库提供一种在线数据迁移策略，最大限度地减少停机时间。使用 Oracle Active Data Guard 备用数据库作为从主数据库分流数据迁移的来源。在满载期间可以暂停从 Oracle 主数据库到备用数据库的复制，以避免出现 ORA-01555 错误。

在 Oracle 中，数据类型为 NUMBER 的主键 (PKsFKs) 或外键 () 中的表列通常用于存储整数。我们建议您在 PostgreSQL 中将它们转换为 INT 或 BIGINT，以获得更好的性能。您可使用 AWS Schema Conversion Tool (AWS SCT) 更改 PK 和 FK 列的默认数据类型映射。（有关更多信息，请参阅 AWS Blog 文章将数字数据类型从 Oracle 转换到 PostgreSQL。）这种示例中的数据迁移使用 AWS Database Migration Service (AWS DMS) 进行满载和更改数据捕获 (CDC)。

你也可以使用这种模式将本地 Oracle 数据库迁移到适用于 PostgreSQL 的亚马逊关系数据库服务 (Amazon RDS)，或者将托管在亚马逊弹性计算云 (亚马逊) 上的 Oracle 数据库迁移到适用于 PostgreSQL 的亚马逊 RDS 或 Aurora PostgreSQL 兼容 EC2 Aurora PostgreSQL。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 本地数据中心中的 Oracle 源数据库，配置了 Active Data Guard 备用数据库
- AWS Direct Connect 配置至 本地数据中心和 Amazon Web Services Cloud 之间
- 熟悉使用 Oracle 数据库作为 AWS DMS 的源数据库
- 熟悉使用 PostgreSQL 数据库作为 AWS DMS 的目标数据库

限制

- Amazon Aurora 数据库集群最多可创建 128 TiB 存储空间。Amazon RDS for PostgreSQL 数据库实例可使用高达 64 TiB 的存储空间创建。有关最新存储信息，请参阅 AWS 文档中的 Amazon Aurora 存储和可靠性以及 Amazon RDS 数据库实例存储。

产品版本

- AWS DMS 支持版本为 10.2 及更高版本（对于版本 10.x）、11g 直至 12.2、18c 以及 19c 的所有 Oracle 数据库版本。有关支持的版本的最新列表，请参阅 AWS 文档中的使用 Oracle 数据库作为 AWS DMS 的源数据库。

架构

源技术堆栈

- 配置了 Oracle Active Data Guard 备用数据库的本地 Oracle 数据库

目标技术堆栈

- Aurora PostgreSQL-Compatible

数据迁移架构

工具

- AWS DMS - AWS Database Migration Service (AWS DMS) 支持多种源数据库和目标数据库。有关支持的 Oracle 源数据库和目标数据库版本的列表，请参阅 AWS DMS 文档中的使用 Oracle 数据库作为 AWS DMS 的源数据库。如果 AWS DMS 不支持源数据库，则必须选择另一种方法来迁移第 6 阶段的数据（在 Epic s 部分）。 重要说明：由于这是异构迁移，因此必须先检查数据库是否支持商用 off-the-shelf (COTS) 应用程序。如果应用程序是 COTS，请咨询供应商以确认支持 Aurora PostgreSQL-Compatible，然后再继续。有关更多信息，请参阅 AWS 文档中的 AWS DMS Step-by-Step 迁移演练。
- AWS SCT - AWS Schema Conversion Tool (AWS SCT) 通过自动将源数据库架构和大部分自定义代码转换为与目标数据库兼容的格式，以促进异构数据库迁移。该工具转换的自定义代码包括视图、存储进程和函数。该工具无法自动转换的任意代码会被清楚地标记，以便您自己进行转换。

操作说明

计划迁移

Task	描述	所需技能
验证源数据库和目标数据库版本。		数据库管理员
安装 AWS SCT 和驱动程序。		数据库管理员
添加并验证 AWS SCT 必备用户和授权源数据库。		数据库管理员
为工作负载创建 AWS SCT 项目，然后连接至源数据库。		数据库管理员
生成评测报告并评测可行性。		数据库管理员、应用程序所有者

准备目标数据库

Task	描述	所需技能
创建 Aurora PostgreSQL-Compatible 目标数据库。		数据库管理员
从源数据库中提取用户、角色和权限列表。		数据库管理员
将现有数据库用户映射到新的数据库用户。		应用程序所有者
在目标数据库中创建用户。		数据库管理员
将上一步的角色应用至目标 Aurora PostgreSQL-Compatible 数据库。		数据库管理员

Task	描述	所需技能
查看源数据库中的数据库选项、参数、网络文件和数据库链接，然后评估其对目标数据库的适用性。		数据库管理员、应用程序所有者
将所有相关设置应用至目标数据库。		数据库管理员

准备数据库对象代码转换

Task	描述	所需技能
为目标数据库配置 AWS SCT 连接。		数据库管理员
在 AWS SCT 中转换架构，并将转换后的代码保存为 .sql 文件。		数据库管理员、应用程序所有者
手动转换无法自动转换的任何数据库对象。		数据库管理员、应用程序所有者
优化数据库代码转换。		数据库管理员、应用程序所有者
根据对象类型将该 .sql 文件分成多个 .sql 文件。		数据库管理员、应用程序所有者
验证目标数据库中的 SQL 脚本。		数据库管理员、应用程序所有者

准备数据迁移

Task	描述	所需技能
创建 AWS DMS 复制实例。		数据库管理员
创建源端点和目标端点。	如果 PKs 和的数据类型从 Oracle 中的 NUMBER 转换 FKs 为 PostgreSQL 中的 BIGINT，请考虑在创建源端点时指定连接`numberDataTypeScale=-2` 属性。	数据库管理员

迁移数据 - 满载

任务	描述	所需技能
在目标数据库中创建架构与表。		数据库管理员
通过对表进行分组或根据表大小拆分大表来创建 AWS DMS 满负荷任务。		数据库管理员
在短时间内停止源 Oracle 数据库的应用程序。		应用程序所有者
验证 Oracle 备库与主库是否同步，并停止从主库到备库的复制。		数据库管理员、应用程序所有者
在源 Oracle 数据库上启动应用程序。		应用程序所有者
从 Oracle 备用数据库到 Aurora PostgreSQL-Compatible 的数据库，并行启动 AWS DMS 满负荷任务。		数据库管理员

任务	描述	所需技能
在完全加载完成后创建 PKs 和二级索引。		数据库管理员
验证数据。		数据库管理员

迁移数据 – CDC

Task	描述	所需技能
在 Oracle 备用数据库与主数据库同步时，以及在上一个作业中应用程序重新启动之前，通过指定自定义 CDC 开始时间或系统更改编号 (SCN) 来创建 AWS DMS 持续复制任务。		数据库管理员
并行启动 AWS DMS 任务，将正在进行的更改从 Oracle 备用数据库复制到 Aurora PostgreSQL-Compatible 数据库。		数据库管理员
重新建立从 Oracle 主数据库到备用数据库复制。		数据库管理员
当 Aurora PostgreSQL-Compatible 目标数据库与源 Oracle 数据库几乎同步时，监控日志并停止 Oracle 数据库上的应用程序。		数据库管理员、应用程序所有者
当目标与源 Oracle 数据库完全同步时，停止 AWS DMS 任务。		数据库管理员

Task	描述	所需技能
在目标数据库中创建 FKs 和校验数据。		数据库管理员
在目标数据库中创建函数、视图、触发器、序列以及其他对象类型。		数据库管理员
在目标数据库应用角色授权。		数据库管理员

迁移应用程序

Task	描述	所需技能
使用 AWS SCT 分析并转换应用程序代码中的 SQL 语句。		应用程序所有者
在 AWS 上创建新应用程序服务器。		应用程序所有者
将应用程序代码迁移至新服务器。		应用程序所有者
为目标数据库和驱动程序配置应用程序服务器。		应用程序所有者
修复应用程序中特定于源数据库引擎的任何代码。		应用程序所有者
针对目标数据库优化应用程序代码。		应用程序所有者

割接

Task	描述	所需技能
将新应用程序服务器指向目标数据库。		数据库管理员、应用程序所有者
执行健全性检查。		数据库管理员、应用程序所有者
上线。		数据库管理员、应用程序所有者

关闭项目

Task	描述	所需技能
关闭临时 AWS 资源。		数据库管理员、系统管理员
查看和验证项目文档。		数据库管理员、应用程序所有者
收集有关迁移时间、手动与工具使用的百分比、成本节约和类似数据的指标。		数据库管理员、应用程序所有者
关闭项目并提供反馈。		数据库管理员、应用程序所有者

相关资源

参考

- Oracle 数据库至 Aurora PostgreSQL-Compatible：迁移行动手册
- 将 Amazon RDS for Oracle 数据库迁移到 Amazon Aurora MySQL
- AWS DMS 网站
- AWS DMS 文档

- AWS SCT 网站
- AWS SCT 文档
- 从 Oracle 迁移到 Amazon Aurora

教程

- AWS DMS 入门
- Amazon RDS 入门
- AWS Database Migration Service Step-by-Step 演练

使用 AWS DMS 从 SAP ASE 迁移至 Amazon RDS for SQL Server

由 Amit Kumar (AWS) 编写

摘要

此模式提供有关将 SAP Adaptive Server Enterprise (ASE) 数据库迁移到运行 Microsoft SQL Server 的 Amazon Relational Database Service (Amazon RDS) 数据库实例的指导。源数据库可以位于本地数据中心或亚马逊弹性计算云 (Amazon EC2) 实例上。该模式使用 AWS Database Migration Service (AWS DMS) 迁移数据，并使用（可选）计算机辅助软件工程 (CASE) 工具转换数据库架构。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 本地数据中心或 EC2 实例上的 SAP ASE 数据库
- 已启动和运行的目标 Amazon RDS for SQL Server 数据库

限制

- 数据库大小限制：64 TB

产品版本

- 仅限 SAP ASE 15.7 或 16.x 版。有关最新信息，请参阅使用 SAP 数据库作为 AWS DMS 源。
- 对于Amazon RDS 目标数据库，AWS DMS 支持 Amazon RDS 上的 Microsoft SQL Server 版本，适用于 Enterprise、Standard、Web 和 Express 版本。有关支持版本的最新列表，请参阅 AWS DMS 文档。建议使用最新版本的 AWS DMS，以获得最全面的版本和功能支持。

架构

源技术堆栈

- 本地或 Amazon EC2 实例上的 SAP ASE 数据库

目标技术堆栈

- Amazon RDS for SQL Server 数据库实例

源架构和目标架构

从亚马逊上的 SAP ASE 数据库 EC2 到 Amazon RDS for SQL Server 数据库实例：

从本地 SAP ASE 数据库至 Amazon RDS for SQL Server 数据库实例：

工具

- AWS Datab ase Migration Service (AWS DMS) 是一项网络服务，可用于将数据从本地、Amazon RDS 数据库实例或实例上的数据库迁移到 AWS 服务上的数据库， EC2 例如 Amazon RDS for SQL Server 或 EC2 实例。您还可以将数据库从 Amazon Web Services 迁移到本地数据库。您可以在异构或同构数据库引擎间迁移数据。
- 对于架构转换，你可以选择使用 erwin Data Modeler 或 SAP。 PowerDesigner

操作说明

计划迁移

Task	描述	所需技能
验证源数据库和目标数据库版本。		数据库管理员
确定存储需求（存储类型和容量）。		数据库管理员， SysAdmin
根据容量、存储功能和网络功能选择正确的实例类型。		数据库管理员， SysAdmin
确定源数据库和目标数据库的网络访问安全要求。		数据库管理员， SysAdmin
确定应用程序迁移策略。		DBA、 SysAdmin、应用程序所有者

配置基础设施

Task	描述	所需技能
创建虚拟私有云（VPC）和子网。		SysAdmin
创建安全组和网络访问控制列表 (ACLs)。		SysAdmin
配置和启动运行 Amazon RDS 数据库实例。		SysAdmin

迁移数据 - 选项 1

Task	描述	所需技能
手动迁移数据库架构或使用 CASE 工具，例如 erwin Data Modeler 或 SAP。 PowerDesigner		数据库管理员

迁移数据 - 选项 2

Task	描述	所需技能
使用 AWS DMS 迁移数据。		数据库管理员

迁移应用程序

Task	描述	所需技能
遵循应用程序迁移策略。		DBA、 SysAdmin、应用程序所有者

割接

Task	描述	所需技能
将应用程序客户端切换至新基础设施。		DBA、 SysAdmin、应用程序所有者

关闭项目

Task	描述	所需技能
关闭临时 AWS 资源。		数据库管理员 ， SysAdmin
查看和验证项目文档。		DBA、 SysAdmin、应用程序所有者
收集关于迁移时间、手动任务与自动任务的百分比以及成本节省等指标。		DBA、 SysAdmin、应用程序所有者
关闭项目并提供反馈。		DBA、 SysAdmin、应用程序所有者

相关资源

参考

- AWS DMS 网站
- Amazon RDS 定价
- 使用 SAP ASE 数据库作为 AWS DMS 源
- RDS Custom for SQL Server 的限制

教程和视频

- AWS DMS 入门
- Amazon RDS 入门

- AWS DMS（视频）
- Amazon RDS（视频）

使用 AWS DMS 将本地 Microsoft SQL Server 数据库迁移至 Amazon Redshift

由 Marcelo Fernandes (AWS) 编写

摘要

此模式提供有关使用 AWS Data Migration Service (AWS DMS) 将本地 Microsoft SQL Server 数据库迁移到 Amazon Redshift 的指导。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 本地数据中心的源 Microsoft SQL Server 数据库
- 已完成使用 Amazon Redshift 数据库作为 AWS DMS 目标的先决条件，如 AWS DMS 文档中所述

产品版本

- SQL Server 2005-2019、Enterprise、Standard、Workgroup、Developer 和 Web 版本。有关支持的版本的最新列表，请参阅 AWS 文档中的使用 Microsoft SQL Server 数据库作为 AWS DMS 的源数据库。

架构

源技术堆栈

- 本地 Microsoft SQL Server 数据库

目标技术堆栈

- Amazon Redshift

数据迁移架构

工具

- AWS DMS 是数据迁移服务，支持多种不同的源数据库和目标数据库。有关支持与 AWS DMS 一起使用的 Microsoft SQL Server 数据库版本和版本的信息，请参阅 AWS DMS 文档中使用 Microsoft SQL Server 数据库作为 AWS DMS 的源。如果 AWS DMS 不支持源数据库，则必须选择另一种方法进行数据迁移。

操作说明

计划迁移

Task	描述	所需技能
验证源数据库和目标数据库版本和引擎。		数据库管理员
确定目标服务器实例的硬件要求。		数据库管理员、系统管理员
确定存储需求（存储类型和容量）。		数据库管理员、系统管理员
根据容量、存储功能和网络功能选择正确的实例类型。		数据库管理员、系统管理员
确定源数据库和目标数据库的网络访问安全要求。		数据库管理员、系统管理员
确定应用程序迁移策略。		数据库管理员、应用程序所有者、系统管理员

配置基础设施

Task	描述	所需技能
创建虚拟私有云（VPC）。	有关更多信息，请参阅 AWS 文档中的在 VPC 中使用数据库实例。	系统管理员

Task	描述	所需技能
创建安全组。		系统管理员
配置和启动 Amazon Redshift 集群。	有关更多信息，请参阅 Amazon Redshift 文档中的 创建示例 Amazon Redshift 集群。	数据库管理员、系统管理员

迁移数据

Task	描述	所需技能
使用 AWS DMS 迁移来自 Microsoft SQL Server 数据库的数据。		数据库管理员

迁移应用程序

Task	描述	所需技能
遵循应用程序迁移策略。		数据库管理员、应用程序所有者、系统管理员

割接

Task	描述	所需技能
将应用程序客户端切换至新基础设施。		数据库管理员、应用程序所有者、系统管理员

关闭项目

Task	描述	所需技能
关闭临时资源。		数据库管理员、系统管理员
查看和验证项目文档。		数据库管理员、应用程序所有者、系统管理员
收集关于迁移时间、手动任务与自动任务的百分比以及成本节省等指标。		数据库管理员、应用程序所有者、系统管理员
关闭项目并提供反馈。		数据库管理员、应用程序所有者、系统管理员

相关资源

参考

- AWS DMS 文档
- Amazon Redshift 文档
- Amazon Redshift 定价

教程和视频

- AWS DMS 入门
- Amazon Redshift 入门
- 将 Amazon Redshift 数据库作为 AWS Database Migration Service 目标
- AWS DMS（视频）

使用 AWS SCT 数据提取代理将本地 Microsoft SQL Server 数据库迁移至 Amazon Redshift

由 Neha Thakur(AWS) 编写

摘要

此模式概述了使用 AWS Schema Conversion Tool (AWS SCT) 数据提取代理将本地 Microsoft SQL Server 源数据库迁移到 Amazon Redshift 目标数据库的步骤。代理是外部程序，它与 AWS SCT 集成，但在其他地方执行数据转换，并代表您与其他 Amazon Web Services 交互。

先决条件和限制

先决条件

- 用于本地数据中心数据仓库工作负载的 Microsoft SQL Server 源数据库
- 一个有效的 Amazon Web Services account

产品版本

- Microsoft SQL Server 版本 2008 或更高版本。有关受支持版本的列表，请参阅 AWS SCT 文档。

架构

技术堆栈源

- 本地 Microsoft SQL Server 数据库

技术堆栈目标

- Amazon Redshift

数据迁移架构

工具

- AWS Schema Conversion Tool (AWS SCT) 通过以下方法来处理异构数据库的迁移工作：将源数据库架构和大部分的自定义代码自动转换成与目标数据库兼容的格式。当源数据库和目标数据库非常不

同时，您可以使用 AWS SCT 代理执行其他数据转换。有关更多信息，请参阅 AWS 文档中的将数据从本地数据仓库迁移至 Amazon Redshift。

最佳实践

- AWS SCT 最佳实践
- Amazon Redshift 最佳实践

操作说明

准备迁移

Task	描述	所需技能
验证源数据库和目标数据库的版本和引擎。		数据库管理员
确定目标服务器实例的硬件要求。		数据库管理员，SysAdmin
识别存储需求（存储类型和容量）。		数据库管理员，SysAdmin
选择适当的实例类型（容量、存储功能、网络功能）。		数据库管理员，SysAdmin
确定源数据库和目标数据库的网络访问安全要求。		数据库管理员，SysAdmin
选择应用程序迁移策略。		DBA、SysAdmin、应用程序所有者

配置基础设施

Task	描述	所需技能
创建虚拟私有云（VPC）和子网。		SysAdmin
创建安全组。		SysAdmin
配置并启动 Amazon Redshift 集群。		SysAdmin

迁移数据

Task	描述	所需技能
使用 AWS SCT 数据提取代理迁移数据。		数据库管理员

迁移应用程序

Task	描述	所需技能
遵循选定的应用程序迁移策略。		DBA、 SysAdmin、应用程序所有者

割接至目标数据库

Task	描述	所需技能
将应用程序客户端切换至新基础设施。		DBA、 SysAdmin、应用程序所有者

关闭项目

Task	描述	所需技能
关闭临时 AWS 资源。		数据库管理员 ， SysAdmin
查看和验证项目文档。		DBA、 SysAdmin、应用程序所有者
收集关于迁移时间、手动任务与自动任务的百分比以及成本节省等指标。		DBA、 SysAdmin、应用程序所有者
关闭项目并提供任何反馈。		DBA、 SysAdmin、应用程序所有者

相关资源

参考

- AWS SCT 用户指南
- 使用数据提取代理
- Amazon Redshift 定价

教程和视频

- AWS Schema Conversion Tool 入门
- Amazon Redshift 入门

使用 AWS SCT 数据提取代理将 Teradata 数据库迁移到 Amazon Redshift

由 Sergey Dmitriev (AWS) 编写

摘要

此模式将指导您完成将 Teradata 数据库（用作本地数据中心的数据仓库）迁移到 Amazon Redshift 数据库的步骤。该模式使用 AWS Schema Conversion Tool（AWS SCT）数据提取代理。代理是外部程序，它与 AWS SCT 集成，但在其他地方执行数据转换，并代表您与其他 Amazon Web Services 交互。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 本地数据中心中的 Teradata 源数据库

产品版本

- Teradata 版本 13 和更高版本。有关支持版本的最新列表，请参阅 AWS SCT 文档。

架构

源技术堆栈

- 本地 Teradata 数据库

目标技术堆栈

- Amazon Redshift 集群

数据迁移架构

工具

- AWS SCT – AWS Schema Conversion Tool (AWS SCT) 通过自动将源数据库架构和大部分自定义代码转换为与目标数据库兼容的格式来处理异构数据库迁移。当源数据库和目标数据库彼此差异很大

时，您可以使用 AWS SCT 代理执行额外的数据转换。有关更多信息，请参阅 AWS 文档中的将数据从本地数据仓库迁移至 Amazon Redshift。

操作说明

准备迁移

Task	描述	所需技能
验证源数据库和目标数据库的版本和引擎。		数据库管理员
确定目标服务器实例的硬件要求。		数据库管理员，SysAdmin
识别存储需求（存储类型和容量）。		数据库管理员，SysAdmin
选择适当的实例类型（容量、存储功能、网络功能）。		数据库管理员，SysAdmin
确定源数据库和目标数据库的网络访问安全要求。		数据库管理员，SysAdmin
选择应用程序迁移策略。		DBA、SysAdmin、应用程序所有者

配置基础设施

Task	描述	所需技能
创建虚拟私有云（VPC）和子网。		SysAdmin
创建安全组。		SysAdmin
配置并启动 Amazon Redshift 集群。		SysAdmin

迁移数据

Task	描述	所需技能
使用 AWS SCT 数据提取代理迁移数据。	有关使用 AWS SCT 数据提取代理的详细信息，请参阅参考和帮助部分中的链接。	数据库管理员

迁移应用程序

Task	描述	所需技能
遵循选定的应用程序迁移策略。		DBA、 SysAdmin、应用程序所有者

割接到目标 Amazon Redshift 数据库

Task	描述	所需技能
将应用程序客户端切换至新基础设施。		DBA、 SysAdmin、应用程序所有者

关闭项目

Task	描述	所需技能
关闭临时 AWS 资源。		数据库管理员 ， SysAdmin
查看和验证项目文档。		DBA、 SysAdmin、应用程序所有者
收集与迁移时间、手动任务与工具任务的百分比、成本节约等相关的指标。		DBA、 SysAdmin、应用程序所有者
关闭项目并提供任何反馈。		

相关资源

参考

- AWS SCT 用户指南
- 使用数据提取代理
- Amazon Redshift 定价
- 将 Teradata RESET WHEN 功能转换为 Amazon Redshift SQL (AWS Prescriptive Guidance)
- 将 Teradata NORMALIZE 临时功能转换为 Amazon Redshift SQL（AWS Prescriptive Guidance）

教程

- AWS Schema Conversion Tool 入门
- Amazon Redshift 入门

使用 AWS SCT 数据提取代理将本地 Vertica 数据库迁移至 Amazon Redshift

由 Sergey Dmitriev (AWS) 编写

摘要

此模式为使用 AWS Schema Conversion Tool (AWS SCT) 数据提取代理将本地 Vertica 数据库迁移至 Amazon Redshift 集群提供了指导。代理是外部程序，它与 AWS SCT 集成，但在其他地方执行数据转换，并代表您与其他 Amazon Web Services 交互。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 用于本地数据中心数据仓库工作负载的 Vertica 源数据库
- Amazon Redshift 目标集群

产品版本

- Vertica 版本 7.2.2 及更高版本。有关支持版本的最新列表，请参阅 AWS SCT 文档。

架构

源技术堆栈

- 本地 Vertica 数据库

目标技术堆栈

- Amazon Redshift 集群

数据迁移架构

工具

- AWS Schema Conversion Tool (AWS SCT) 通过以下方法来处理异构数据库的迁移工作：将源数据库架构和大部分的自定义代码自动转换成与目标数据库兼容的格式。当源数据库和目标数据库彼此

差异很大时，您可以使用 AWS SCT 代理执行额外的数据转换。有关更多信息，请参阅 AWS 文档中的将数据从本地数据仓库迁移至 Amazon Redshift。

操作说明

准备迁移

Task	描述	所需技能
验证源数据库和目标数据库版本。		数据库管理员
识别存储需求（存储类型和容量）。		数据库管理员，SysAdmin
选择适当的实例类型（容量、存储功能、网络功能）。		数据库管理员，SysAdmin
确定源数据库和目标数据库的网络访问安全要求。		数据库管理员，SysAdmin
选择应用程序迁移策略。		DBA、SysAdmin、应用程序所有者

配置基础设施

Task	描述	所需技能
创建虚拟私有云（VPC）和子网。		SysAdmin
创建安全组。		SysAdmin
配置和启动 Amazon Redshift 集群。		SysAdmin

迁移数据

Task	描述	所需技能
使用 AWS SCT 数据提取代理迁移数据。	有关使用 AWS SCT 数据提取代理的详细信息，请参阅“参考和帮助”部分中的链接。	数据库管理员

迁移应用程序

Task	描述	所需技能
遵循选定的应用程序迁移策略。		DBA、 SysAdmin、应用程序所有者

割接至目标数据库

Task	描述	所需技能
将应用程序客户端切换至新基础设施。		DBA、 SysAdmin、应用程序所有者

关闭项目

Task	描述	所需技能
关闭临时 AWS 资源。		数据库管理员 ， SysAdmin
查看和验证项目文档。		DBA、 SysAdmin、应用程序所有者
收集与迁移时间、手动任务与工具任务的百分比、成本节约等相关的指标。		DBA、 SysAdmin、应用程序所有者
关闭项目并提供任何反馈。		

相关资源

参考

- AWS SCT 用户指南
- 使用数据提取代理
- Amazon Redshift 定价

教程和视频

- AWS Schema Conversion Tool 入门
- Amazon Redshift 入门

将遗留应用程序从 Oracle Pro*C 迁移到 ECPG

创建者：Sai Parthasaradhi (AWS) 和 Mahesh Balumuri (AWS)

摘要

大多数具有嵌入式 SQL 代码的遗留应用程序使用 Oracle Pro*C 预编译器来访问数据库。当您将这些 Oracle 数据库迁移到 Amazon Relational Database Service (Amazon RDS) for PostgreSQL 或 Amazon Aurora PostgreSQL-Compatible Edition 时，您必须将应用程序代码转换为与 PostgreSQL 中的预编译器兼容的格式（称为 ECPG）。此模式描述了如何将 Oracle Pro*C 代码转换至 PostgreSQL ECPG 中的等效代码。

有关 Pro*C 的更多信息，请参阅 Oracle 文档。有关 ECPG 的简要介绍，请参阅其他信息部分。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- Amazon RDS for PostgreSQL 或 Aurora PostgreSQL-Compatible 数据库
- 本地运行的 Oracle 数据库

工具

- 下一部分中列出的 PostgreSQL 程序包。
- AWS CLI – AWS 命令行界面（AWS CLI）是一种开源工具，用于通过命令行 Shell 中的命令与 Amazon Web Services 交互。仅需最少的配置，即可使用 AWS CLI 开始运行命令，以便从终端程序中的命令提示符实现与基于浏览器的 AWS 管理控制台所提供的功能等同的功能。

操作说明

在 CentOS 或 RHEL 设置构建环境

Task	描述	所需技能
安装 PostgreSQL 程序包。	通过使用以下命令安装所需 PostgreSQL 程序包。 `yum update -y`	应用程序开发者、 DevOps 工程师

Task	描述	所需技能

	`yum install -y yum-utils`
	`rpm -ivh https://download.postgresql.org/pub/repos/yum/reporpms/EL-8-x86_64/pgdg-redhat-repo-latest.noarch.rpm`
	`dnf -qy module disable postgresql`	

安装标头文件和库。	使用以下命令安装包含头文件和库的 `postgresql12-devel` 程序包。在开发环境和运行时系统环境中都安装该包，以避免运行时系统环境中出现错误。
	`dnf -y install postgresql12-devel`
	`yum install ncompress zip ghostscript jq unzip wget git -y`
	仅对于开发环境，还运行以下命令。
	`yum install zlib-devel make -y`
	`ln -s /usr/pgsql-12/bin/ecpg /usr/bin/`
		应用程序开发者、 DevOps 工程师

配置环境路径变量。	为 PostgreSQL 客户端库设置环境路径。 `export PATH=$PATH:/usr/pgsql-12/bin`	应用程序开发者、 DevOps 工程师

Task	描述	所需技能

必要时安装其他软件。	如果需要，可以在 Oracle 中安装 pgLoader 以替换 SQL*Loader。
	
	`wget -O /etc/yum.repos.d/pgloader-ccl.repo https://dl.packager.io/srv/opf/pgloader-ccl/master/installer/el/7.repo`
	`yum install pgloader-ccl -y`
	`ln -s /opt/pgloader-ccl/bin/pgloader /usr/bin/`
	
	如果您要从 Pro*C 模块调用任何 Java 应用程序，请安装 Java。
	
	`yum install java -y`
	
	安装 ant 以编译 Java 代码。
	
	`yum install ant -y`
		应用程序开发者、 DevOps 工程师

Task	描述	所需技能

安装 Amazon CLI。	安装 AWS CLI 以运行命令，从您的应用程序与 Amazon Web Services （例如 AWS Secrets Manager 和 Amazon Simple Storage Service (Amazon S3) ）交互。
	`cd /tmp/`
	`curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"`
	`unzip awscliv2.zip`
	`./aws/install -i /usr/local/aws-cli -b /usr/local/bin --update`
		应用程序开发者、 DevOps 工程师

确定要转换的程序。	确定要从 Pro*C 转换为 ECPG 的应用程序。	应用程序开发人员、应用程序所有者

将 Pro*C 代码转换为 ECPG

Task	描述	所需技能
删除不需要的标头。	删除 PostgreSQL 中不需要的 `include` 标头，例如 `oci.h`、`oratypes` 和 `sqlda`。	应用程序所有者、应用程序开发人员
更新变量声明。	为用作主机变量的所有变量声明添加 `EXEC SQL` 语句。 从您的应用程序中删除如下 `EXEC SQL VAR` 声明。	应用程序开发人员、应用程序所有者

Task	描述	所需技能
	`EXEC SQL VAR query IS STRING(2048);`	

Task	描述	所需技能

更新 ROWNUM 功能。	该 ROWNUM 函数在 PostgreSQL 中不可用。将其替换为 SQL 查询中的 ROW_NUMBER 窗口函数。
	
	Pro*C 代码：
	
	`SELECT SUBSTR(RTRIM(FILE_NAME,'.txt'),12) INTO :gcpclFileseq FROM (SELECT FILE_NAME FROM DEMO_FILES_TABLE WHERE FILE_NAME LIKE '%POC%' ORDER BY FILE_NAME DESC) FL2 WHERE ROWNUM <=1 ORDER BY ROWNUM;`
	
	ECPG 代码：
	
	`SELECT SUBSTR(RTRIM(FILE_NAME,'.txt'),12) INTO :gcpclFileseq FROM (SELECT FILE_NAME , ROW_NUMBER() OVER (ORDER BY FILE_NAME DESC) AS ROWNUM FROM demo_schema.DEMO_FILES_TABLE WHERE FILE_NAME LIKE '%POC%' ORDER BY FILE_NAME DESC) FL2`
		应用程序开发人员、应用程序所有者

Task	描述	所需技能
	`WHERE ROWNUM <=1 ORDER BY ROWNUM;`	

更新函数参数，以使用别名变量。	在 PostgreSQL 中，函数参数不能用作主机变量。使用别名变量覆盖它们。
	
	Pro*C 代码：
	
	`int processData(int referenceId){ EXEC SQL char col_val[100]; EXEC SQL select column_name INTO :col_val from table_name where col=:referenceId; }`
	
	ECPG 代码：
	
	`int processData(int referenceIdParam){ EXEC SQL int referenceId = referenceIdParam; EXEC SQL char col_val[100]; EXEC SQL select column_name INTO :col_val from table_name where col=:referenceId; }`
		应用程序开发人员、应用程序所有者

Task	描述	所需技能

更新结构类型。	如果将 `struct` 类型变量用作主机变量，则通过 `typedef` 在 `EXEC SQL BEGIN` 和 `END` 块中定义 `struct` 类型。如果 `struct` 类型是在标头 (`.h`) 文件中定义的，则使用 `EXEC SQL` 包含语句纳入这些文件。
	
	Pro*C 代码：
	
	标头文件 (`demo.h`)
	
	`struct s_partition_ranges { char sc_table_group[31]; char sc_table_name[31]; char sc_range_value[10]; }; struct s_partition_ranges_ind { short ss_table_group; short ss_table_name; short ss_range_value; };`
	
	ECPG 代码：
	
	标头文件 (`demo.h`)
	
	`EXEC SQL BEGIN DECLARE SECTION;`
		应用程序开发人员、应用程序所有者

Task	描述	所需技能

	typedef struct
	{
	  char sc_table_
	group[31];
	  char sc_table_
	name[31];
	  char sc_range_
	value[10];
	} s_partition_ranges;
	typedef struct
	{
	  short ss_table_
	group;
	  short ss_table_
	name;
	  short ss_range_
	value;
	} s_partition_ranges
	_ind;
	EXEC SQL END DECLARE
	 SECTION;
	
	Pro*C 文件 (demo.pc)
	
	#include "demo.h"
	struct s_partiti
	on_ranges gc_partit
	ion_data[MAX_PART_
	TABLE] ;
	struct s_partiti
	on_ranges_ind
	 gc_partition_data_
	ind[MAX_PART_TABLE] ;
	
	ECPG 文件 (demo.pc)
	
	exec sql include
	 "demo.h"
	EXEC SQL BEGIN DECLARE
	 SECTION;	

Task	描述	所需技能
	`s_partition_ranges gc_partition_data[ MAX_PART_TABLE] ; s_partition_ranges_ind gc_partition_data_ ind[MAX_PART_TABLE] ; EXEC SQL END DECLARE SECTION;`	

修改逻辑以从游标中获取。	要使用数组变量从游标中提取多行，请更改代码以使用 FETCH FORWARD。
	
	Pro*C 代码：
	
	`EXEC SQL char aPoeFiles [MAX_FILES][FILENA ME_LENGTH]; EXEC SQL FETCH filename_ cursor into :aPoeFile s;`
	
	ECPG 代码：
	
	`EXEC SQL char aPoeFiles [MAX_FILES][FILENA ME_LENGTH]; EXEC SQL int fetchSize = MAX_FILES; EXEC SQL FETCH FORWARD :fetchSiz e filename_cursor into :aPoeFiles;`
		应用程序开发人员、应用程序所有者

Task	描述	所需技能

修改没有返回值的数据包调用。	没有返回值的 Oracle 包函数应该使用指示变量来调用。如果您的应用程序包含多个具有相同名称的函数，或者未知类型的函数生成运行时系统错误，请将值强制转换为数据类型。
	
	Pro*C 代码：
	
	`void ProcessData (char *data , int id) { EXEC SQL EXECUTE BEGIN pkg_demo. process_data (:data, :id); END; END-EXEC; }`
	
	ECPG 代码：
	
	`void ProcessData (char *dataParam, int idParam ) { EXEC SQL char *data = dataParam; EXEC SQL int id = idParam; EXEC SQL short rowInd; EXEC SQL short rowInd = 0;`
		应用程序开发人员、应用程序所有者

Task	描述	所需技能
	EXEC SQL SELECT pkg_demo.process_data (inp_data => :data::text, inp_id => :id) INTO :rowInd; }	

Task	描述	所需技能

重写 SQL_CURSOR 变量。	重写 SQL_CURSOR 变量及其实现。
	
	Pro*C 代码：
	
	`/* SQL Cursor */`
	`SQL_CU`
	`RSOR demo_cursor;`
	`EXEC SQL`
	`ALLOCATE :demo_cursor;`
	`EXEC SQL EXECUTE`
	`BEGIN`
	`pkg_demo.`
	`get_cursor(`
	`demo_cur=`
	`>:demo_cursor`
	`);`
	`END;`
	`END-EXEC;`
	
	ECPG 代码：
	
	`EXEC SQL DECLARE`
	`demo_cursor CURSOR FOR`
	`SELECT`
	`* from`
	`pkg_demo.open_file`
	`name_rc(`
	`demo_cur=`
	`>refcursor`
	`) ;`
	`EXEC SQL char open_file`
	`name_rcInd[100];`
	`# As the below function`
	`returns cursor_name`
	`as`
	`# return we need to`
	`use char[] type as`
	`indicator.`
		应用程序开发人员、应用程序所有者

Task	描述	所需技能
	`EXEC SQL SELECT pkg_demo.get_cursor ( demo_cur=>'demo_cursor' ) INTO :open_filename_rcInd;`	
应用常见迁移模式。	• 更改 SQL 查询，以使其与 PostgreSQL 兼容。 • 如果 ECPG 不支持匿名方块，则请将其移至数据库。 • 删除 `dbms_application_info` 逻辑，PostgreSQL 不支持这种逻辑。 • 在光标关闭后移动 `EXEC SQL COMMIT` 语句。如果您在循环中提交查询以从游标中获取记录，游标将关闭并显示游标不存在错误。 • 有关处理 ECPG 中的异常和错误代码的信息，请参阅 PostgreSQL 文档中的错误处理。	应用程序开发人员、应用程序所有者
如果需要，启用调试。	要在调试模式下运行 ECPG 程序，请在主函数块中添加以下命令。 `ECPGdebug(1, stderr);`	应用程序开发人员、应用程序所有者

编译 ECPG 程序

Task	描述	所需技能

为 ECPG 创建可执行文件。	如果您有一个名为 `prog1.pgc` 的嵌入式 SQL C 源文件，则可以使用以下命令序列创建可执行程序。
	`ecpg prog1.pgc`
	`cc -I/usr/local/pgsql/include -c prog1.c`
	`cc -o prog1 prog1.o -L/usr/local/pgsql/lib -lecpg`
		应用程序开发人员、应用程序所有者

创建一个 make 文件用于编译。	创建一个 make 文件以编译 ECPG 程序，如以下示例文件中所示。
	`CFLAGS ::= $(CFLAGS) -I/usr/pgsql-12/include -g -Wall`
	`LDFLAGS ::= $(LDFLAGS) -L/usr/pgsql-12/lib -Wl,-rpath,/usr/pgsql-12/lib`
	`LDLIBS ::= $(LDLIBS) -lecpg`
	`PROGRAMS = test`
	`.PHONY: all clean`
	`%.c: %.pgc`
	`  ecpg $<`
	`all: $(PROGRAMS)`
	`clean:`
	`  rm -f $(PROGRAMS) $(PROGRAMS:%=%.c) $(PROGRAMS:%=%.o)`
		应用程序开发人员、应用程序所有者

测试应用程序

Task	描述	所需技能
测试代码。	测试转换后的应用程序代码，以确保其正常运行。	应用程序开发人员、应用程序所有者、测试工程师

相关资源

- ECPG-C 语言中的嵌入式 SQL（PostgreSQL 文档）
- 错误处理（PostgreSQL 文档）
- 为什么要使用 Oracle Pro*C/C++ 预编译器（Oracle 文档）

其他信息

PostgreSQL 有一个嵌入式 SQL 预编译器 ECPG，相当于 Oracle Pro*C 预编译器。ECPG 通过用特殊函数调用替换 SQL 调用，将嵌入 SQL 语句的 C 程序转换为标准 C 代码。然后可以使用任何 C 编译器工具链处理输出文件。

输入和输出文件

ECPG 将您在命令行上指定的每个输入文件转换为相应的 C 输出文件。如果输入文件名没有文件扩展名，则假定为 .pgc。将文件扩展名替换为 `.c`，以构造输出文件名。但是，可以使用 `-o` 选项覆盖默认输出文件名。

如果您使用破折号 (`-`) 作为输入文件名，ECPG 会从标准输入中读取程序并写入标准输出，除非您使用该 `-o` 选项将其覆盖。

标头文件

当 PostgreSQL 编译器编译预处理的 C 代码文件时，它会在 PostgreSQL `include` 目录中查找 ECPG 标头文件。因此，您可能必须使用 `-I` 选项将编译器指向正确的目录（例如 `-I/usr/local/pgsql/include`）。

库

使用 C 代码和嵌入式 SQL 的程序必须链接到 `libecpg` 库。例如，您可使用链接器选项 `-L/usr/local/pgsql/lib -lecpg`。

转换后的 ECPG 应用程序通过嵌入式 SQL 库 (`ecpglib`) 调用 `libpq` 库中的函数，并使用标准的前端/后端协议与 PostgreSQL 服务器通信。

将虚拟生成的列从 Oracle 迁移至 PostgreSQL

由 Veeranjaneyulu Grandhi (AWS)、Rajesh Madiwale (AWS) 和 Ramesh Pathuri (AWS) 编写

摘要

在版本 11 及以前版本中，PostgreSQL 不提供直接等同于 Oracle 虚拟列的功能。从 Oracle 数据库迁移到 PostgreSQL 版本 11 或以前版本时，处理虚拟生成的列很困难，原因有两个：

- 迁移时虚拟列不可见。
- PostgreSQL 不支持版本 12 之前的 `generate` 表达式。

但是，也有一些变通方法可模拟类似的功能。当您使用 AWS Database Migration Service (AWS DMS) 将数据从 Oracle 数据库迁移至 PostgreSQL 版本 11 及以前版本时，您可以使用触发函数在虚拟生成的列中填充值。此模式提供了可用于此目的 Oracle 数据库和 PostgreSQL 代码的示例。在 AWS 上，对于您的 PostgreSQL 数据库，您可以使用 Amazon Relational Database Service (Amazon RDS) for PostgreSQL 或 Amazon Aurora PostgreSQL-Compatible Edition。

从 PostgreSQL 版本 12 开始支持生成的列。生成的列可以按其他列值即时计算，也可以计算和存储。PostgreSQL 生成列与 Oracle 虚拟列类似。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 源 Oracle 数据库
- 目标 PostgreSQL 数据库（在 Amazon RDS for PostgreSQL 或 Aurora PostgreSQL-Compatible 上）
- PL/pgSQL 编码专业知识

限制

- 仅适用于 12 之前的 PostgreSQL 版本。
- 适用于 Oracle 数据库版本 11g 或更高版本。
- 数据迁移工具不支持虚拟列。
- 仅适用于同一表中定义的列。
- 如果虚拟生成的列引用确定性的用户定义函数，则其不能将其用作分区键列。

- 表达式输出必须是标量值。它无法返回 Oracle 提供的数据类型、用户定义的类型 LOB 或 LONG RAW。
- 针对虚拟列定义的索引，等同于 PostgreSQL 中基于函数的索引。
- 必须收集表格的统计信息。

工具

- pgAdmin 4 是一种适用于 PostgreSQL 的开源管理工具。该工具提供了图形界面，可简化数据库对象的创建、维护和使用。
- Oracle SQL Developer 是免费的集成开发环境，用于在传统部署和云部署中在 Oracle 数据库中使用 SQL。

操作说明

创建源数据库和目标数据库表

Task	描述	所需技能
创建源 Oracle 数据库表。	在 Oracle 数据库，使用以下语句创建包含虚拟生成的列的表。 `CREATE TABLE test.generated_column ( CODE NUMBER, STATUS VARCHAR2(12) DEFAULT 'PreOpen', FLAG CHAR(1) GENERATED ALWAYS AS (CASE UPPER(STATUS) WHEN 'OPEN' THEN 'N' ELSE 'Y' END) VIRTUAL VISIBLE );` 在此源表，STATUS 列中的数据通过 AWS DMS 迁移到目标数据库。但是，FLAG 列是使用	数据库管理员，应用程序开发人员

Task	描述	所需技能
	generate by 功能填充的，因此 AWS DMS 在迁移期间看不到此列。要实现 generated by 功能，您必须使用目标数据库中的触发器和函数填充 FLAG 列中的值，如下一个操作说明所示。	
在 AWS 创建目标 PostgreSQL 表。	使用以下语句在 AWS 上创建一个 PostgreSQL 表。 `CREATE TABLE test.generated_column ( code integer not null, status character varying(12) not null , flag character(1) );` 在此表中，status 列是标准列。flag 列将是根据该列中的数据生成的 status 列。	数据库管理员，应用程序开发人员

创建触发函数来处理 PostgreSQL 中的虚拟列

Task	描述	所需技能
创建 PostgreSQL 触发器。	在 PostgreSQL 中创建触发器。 `CREATE TRIGGER tgr_gen_column AFTER INSERT OR UPDATE OF status ON test.generated_column`	数据库管理员，应用程序开发人员

Task	描述	所需技能

	`FOR EACH ROW`
	`EXECUTE FUNCTION`
	`test.tgf_gen_colum`
	`n();`	

Task	描述	所需技能
创建 PostgreSQL 触发器函数。	在 PostgreSQL 中，为触发器创建函数。此函数填充由应用程序或 AWS DMS 插入或更新虚拟列，并验证数据。 `CREATE OR REPLACE FUNCTION test.tgf_gen_column() RETURNS trigger AS $VIRTUAL_COL$ BEGIN IF (TG_OP = 'INSERT') THEN IF (NEW.flag IS NOT NULL) THEN RAISE EXCEPTION 'ERROR: cannot insert into column "flag"' USING DETAIL = 'Column "flag" is a generated column.'; END IF; END IF; IF (TG_OP = 'UPDATE') THEN IF (NEW.flag::VARCHAR != OLD.flag::varchar) THEN RAISE EXCEPTION 'ERROR: cannot update column "flag"' USING DETAIL = 'Column "flag" is a generated column.'; END IF; END IF; IF TG_OP IN ('INSERT','UPDATE') THEN IF (old.flag is NULL) OR (coalesce(old.stat`	数据库管理员，应用程序开发人员

Task	描述	所需技能
	us,'') != coalesce(new.status,'')) THEN UPDATE test.gene rated_column SET flag = (CASE UPPER(status) WHEN 'OPEN' THEN 'N' ELSE 'Y' END) WHERE code = new.code; END IF; END IF; RETURN NEW; END $VIRTUAL_COL$ LANGUAGE plpgsql;	

使用 AWS DMS 测试数据迁移

Task	描述	所需技能
创建复制实例。	要创建复制实例，请按照 AWS DMS 文档中的说明进行操作。复制实例应与源数据库和目标数据库位于同一虚拟私有云（VPC）中。	数据库管理员，应用程序开发人员
创建源和目标端点。	要创建端点，请按照 AWS DMS 文档中的说明进行操作。	数据库管理员，应用程序开发人员
测试端点连接。	您可以通过指定 VPC 和复制实例并选择运行测试来测试端点连接。	数据库管理员，应用程序开发人员
创建和启动满载任务。	有关说明，请参阅 AWS DMS 文档中的创建任务和满载任务设置。	数据库管理员，应用程序开发人员

Task	描述	所需技能
验证虚拟列数据。	比较源数据库和目标数据库中虚拟列数据。您可手动验证数据，也可以为此步骤编写脚本。	数据库管理员，应用程序开发人员

相关资源

- AWS Database Migration Service 入门（AWS DMS 文档）
- 使用 Oracle 数据库作为 AWS DMS 的来源（AWS DMS 文档）
- 使用 PostgreSQL 数据库作为 AWS DMS 的目标（AWS DMS 文档）
- 在 PostgreSQL 中生成列（PostgreSQL 文档）
- 触发器函数（PostgreSQL 文档）
- Oracle 数据库中的虚拟列（Oracle 文档）

在 Aurora PostgreSQL-Compatible 上设置 Oracle UTL_FILE 功能

由 Rakesh Raghav (AWS) 和 anuradha chintha (AWS) 编写

摘要

在从 Oracle 迁移到 Amazon Web Services (AWS) 云上的 Amazon Aurora PostgreSQL 兼容版的过程中，您可能会遇到多种挑战。例如，迁移依赖于 Oracle UTL_FILE 实用程序的代码始终是一项挑战。在 Oracle PL/SQL 中，UTL_FILE 软件包与底层操作系统一起用于文件操作，例如读取和写入。该 UTL_FILE 实用程序适用于服务器和客户机系统。

Amazon Aurora PostgreSQL-Compatible 的是一款托管式数据库产品。因此，无法访问 Database Server 上的文件。此模式将引导您完成整合 Amazon Simple Storage Service (Amazon S3) 和 Amazon Aurora PostgreSQL 兼容版，以实现 UTL_FILE 功能子网。使用此集成，我们可以创建和使用文件，而无需使用第三方提取、转换、加载（ETL）工具或服务。

或者，您可以设置亚马逊 CloudWatch 监控和亚马逊 SNS 通知。

建议在生产环境中实施该解决方案之前，对方案进行全面测试。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- AWS Database Migration Service (AWS DMS) 专业知识
- PL/pgSQL 编码专长
- Amazon Aurora PostgreSQL-Compatible 集群
- 一个 S3 存储桶

限制

此模式不提供替代 Oracle UTL_FILE 实用程序的功能。但是，可以进一步增强步骤和示例代码，以实现数据库现代化目标。

产品版本

- Amazon Aurora PostgreSQL-Compatible 版本 11.9。

架构

目标技术堆栈

- Amazon Aurora（兼容 PostgreSQL）
- Amazon CloudWatch
- Amazon Simple Notification Service (Amazon SNS)
- Amazon S3

目标架构

下图高度概括此解决方案。

1. 文件将从应用程序上传至 S3 存储桶。
2. 该 `aws_s3` 扩展使用 PL/pgSQL 访问数据，并将数据上传到兼容 Aurora PostgreSQL 的版本。

工具

- Amazon Aurora PostgreSQL-Compatible – Amazon Aurora PostgreSQL-Compatible Edition 是一个完全托管式、兼容 PostgreSQL 和 ACID 的关系数据库引擎。您已了解了 MySQL 和 PostgreSQL 不仅具有高端商用数据库的速度和可靠性，同时还具有开源数据库的简单性和成本效益。
- AWS CLI - AWS 命令行界面（AWS CLI）是用于管理 Amazon Web Services 的统一工具。只通过一个工具进行下载和配置，您就可以使用命令行控制多个 Amazon Web Services 并利用脚本来自动执行这些服务。
- 亚马逊 CloudWatch — 亚马逊 CloudWatch 监控亚马逊 S3 的资源和使用情况。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项面向互联网的存储服务。Amazon S3 提供了一个存储层，用于接收和存储文件，以便在兼容 Aurora PostgreSQL 的集群之间使用和传输这些文件。
- aws_s3 – 该 `aws_s3` 扩展程序集成了 Amazon S3 和 Aurora PostgreSQL-Compatible。
- Amazon SNS – Amazon Simple Notification Service（Amazon SNS）可协调和管理发布者和客户端之间消息的传送或发送。在这种模式中，Amazon SNS 用于发送通知。
- pgAdmin – pgAdmin 是 Postgres 的开源管理工具。pgadmin 4 提供了一个用于创建、维护和使用数据库对象的图形界面。

代码

为了实现所需功能，该模式创建了多个与 UTL_FILE 命名类似的函数。其他信息 部分包含这些函数的代码库。

在代码中，将 testaurorabucket 替换为您的 S3 存储桶名称。将 us-east-1 替换为您测试 S3 存储桶所在的 AWS 区域。

操作说明

集成 Amazon S3 和 Aurora PostgreSQL-Compatible

Task	描述	所需技能
设置 IAM policy。	用于访问 S3 存储桶的 AWS Identity and Access Management (IAM) 角色。关于此代码，请参阅更多信息章节。	AWS 管理员，数据库管理员
将 Amazon S3 访问角色添加至 Aurora PostgreSQL。	创建两个 IAM 角色：一个角色用于对 Amazon S3 进行读取访问，一个角色用于对 Amazon S3 进行写入访问。将角色附加到兼容 Aurora PostgreSQL 的集群。 • S3Export 功能只有一个角色 • S3Import 功能只有一个角色 有关更多信息，请参阅 Aurora PostgreSQL-Compatible 文档，介绍如何将数据导入和导出到 Amazon S3。	AWS 管理员，数据库管理员

在 Aurora PostgreSQL-Compatible 中设置扩展

Task	描述	所需技能
创建 aws_commons 扩展。	`aws_commons` 扩展是 `aws_s3` 扩展的依赖项。	数据库管理员、开发人员
创建 aws_s3 扩展。	`aws_s3` 扩展程序与 Amazon S3 交互。	数据库管理员、开发人员

验证 Amazon S3 和 Aurora PostgreSQL-Compatible 的集成

Task	描述	所需技能
将 Amazon S3 中的数据导入到 Aurora PostgreSQL	要测试将文件导入到 Aurora PostgreSQL 兼容版中，请创建一个示例 CSV 文件并将其上传到 S3 存储桶中。基于 CSV 文件创建表定义，然后使用 `aws_s3.table_import_from_s3` 函数将文件加载到表中。	数据库管理员、开发人员
测试将文件从 Aurora PostgreSQL 导出至 Amazon S3。	要测试从兼容 Aurora PostgreSQL 的导出文件，请创建一个测试表，在其中填充数据，然后使用 `aws_s3.query_export_to_s3` 函数导出数据。	数据库管理员、开发人员

若要模拟 UTL_FILE 实用程序，请创建包装函数

Task	描述	所需技能
创建 utl_file_utility 架构。	该架构将包装器函数汇聚在一起。运行以下命令以创建 EIP。 `CREATE SCHEMA utl_file_utility;`	数据库管理员、开发人员
创建 file_type 类型。	使用以下代码创建 `file_type` 类型。 `CREATE TYPE utl_file_utility.file_type AS ( p_path character varying(30), p_file_name character varying );`	数据库管理员/开发人员
创建 init 函数	`init` 函数初始化公共变量，例如 `bucket` 或 `region`。关于此代码，请参阅更多信息章节。	数据库管理员/开发人员
创建包装器函数。	创建包装函数 `fopen`、`put_line` 和 `fclose`。关于代码，请参阅更多信息章节。	数据库管理员、开发人员

测试包装器函数

Task	描述	所需技能
在写入模式测试包装器函数。	要在写入模式下测试包装器函数，请使用其他信息部分中提供的代码。	数据库管理员、开发人员
在追加模式下测试包装器函数。	要在追加模式下测试包装器函数，请使用其他信息部分中提供的代码。	数据库管理员、开发人员

相关资源

- Amazon S3 集成
- Amazon S3
- Aurora
- Amazon CloudWatch
- Amazon SNS

其他信息

设置 IAM policy

创建以下策略。

策略名称

S3 IntRead

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3integrationtest
",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
```

```
                "s3:ListBucket"
            ],
            "Resource": [
         "arn:aws:s3:::testaurorabuc
ket/*",
         "arn:aws:s3:::testaurorabuc
ket"
            ]
        }
    ]
}
```

S3 IntWrite

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3integrationtest
",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",

                "s3:ListBucket"
            ],
            "Resource": [
    "arn:aws:s3:::testaurorabucket/
*",                "arn:aws:s3:::test
aurorabucket"
            ]
        }
    ]
}
```

创建 init 函数

若要初始化常用变量，例如 bucket 或 region，请使用以下代码创建 init 函数。

```
CREATE OR REPLACE FUNCTION utl_file_utility.init(
    )
    RETURNS void
    LANGUAGE 'plpgsql'
```

```
    COST 100
    VOLATILE
AS $BODY$
BEGIN
      perform set_config
      ( format( '%s.%s','UTL_FILE_UTILITY', 'region' )
      , 'us-east-1'::text
      , false );

      perform set_config
      ( format( '%s.%s','UTL_FILE_UTILITY', 's3bucket' )
      , 'testaurorabucket'::text
      , false );
END;
$BODY$;
```

创建包装器函数

创建 fopen、put_line 和 fclose 包装器函数。

fopen

```
CREATE OR REPLACE FUNCTION utl_file_utility.fopen(
    p_file_name character varying,
    p_path character varying,
    p_mode character DEFAULT 'W'::bpchar,
    OUT p_file_type utl_file_utility.file_type)
    RETURNS utl_file_utility.file_type
    LANGUAGE 'plpgsql'

    COST 100
    VOLATILE
AS $BODY$
declare
    v_sql character varying;
    v_cnt_stat integer;
    v_cnt integer;
    v_tabname character varying;
    v_filewithpath character varying;
    v_region character varying;
    v_bucket character varying;

BEGIN
```

```
   /*initialize common variable */
   PERFORM utl_file_utility.init();
   v_region := current_setting( format( '%s.%s', 'UTL_FILE_UTILITY', 'region' ) );
   v_bucket :=  current_setting( format( '%s.%s', 'UTL_FILE_UTILITY', 's3bucket' ) );

   /* set tabname*/
   v_tabname := substring(p_file_name,1,case when strpos(p_file_name,'.') = 0 then
length(p_file_name) else strpos(p_file_name,'.') - 1 end );
   v_filewithpath := case when NULLif(p_path,'') is null then p_file_name else
concat_ws('/',p_path,p_file_name) end ;
   raise notice 'v_bucket %, v_filewithpath % , v_region %', v_bucket,v_filewithpath,
v_region;

   /* APPEND MODE HANDLING; RETURN EXISTING FILE DETAILS IF PRESENT ELSE CREATE AN
EMPTY FILE */
   IF p_mode = 'A' THEN
       v_sql := concat_ws('','create temp table if not exists ', v_tabname,' (col1
text)');
       execute v_sql;

       begin
       PERFORM aws_s3.table_import_from_s3
           ( v_tabname,
           '',
           'DELIMITER AS ''#''',
           aws_commons.create_s3_uri
           (     v_bucket,
               v_filewithpath ,
               v_region)
           );
       exception
           when others then
            raise notice 'File load issue ,%',sqlerrm;
            raise;
       end;
       execute concat_ws('','select count(*) from ',v_tabname) into v_cnt;

       IF v_cnt > 0
       then
           p_file_type.p_path := p_path;
           p_file_type.p_file_name := p_file_name;
       else
           PERFORM aws_s3.query_export_to_s3('select ''''',
```

```
                            aws_commons.create_s3_uri(v_bucket, v_filewithpath,
 v_region)
                              );

            p_file_type.p_path := p_path;
            p_file_type.p_file_name := p_file_name;
        end if;
        v_sql := concat_ws('','drop table ', v_tabname);
        execute v_sql;
    ELSEIF p_mode = 'W' THEN
            PERFORM aws_s3.query_export_to_s3('select ''''',
                            aws_commons.create_s3_uri(v_bucket, v_filewithpath,
 v_region)
                              );
            p_file_type.p_path := p_path;
            p_file_type.p_file_name := p_file_name;
    END IF;

EXCEPTION
        when others then
            p_file_type.p_path := p_path;
            p_file_type.p_file_name := p_file_name;
            raise notice 'fopenerror,%',sqlerrm;
            raise;
END;
$BODY$;
```

put_line

```
CREATE OR REPLACE FUNCTION utl_file_utility.put_line(
    p_file_name character varying,
    p_path character varying,
    p_line text,
    p_flag character DEFAULT 'W'::bpchar)
    RETURNS boolean
    LANGUAGE 'plpgsql'

    COST 100
    VOLATILE
AS $BODY$
/*****************************************************************************
* Write line, p_line in windows format to file, p_fp - with carriage return
* added before new line.
```

```
**************************************************************************/
declare
    v_sql varchar;
    v_ins_sql varchar;
    v_cnt INTEGER;
    v_filewithpath character varying;
    v_tabname  character varying;
    v_bucket character varying;
    v_region character varying;

BEGIN
 PERFORM utl_file_utility.init();

/* check if temp table already exist */

 v_tabname := substring(p_file_name,1,case when strpos(p_file_name,'.') = 0 then
 length(p_file_name) else strpos(p_file_name,'.') - 1 end );

 v_sql := concat_ws('','select count(1) FROM pg_catalog.pg_class c LEFT JOIN
 pg_catalog.pg_namespace n ON n.oid = c.relnamespace where n.nspname like ''pg_temp_
%'''
                         ,' AND pg_catalog.pg_table_is_visible(c.oid) AND
 Upper(relname) = Upper( '''
                         ,  v_tabname ,''' ) ');

 execute v_sql into v_cnt;

  IF v_cnt = 0 THEN
        v_sql := concat_ws('','create temp table ',v_tabname,' (col text)');
       execute v_sql;
       /* CHECK IF APPEND MODE */
       IF upper(p_flag) = 'A' THEN
           PERFORM utl_file_utility.init();
           v_region := current_setting( format( '%s.%s', 'UTL_FILE_UTILITY',
 'region' ) );
           v_bucket :=  current_setting( format( '%s.%s', 'UTL_FILE_UTILITY',
 's3bucket' ) );

           /* set tabname*/
           v_filewithpath := case when NULLif(p_path,'') is null then p_file_name else
 concat_ws('/',p_path,p_file_name) end ;

           begin
              PERFORM aws_s3.table_import_from_s3
```

```
                    ( v_tabname,
                         '',
                      'DELIMITER AS ''#''',
                       aws_commons.create_s3_uri
                          ( v_bucket,
                              v_filewithpath,
                              v_region    )
                    );
            exception
                when others then
                    raise notice  'Error Message : %',sqlerrm;
                    raise;
            end;
        END IF;
    END IF;
    /* INSERT INTO TEMP TABLE */
    v_ins_sql := concat_ws('','insert into ',v_tabname,' values(''',p_line,''')');
    execute v_ins_sql;
    RETURN TRUE;
    exception
            when others then
                raise notice  'Error Message : %',sqlerrm;
                raise;
END;
$BODY$;
```

fclose

```
CREATE OR REPLACE FUNCTION utl_file_utility.fclose(
    p_file_name character varying,
    p_path character varying)
    RETURNS boolean
    LANGUAGE 'plpgsql'

    COST 100
    VOLATILE
AS $BODY$
DECLARE
    v_filewithpath character varying;
    v_bucket character varying;
    v_region character varying;
    v_tabname character varying;
    v_sql character varying;
```

```
BEGIN
      PERFORM utl_file_utility.init();

    v_region := current_setting( format( '%s.%s', 'UTL_FILE_UTILITY', 'region' ) );
    v_bucket :=  current_setting( format( '%s.%s', 'UTL_FILE_UTILITY', 's3bucket' ) );

    v_tabname := substring(p_file_name,1,case when strpos(p_file_name,'.') = 0 then
 length(p_file_name) else strpos(p_file_name,'.') - 1 end );
    v_filewithpath := case when NULLif(p_path,'') is null then p_file_name else
 concat_ws('/',p_path,p_file_name) end ;

    raise notice 'v_bucket %, v_filewithpath % , v_region %', v_bucket,v_filewithpath,
 v_region ;

    /* exporting to s3 */
    perform aws_s3.query_export_to_s3
        (concat_ws('','select * from ',v_tabname,'  order by ctid asc'),
            aws_commons.create_s3_uri(v_bucket, v_filewithpath, v_region)
        );
    v_sql := concat_ws('','drop table ', v_tabname);
    execute v_sql;
    RETURN TRUE;
EXCEPTION
       when others then
     raise notice 'error fclose %',sqlerrm;
     RAISE;
END;
$BODY$;
```

测试您的设置与包装器功能

使用下面的代码示例测试设置。

测试写入模式

以下代码在 S3 存储桶中写入名为 `s3inttest` 的文件。

```
do $$
declare
l_file_name varchar := 's3inttest' ;
l_path varchar := 'integration_test' ;
l_mode char(1) := 'W';
l_fs utl_file_utility.file_type ;
l_status boolean;
```

```
begin
select * from
utl_file_utility.fopen( l_file_name, l_path , l_mode ) into l_fs ;
raise notice 'fopen : l_fs : %', l_fs;

select * from
utl_file_utility.put_line( l_file_name, l_path ,'this is test file:in s3bucket: for
 test purpose', l_mode ) into l_status ;
raise notice 'put_line : l_status %', l_status;

select * from utl_file_utility.fclose( l_file_name , l_path ) into l_status ;
raise notice 'fclose : l_status %', l_status;

end;
$$
```

测试追加模式

以下代码将行追加到上一个测试中创建的 s3inttest 文件上。

```
do $$
declare
l_file_name varchar := 's3inttest' ;
l_path varchar := 'integration_test' ;
l_mode char(1) := 'A';
l_fs utl_file_utility.file_type ;
l_status boolean;

begin
select * from
utl_file_utility.fopen( l_file_name, l_path , l_mode ) into l_fs ;
raise notice 'fopen : l_fs : %', l_fs;


select * from
utl_file_utility.put_line( l_file_name, l_path ,'this is test file:in s3bucket: for
 test purpose : append 1', l_mode ) into l_status ;
raise notice 'put_line : l_status %', l_status;

select * from
utl_file_utility.put_line( l_file_name, l_path ,'this is test file:in s3bucket : for
 test purpose : append 2', l_mode ) into l_status ;
raise notice 'put_line : l_status %', l_status;
```

```
select * from utl_file_utility.fclose( l_file_name , l_path ) into l_status ;
raise notice 'fclose : l_status %', l_status;

end;
$$
```

Amazon SNS 通知

或者，您可以在 S3 存储桶上设置亚马逊 CloudWatch 监控和 Amazon SNS 通知。有关更多信息，请参阅监控 Amazon S3 和设置 Amazon SNS 通知。

从 Oracle 迁移至 Amazon Aurora PostgreSQL 后验证数据库对象

由 Venkatramana Chintha (AWS) 和 Eduardo Valentim (AWS) 编写

摘要

此模式描述了一种在将 Oracle 数据库迁移到兼容 Amazon Aurora PostgreSQL 的版本后验证对象 step-by-step的方法。

此模式概述了数据库对象验证的使用场景和步骤；有关更多详细信息，请参阅 AWS 数据库博客上的 AWS SCT 和 AWS DM S 在迁移后验证数据库对象。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 已迁移至 PostgreSQL-Compatibl 的数据库的本地 Oracle 数据库。
- 适用亚马逊RDSDataFullAccess政策的登录凭证，适用于兼容 Aurora PostgreSQL 的数据库。
- 此模式使用 Aurora Serverless 数据库集群查询编辑器，该编辑器可在 Amazon Relational Database Service (Amazon RDS) 控制台中找到。但是，您可将此模式与任何其他查询编辑器一起使用。

限制

- Oracle SYNONYM 对象在 PostgreSQL 中不可用，但可以通过视图或 SET search_path 查询进行部分验证。
- Amazon RDS 查询编辑器仅在某些 AWS 区域 以及某些 MySQL 和 PostgreSQL 版本中可用。

架构

工具

工具

- Amazon Aurora PostgreSQL-Compatible Edition – Aurora PostgreSQL-Compatible 是一个完全托管式、兼容 PostgreSQL 和 ACID 的关系数据库引擎，结合了高端商用数据库的速度和可靠性，同时还具有开源数据库的成本效益。

- Amazon RDS – Amazon Relational Database Service(Amazon RDS)能够在 Amazon Web Services Cloud 中轻松设置、操作和扩展关系数据库。它为行业标准的关系数据库提供了经济高效、可调整大小的容量，并管理常见的数据库管理任务。
- Query Editor for Aurora Severless – Query 编辑器可帮助在 Amazon RDS 控制台中运行 SQL 查询。您可以在 Aurora Serverless 数据库集群上运行任何有效的 SQL 语句，包括数据操作和数据定义语句。

要验证对象，请使用“附件部分中的“对象验证脚本”文件中的完整脚本。请参考下表。

Oracle 对象	待用脚本
程序包	查询 1
表	查询 3
视图	查询 5
序列	查询 7
触发	查询 9
主键	查询 11
索引	查询 13
检查约束	查询 15
外键	查询 17

PostgreSQL 对象	待用脚本
程序包	查询 2
表	查询 4
视图	查询 6
序列	查询 8

触发	查询 10
主键	查询 12
索引	查询 14
检查约束	查询 16
外键	查询 18

操作说明

校验源 Oracle 数据库中的对象

Task	描述	所需技能
在源 Oracle 数据库运行 “软件包” 验证查询。	从 “附件”部分下载并打开“对象验证脚本”文件。通过您的客户端程序连接至源 Oracle 数据库。在“对象验证脚本” 文件中运行“查询 1”验证脚本。重要：在查询中输入您的 Oracle 用户名而非“your_schema”。务必记录查询结果。	开发人员、数据库管理员
运行 “表” 验证查询。	从 “对象验证脚本” 文件运行 “查询 3” 脚本。务必记录查询结果。	开发人员、数据库管理员
运行 “视图”验证查询。	从 “对象验证脚本” 文件运行 “查询 5” 脚本。务必记录查询结果。	开发人员、数据库管理员
运行“序列”计数验证。	从 “对象验证脚本” 文件运行 “查询 7” 脚本。务必记录查询结果。	开发人员、数据库管理员

Task	描述	所需技能
运行“触发器”验证查询。	从 “对象验证脚本” 文件运行 “查询 9” 脚本。务必记录查询结果。	开发人员、数据库管理员
运行“主键”验证查询。	从 “对象验证脚本” 文件运行 “查询 11” 脚本。务必记录查询结果。	开发人员、数据库管理员
运行 “索引” 验证查询。	从 “对象验证脚本” 文件运行 “查询 13” 脚本。务必记录查询结果。	开发人员、数据库管理员
运行“检查约束”验证查询。	从 “对象验证脚本” 文件运行 “查询 15” 脚本。务必记录查询结果。	开发人员、数据库管理员
运行“外键”验证查询。	从 “对象验证脚本” 文件运行 “查询 17” 脚本。务必记录查询结果。	开发人员、数据库管理员

验证目标 Aurora PostgreSQL-Compatible 中的对象

Task	描述	所需技能
使用查询编辑器连接至与 Aurora PostgreSQL 兼容的目标数据库。	登录 AWS 管理控制台 并打开 Amazon RDS 控制台。在右上角，选择在其中创建 Aurora Postready 的 AWS 区域。在导航窗格中，选择“数据库”，然后选择目标 Aurora PostgreSQL-Compatible 数据库 在“操作”中，选择“查询”。重要说明：如果您之前未连接到数据库，则“Connect to database (连接到数据库)”页面将打开。	开发人员、数据库管理员

Task	描述	所需技能
	然后，您需要输入数据库信息，如用户名和密码。	
运行 “程序包” 验证查询。	在“附件”部分，从 “对象验证脚本” 文件运行 “查询 2” 脚本。务必记录查询结果。	开发人员、数据库管理员
运行 “表” 验证查询。	返回 PostgreSQL-Compatible 的数据库的查询编辑器，然后运行“对象验证脚本”文件中的 “查询 4” 脚本。务必记录查询结果。	开发人员、数据库管理员
运行 “视图”验证查询。	返回 PostgreSQL-Compatible 的数据库的查询编辑器，然后运行“对象验证脚本”文件中的 “查询 6” 脚本。务必记录查询结果。	开发人员、数据库管理员
运行“序列”计数验证。	返回 PostgreSQL-Compatible 的数据库的查询编辑器，然后运行“对象验证脚本”文件中的 “查询 8” 脚本。务必记录查询结果。	开发人员、数据库管理员
运行“触发器”验证查询。	返回 PostgreSQL-Compatible 的数据库的查询编辑器，然后运行“对象验证脚本”文件中的 “查询 10” 脚本。务必记录查询结果。	开发人员、数据库管理员
运行“主键”验证查询。	返回 PostgreSQL-Compatible 的数据库的查询编辑器，然后运行“对象验证脚本”文件中的 “查询 12” 脚本。务必记录查询结果。	开发人员、数据库管理员

Task	描述	所需技能
运行 “索引” 验证查询。	返回 PostgreSQL-Compatible 的数据库的查询编辑器，然后运行“对象验证脚本”文件中的 “查询 14” 脚本。务必记录查询结果。	开发人员、数据库管理员
运行“检查约束”验证查询。	从 “对象验证脚本” 文件运行 “查询 16” 脚本。务必记录查询结果。	开发人员、数据库管理员
运行“外键”验证查询。	从 “对象验证脚本” 文件运行 “查询 18” 脚本。务必记录查询结果。	开发人员、数据库管理员

比较源数据库与目标数据库验证记录

Task	描述	所需技能
比较和验证两个查询结果。	比较 Oracle 和 Aurora PostgreSQL-Compatible 数据库的查询结果，以验证所有对象。如果都匹配，则所有对象都已成功验证。	开发人员、数据库管理员

相关资源

- 在迁移后通过 AWS SCT 和 AWS DMS 验证数据库对象
- Amazon Aurora 功能：PostgreSQL-Compatible Edition

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

更换主机

主题

- 加快 Microsoft 工作负载的发现和迁移到 AWS
- 在 AWS Managed Services on Windows 上自动执行工作负载前摄取活动
- 在重新托管迁移到的过程中，为防火墙请求创建批准流程 AWS
- 将 EC2 Windows 实例提取并迁移到 AWS Managed Services 账户
- 将 Couchbase 服务器数据库迁移到亚马逊 EC2
- 使用日志传送将适用于 LUW 的 Db2 迁移 EC2 到 Amazon 以减少中断时间
- 将适用于 LUW 的 Db2 迁移到 EC2 具有高可用性灾难恢复功能的 Amazon
- 使用 Pow VMware VMs erCLI 通过 HCX 自动化进行迁移
- 将 F5 BIG-IP 工作负载迁移至 Amazon Web Services Cloud 上的 F5 BIG-IP VE
- 使用二进制方法将本地 Go Web 应用程序迁移至 AWS Elastic Beanstalk
- 将本地 SFTP 服务器迁移到使用 AWSAWS Transfer for SFTP
- 使用 AWS 应用程序迁移服务将本地虚拟机迁移到 Amazon EC2
- 使用 AWS SFTP 将小型数据集从本地迁移至 Amazon S3
- 从 Oracle 迁移 GlassFish 到 AWS Elastic Beanstalk
- 将本地 Oracle 数据库迁移到亚马逊上的 Oracle EC2
- 使用 Oracle 数据泵将本地 Oracle 数据库迁移到亚马逊 EC2
- 使用 AWS MGN 将 RHEL BYOL 系统迁移至 AWS License-Included 实例
- 将本地 SAP ASE 数据库迁移到亚马逊 EC2
- 将本地微软 SQL Server 数据库迁移到亚马逊 EC2
- 将本地 MySQL 数据库迁移到亚马逊 EC2
- 使用 Application Migration Service 缩短同构 SAP 迁移割接时间
- 在 Amazon Web Services Cloud 中重新托管本地工作负载：迁移核对清单
- 使用 Amazon 为 SQL Server Always On FCI 设置多可用区基础架构 FSx
- 使用 BMC Discovery 查询提取迁移数据以进行迁移规划

加快 Microsoft 工作负载的发现和迁移到 AWS

由 Ali Alzand 创作

摘要

此模式向您展示如何使用迁移验证工具包 PowerShell 模块来发现您的 Microsoft 工作负载并将其迁移到 AWS。该模块的工作原理是，对与任何 Microsoft 工作负载关联的常见任务执行多项检查和验证。例如，该模块会检查可能连接了多个磁盘的实例或使用许多 IP 地址的实例。有关该模块可以执行的检查的完整列表，请参阅模块 GitHub 页面上的 “检查” 部分。

迁移验证工具包 PowerShell 模块可以帮助您的组织减少在发现 Microsoft 工作负载上正在运行哪些应用程序和服务时所花费的时间和精力。该模块还可以帮助您确定工作负载的配置，以便您可以了解 AWS 是否支持您的配置。该模块还提供后续步骤和缓解操作的建议，这样您就可以避免在迁移之前、期间或之后出现任何配置错误。

先决条件和限制

先决条件

- 本地管理员帐户
- PowerShell 4.0

限制

- 仅适用于微软 Windows Server 2012 R2 或更高版本

工具

工具

- PowerShell 4.0

代码存储库

此模式的迁移验证工具包 PowerShell 模块可在-microsoft GitHub migration-validator-toolkit-for-workdloads 存储库中找到。

操作说明

在单个目标上运行迁移验证工具包 PowerShell 模块

Task	描述	所需技能

下载、提取、导入和调用模块。	选择以下方法之一来下载和部署模块：
	• 运行 PowerShell 脚本
	• 下载并解压.zip 文件
	• 克隆 GitHub 存储库
	
	运行 PowerShell 脚本
	
	在中 PowerShell，运行以下示例代码：
	
	`#MigrationValidatorToolkit`
	`$uri = 'https://github.com/aws-samples/migration-validator-toolkit-for-microsoft-workloads/archive/refs/heads/main.zip'`
	`$destination = (Get-Location).Path`
	`if ((Test-Path -Path "$destination\MigrationValidatorToolkit.zip" -PathType Leaf) -or (Test-Path -Path "$destination\MigrationValidatorToolkit")) {`
	`write-host "File $destination\MigrationValidatorToolkit.zip or folder`
		系统管理员

Task	描述	所需技能

	``
	$destination\Migra
	tionValidatorToolkit
	found, exiting"
	}else {
	Write-host "Enable
	TLS 1.2 for this
	PowerShell session
	only."
	[Net.ServicePointM
	anager]::SecurityP
	rotocol = [Net.Secu
	rityProtocolType]:
	:Tls12
	$webClient =
	New-Object System.Ne
	t.WebClient
	Write-host
	"Downloading Migration
	ValidatorToolkit.zip"
	$webClient.Downloa
	dFile($uri, "$destina
	tion\MigrationVali
	datorToolkit.zip")
	Write-host
	"MigrationValidato
	rToolkit.zip download
	successfully"
	Add-Type -Assembly
	"system.io.compres
	sion.filesystem"
	[System.IO.Compres
	sion.ZipFile]::Ext
	ractToDirectory("$
	destination\Migrat
	ionValidatorToolki
	t.zip","$destinati
	on\MigrationValida
	torToolkit")
	Write-host
	"Extracting Migration
	ValidatorToolkit.zip
	complete successfully"
	``	

Task	描述	所需技能

	<pre>Import-Module "$destination\MigrationValidatorToolkit\migration-validator-toolkit-for-microsoft-workloads-main\MigrationValidatorToolkit.psm1"; Invoke-MigrationValidatorToolkit
	}</pre>
	该代码从.zip 文件下载模块。然后，代码提取、导入和调用该模块。
	
	下载并解压.zip 文件
	
	1. 下载 .zip 文件（下载）。
	2. 将 .zip 文件解压缩。
	3. 按照本指南的 “手动调用模块” 故事中的步骤进行操作。
	
	克隆 GitHub 存储库
	
	1. 要克隆 GitHubmigration-validator-toolkit-for微软工作负载存储库，请在终端窗口中运行以下 Git 命令：
	
	<pre>git clone https://github.com/aws-samples/migration-validator-toolkit-for-microsoft-workloads.git</pre>	

Task	描述	所需技能
	2. 按照本指南的 “手动调用模块” 故事中的步骤进行操作。	

Task	描述	所需技能

手动调用模块。	1. 转到存储已下载模块的目录。
	2. 要生成您选择的输出，请在中以管理员身份运行以下命令之一 PowerShell：
	
	格式表格式：
	`Import-Module .\MigrationValidatorToolkit.psm1;Invoke-MigrationValidatorToolkit`
	
	格式列表格式：
	`Import-Module .\MigrationValidatorToolkit.psm1;Invoke-MigrationValidatorToolkit -List`
	
	输出GridView格式：
	`Import-Module .\MigrationValidatorToolkit.psm1;Invoke-MigrationValidatorToolkit -GridView`
	
	ConvertTo-csv 格式：
	`Import-Module .\MigrationValidatorToolkit.psm1;Invoke-MigrationValidatorToolkit -csv`
		系统管理员

在多个目标上运行迁移验证工具包 PowerShell 模块

Task	描述	所需技能
下载.zip 文件或克隆 GitHub 存储库。	请选择以下选项之一： • 下载 zip 文件。（下载）。 • 要克隆 GitHubmigration-validator-toolkit-for微软工作负载存储库，请在终端窗口中运行以下 Git 命令： `git clone https://github.com/aws-samples/migration-validator-toolkit-for-microsoft-workloads.git`	系统管理员
更新 server.csv 列表。	如果您下载了.zip 文件，请按照以下步骤操作： 1. 将 .zip 文件解压缩。 2. 转到 `Migration ValidatorToolkit\Inputs\` 目录。 3. `serverlist.csv` 使用目标计算机的主机名进行更新。	系统管理员
调用模块。	您可以使用域内任何使用对目标计算机具有管理员访问权限的域用户的计算机。 1. 将源代码下载为.zip 文件并解压文件。	系统管理员

Task	描述	所需技能
	2. 以 PowerShell管理员身份运行以下命令： `Import-Module .\MigrationValidatorToolkit.psm1;Invoke-DomainComputers` 输出.csv 文件以`MigrationValidatorToolkit\Outputs\folder` 前缀名称`DomainComputers_MigrationAutomations_YYYY-MM-DDTHH-MM-SS` 保存。	

故障排除

事务	解决方案
`MigrationValidatorToolkit` 将有关执行、命令和错误的信息写入正在运行的主机上的日志文件。	您可以在以下位置手动查看日志文件： 1. 转到 `MigrationValidatorToolkit\logs\` 目录。 2. 找到日志文件。日志文件名的格式为：`ComputerName_MigrationValidatorToolkit_YYYY-MM-SSTHH-MM-SS.log`

相关资源

- 将微软工作负载迁移到 AWS 的选项、工具和最佳实践（AWS Prescriptive Guidence）
- 微软迁移模式（AWS Prescriptive Guidence）

- AWS 上的免费云迁移服务（AWS 文档）
- 预定义的发布后操作（应用程序营销文档）

其他信息

常见问题

在哪里可以运行迁移验证工具包 PowerShell 模块？

你可以在微软 Windows Server 2012 R2 或更高版本上运行该模块。

我什么时候运行这个模块？

我们建议您在迁移过程的评估阶段运行该模块。

该模块会修改我现有的服务器吗？

不是。 此模块中的所有操作都是只读的。

运行该模块需要多长时间？

运行该模块通常需要 1-5 分钟，但这取决于服务器的资源分配。

该模块需要什么权限才能运行？

您必须使用本地管理员帐户运行该模块。

我可以在物理服务器上运行该模块吗？

是的，只要操作系统是微软 Windows Server 2012 R2 或更高版本即可。

如何为多台服务器大规模运行该模块？

要在多台加入域的计算机上大规模运行该模块，请按照本指南中 “在多个目标上运行迁移验证器工具包” PowerShell 模块中的步骤进行操作。对于未加入域的计算机，请使用远程调用或按照本指南中在单个目标长篇故事上运行迁移验证器工具包 PowerShell 模块中的步骤在本地运行该模块。

在 AWS Managed Services on Windows 上自动执行工作负载前摄取活动

由 Jacob Zhang (AWS)、Calvin Yeh (AWS) 和 Dwayne Bordelon (AWS) 编写

摘要

在 Amazon Web Services (AWS) Cloud 上，AWS Managed Services(AMS) 使用 AMS 工作负载摄取 (WIGS) 将现有工作负载迁移至 AMS 托管 VPC 中。此模式描述了一种自动执行常见的工作负载前摄取活动的解决方案，例如升级.NET PowerShell 和 Windows 以及运行 AMS 维护的 Windows WIGS 摄取前验证。该模式还为运行结果提供了统一用户界面。它将执行摄取前活动的 AWS Systems Manager 命令文档打包到 AWS 模板中。 CloudFormation 该模板可以重复部署，无需访问 Systems Manager 本身，也无需与 AMS 的自动化冲突。

商业背景

迁移到 AMS 需要使用包含 AMS 组件的 AMS 托管亚马逊系统映像（EC2) 配置新的亚马逊弹性计算云 (AmazonAMIs) 实例。在现有数据中心运行的任何工作负载或应用程序都必须重新部署到从这些 AMS AMIs 启动的新 EC2 实例。为了避免在此过程中可能出现的大量手动工作，AMS 团队构建了 AMS 工作负载摄取 (WIGS) 工作流，将您的自定义映像加载到 AMS。

在 WIGS 流程开始之前，Windows 实例必须满足先决条件。Windows PowerShell 脚本通常用于执行必要的准备工作（WIGS 准备）和检查实例是否已准备就绪 WIGs （WIGS 摄取前验证）。准备和验证过程需要工程师在每台服务器上花费 15-30 分钟，手动登录并逐一运行脚本。

业务驱动因素

传统上，使用 Systems Manager 可以自动执行操作任务，例如运行 Windows PowerShell 脚本。然而，由于 AMS 的自动化与用户的自动化之间的风险较高且频繁发生冲突，AMS 通常不会授予其用户对 Systems Manager 的访问权限。

对于使用 AWS 应用程序迁移服务 (AWS MGN) 进行大规模迁移，中的Windows PowerShell 脚本`C:\Program Files (x86)\AWS Replication Agent\post_launch folder`通常会在启动测试或直接转换实例时自动运行。但是，如果在实例启动期间立即运行这些脚本，则经常会与 AMS 的自动化发生冲突。因此，启动可能会失败，而不会提供排除故障所需运行结果。

该模式解决了这些问题并提供了一个有效的自动化解决方案。

先决条件和限制

先决条件

- 已完成 AMS 搭载的有效 Amazon Web Services account。

- 创建 Amazon Simple Storage Service (Amazon S3) 存储桶 如果账户中没有您可以控制的 S3 存储桶，请使用更改请求 (RFC) 创建一个。
- 从存储库下载的 Pre WIGs _cfn.json 模板。ams-auto-prewigs-windows
- 应用此模式的服务器必须满足以下要求：
  - 运行 Windows Server 2012 或更高版本。
  - 已在沙盒 VPC 迁移子网中启动或准备启动。
  - 安装 AWS Systems Manager Agent (SSM Agent)。
  - 连接 AWS Identity and Acccess Management (IAM) 实例配置文件 实例配置文件必须有权从同一 Amazon Web Services account 的 S3 存储桶下载文件。满足上述要求的实例配置文件通常已在迁移的早期设置期间建立。
  - 可从 AWS Systems Manager Fleet Manager 中查看。

限制

- WIGS 前的活动根据您的环境和业务要求而有所不同。您可能需要对此模式进行细微修改，以满足您的特定需求。

产品版本

- 该模式已在 Windows Server 2012、2012 R2、2016 和 2019 上进行了测试。从理论上讲，它适用于上述版本的 Windows。它不适用于早期的 Windows 版本。

架构

架构图如下：

1. 沙盒 VPC，其迁移子网包含尚未准备的服务器。
2. 存储 CloudFormation 模板使用的脚本的 S3 存储桶。
3. 该 CloudFormation 模板部署了 Systems Manager 命令文档。该过程将迭代进行，直到步骤完成。
4. 实例已准备就绪，并制作了 W RFCs IGS。
5. 在 AMS 托管 VPC 中，AMS 托管子网包含工作负载摄取后的服务器。

工作原理

- 此模式打包到一个 AWS CloudFormation 模板中，该模板允许基础设施即代码 (IaC) 可重复部署。您只需为需要此自动化的每个 Amazon Web Services account 部署此模板一次。
- 自动化适用于部署此模式的 AWS 账户AutoPreWIGs中具有标签密钥的所有 EC2 实例。首次AutoPreWIGs启动带有标签密钥的 Amazon EC2 Windows 实例时，自动化会执行以下任务。
  1. 将 Windows 升级 PowerShell 到版本 5.1，将.NET 升级到版本 4.5.2。该实例可能会重启多次，具体取决于其现有的 Windows PowerShell 和.NET 版本。每次重新启动后，升级都会继续，直到完成。此步骤使用 CloudFormation 模板中根据 Windows PowerShell 脚本修改的嵌入式代码，以及有关服务器重启的特定Systems Manager指南。
  2. 从亚马逊 S3 下载并运行您自定义的 Windows PowerShell 脚本，用于为 WIGS 准备亚马逊 EC2 Windows 实例。有关更多信息，请参阅操作说明部分。
  3. 安装来自 AWS 的 Windows WIGS 摄取前验证模块 PowerShell 。
  4. 运行 Windows WIGS 摄取前验证并使结果可在 Systems Manager 状态管理器中查看。

工具

- AWS CloudFormation — AWS CloudFormation 是一项可帮助您建模和设置 AWS 资源的服务。您可以使用描述您想要的所有 AWS 资源及其依赖关系的，这样您就可以启动这些资源并将其配置为堆栈。，此模式使用 CloudFormation 模板自动部署此模式中的资源。
- AWS Managed Services – AWS Managed Services (AMS) 是一项企业服务，可为您的 AWS 基础设施提供持续管理。在 AMS 环境中对基础设施所做的更改必须通过 RFC 进行。
- AWS Systems Manager – AWS Systems Manager（之前被称为 SSM）是一项 Amazon Web Services，您可用它在 AWS 上查看和控制您的基础设施。通过使用 Systems Manager 控制台，您可以查看来自多个 Amazon Web Services 的操作数据并在 AWS 资源之间自动执行操作任务。此模式使用 Systems Manager 运行并查看预 WIGS 活动的运行结果。
- Amazon S3 – Amazon Simple Storage Service（Amazon S3）是一种对象存储服务，提供行业领先的可扩展性、数据可用性、安全性和性能。此模式使用 Amazon S3 来存储 CloudFormation 模板和下载的 Windows PowerShell 脚本。

操作说明

创建自定义 Windows PowerShell 脚本以自动执行其他任务

Task	描述	所需技能
根据业务需求对服务器执行必要的更改。	如果您需要在服务器摄取之前将更改自动应用于服务器，请创建一个名为的 Windows PowerShell 脚本。`ingestion-prep.ps1` **Important** 该脚本不得包含重新启动服务器的指令，也不得要求管理员权限。	PowerShell 脚本
移除 AMS 不支持的软件。	AMS 需要在 WIGS 运行之前移除某些软件，例如防病毒应用程序和 VMware 工具。在 `ingestion-prep.ps1` 脚本中包括卸载。有关不支持的软件的更多信息，请参见 AWS 文档。	PowerShell 脚本

将 CloudFormation 模板和可选的 Windows PowerShell 脚本上传到亚马逊 S3

Task	描述	所需技能
在 S3 中创建文件夹。	在部署此模式的同一 Amazon Web Services account 的 S3 存储桶中，创建一个文件夹。	常规 AWS
上传脚本。	将您在上一篇长篇故事中创建的`PreWIGs_C`	常规 AWS

Task	描述	所需技能
	FN.json CloudFormation 模板和 ingestion-prep.ps1 Windows PowerShell 脚本上传到 Amazon S3 文件夹。	

部署堆 CloudFormation 栈

Task	描述	所需技能
选择更改类型。	导航到 AMS 控制台，以创建 RFC。使用 “从 CloudFormation (CFN) 模板创建堆栈” 更改类型。	常规 AMS
为 CloudFormation 模板的路径设置运行参数。	在执行配置部分中，展开其他配置。在CloudFormation 模板 S3 端点框中，将 URL 粘贴到 CloudFormation 模板中。	常规 AMS
指定 Amazon S3 文件夹路径。	在参数下，使用ScriptSource作为名称。对于值，输入包含 Windows PowerShell 脚本的 S3 文件夹的路径。确保使用 https://xxx URL 而不是 s3://xxx URI，并在末尾加上 /。	常规 AMS
部署堆栈。	选择 Create（创建）以部署堆栈。	常规 AMS
将 RFC 升级至 AMS Ops。	RFC 必须由 AMS 运营团队手动实施，因为它使用 Systems Manager 来部署资源并需要安全审查。您创建了 RFC 之后，系统即会自动拒绝它。选	常规 AMS

Task	描述	所需技能
	择 RFC，然后在 RFC 中添加一条对应信息，说明请手动执行。记下 RFC ID，并通过服务请求将其升级。	

将自动化应用到实例

Task	描述	所需技能
将 AutoPreWIGs 标签添加到实例。	记 IDs 下您要应用此自动化的所有实例，并等待至少 30 分钟，该实例才能完成 AMS 实施的自动化。提交自动 RFC 以添加以密钥AutoPreWIGs为标签并将任意字符串（例如 1）作为值。 添加标签后几分钟就会应用自动化。	常规 AMS
验证自动化结果。	打开 Systems Manager 控制台，选择 State Manager。选择名为 ams-prewig-prep-and-validation-Association 的关联 ID。在执行历史选项卡，您可看到自动化的结果。	常规 AMS
修正所有错误。	如果自动化失败，请选择其执行 ID。您可以查看每个 EC2 实例的运行结果。要查看自动化每个步骤的详细信息，请选择输出。如果特定步骤失败，请使用输出和错误部分中的信息来诊断问题。	迁移工程师

Task	描述	所需技能
移除 AutoPreWIGs 标签。	**Important** 修复错误后（如果有），请提交自动的 RFC 以删除AutoPreWIGs标签。如果不移除标签，WIGS 就会失败。	常规 AMS

摄取准备好的实例

Task	描述	所需技能
提交参加 W RFCs IGS。	现在，实例已准备好接受工作负载，请提交 WIGS RFCs 。	常规 AMS

相关资源

- AMS 工作负载摄取 (WIGS)
- 迁移工作负载：Windows 摄取前验证
- AWS Application Migration Service 快速入门指南
- 开始使用 AWS CloudFormation
- 设置 AWS Systems Manager

在重新托管迁移到的过程中，为防火墙请求创建批准流程 AWS

由 Srikanth Rangavajhala (AWS) 编写

摘要

如果要使用云迁移工厂AWS Application Migration Service或开启云迁移工厂 AWS Cloud，则前提条件之一是必须保持 TCP 端口 443 和 1500 处于打开状态。 AWS通常，打开这些防火墙端口需要获得信息安全 (InfoSec) 团队的批准。

此模式概述了在重新托管迁移到防火墙期间获得 InfoSec 团队批准的 AWS Cloud流程。您可以使用此流程来避免 InfoSec 团队拒绝您的防火墙请求，这可能会变得昂贵且耗时。防火墙请求流程有两个审查和批准步骤，由 AWS 迁移顾问和负责人与您 InfoSec 和应用程序团队合作打开防火墙端口。

此模式假设您正在与贵组织的 AWS 顾问或迁移专家一起计划重新托管迁移。如果您的组织没有防火墙审批流程或防火墙请求一揽子审批表单，则可以使用此模式。有关此内容的更多信息，请参阅此模式的限制部分。有关应用程序迁移服务的网络要求的更多信息，请参阅应用程序迁移服务文档中的网络要求。

先决条件和限制

先决条件

- 由贵组织的 AWS 顾问或迁移专家进行计划中的重新托管迁移
- 迁移堆栈所需端口和 IP 信息
- 现有和未来状态架构图
- 有关本地和目标基础架构、端口和流 zone-to-zone量的防火墙信息
- 防火墙请求审查清单（随附）
- 根据贵组织要求配置的防火墙请求文档
- 防火墙审阅者和审批者的联系人列表，包括以下角色：
  - 防火墙请求提交者 — AWS 迁移专家或顾问。防火墙请求提交者也可以是您组织中的迁移专家。
  - 防火墙请求审阅者 — 通常，这是来自的单点联系人 (SPOC)。 AWS
  - 防火墙请求批准者- InfoSec 团队成员。

限制

- 此模式描述了一个通用防火墙请求审批过程。各个组织的要求可能有所不同。
- 请确保跟踪对防火墙请求文档的更改。

下表显示了此模式的用例。

您的组织是否有现有的防火墙审批流程？	您的组织是否有现有的防火墙请求表单？	建议采取的措施
支持	是	与 AWS 顾问或迁移专家合作，实施贵组织的流程。
否	是	使用此模式的防火墙审批流程。请贵组织的 AWS 顾问或迁移专家提交防火墙申请一揽子批准表。
否	否	使用此模式的防火墙审批流程。请贵组织的 AWS 顾问或迁移专家提交防火墙申请一揽子批准表。

架构

下图显示了防火墙请求审批过程的步骤。

工具

您可以使用诸如 Palo Alto Network s 之类的扫描器工具 SolarWinds，也可以分析和验证防火墙和 IP 地址。

操作说明

分析防火墙请求

Task	描述	所需技能
分析端口和 IP 地址。	防火墙请求提交者完成初步分析，以了解所需防火墙端口和 IP 地址。完成后，他们会要求	Amazon Web Services Cloud 工程师、迁移专家

Task	描述	所需技能
	您的 InfoSec 团队打开所需的端口并映射 IP 地址。	

验证防火墙请求

Task	描述	所需技能
验证防火墙信息。	AWS Cloud 工程师安排与您的 InfoSec 团队会面。在此会议期间，工程师将检查并验证防火墙请求信息。 通常情况下，防火墙请求提交者与防火墙请求者是同一个人。如果发现任何问题或提出任何建议，则此验证阶段可以根据审批者给出的反馈进行迭代。	Amazon Web Services Cloud 工程师、迁移专家
更新防火墙请求文档。	InfoSec 团队分享反馈后，将编辑、保存并重新上传防火墙请求文档。本文档在每次迭代后都会更新。 我们建议您将此文档存储在受版本控制的存储文件夹中。这意味着所有更改都会被跟踪并正确应用。	Amazon Web Services Cloud 工程师、迁移专家

提交防火墙请求

Task	描述	所需技能
提交防火墙请求。	在防火墙请求批准者批准防火墙全面批准请求后，AWS	Amazon Web Services Cloud 工程师、迁移专家

Task	描述	所需技能
	Cloud 工程师提交防火墙请求。该请求指定必须打开的端口以及映射和更新所需的 IP 地址 AWS 账户。 您可以在提交防火墙请求后提出建议或提供反馈。我们建议您自动执行此反馈流程，并通过定义的工作流机制发送任何编辑内容。	

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

将 EC2 Windows 实例提取并迁移到 AWS Managed Services 账户

由 Anil Kunapareddy (AWS) 和 Venkatramana Chintha (AWS) 编写

摘要

此模式解释了将亚马逊弹性计算云 (Amazon EC2) Windows 实例迁移和提取到亚马逊网络服务 (AWS) 托管服务 (AMS) 账户的 step-by-step过程。AMS 可帮助您更高效、更安全地管理实例。AMS 可提供操作灵活性，增强安全性和合规性，并帮助您优化容量和降低成本。

这种模式从您的 AMS 账户中迁移到暂存子网的 EC2 Windows 实例开始。多种迁移服务和工具可用于执行此任务，例如 AWS 应用程序迁移服务。

若要对 AMS 托管环境进行更改，您为特定操作或操作创建并提交更改请求 (RFC)。使用 AMS 工作负载摄取 (WIGS) RFC，您可以将实例摄取到 AMS 账户并创建自定义亚马逊机器映像（AMI）。然后，您可以通过提交另一个 RFC 来创建堆栈，从而创建 AMS 托管的 EC2 实例。 EC2 有关更多信息，请参阅 AMS 文档中的 AMS 工作负载摄取。

先决条件和限制

先决条件

- 由 AMS 托管的活跃 Amazon Web Services account
- 现有的登录区
- 在 AMS 托管的 VPC 中进行更改的权限
- 您的 AM EC2 S 账户中暂存子网中的亚马逊 Windows 实例
- 完成使用 AMS WIGS 迁移工作负载的一般先决条件
- 完成使用 AMS WIGS 迁移工作负载的 Windows 先决条件

限制

- 这种模式适用于运行 Windows 服务器的 EC2 实例。此模式不适用于运行其他操作系统 (例如 Linux) 的实例。

架构

源技术堆栈

您的 AM EC2 S 账户中暂存子网中的 Amazon Windows 实例

目标技术堆栈

由 AWS Managed Services (AMS) 管理的亚马逊 EC2 Windows 实例

目标架构

工具

Amazon Web Services

- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要使用 Amazon EC2 启动任意数量或数量的虚拟服务器，也可以进行横向扩展或扩展。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Managed Services (AMS) 通过为您的 AWS 基础设施提供持续管理，包括监控、事件管理、安全指导、补丁支持和 AWS 工作负载备份，帮助您更高效、更安全地运营。

其他服务

- PowerShell是一款在 Windows、Linux 和 macOS 上运行的微软自动化和配置管理程序。

操作说明

在实例上配置设置

Task	描述	所需技能
更改 DNS 客户端设置。	1. 在源 EC2 实例上，以管理员身份打开命令提示符，键入`gpedit.msc` ，然后按 Enter。 2. 在“本地组策略编辑器”中，导航至计算机配置、管理模板、网络、DNS 客户端。 3. 对于主 DNS 后缀，选择未配置。	迁移工程师

Task	描述	所需技能
	4. 对于主 DNS 后缀下放，选择未配置。	
更改 Windows 更新设置。	1. 在“本地组策略编辑器”中，导航至计算机配置、管理模板、Windows 组件、Windows 更新。 2. 在指定内部网 Microsoft 更新服务位置，选择未配置。 3. 在配置自动更新中，选择未配置。 4. 在自动更新检测频率中，选择未配置。 5. 关闭本地组策略编辑器。	迁移工程师
启用防火墙。	1. 在源 EC2 实例上，以管理员身份打开命令提示符，键入`services.msc` ，然后按 Enter。 2. 在 Windows Services 中，启用防火墙。 3. 关闭 Windows Service。	迁移工程师

准备 AMS WIGS 实例

Task	描述	所需技能
清理和准备实例。	1. 使用堡垒主机和本地凭据，创建与暂存子 EC2 网中实例的远程桌面协议 (RDP) 连接。	迁移工程师

Task	描述	所需技能
	2. 移除 AMS 中不需要的旧版软件、防病毒软件和备份解决方案。	

修复 sppnp.dll 文件。	1. 转到 `C:\Windows\System32\sppnp.dll` 。
	2. 将 `sppnp.dll` 重命名为 `sppnp_old.dll` 。
	3. 使用 PowerShell 和管理员凭据输入以下命令：
	`dism /online /cleanup-image /restorehealth`
	`sfc /scannnow`
	4. 重启 EC2 Windows 实例。
		迁移工程师

运行 WiG 前验证脚本。	1. 从 AMS 文档中的迁移工作负载：Windows 预摄取验证下载 Windows WIGS 预摄取验证 zip 文件 (`windows-prewings-valication.zip`)。 2. 运行 Windows WIG 前验证脚本，并验证结果。 3. 如果验证失败，请修复问题，然后重新运行验证脚本，直至验证成功。	迁移工程师

Task	描述	所需技能
创建故障安全 AMI。	WiG 前验证通过后，请按如下方式创建预摄取 AMI： 1. 选择部署、高级堆栈组件、AMI、创建。 2. 在创建过程中，添加标签 `Key=Name, Value=APPLICATION-ID_IngestReady` 。 3. 等待 AMI 创建完成后继续。 有关更多信息，请参阅 AMS 文档中的 AMI | 创建。	迁移工程师

摄取和验证实例

Task	描述	所需技能
提交 RFC，以创建工作负载摄取堆栈。	提交变更申请 (RFC)，以启动 AMS WIGS。有关说明，请参阅 AMS 文档中的工作负载摄取堆栈：创建。这将启动工作负载摄取并安装 AMS 所需的所有软件，包括备份工具、Amazon EC2 管理软件和防病毒软件。	迁移工程师
验证迁移是否成功。	工作负载摄取完成后，您可看到 AMS 托管的实例和 AMS 摄取的 AMI。 1. 使用域凭证登录 AMS 管理实例。 2. 按如下方式验证域加入：	迁移工程师

Task	描述	所需技能
	a. 在 Windows 资源管理器中，右键单击此 PC，然后选择属性。 b. 在“设备规格”部分，确认域名显示在完整设备名称中。 3. 验证源和目标磁盘驱动器。	

在目标 AMS 账户启动实例

Task	描述	所需技能
提交 RFC 以创建 EC2 堆栈。	1. 使用 Windows 实例的 AMI，按照 AMS 文档中 EC2 创建堆栈 EC2 实例中的说明为堆栈准备 RFC。在 EC2 堆栈 RFC 中，提供所有参数，包括服务器名称、标签、目标 VPC、目标子网、实例类型、目标安全组、摄取 AMI 和角色。 2. 提交 EC2 堆栈的 RFC，然后等待实例成功创建。	迁移工程师

相关资源

AWS Prescriptive Guidance

- 在 AWS Managed Services on Windows 上自动执行工作负载前摄取活动
- 使用 Python 在 AMS 中自动创建 RFC

AWS 文档

- AMS 工作负载摄取
- 迁移如何改变您的资源
- 迁移工作负载：标准流程

营销资源

- AWS 托管服务
- AWS 托管服务 FAQs
- AWS 托管服务资源
- AWS 托管服务功能

将 Couchbase 服务器数据库迁移到亚马逊 EC2

由 Subhani Shaik (AWS) 编写

摘要

此模式描述了如何将 Couchbase Server 从本地环境迁移到亚马逊弹性计算云 (Amazon EC2) 上。AWS

Couchbase Server 是一个分布式 NoSQL（JSON 文档）数据库，可提供关系数据库功能。将 Couchbase Server 数据库迁移到 AWS 可以提高可扩展性、提高性能、提高成本效益、增强安全性、简化管理和覆盖全球，从而使需要高可用性和低延迟数据访问的应用程序受益。您还可以通过 AWS 托管服务访问高级功能。

开启的 Couchbase 服务器 AWS 提供以下主要功能：

- 内存优先架构
- 高可用性、灾难恢复和负载平衡
- 多主机、多区域部署以实现最佳性能

有关主要优势的更多信息，请参阅 “其他信息” 部分和 Couchbase 网站。

先决条件和限制

先决条件

- AWS 账户 具有虚拟私有云 (VPC)、两个可用区、私有子网和一个安全组的处于活动状态。有关说明，请参阅亚马逊虚拟私有云（亚马逊 VPC）文档中的创建 VPC。
- 在源环境和目标环境之间启用了连接。有关 Couchbase Server 使用的 TCX 端口的信息，请参阅 Couchbase 文档。

架构

下图显示了将 Couchbase Server 迁移到的高级架构。 AWS

从本地 Couchbase 集群中，数据使用通过客户网关移动。AWS Direct Connect数据通过路由器和 AWS Direct Connect 路由，然后通过 AWS Virtual Private Network (AWS VPN) 网关到达 VPC。VPC

包含一个正在运行 Couchbase 服务器的 EC2 实例。该 AWS 基础设施还包括用于访问控制的AWS Identity and Access Management (IAM)、用于数据加密的 AWS Key Management Service (AWS KMS)、用于区块存储的亚马逊弹性区块存储 (Amazon EBS) B lock Store，以及用于数据存储的亚马逊简单存储服务 (Amazon S3) Service。

工具

AWS 服务

- 亚马逊弹性计算云 (Amazon EC2) 在中提供可扩展的计算容量 AWS Cloud。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- AWS Direct Connect通过标准以太网光纤电缆将您的内部网络链接到某个 AWS Direct Connect 位置。通过此连接，您可以直接创建面向公众的虚拟接口， AWS 服务 同时绕过网络路径中的互联网服务提供商。

最佳实践

- 在不同的操作平台上@@ 安装和配置 Couchbase
- 在上部署 Couchbase 服务器@@ 的最佳实践 AWS
- 创建 Couchbase 集群
- Couchbase 应用程序的@@ 性能最佳实践
- Couchbase 服务器@@ 的安全最佳实践
- Couchbase 服务器数据库的@@ 存储最佳实践

操作说明

为 Couchbase EC2 服务器部署 Amazon 实例

Task	描述	所需技能
打开 Amazon EC2 控制台。	登录AWS Management Console并打开 Amazon EC2 控制台。	DevOps 工程师，Couchbase 管理员
部署 Amazon EC2 实例。	启动与本地 Couchbase 服务器配置相匹配的 EC2 实例。有关如何部署 EC2 实例的更多信	DevOps 工程师，Couchbase 管理员

Task	描述	所需技能
	息，请参阅亚马逊 EC2 文档中的启动亚马逊 EC2 实例。	

在亚马逊上安装和配置 Couchbase 服务器 EC2

Task	描述	所需技能
安装 Couchbase 集群。	在亚马逊上安装 Couchbase 服务器之前，请查看 Couchbase 服务器部署指南。 EC2 要安装 Couchbase 服务器，请参阅 C ou chbase 服务器文档	Couchbase 管理员
配置集群。	要配置集群，请参阅 Couchbase 文档中的集群配置选项。	Couchbase 管理员

添加一个新节点并重新平衡 Couchbase 集群

Task	描述	所需技能
为 EC2 实例添加节点。	将安装了 Couchbase 的新部署 EC2 实例添加到现有的本地集群中。有关说明，请参阅 Couchbase 服务器文档中的添加节点和重新平衡。	Couchbase 管理员
重新平衡集群。	重新平衡过程使新添加的带有该 EC2 实例的节点成为 Couchbase 集群的活跃成员。有关说明，请参阅 Couch base 服务器文档中的添加节点和重新平衡	Couchbase 管理员

重新配置连接

Task	描述	所需技能
移除本地节点并重新平衡。	现在，您可以从集群中移除本地节点。移除节点后，按照再平衡过程在集群中的可用节点之间重新分配数据、索引、事件处理和查询处理。有关说明，请参阅 Couchbase 服务器文档中的移除节点并重新平衡。	Couchbase 管理员
更新连接参数。	更新应用程序的连接参数以使用新的 Amazon EC2 IP 地址，这样您的应用程序就可以连接到新节点。	Couchbase 应用程序开发人员

相关资源

- Couchbase 服务器服务
- 使用部署 Couchbase 服务器 AWS Marketplace
- Connect 到 Couchbase 服务器
- 管理存储桶
- 跨数据中心复制 (XDCR)
- Couchbase Inc. 许可协议

其他信息

主要优势

将 Couchbase 数据库迁移到 AWS 具有以下优势：

可扩展性。您可以根据需要向上或向下扩展 Couchbase 集群，而无需管理物理硬件，因此您可以轻松适应不断波动的数据量和应用程序使用情况。 AWS 提供：

- 垂直和水平缩放选项

- 全球部署能力
- 跨越负载均衡 AWS 区域
- 数据库扩展解决方案
- 内容交付优化

性能优化。 AWS 提供高性能的网络基础设施和优化的实例类型，确保您的 Couchbase 数据库实现快速数据访问和低延迟。

- 高性能计算 (HPC) 选项
- 通过 Amazon 交付全球内容 CloudFront
- 多种存储选项
- 高级数据库服务，包括亚马逊关系数据库服务 (Amazon RDS) 和亚马逊 DynamoDB
- 低延迟连接 AWS Direct Connect

成本优化。选择适当的实例类型和配置，根据您的工作负载平衡性能和成本。只需为您使用的资源付费。这样就无需管理本地硬件并利用规模 AWS Cloud 经济，从而有可能降低运营成本。

- 当您在上使用 Couchbase 时，预留实例可以帮助您提前计划并大幅降低成本。 AWS
- 自动扩展可防止过度配置，并帮助您优化利用率和成本效率。

增强安全性。受益于强大的安全功能 AWS，例如数据加密、访问控制和安全组，以帮助保护您存储在 Couchbase 中的敏感数据。其他好处：

- 责任AWS 共担模型明确区分了云安全（AWS 责任）和云安全（客户责任）。
- AWS 合规性支持主要的安全标准。
- AWS 提供高级加密选项。
- AWS Identity and Access Management (IAM) 可帮助您管理对资源的安全访问。

简化了管理。 AWS 为 Couchbase 提供托管服务，因此您可以专注于应用程序开发而不是管理底层基础架构。

全球影响力。您可以跨多个集群部署 Couchbase 集群 AWS 区域 ，为世界各地的用户实现低延迟。您可以将数据库完全部署在云端或混合环境中。借助内置的企业级安全性以及从边缘到云端的快速、高效

的数据双向同步，您可以保护您的数据。同时，您可以使用一致的编程模型来构建 Web 和移动应用程序，从而简化开发。

业务连续性：

- 数据备份和恢复。如果出现问题，您可以使用AWS Backup来确保数据弹性并便于恢复。有关灾难恢复选项，请参阅 Well-Architect AWS ed Framework 文档。
- Couchbase 多区域部署：要在多区域 AWS 环境中部署 Couchbase 数据库，您可以在中订阅 Couchbase Server AWS Marketplace，使用AWS CloudFormation模板在每个区域中创建单独的 Couchbase 集群，然后配置跨区域复制以同步跨区域的数据。此配置可确保跨多个区域的高可用性和地理冗余。有关更多信息，请参阅 Couchbase 文档 AWS Marketplace中的使用部署 Couchbase 服务器。

基础设施敏捷性：

- 快速资源调配和取消配置
- 全球基础设施覆盖面
- 根据需求@@ 自动扩展
- 基础设施即代码 (IaC) 实现一致部署
- 针对不同工作负载进行了优化的多种实例类型

创新支持：

- 访问最新技术，包括 AI/ML、物联网和分析
- 托管服务，可减少运营开销
- 现代应用程序开发实践
- 无服务器计算选项

卓越运营：

- 集中监控和记录
- 自动资源管理
- 预测性维护功能
- 增强了对资源使用情况的可见性
- 简化的部署流程

现代化机会：

- 微服务架构
- DevOps实践实施
- 云原生应用程序开发
- 旧版应用程序现代化

竞争优势：

- 缩短上市时间
- 改善了客户体验
- 数据驱动的决策
- 增强的商业智能

使用日志传送将适用于 LUW 的 Db2 迁移 EC2 到 Amazon 以减少中断时间

由 Feng Cai (AWS)、Ambarish Satarkar (AWS) 和 Saurabh Sharma (AWS) 编写

摘要

当客户将他们的 IBM Db2 for LUW（Linux、UNIX 和 Windows）工作负载迁移到亚马逊网络服务 (AWS) 时，使用带有自带许可 (BYOL EC2) 模式的亚马逊弹性计算云 (Amazon) 是最快的方法。但是，将大量数据从本地 Db2 迁移到 AWS 可能是一项挑战，尤其是在停机时间较短的情况下。许多客户尝试将停机窗口设置为 30 分钟以下，这样留给数据库本身的时间就很少了。

此模式涵盖如何使用事务日志传送在较短的中断窗口内完成 Db2 迁移。此方法适用于小端 Linux 平台上的 Db2。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 在与本地文件系统布局相匹配的 EC2 实例上运行的 Db2 实例
- 实例可以访问的亚马逊简单存储服务 (Amazon S3) Service 存储桶 EC2
- AWS 身份和访问管理 (IAM) 策略和角色，用于对 Amazon S3 进行编程调用
- Amazon EC2 和本地服务器上的时区和系统时钟同步
- 通过 AWS Site-to-SiteVPN 或 AWS Di rect Connect 连接到 AWS 的本地网络

限制

- Db2 本地实例和 Amazon EC2 必须位于同一个平台系列上。
- 必须记录 Db2 本地工作负载。若要阻止任何未记录的事务，请在数据库配置中设置 `blocknonlogged=yes`。

产品版本

- 适用于 LUW 版本 11.5.9 及更高版本的 Db2

架构

源技术堆栈

- Linux 上的 Db2 x86_64

目标技术堆栈

- Amazon EBS
- Amazon EC2
- AWS Identity and Access Management (IAM)
- Amazon S3
- AWS Site-to-Site VPN 或直接连接

目标架构

下图显示了一个在本地运行的 Db2 实例，该实例通过虚拟专用网络 (VPN) 连接到 Amazon 上的 Db2。EC2虚线代表您的数据中心和 Amazon Web Services Cloud 之间的 VPN 隧道。

工具

Amazon Web Services

- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- AWS Direct Connect 通过标准的以太网光纤电缆将内部网络链接到 Direct Connect 位置。通过此连接，您可以直接创建连接到公有 Amazon Web Services 的虚拟接口，同时绕过网络路径中的互联网服务提供商。
- 亚马逊弹性区块存储 (Amazon EBS) E lastic Block Store 提供块级存储卷，用于亚马逊弹性计算云 (Amazon) 实例。 EC2
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Site-to-Site VPN 可帮助您在您在 AWS 上启动的实例和您自己的远程网络之间传递流量。

其他工具

- db2cli 是 Db2 交互式 CLI 命令。

最佳实践

- 在目标数据库上，使用 Amazon S3 网关端点访问 Amazon S3 中的数据库备份映像和日志文件。
- 在源数据库上，使用 PrivateLink 适用于 Amazon S3 的 AWS 将数据库备份映像和日志文件发送到 Amazon S3。

操作说明

设置环境变量

Task	描述	所需技能
设置环境变量。	此模式使用以下名称： • 实例名称：db2inst1 • 数据库名称：SAMPLE 您可更改它们以适应您的环境。	数据库管理员

配置本地 Db2 服务器

Task	描述	所需技能

设置 AWS CLI。	要下载并安装最新版本的 AWS CLI，请运行以下命令：
	`$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"`
	`unzip awscliv2.zip`
		Linx 管理员

Task	描述	所需技能
	`sudo ./aws/install`	

设置 Db2 归档日志的本地目标。	要使 Amazon EC2 上的目标数据库与本地源数据库保持同步，需要从源数据库检索最新的事务日志。
	
	在此设置中，/db2logs 由 LOGARCHMETH2 在源上设置作为暂存区域。此目录中的存档日志将同步到 Amazon S3 中，并由亚马逊上的 Db2 访问。 EC2使用 LOGARCHMETH2 模式，原因是 LOGARCHMETH1 可能已配置为使用 AWS CLI 命令无法访问的第三方供应商工具。要检索日志，请运行以下命令：
	
	`db2 connect to sample`
	`db2 update db cfg for SAMPLE using LOGARCHMETH2 disk:/db2logs`
		数据库管理员

运行在线数据库备份。	运行在线数据库备份，并将其保存到本地备份文件系统： `db2 backup db sample online to /backup`	数据库管理员

设置 S3 存储桶和 IAM policy

Task	描述	所需技能
创建 S3 存储桶。	为本地服务器创建 S3 存储桶，以便将备份 Db2 映象和日志文件发送到 AWS 上。Amazon 也将访问该存储桶 EC2： `aws s3api create-bucket --bucket logshipmig-db2 --region us-east-1`	AWS 系统管理员
创建一个 IAM 策略。	该`db2bucket.json` 文件包含访问 Amazon S3 存储桶的 IAM 策略： `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "kms:GenerateDataKey", "kms:Decrypt", "s3:PutObject", "s3:GetObject", "s3:AbortMultipartUpload", "s3:ListBucket",`	AWS 管理员、AWS 系统管理员

Task	描述	所需技能

	<pre>"s3:DeleteObject",
	
	"s3:GetObjectVersion",
	
	"s3:ListMultipartUploadParts"
	],
	"Resource":
	[
	
	"arn:aws:s3:::logshipmig-db2/*",
	
	"arn:aws:s3:::logshipmig-db2"
	]
	}
	]
	}</pre>
	要创建策略，请使用以下 AWS CLI 命令：
	
	<pre>aws iam create-policy \
	--policy-name
	db2s3policy \
	--policy-document
	file://db2bucket.json</pre>
	JSON 输出显示了策略的亚马逊资源名称 (ARN)，其中`aws_account_id` 代表您的账户 ID：	

Task	描述	所需技能
	`"Arn": "arn:aws:iam::aws_account_id:policy/db2s3policy"`	
将 IAM 策略附加到 EC2 实例使用的 IAM 角色。	在大多数 AWS 环境中，正在运行的 EC2 实例都由您的系统管理员设置的 IAM 角色。如果未设置 IAM 角色，请创建该角色并在 EC2 控制台上选择 Modify IAM 角色以将该角色与托管 Db2 数据库的 EC2 实例相关联。将 IAM 策略附加到带有策略的 IAM 角色 ARN： `aws iam attach-role-policy \ --policy-arn "arn:aws:iam::aws_account_id:policy/db2s3policy" \ --role-name db2s3role` 附加策略后，任何与 IAM 角色关联的 EC2 实例都可以访问 S3 存储桶。	AWS 管理员、AWS 系统管理员

将源数据库备份映象和日志文件发送到 Amazon S3

Task	描述	所需技能
在本地 Db2 服务器上配置 AWS CLI。	使用在前面的步骤中`Secret Access Key`生成的`Access Key ID`和配置 AWS CLI：	AWS 管理员、AWS 系统管理员

Task	描述	所需技能

	`$ aws configure`
	`AWS Access Key ID`
	` [None]: **********`
	`AWS Secret Access Key`
	` [None]: ********`
	`**********`
	`Default region name`
	` [None]: us-east-1`
	`Default output format`
	` [None]: json`	

将备份映象发送到 Amazon S3。	早些时候，联机数据库备份已保存至 /backup 本地目录中。要将该备份映像发送到 S3 存储桶，请运行以下命令：
	`aws s3 sync /backup`
	` s3://logshipmig-db2/`
	`SAMPLE_backup`
		AWS 管理员、迁移工程师

将 Db2 存档日志发送至 Amazon S3。	将本地 Db2 存档日志与 Amazon 上的目标 Db2 实例可以访问的 S3 存储桶同步：EC2
	`aws s3 sync /db2logs`
	` s3://logshipmig-db2/`
	`SAMPLE_LOG`
	使用 cron 或其他计划工具定期以运行此命令。频率取决于源数据库归档事务日志文件的频率。
		AWS 管理员、迁移工程师

将亚马逊上的 Db2 连接 EC2 到 Amazon S3 并开始数据库同步

Task	描述	所需技能

创建 PKCS12 密钥库。	Db2 使用公钥加密标准 (PKCS) 加密密钥库，保障 AWS 访问密钥的安全。创建密钥库并配置源 Db2 实例以使用它：
	
	`gsk8capicmd_64 -keydb -create -db "/home/db2inst1/.keystore/db2s3.p12" -pw "<password>" -type pkcs12 -stash`
	
	`db2 "update dbm cfg using keystore_location /home/db2inst1/.keystore/db2s3.p12 keystore_type pkcs12"`
		数据库管理员

创建 Db2 存储访问别名。	要创建存储访问别名，请使用以下脚本语法：
	
	`db2 "catalog storage access alias <alias_name> vendor S3 server <S3 endpoint> container '<bucket_name>'"`
	
	例如，您的脚本可能如下所示：
	
	`db2 "catalog storage access alias DB2AWSS3 vendor S3 server`
		数据库管理员

Task	描述	所需技能
	`s3.us-east-1.amazonaws.com container 'logshipmig-db2'"`	

Task	描述	所需技能

设置暂存区域。	默认情况下，Db2 使用 DB2_OBJECT_STORAGE_LOCAL_STAGING_PATH 作为过渡区域，向 Amazon S3 上传和从 Amazon S3 下载文件。默认路径位于实例主目录 sqllib/tmp/RemoteStorage.xxxx 下，xxxx 引用 Db2 分区号。请注意，暂存区域必须有足够的容量来容纳备份映象和日志文件。您可使用注册表将暂存区域指向其他目录。
	
	我们还建议使用DB2_ENABLE_COS_SDK=ON DB2_OBJECT_STORAGE_SETTINGS=EnableStreamingRestore 、和awssdk库链接绕过 Amazon S3 暂存区进行数据库备份和恢复：
	
	`#By root:`
	`cp -rp /home/db2inst1/sqllib/lib64/awssdk/RHEL/7.6/* /home/db2inst1/sqllib/lib64/`
	
	`#By db2 instance owner:`
	`db2set DB2_OBJECT_STORAGE_LOCAL_STAGING_PATH=/db2stage`
	`db2set DB2_ENABLE_COS_SDK=ON`
	`Db2set DB2_OBJECT_STORAGE_SETTINGS`
		数据库管理员

Task	描述	所需技能

	`=EnableStreamingRestore`
	`db2stop`
	`db2start`	

从备份映象恢复数据库。	EC2 从 S3 存储桶中的备份映像还原 Amazon 上的目标数据库： `db2 restore db sample from DB2REMOTE://DB2AWSS3/logshipmig-db2/SAMPLE_backup replace existing`	数据库管理员

Task	描述	所需技能

前滚数据库。	恢复完成后，目标数据库进入向前滚挂起状态。配置LOGARCHMETH1 和，以LOGARCHMETH2 便 Db2 知道从何处获取事务日志文件：
	`db2 update db cfg for SAMPLE using LOGARCHMETH1 'DB2REMOTE://DB2AWSS3//SAMPLE_LOGS/'`
	`db2 update db cfg for SAMPLE using LOGARCHMETH2 OFF`
	启动数据库向前滚动：
	`db2 ROLLFORWARD DATABASE sample to END OF LOGS`
	此命令处理已传输至 S3 存储桶的所有日志文件。根据本地 Db2 服务器上 s3 sync 命令的频率定期运行该命令。例如，如果 s3 sync 每小时运行一次，并且同步所有日志文件需要 10 分钟，则将命令设置为每小时后运行 10 分钟。
		数据库管理员

在转换窗口期间，让 Amazon 上的 Db2 上 EC2 线

Task	描述	所需技能
将目标数据库联机。	在割接窗口中，执行下列操作之一：	数据库管理员

Task	描述	所需技能

	• 将本地数据库放入 ADMIN MODE，然后运行 s3 sync 命令强制存档最后一个事务日志。
	• 关闭数据库。
	
	将最后一个事务日志同步到 Amazon S3 后，最后一次运行该ROLLFORWARD 命令：
	
	`db2 rollforward DB sample to END OF LOGS`
	`db2 rollforward DB sample complete`
	
	`Rollforward Status`
	`....`
	`Rollforward status = not pending`
	`....`
	`DB20000I The ROLLFORWARD command completed successfully.`
	
	`db2 activate db sample`
	`DB20000I The ACTIVATE DATABASE command completed successfully.`
	
	将目标数据库联机，并将应用程序连接指向 Amazon EC2 上的 Db2。	

故障排除

事务	解决方案
如果多个数据库在不同的主机 (DEV、QA、PROD) 上具有相同的实例名和数据库名称，则备份和日志可能会进入同一个子目录。	为 DEV、QA 和 PROD 使用不同的 S3 存储桶，并添加主机名作为子目录前缀以避免混淆。
如果在同一位置有多个备份映像，则恢复时会出现以下错误： `SQL2522N More than one backup file matches the time stamp value provided for the backed up database image.`	在`restore`命令中，添加备份的时间戳： `db2 restore db sample from DB2REMOTE://DB2AWSS3/logshipmig-db2/SAMPLE_backup taken at 20230628164042 replace existing`

相关资源

- 不同操作系统和硬件平台之间的 Db2 备份与恢复操作
- 设置 Db2 存储访问别名和远程 DB2
- db2 前滚命令
- Db2 辅助日志存档方法

将适用于 LUW 的 Db2 迁移到 EC2 具有高可用性灾难恢复功能的 Amazon

由 Feng Cai (AWS)、Aruna Gangireddy (AWS) 和 Venkatesan Govindan (AWS) 编写

摘要

当客户将 IBM Db2 LUW（Linux、UNIX 和 Windows）工作负载迁移到亚马逊网络服务 (AWS) 时，使用带有自带许可 (BYOL EC2) 模式的亚马逊弹性计算云 (Amazon) 是最快的方法。但是，将大量数据从本地 Db2 迁移到 AWS 可能是一项挑战，尤其是在停机时间较短的情况下。许多客户尝试将停机窗口设置为 30 分钟以下，这样留给数据库本身的时间就很少了。

此模式涵盖如何使用 Db2 高可用性灾难恢复 (HADR) 在较短的中断窗口内完成 Db2 迁移。此方法适用于小端 Linux 平台上且不使用数据分区功能 (DPF) 的 Db2 数据库。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 在 Amazon 实例上运行的 Db2 EC2 实例，该实例与本地文件系统布局相匹配
- 实例可以访问的亚马逊简单存储服务 (Amazon S3) Service 存储桶 EC2
- AWS 身份和访问管理 (IAM) 策略和角色，用于对 Amazon S3 进行编程调用
- Amazon EC2 和本地服务器上的时区和系统时钟同步
- 通过 AWS Site-to-SiteVPN 或 AWS Di rect Connect 连接到 AWS 的本地网络
- 本地服务器与 Amazon 之间通过 HADR EC2 端口进行通信

限制

- Db2 本地实例和 Amazon EC2 必须位于同一个平台系列上。
- 分区数据库环境不支持 HADR。
- HADR 不支持对数据库日志文件使用原始 I/O（直接磁盘访问）。
- HADR 不支持无限的日志记录。
- `LOGINDEXBUILD` 必须设置为 `YES`，这将增加重建索引的日志使用量。
- 必须记录 Db2 本地工作负载。在数据库配置中设置 `blocknonlogged=yes` 以阻止任何未记录的事务。

产品版本

- 适用于 LUW 版本 11.5.9 及更高版本的 Db2

架构

源技术堆栈

- Linux 上的 Db2 x86_64

目标技术堆栈

- Amazon EC2
- AWS Identity and Access Management (IAM)
- Amazon S3
- AWS Site-to-Site VPN

目标架构

在下图中，本地 Db2 作为主服务器在 `db2-server1` 运行。它包含两个 HADR 备用目标。一个备用目标位于本地，并且是可选。另一个备用目标`db2-ec2`在亚马逊上 EC2。将数据库切换到 AWS 后，`db2-ec2`将成为主数据库。

1. 日志从主本地数据库流式传输至备用本地数据库。
2. 使用 Db2 HADR，日志通过 Site-to-Site VPN 从主本地数据库流式传输到 Amazon 上的 Db2。 EC2
3. Db2 备份和存档日志从主本地数据库发送至 AWS 上的 S3 存储桶。

工具

Amazon Web Services

- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- AWS Direct Connect 通过标准的以太网光纤电缆将内部网络链接到 Direct Connect 位置。通过此连接，您可以直接创建连接到公有 Amazon Web Services 的虚拟接口，同时绕过网络路径中的互联网服务提供商。

- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Site-to-Site VPN 可帮助您在您在 AWS 上启动的实例和您自己的远程网络之间传递流量。

其他工具

- db2cli 是 Db2 交互式 CLI 命令。

最佳实践

- 在目标数据库上，使用 Amazon S3 网关端点访问 Amazon S3 中的数据库备份映像和日志文件。
- 在源数据库上，使用 PrivateLink 适用于 Amazon S3 的 AWS 将数据库备份映像和日志文件发送到 Amazon S3。

操作说明

设置环境变量

Task	描述	所需技能
设置环境变量。	此模式使用以下名称和端口： 1. Db2 本地主机名：`db2-server1` 2. HADR 备用主机名：`db2-server2` （如果 HADR 当前在本地运行） 3. Amazon EC2 主机名：`db2-ec2` 4. 实例名称：`db2inst1` 5. 数据库名称：`SAMPLE`	数据库管理员

Task	描述	所需技能
	6. HADR 端口： • db2-server1: 50010 • db2-server2: 50011 • db2-ec2: 50012 您可更改它们以适应您的环境。	

配置本地 Db2 服务器

Task	描述	所需技能

设置 AWS CLI。	要下载并安装最新版本的 AWS CLI，请运行以下命令：
	
	`$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"`
	`unzip awscliv2.zip`
	`sudo ./aws/install`
		Linx 管理员

设置 Db2 归档日志的本地目标。	大量更新批处理作业和网络速度减慢等情况可能会导致 HADR 备用服务器出现延迟。为了赶上进度，备用服务器需要来自主服务器的事务日志。请求日志位置顺序如下： • 主服务器上的活动日志目录 • 备用服务器上的 LOGARCHMETH1 或 LOGARCHMETH2 位置	数据库管理员

Task	描述	所需技能

	• 主服务器上的 LOGARCHMETH1 或 LOGARCHMETH2 位置
	
	在此设置中，/db2logs 由 LOGARCHMETH2 在源上设置作为暂存区域。此目录中的存档日志将同步到 Amazon S3 中，并由亚马逊上的 Db2 访问。 EC2LOGARCHMETH2 之所以使用该模式，是因为LOGARCHMETH1 可能已配置为使用 AWS CLI 命令无法访问的第三方供应商工具：
	
	`db2 connect to sample`
	`db2 update db cfg for SAMPLE using LOGARCHMETH2 disk:/db2logs`	

运行在线数据库备份。	运行在线数据库备份，并将其保存到本地备份文件系统： `db2 backup db sample online to /backup`	数据库管理员

设置 S3 存储桶和 IAM policy

Task	描述	所需技能
创建 S3 存储桶。	为本地服务器创建 S3 存储桶，以便将备份 Db2 映象和日志文件发送到 AWS	AWS 管理员

Task	描述	所需技能
	上。Amazon 将访问该存储桶 EC2： `aws s3api create-bucket --bucket hadrmig-db2 --region us-east-1`	

Task	描述	所需技能
创建一个 IAM 策略。	该db2bucket.json 文件包含用于访问 S3 存储桶的 IAM 策略： { "Version": "2012-10-17", "Statement": [{ "Effect": "Allow", "Action": ["kms:GenerateDataKey", "kms:Decrypt", "s3:PutObject", "s3:GetObject", "s3:AbortMultipartUpload", "s3:ListBucket", "s3:DeleteObject", "s3:GetObjectVersion", "s3:ListMultipartUploadParts"], "Resource": ["arn:aws:s3:::hadrmig-db2/*",	AWS 管理员、AWS 系统管理员

Task	描述	所需技能

	<pre>"arn:aws:s3:::hadr
	mig-db2"
	]
	}
	]
	}</pre>
	要创建策略，请使用以下 AWS CLI 命令：
	<pre>aws iam create-policy \
	--policy-name
	db2s3hapolicy \
	--policy-document
	file://db2bucket.j
	son</pre>
	JSON 输出显示了策略的亚马逊资源名称 (ARN)，其中`aws_account_id` 代表您的账户 ID：
	<pre>"Arn": "arn:aws:
	iam::aws_account_i
	d:policy/db2s3hapo
	licy"</pre>	

Task	描述	所需技能
将 IAM 策略附加到 IAM 角色。	通常，运行 Db2 的 EC2 实例将具有由系统管理员分配的 IAM 角色。如果未分配 IAM 角色，则可以在亚马逊 EC2 控制台上选择修改 IAM 角色。 将 IAM 策略附加到与 EC2 实例关联的 IAM 角色。附加策略后，EC2 实例可以访问 S3 存储桶： `aws iam attach-role-policy --policy-arn "arn:aws:iam::aws_account_id:policy/db2s3hapolicy" --role-name db2s3harole`	

将源数据库备份映象和日志文件发送到 Amazon S3

Task	描述	所需技能

在本地 Db2 服务器配置 AWS CLI。	使用您之前生成的Access Key ID和Secret Access Key配置 AWS CLI：
	
	`$ aws configure`
	`AWS Access Key ID [None]: **********`
	`AWS Secret Access Key [None]: ********** **********`
	`Default region name [None]: us-east-1`
	`Default output format [None]: json`
		AWS 管理员、AWS 系统管理员

Task	描述	所需技能
将备份映象发送到 Amazon S3。	早些时候，联机数据库备份已保存至 /backup 本地目录中。要将该备份映像发送到 S3 存储桶，请运行以下命令： `aws s3 sync /backup s3://hadrmig-db2/SAMPLE_backup`	AWS 管理员、AWS 系统管理员
将 Db2 存档日志发送至 Amazon S3。	将本地 Db2 存档日志与亚马逊上的目标 Db2 实例可以访问的 Amazon S3 存储桶同步： EC2 `aws s3 sync /db2logs s3://hadrmig-db2/SAMPLE_LOGS` 使用 cron 或其他计划工具定期以运行此命令。频率取决于源数据库归档事务日志文件的频率。	

将亚马逊上的 Db2 连接 EC2 到 Amazon S3 并开始初始数据库同步

Task	描述	所需技能
创建 PKCS12 密钥库。	Db2 使用公钥加密标准 (PKCS) 加密密钥库，保障 AWS 访问密钥的安全。创建密钥库，并将源 Db2 配置为使用它： `gsk8capicmd_64 -keydb -create -db "/home/db2inst1/.keystore/d`	数据库管理员

Task	描述	所需技能

	`b2s3.p12" -pw "<password>" -type pkcs12 -stash`
	
	`db2 "update dbm cfg using keystore_location /home/db2inst1/.keystore/db2s3.p12 keystore_type pkcs12"`	

创建 Db2 存储访问别名。	Db2 使用存储访问别名通过 INGEST、LOAD、BACKUP DATABASE 或 RESTORE DATABASE 命令直接访问 Amazon S3。
	
	因为您为 EC2 实例分配了 IAM 角色，USER并且PASSWORD不是必需的：
	
	`db2 "catalog storage access alias <alias_name> vendor S3 server <S3 endpoint> container '<bucket_name>'"`
	
	例如，您的脚本可能如下所示：
	
	`db2 "catalog storage access alias DB2AWSS3 vendor S3 server s3.us-east-1.amazonaws.com container 'hadrmig-db2'"`
		数据库管理员

Task	描述	所需技能

设置暂存区域。	我们建议使用DB2_ENABLE_COS_SDK=ON DB2_OBJECT_STORAGE_SETTINGS=EnableStreamingRestore 、和awssdk库链接绕过 Amazon S3 暂存区进行数据库备份和恢复：
	`#By root:`
	`cp -rp /home/db2inst1/sqllib/lib64/awssdk/RHEL/7.6/* /home/db2inst1/sqllib/lib64/`
	
	`#By db2 instance owner:`
	`db2set DB2_OBJECT_STORAGE_LOCAL_STAGING_PATH=/db2stage`
	`db2set DB2_ENABLE_COS_SDK=ON`
	`db2set DB2_OBJECT_STORAGE_LOCAL_STAGING_PATH=/db2stage`
	`db2stop`
	`db2start`
		数据库管理员

Task	描述	所需技能

从备份映象恢复数据库。	EC2 从 S3 存储桶中的备份映像还原 Amazon 上的目标数据库：
	
	`db2 create db sample on /data1`
	`db2 restore db sample from DB2REMOTE://DB2AWSS3/hadrmig-db2/SAMPLE_backup replace existing`
		数据库管理员

在本地无 HADR 的情况下设置 HADR

Task	描述	所需技能
配置本地 Db2 服务器为主服务器。	将 db2-server1 （本地源）的 HADR 的数据库配置设置为主数据库。设置HADR_SYNCMODE 为交易响应时间最短的SUPERASYNC 模式： `db2 update db cfg for sample using HADR_LOCAL_HOST db2-server1 HADR_LOCAL_SVC 50010 HADR_REMOTE_HOST db2-ec2 HADR_REMOTE_SVC 50012 HADR_REMOTE_INST db2inst1 HADR_SYNCMODE SUPERASYNC DB20000I The UPDATE DATABASE CONFIGURATION command`	数据库管理员

Task	描述	所需技能
	`completed successfully` 本地数据中心与 AWS 之间预计会有一些网络延迟。(您可根据网络可靠性设置不同 `HADR_SYNCMODE` 值。有关更多信息，请参阅相关资源部分)。	

更改目标数据库日志归档目标。	更改目标数据库日志存档目标以匹配 Amazon EC2 环境：
	
	`db2 update db cfg for SAMPLE using LOGARCHMETH1 'DB2REMOTE://DB2AWSS3//SAMPLE_LOGS/' LOGARCHMETH2 OFF`
	`DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully`
		数据库管理员

Task	描述	所需技能
在亚马逊 EC2 服务器上为 Db2 配置 HADR。	更新处于待机状态的 HADR db2-ec2 的数据库配置： `db2 update db cfg for sample using HADR_LOCAL_HOST db2-ec2 HADR_LOCAL_SVC 50012 HADR_REMOTE_HOST db2-server1 HADR_REMOTE_SVC 50010 HADR_REMOTE_INST db2inst1 HADR_SYNCMODE SUPERASYNC DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully`	数据库管理员

Task	描述	所需技能

验证 HADR 设置。	验证源 Db2 服务器和目标 Db2 服务器上的 HADR 参数。
	
	要验证是否已启db2-server1 用，请运行以下命令：
	
	`db2 get db cfg for sample|grep HADR`
	`HADR database role`
	
	`= PRIMARY`
	`HADR local host name (HADR_LOCAL_HOST) = db2-server1`
	`HADR local service name (HADR_LOCAL_SVC) = 50010`
	`HADR remote host name (HADR_REMOTE_HOST) = db2-ec2`
	`HADR remote service name (HADR_REMOTE_SVC) = 50012`
	`HADR instance name of remote server (HADR_REMOTE_INST) = db2inst1`
	`HADR timeout value`
	
	`(HADR_TIMEOUT) = 120`
	`HADR target list (HADR_TARGET_LIST) =`
	`HADR log write synchronization mode (HADR_SYNCMODE) = NEARSYNC`
	`HADR spool log data limit (4KB)`
		数据库管理员

Task	描述	所需技能
	(HADR_SPOOL_LIMIT) = AUTOMATIC(52000) HADR log replay delay (seconds) (HADR_REPLAY_DELAY) = 0 HADR peer window duration (seconds) (HADR_PEER_WINDOW) = 0 HADR SSL certificate label (HADR_SSL_LABEL) = HADR SSL Hostname Validation (HADR_SSL_HOST_VAL) = OFF 要验证是否已启db2-ec2用，请运行以下命令： db2 get db cfg for sample|grep HADR HADR database role = STANDBY HADR local host name (HADR_LOCAL_HOST) = db2-ec2 HADR local service name (HADR_LOCAL_SVC) = 50012 HADR remote host name (HADR_REMOTE_HOST) = db2-server1 HADR remote service name (HADR_REMOTE_SVC) = 50010 HADR instance name of remote server	

Task	描述	所需技能
	(HADR_REMOTE_INST) = db2inst1 HADR timeout value (HADR_TIMEOUT) = 120 HADR target list (HADR_TARGET_LIST) = HADR log write synchronization mode (HADR_SYNCMODE) = SUPERASYNC HADR spool log data limit (4KB) (HADR_SPOOL_LIMIT) = AUTOMATIC(52000) HADR log replay delay (seconds) (HADR_REPLAY_DELAY) = 0 HADR peer window duration (seconds) (HADR_PEER_WINDOW) = 0 HADR SSL certificate label (HADR_SSL_LABEL) = HADR SSL Hostname Validation (HADR_SSL_HOST_VAL) = OFF HADR_LOCAL_HOST 、HADR_LOCAL_SVC 、HADR_REMOTE_HOST 、和 HADR_REMOTE_SVC 参数表示一个主 HADR 设置和一个备用 HADR 设置。	

Task	描述	所需技能

启动 Db2 HADR 实例。	首先在备用服务器db2-ec2上启动 Db2 HADR 实例：
	`db2 start hadr on db sample as standby`
	`DB20000I The START HADR ON DATABASE command completed successfully.`
	在主（源）服务器上启动 Db2 HADR：db2-server1
	`db2 start hadr on db sample as primary`
	`DB20000I The START HADR ON DATABASE command completed successfully.`
	本地和亚马逊上的 Db2 之间的 HADR 连接现 EC2 已成功建立。Db2 主服务器 db2-server1 开始将事务日志记录实时传输至 db2-ec2。
		数据库管理员

当本地有 HADR 时设置 HADR

Task	描述	所需技能
在 Amazon 上添加 Db2 EC2 作为辅助备用服务器。	如果 HADR 正在本地 Db2 实例上运行，则可以通过在上运行以下命令将 Amazon 上的 Db2 添加 EC2 为辅助备用HADR_TARGET_LIST 实例：db2-ec2	数据库管理员

Task	描述	所需技能
	db2 update db cfg for sample using HADR_LOCAL_HOST db2-ec2 HADR_LOCAL_SVC 50012 HADR_REMOTE_HOST db2-server1 HADR_REMOTE_SVC 50010 HADR_REMOTE_INST db2inst1 HADR_SYNCMODE SUPERASYNC DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully. db2 update db cfg for sample using HADR_TARGET_LIST "db2-server1:50010|db2-server2:50011" DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.	

Task	描述	所需技能
将辅助备用信息添加至本地服务器。	更新 HADR_TARGET_LIST 两台本地服务器（主服务器和备用服务器）。 开启db2-server1 ，运行以下代码： db2 update db cfg for sample using HADR_TARGET_LIST "db2-server2:50011|db2-ec2:50012" DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully. SQL1363W One or more of the parameters submitted for immediate modification were not changed dynamically. For these configuration parameters, the database must be shutdown and reactivated before the configuration parameter changes become effective. 开启db2-server2 ，运行以下代码： db2 update db cfg for sample using HADR_TARGET_LIST "db2-serv	数据库管理员

Task	描述	所需技能
	er1:50010|db2-ec2: 50012" DB20000I The UPDATE DATABASE CONFIGURATION command completed successfu lly. SQL1363W One or more of the parameter s submitted for immediate modificat ion were not changed dynamically. For these configura tion parameters, the database must be shutdown and reactivated before the configuration parameter changes become effective.	

Task	描述	所需技能

验证 HADR 设置。	验证源 Db2 服务器和目标 Db2 服务器上的 HADR 参数。
	
	开启db2-server1 ，运行以下代码：
	
	`db2 get db cfg for sample|grep HADR`
	`HADR database role = PRIMARY`
	`HADR local host name (HADR_LOCAL_HOST) = db2-server1`
	`HADR local service name (HADR_LOCAL_SVC) = 50010`
	`HADR remote host name (HADR_REMOTE_HOST) = db2-server2`
	`HADR remote service name (HADR_REMOTE_SVC) = 50011`
	`HADR instance name of remote server (HADR_REMOTE_INST) = db2inst1`
	`HADR timeout value (HADR_TIMEOUT) = 120`
	`HADR target list (HADR_TARGET_LIST) = db2-server2:50011|db2-ec2:50012`
	`HADR log write synchronization mode`	

Task	描述	所需技能
	(HADR_SYNCMODE) = NEARSYNC HADR spool log data limit (4KB) (HADR_SPOOL_LIMIT) = AUTOMATIC(52000) HADR log replay delay (seconds) (HADR_REPLAY_DELAY) = 0 HADR peer window duration (seconds) (HADR_PEER_WINDOW) = 0 HADR SSL certificate label (HADR_SSL_LABEL) = HADR SSL Hostname Validation (HADR_SSL_HOST_VAL) = OFF 开启db2-server2 ，运行以下代码： db2 get db cfg for sample|grep HADR HADR database role = STANDBY HADR local host name (HADR_LOCAL_HOST) = db2-server2 HADR local service name (HADR_LOCAL_SVC) = 50011 HADR remote host name (HADR_REMOTE_HOST) = db2-server1 HADR remote service name	

Task	描述	所需技能
	(HADR_REMOTE_SVC) = 50010 HADR instance name of remote server (HADR_REMOTE_INST) = db2inst1 HADR timeout value (HADR_TIMEOUT) = 120 HADR target list (HADR_TARGET_LIST) = db2-server1:50010|db2-ec2:50012 HADR log write synchronization mode (HADR_SYNCMODE) = NEARSYNC HADR spool log data limit (4KB) (HADR_SPOOL_LIMIT) = AUTOMATIC(52000) HADR log replay delay (seconds) (HADR_REPLAY_DELAY) = 0 HADR peer window duration (seconds) (HADR_PEER_WINDOW) = 0 HADR SSL certificate label (HADR_SSL_LABEL) = HADR SSL Hostname Validation (HADR_SSL_HOST_VAL) = OFF 开启db2-ec2，运行以下代码： db2 get db cfg for sample|grep HADR	

Task	描述	所需技能
	HADR database role = STANDBY HADR local host name (HADR_LOCAL_HOST) = db2-ec2 HADR local service name (HADR_LOCAL_SVC) = 50012 HADR remote host name (HADR_REMOTE_HOST) = db2-server1 HADR remote service name (HADR_REMOTE_SVC) = 50010 HADR instance name of remote server (HADR_REMOTE_INST) = db2inst1 HADR timeout value (HADR_TIMEOUT) = 120 HADR target list (HADR_TARGET_LIST) = db2-server1:50010|db2-server2:50011 HADR log write synchronization mode (HADR_SYNCMODE) = SUPERASYNC HADR spool log data limit (4KB) (HADR_SPOOL_LIMIT) = AUTOMATIC(52000) HADR log replay delay (seconds) (HADR_REPLAY_DELAY) = 0	

Task	描述	所需技能
	HADR peer window duration (seconds) (HADR_PEER_WINDOW) = 0 HADR SSL certificate label (HADR_SSL_LABEL) = HADR SSL Hostname Validation (HADR_SSL_HOST_VAL) = OFF HADR_LOCAL_HOST 、HADR_LOCAL_SVC 、HADR_REMOTE_HOST 、HADR_REMOTE_SVC 、HADR_TARGET_LIST 参数表示一个主 HADR 设置和两个备用 HADR 设置。	

Task	描述	所需技能

停止和启动 Db2 HADR。	HADR_TARGET_LIST 现在已在所有三台服务器上进行设置。每个 Db2 服务器都了解其他两个。停止并重启 HADR（短暂停机）以利用新配置。
	
	开启db2-server1 ，运行以下命令：
	
	`db2 stop hadr on db sample`
	`db2 deactivate db sample`
	`db2 activate db sample`
	
	开启db2-server2 ，运行以下命令：
	
	`db2 deactivate db sample`
	`db2 start hadr on db sample as standby`
	`SQL1766W The command completed successfully`
	
	开启db2-ec2，运行以下命令：
	
	`db2 start hadr on db sample as standby`
	`SQL1766W The command completed successfully`
	
	开启db2-server1 ，运行以下命令：
		数据库管理员

Task	描述	所需技能

	`db2 start hadr on db sample as primary`
	`SQL1766W The command completed successfully`
	
	本地和亚马逊上的 Db2 之间的 HADR 连接 EC2 现已成功建立。Db2 主服务器 `db2-server1` 开始将事务日志记录实时流式传输至 `db2-server2` 和 `db2-ec2`。	

在转换窗口期间，将 Amazon 上的 Db2 设 EC2 为主数据库

Task	描述	所需技能

确保备用服务器上无 HADR 延迟。	从主服务器 `db2-server1` 检查 HADR 状态。`HADR_STATE` 处于 `REMOTE_CATCHUP` 状态时不要惊慌，当 `HADR_SYNCMODE` 设置为 `SUPERASYNC` 时，这是正常的。an `PRIMARY_LOG_TIME` d `STANDBY_REPLAY_LOG_TIME` 显示它们处于同步状态：
	
	`db2pd -hadr -db sample`
	
	`HADR_ROLE = PRIMARY`
	
	`REPLAY_TYPE = PHYSICAL`
	
	`HADR_SYNCMODE = SUPERASYNC`
		数据库管理员

Task	描述	所需技能

	<pre> STANDBY_ID = 2
	
	LOG_STREAM_ID = 0
	
	      HADR_STATE =
	REMOTE_CATCHUP
	.....
	
	PRIMARY_LOG_TIME =
	10/26/2022 02:11:32.
	000000 (1666750292)
	
	STANDBY_LOG_TIME =
	10/26/2022 02:11:32.
	000000 (1666750292)
	             STANDBY_R
	EPLAY_LOG_TIME =
	10/26/2022 02:11:32.
	000000 (1666750292)</pre>	

Task	描述	所需技能

运行 HADR 接管。	若要完成迁移，请运行 HADR takeover 命令创建 db2-ec2 主数据库。使用命令验证db2pd该HADR_ROLE 值：
	
	`db2 TAKEOVER HADR ON DATABASE sample`
	`DB20000I The TAKEOVER HADR ON DATABASE command completed successfully.`
	
	`db2pd -hadr -db sample`
	`Database Member 0 -- Database SAMPLE -- Active -- Up 0 days 00:03:25 -- Date 2022-10-26-02.46.45.048988`
	
	`HADR_ROLE = PRIMARY`
	`REPLAY_TYPE = PHYSICAL`
	
	要完成向 AWS 的迁移，请将应用程序连接指向 Amazon EC2 上的 Db2。	

故障排除

事务	解决方案
如果您出于防火墙和安全原因使用 NAT，则主机可能有两个 IP 地址（一个内部地址和一个外部地址），这可能会导致 HADR IP 地址检查失	要在 NAT 环境中支持 HADR，您可使用内部地址和外部地址 HADR_LOCAL_HOST 进行配置。例如，如果 Db2 服务器有内部名称 host1 和

事务	解决方案
败。该START HADR ON DATABASE命令将返回以下消息： HADR_LOCAL_HOST:HADR_LOCAL_SVC (-xx-xx-xx-xx.:50011 (xx.xx.xx.xx:50011)) on remote database is different from HADR_REMOTE_HOST:HADR_REMOTE_SVC (xx-xx-xx-xx.:50011 (x.x.x.x:50011)) on local database.	外部名称 host1E，则 HADR_LOCAL_HOST 可以是 HADR_LOCAL_HOST: "host1 | host1E"。

相关资源

- 不同操作系统和硬件平台之间的 Db2 备份与恢复操作
- 设置 Db2 存储访问别名和远程 DB2
- Db2 高可用性灾难恢复
- hadr_syncmode — 对等状态配置参数中日志写入的 HADR 同步模式

使用 Pow VMware VMs erCLI 通过 HCX 自动化进行迁移

由 Giri Nadiminty (AWS)、Hassan Adekoya (AWS) 和 Naveen Deshwal 编写

摘要

注意：自 2024 年 4 月 30 日起，VMware Cloud on AWS 不再由其渠道 AWS 合作伙伴转售。该服务将继续通过博通提供。我们鼓励您联系您的 AWS 代表了解详情。

此模式描述了如何使用由 VMware PowerCLI 脚本支持的 VMware 混合 VMware 云扩展 (VMsHCX) 自动化将 VMware 本地虚拟机 () 迁移到 AWS 上的云。PowerCLI 是一款基于 Windows 的命令行工具。PowerShell它可以帮助您管理 VMware 软件，并自动执行基础架构和迁移任务。

您可以调整这种模式，以便在 vCenter、软件定义的数据中心 (SDDCs) 和云环境的任意组合之间进行迁移。此模式中包含的 PowerCLI 脚本使用自动化（而不是鼠标点击）执行所有虚拟机配置和调度任务，因此它们可以节省迁移活动的时间，并有助于降低人为错误的风险。

先决条件和限制

先决条件

- 带有 SDDC 的 AWS VMware 云账户
- 现有本地或基于云的 vCenter 或 SDDC
- 具有源和目标 vCenter 必要权限的用户帐户或 SDDCs
- 在源和@@ 目标 vCenter 之间配置的 HCX 网络扩展 (HCX-NE) 的 HCX 站点配对或 SDDCs
- VMware PowerCLI 已安装在你选择的服务器上

限制

- 如果源 vCenter 使用跨 vCenter NSX，则 PowerCLI 模块将无法运行。使用带有 HCX API 的脚本方法（例如 Python）而不是 PowerCLI。
- 如果迁移 VMs 需要新的名称或 IP 地址，请使用带有 HCX API 的脚本方法（例如 Python）。
- 此模式不会填充 .csv 文件，这是必要条件。您可以使用 VMware vRealize Network Insight (vRNI) 或其他方法填充文件。

产品版本

- VMware vSphere 版本 5 或更高版本
- VMware HCX 版本 4.4 或更高版本
- VMware PowerCLI 版本 12.7 或更高版本

架构

源技术堆栈

- 本地或云端 VMware

目标技术堆栈

- VMware AWS 上的云

目标架构

工具

Amazon Web Services

- VMware Cloud on AWS 是一项由 AWS 联合设计的服务 VMware ，旨在帮助您将 VMware 基于 vSphere 的本地环境迁移和扩展到 AWS 云。

其他工具

- VMware 混合云扩展 (HCX) 是一种实用工具，用于在不更改底层平台的情况下将工作负载从本地 VMware 环境迁移到 VMware Cloud on AWS。注意：该产品以前被称为 Hybrid Cloud Extension 和 NSX Hybrid Connect。此示例使用 HCX 进行虚拟机迁移。
- VMware PowerCLI 是一款用于自动管理 vSphere 和 v VMware Cloud 的命令行工具。你可以使用 cmdlet 在 Windows PowerShell 中运行 Power PowerShell CLI 命令。此模式使用 PowerCLI 运行迁移命令。

代码

简单、独立脚本

我们建议您使用此单机脚本执行初始测试，以验证配置选项是否被接受并按预期运行。有关说明，请参阅操作说明部分。

```
<# Manual Variables #>
$HcxServer = "[enterValue]"
$SrcNetworkName = "[enterValue]"
$DstNetworkName = "[enterValue]"
$DstComputeName = "[enterValue]"
$DstDSName = "[enterValue]"
$DstFolderName = "[enterValue]"
$vmName = "[enterValue]"

<# Environment Setup #>
Connect-HCXServer -Server $HcxServer
$HcxDstSite = Get-HCXSite -Destination
$HcxSrcSite = Get-HCXSite -Source
$SrcNetwork = Get-HCXNetwork -Name $SrcNetworkName -Type VirtualWire -Site $HcxSrcSite
$DstNetwork = Get-HCXNetwork -Name $DstNetworkName -Type NsxtSegment -Site $HcxDstSite
$DstCompute = Get-HCXContainer -Name $DstComputeName -Site $HcxDstSite
$DstDS = Get-HCXDatastore -Name $DstDSName -Site $HcxDstSite
$DstFolder = Get-HCXContainer -name $DstFolderName -Site $HcxDstSite
$vm = Get-HCXVM -Name $vmName

<# Migration #>
$NetworkMapping = New-HCXNetworkMapping -SourceNetwork $SrcNetwork -DestinationNetwork
 $DstNetwork
$NewMigration = New-HCXMigration -VM $vm -MigrationType vMotion -SourceSite $HcxSrcSite
 -DestinationSite $HcxDstSite -Folder $DstFolder -TargetComputeContainer $DstCompute
 -TargetDatastore $DstDS -NetworkMapping $NetworkMapping -DiskProvisionType Thin
 -UpgradeVMTools $True -RemoveISOs $True -ForcePowerOffVm $True -RetainMac $True -
UpgradeHardware $True -RemoveSnapshots $True
```

功能齐全、基于 .csv 的脚本

测试完成后，您可在生产环境中使用以下脚本。有关说明，请参阅操作说明部分。

```
<# Schedule #>
write-host("Getting Time for Scheduling")
$startTime = [DateTime]::Now.AddDays(12)
$endTime = [DateTime]::Now.AddDays(15)

<# Migration #>
Connect-HCXServer -Server [enterValue]
```

```
write-host("Getting Source Site")
$HcxSrcSite = Get-HCXSite
write-host("Getting Target Site")
$HcxDstSite = Get-HCXSite -Destination
$HCXVMS = Import-CSV .\Import_VM_list.csv
ForEach ($HCXVM in $HCXVMS) {
    $DstFolder = Get-HCXContainer $HCXVM.DESTINATION_VM_FOLDER -Site $HcxDstSite
    $DstCompute = Get-HCXContainer $HCXVM.DESTINATION_COMPUTE  -Site $HcxDstSite
    $DstDatastore = Get-HCXDatastore $HCXVM.DESTINATION_DATASTORE -Site $HcxDstSite
    $SrcNetwork = Get-HCXNetwork $HCXVM.SOURCE_NETWORK -Type VirtualWire -Site
 $HcxSrcSite
    $DstNetwork = Get-HCXNetwork $HCXVM.DESTINATION_NETWORK -Type NsxtSegment -Site
 $HcxDstSite
    $NetworkMapping = New-HCXNetworkMapping -SourceNetwork $SrcNetwork -
DestinationNetwork $DstNetwork
    $NewMigration = New-HCXMigration -VM (Get-HCXVM $HCXVM.VM_NAME) -MigrationType
 Bulk -SourceSite $HcxSrcSite -DestinationSite $HcxDstSite -Folder $DstFolder -
TargetComputeContainer $DstCompute -TargetDatastore $DstDatastore -NetworkMapping
 $NetworkMapping -DiskProvisionType Thin -UpgradeVMTools $True -RemoveISOs $True -
ForcePowerOffVm $True -RetainMac $True -UpgradeHardware $True -RemoveSnapshots $True -
ScheduleStartTime $startTime -ScheduleEndTime $endTime
    Start-HCXMigration -Migration $NewMigration -Confirm:$false
}
```

操作说明

收集手动变量信息

Task	描述	所需技能
查找源和目标 vCenter 和 SDDC 服务器名称。	PowerCLI 脚本需要此操作说明中描述的变量。为便于使用脚本，您可以提前收集这些信息。 在 vSphere 控制台的 HCX 部分，选择基础设施、站点配对。记录显示的源服务器和目标服务器的名称。	云架构师
查找源与目标 HCX 名称。	在 vSphere 控制台的 HCX 部分，选择系统、管理。记录显	云架构师

Task	描述	所需技能
	示的源服务器和目标 HCX 的名称。	
查找源和目标网络名称。	在 vSphere 控制台的 HCX 部分，选择系统、网络扩展。记录源和目标网络的名称 **Note** 或者，在连接到 HCX 服务器后，您可以使用 PowerCLI Get-HCXNetwork 和 Get-Destination 命令获取源和HCXNetwork目标网络名称。	云架构师
在 vSphere 控制台收集更多信息。	在 vSphere 控制台上收集以下信息： • 你 VMs 要迁移的名字 • 目标计算环境（集群/主机） • 目标数据存储 • 目标虚拟机文件夹名称	云架构师

做出迁移决定

Task	描述	所需技能
确定迁移选项。	确定了以下内容： • `MigrationType` — HCX 辅助迁移类型有 vMotion、批量、冷迁和 RAV。您的选择取决于您的停机时间要	云架构师

Task	描述	所需技能
	求、网络带宽、迁移时间范围和工作负载。有关更多信息，请参阅 AWS 博客文章使用混合 VMware 云扩展 (HCX) 在 AWS 上将工作负载迁移到云。 • `DiskProvisionType (Thin, Thick)` • `UpgradeVMTools ($True, $False)` • `RemoveISOs ($True, $False)` • `ForcePowerOffVm ($True, $False)` • `RetainMac ($True, $False)` • `UpgradeHardware ($True, $False)` • `RemoveSnapshots ($True, $False)` 有关每个选项的更多信息，请参阅VMware 开发者文档。	

运行简单的脚本，进行初始测试

Task	描述	所需技能
复制脚本。	该脚本的简单版本自包含在单个文件内。您可将其用于测试单台机器的迁移。	云架构师

Task	描述	所需技能
	复制此模式的 “代码” 部分中的第一个脚本，并将其存储在安装了 VMware PowerCLI 模块的计算机上。（要安装 PowerCLI，请按照VMware 文档中的说明进行操作。）	
设置脚本变量。	设置脚本 `Manual Variables` 部分中的所有变量。	云架构师
设置迁移变量。	在脚本 `Migration` 部分设置所有 `New-HCXMigration` 设置。	云架构师
指定站点。	（可选）如果源或目标有多个站点，请在脚本的 `Environment Setup` 部分手动指定站点。 如果源站点和目标站点仅一个站点，则脚本将自动查找信息。	云架构师
运行 脚本。	在安装了 PowerCLI 的服务器上，从提升的 PowerShell 窗口运行脚本，并在出现提示时输入您的凭据。	云架构师
验证脚本。	确认虚拟机迁移已启动。	云架构师

运行功能齐全的脚本以迁移多个 VMs

Task	描述	所需技能
创建和填充 .csv 文件。	在您的计算机上创建一个名为 `Import_VM_list.csv` 的 .csv 文件，并使用以下示例内容填充该文件： `VM_NAME,DESTINATIO N_VM_FOLDER,DESTIN ATION_COMPUTE,DEST INATION_DATASTORE, SOURCE_NETWORK,DES TINATION_NETWORK [enterValue],[enter Value],[enterValue ],[enterValue],[en terValue],[enterVa lue]` 将.csv 文件中的每一个 `[enterValue]` 都替换为您之前收集的信息。 **Note** 您可以使用 VMware vRealize Network Insight (vRNI) 或其他方法填充.csv 文件。	云架构师
复制脚本。	该脚本的全功能版本使用外部.csv 文件中的信息自动迁移多个文件。 VMs 复制此模式的 “代码” 部分中的第二个脚本，并将其存储在安装了 VMware PowerCLI 模块	云架构师

Task	描述	所需技能
	的计算机上，与.csv 文件位于同一个文件夹中。	
修改脚本。	编辑脚本，以进行以下更改： • 第 7 行：设置 HCX 服务器变量 (`Connect-HCXServer`)。 • 第 12 行：（可选）如果您以不同的方式设置.csv 文件名，请对其进行更新。 • 第 3-4 行：（可选）设置时间表。 • 第 20 行：（可选）在 `Migration` 部分中指定 `New-HCXMigration` 设置。 • 第 9 行和第 11 行：（可选）如果源或目标包含多个站点，请手动指定所需站点。	云架构师
运行 脚本。	在安装了 PowerCLI 的服务器上，从提升的 PowerShell 窗口运行脚本，并在出现提示时输入您的凭据。	云架构师
验证脚本。	确认虚拟机迁移已启动。	云架构师

故障排除

事务	解决方案
脚本失败，显示错误消息：	如果源 vCenter 使用跨 vCenter NSX，则 PowerCLI 模块将无法运行。使用带有 HCX API

事务	解决方案
“所有源网络都未映射至目标！”	的脚本方法（例如 Python）而不是 PowerCLI。这是 PowerCLI 脚本的已知限制。
脚本失败，显示错误消息： “Connect-HCXServer 错误：未授权”	您输入的凭证未提供必要权限。

相关资源

- 使用混合 VMware 云扩展 (HCX) 将工作负载迁移到 AWS 上的云端（AWS 博客文章）
- 选择将您的 VMware 应用程序和工作负载迁移到 AWS 云的迁移方法（AWS Prescriptive Guidence）
- 使用 VMware HCX 将 VMware SDDC 迁移到 AWS 上的 VMware 云端（AWS Prescriptiv e Guidence）
- HCX 模块入门（VMware 博客文章）

将 F5 BIG-IP 工作负载迁移至 Amazon Web Services Cloud 上的 F5 BIG-IP VE

由 Will Bauer (AWS) 创建

摘要

组织希望迁移至 Amazon Web Services (AWS) Cloud，以提高其敏捷性和弹性。将 F5 BIG-IP 安全和流量管理解决方案迁移至 Amazon Web Services Cloud 后，您可以专注于在整个企业架构中实现敏捷性和采用高价值运营模式。

此模式介绍了如何在 Amazon Web Services Cloud 上将 F5 BIG-IP 工作负载迁移至 F5 BIG-IP Virtual Edition（VE）工作负载。将通过为现有环境更换主机并利用更换平台的各个方面（例如服务发现和 API 集成）来迁移工作负载。AWS CloudFormation 模板可加快您的工作负载迁移到 AWS 云的速度。

此模式适用于迁移 F5 安全和流量管理解决方案的技术工程和架构团队，并随附 AWS Prescriptive Guidance 网站上的 在 Amazon Web Services Cloud 上 从 F5 BIG-IP 迁移至 F5 BIG-IP VE 的指南。

先决条件和限制

先决条件

- 现有的本地 F5 BIG-IP 工作负载。
- 现有 BIG-IP VE 版本 F5 许可证。
- 一个有效的 Amazon Web Services account。
- 一种现有的虚拟私有云 (VPC)，配置了通过 NAT 网关或弹性 IP 地址的出口，并配置了对以下终端节点的访问权限：亚马逊简单存储服务 (Amazon S3)、亚马逊弹性计算云（EC2Amazon)、AWS 安全令牌服务 (AWS STS) 和亚马逊。 CloudWatch您还可以修改模块化和可扩展 VPC 架构 Quick Start，将其作为部署的基块。
- 一个或两个可用现有区，具体取决于您的要求。
- 每个可用区内包含三个私有子网。
- AWS CloudFormation 模板，可在 F5 GitHub 存储库中找到。

在迁移过程中，您还可以根据需要使用以下内容：

- F5 云故障转移扩展用于管理弹性 IP 地址映射、辅助 IP 映射以及路由表更改。
- 如果您使用多个可用区，则需要使用 F5 云故障转移扩展来处理与虚拟服务器的弹性 IP 映射。
- 您应考虑使用 F5 应用程序服务 3 (AS3)、F5 应用程序服务模板 (FAST) 或其他基础设施即代码 (IaC) 模型来管理配置。在 IaC 模型中准备配置，并使用代码存储库帮助迁移和持续的管理工作。

专业知识

- 这种模式需要熟悉 VPCs 如何将一个或多个数据中心连接到现有的数据中心。有关这方面的更多信息，请参阅 Amazon Network-to-Amazon VPC 文档中的 VPC 连接选项。
- 还需要熟悉 F5 产品和模块，包括Traffic Management Operating System (TMOS)、Local Traffic Manager (LTM)、Global Traffic Manager (GTM)、Access Policy Manager (APM)、Application Security Manager (ASM)、Advanced Firewall Manager (AFM) 以及 BIG-IQ。

产品版本

- 我们建议您使用 F5 BIG-IP 13.1或更高版本，尽管该模式支持 F5 BIG-IP 12.1或更高版本。

架构

源技术堆栈

- F5 BIG-IP 工作负载

目标技术堆栈

- Amazon CloudFront
- Amazon CloudWatch
- Amazon EC2
- Amazon S3
- Amazon VPC
- AWS Global Accelerator
- AWS STS
- AWS Transit Gateway
- F5 BIG-IP VE

目标架构

工具

- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。
- Amazon 通过全球数据中心网络交付您的网页内容，从而降低延迟并提高性能，从而 CloudFront加快网络内容的分发。
- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和在 AWS 上运行的应用程序的指标。
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Security Token Service (AWS STS) 可帮助您为用户申请临时、权限有限的凭证。
- AWS Transit Gateway 是一个连接虚拟私有云 (VPCs) 和本地网络的中心枢纽。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。

操作说明

发现与评测

Task	描述	所需技能
评测 F5 BIG-IP 性能。	收集并记录虚拟服务器上应用程序的性能指标，以及将要迁移系统的指标。这将有助于正确调整目标 AWS 基础设施规模，从而实现更好的成本优化。	F5 架构师、工程师和网络架构师以及工程师
评估 F5 BIG-IP 操作系统和配置。	评估将迁移哪些对象以及是否需要维护网络结构，例如 VLANs。	F5 架构师、工程师

Task	描述	所需技能
评估 F5 许可证选项。	评估您所需许可证和使用模式此评测应基于您对 F5 BIG-IP 操作系统与配置的评估。	F5 架构师、工程师
评估公用应用程序。	确定哪些需要公有 IP 地址的应用程序。将这些应用程序与所需实例和集群对齐，以满足性能和服务水平协议（SLA）要求。	F5 架构师、云架构师、网络架构师、工程师、应用团队
评估内部应用程序。	评估内部用户将使用的应用程序。确保您了解这些内部用户在组织中的位置，以及这些环境是如何连接至 Amazon Web Services Cloud 的。您还应确保这些应用程序在默认域中使用域名系统（DNS）。	F5 架构师、云架构师、网络架构师、工程师、应用团队
完成 AMI。	并非所有 F5 BIG-IP 版本都是以 Amazon 机器映像 () 的形式创建的。AMIs如果您有特定的快速修复工程 (QFE) 版本，则可以使用 F5 BIG-IP Image Generator Tool。如需了解关此工具的更多信息，请参阅“相关资源”部分。	F5 架构师、云架构师、工程师
最终确定实例类型和架构。	决定实例类型、VPC 架构和互连架构。	F5 架构师、云架构师、网络架构师、工程师

完成与安全和合规相关的活动

Task	描述	所需技能
记录现有 F5 安全策略。	收集并记录现有 F5 安全策略。确保在安全的代码存储库中创建副本。	F5 架构师、工程师
加密 AMI。	（可选）您的组织可能需要静态数据加密。关于创建自带许可（BYOL）映像的更多信息，请参阅“相关资源”部分。	F5 架构师、工程师云架构师、工程师
强化设备。	这将有助于防范潜在漏洞。	F5 架构师、工程师

配置新 AWS 环境

Task	描述	所需技能
创建边缘和安全账户。	登录 AWS 管理控制台，并创建将提供和运营边缘和安全服务的 Amazon Web Services account。这些帐户可能不同于 VPCs 为共享服务和应用程序运行的帐户。此步骤可以作为登录区的一部分。	云架构师、工程师
部署边缘和安全性 VPCs。	设置和配置交付边缘和安全服务 VPCs 所需的内容。	云架构师、工程师
连接至源数据中心。	连接至托管 F5 BIG-IP 工作负载的源数据中心。	云架构师、网络架构师、工程师
部署 VPC 连接。	将 Edge 和安全服务 VPCs 连接到应用程序 VPCs。	网络架构师、工程师

Task	描述	所需技能
部署实例。	使用 “相关资源” 部分中的 AWS CloudFormation 模板部署实例。	F5 架构师、工程师
测试和配置实例故障转移。	确保 AWS Advanced HA iApp 模板或 F5 Cloud Failover Extension 已配置且运行正常。	F5 架构师、工程师

配置联网

Task	描述	所需技能
准备 VPC 拓扑。	打开 Amazon VPC 控制台，确保您的 VPC 具有 F5 BIG-IP VE 部署所需所有子网与保护。	网络架构师、F5 架构师、云架构师、工程师
准备好 VPC 端点。	如果 F5 BIG-IP 工作负载无法访问 TMM 接口上的 NAT 网关或弹性 IP 地址，请为亚马逊、亚马逊 S3 和 AWS STS 准备 VPC 终端节点。 EC2	云架构师、工程师

迁移数据

Task	描述	所需技能
迁移配置。	将 F5 BIG-IP 配置迁移至 Amazon Web Services Cloud 上的 F5 BIG-IP VE。	F5 架构师、工程师
关联次要节点 IPs。	虚拟服务器 IP 地址与分配至实例的辅助 IP 地址关联。分配辅助 IP 地址并，确保选中“允许重新映射/重新分配”。	F5 架构师、工程师

测试配置

Task	描述	所需技能
验证虚拟服务器配置。	测试虚拟服务器。	F5 架构师、应用团队

完成操作

Task	描述	所需技能
创建备份策略。	必须关闭系统才可创建完整快照。有关更多信息，请参阅“相关资源”部分中的“更新 F5 BIG-IP 虚拟机”。	F5 架构师、云架构师、工程师
创建集群故障转移运行手册。	确保故障转移运行手册进程已完成。	F5 架构师、工程师
设置和验证日志记录。	配置 F5 Telemetry Streaming 以将日志发送至所需目标。	F5 架构师、工程师

完成割接

Task	描述	所需技能
割接到新的部署。		F5 架构师、云架构师、网络架构师、工程师、 AppTeams

相关资源

迁移指南

- 从 F5 BIG-IP 迁移至 Amazon Web Services Cloud 上的 F5 BIG-IP VE

F5 资源

- F5 CloudFormation GitHub 存储库中的 AWS 模板
- Amazon Web Services Marketplace 的 F5
- F5 BIG-IP VE 概述
- 快速入门示例 - BIG-IP Virtual Edition with WAF (LTM + ASM)
- AWS 上的 F5 应用程序服务开启：概述（视频）
- F5 应用程序服务 3 扩展用户指南
- F5 云文档
- F5 iControl REST wiki
- F5 单一配置文件（11.x - 15.x）概述
- F5 拓扑实验室
- F5 白皮书
- F5 BIG-IP Image Generator Tool
- 更新 F5 BIG-IP VE 虚拟机
- UCS 存档“平台迁移”选项概述

使用二进制方法将本地 Go Web 应用程序迁移至 AWS Elastic Beanstalk

由 Suhas Basavaraj(AWS) 和 Shumaz Mukhtar Kazi(AWS) 编写

摘要

此模式介绍如何将本地 Go Web 应用程序迁移至 AWS Elastic Beanstalk。应用程序迁移后，Elastic Beanstalk 会为源包构建二进制文件，并将其部署到亚马逊弹性计算云 (Amazon) 实例。 EC2

根据更换主机迁移策略，该模式的方法速度很快，并且不需要更改代码，这意味着更少的测试和迁移时间。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 本地 Go Web 应用程序。
- 包含 Go 应用程序源代码的 GitHub 存储库。如果您不使用 GitHub，还有其他方法可以为 Elasti c Beanstalk 创建应用程序源包。

产品版本

- Elastic Beanstalk 支持最新 Go 版本。有关更多信息，请参阅 Elastic Beanstalk 文档。

架构

源技术堆栈

- 本地 Go Web 应用程序

目标技术堆栈

- AWS Elastic Beanstalk
- 亚马逊 CloudWatch

目标架构

工具

- AWS Elastic Beanstalk 可在 Amazon Web Services Cloud 中快速部署和管理应用程序，用户不必了解运行这些应用程序的基础设施。Elastic Beanstalk 可降低管理的复杂性，但不会影响选择或控制。
- GitHub是一个开源的分布式版本控制系统。

操作说明

创建 Go Web 应用程序源捆绑包 .zip 文件

Task	描述	所需技能
为 Go 应用程序创建源捆绑包。	打开包含 Go 应用程序源代码的 GitHub 存储库并准备源包。源捆绑包在根目录中包含 `application.go` 源文件，该文件托管 Go 应用程序的主软件包。如果您不使用 GitHub，请参阅此模式前面的“先决条件”部分，了解创建应用程序源包的其他方法。	系统管理员、应用程序开发人员
创建配置文件。	在源捆绑包中创建 `.ebextensions` 文件夹，然后在该文件夹中创建 `options.config` 文件。有关更多信息，请参阅 Elastic Beanstalk 文档。	系统管理员、应用程序开发人员
创建源捆绑包 .zip 文件。	运行以下命令。 `git archive -o ../godemoapp.zip HEAD` 这将创建源捆绑包 .zip 文件。下载 .zip 文件并将其另存为本地文件。	系统管理员、应用程序开发人员

Task	描述	所需技能
	Important .zip 文件不能超过 512 MB，也不能包含父文件夹或顶级目录。	

将 Go Web 应用程序迁移至 Elastic Beanstalk

Task	描述	所需技能
选择 Elastic Beanstalk 应用程序。	1. 登录 AWS 管理控制台 并打开 Elastic Beanstalk 控制台。 2. 从区域列表，选择您的 AWS 区域。 3. 在导航窗格中，选择应用程序，然后选择现有 Elastic Beanstalk 应用程序或创建一个。 有关如何创建 Elastic Beanstalk 应用程序的说明，请参阅 Elastic Beanstalk 文档。	系统管理员、应用程序开发人员
初始化 Elastic Beanstalk Web 服务器环境。	1. 在应用程序概述页面，选择创建新环境，然后选择 Web 服务器环境。 2. 填写环境名称和域名字段。 3. 选择平台版本，然后选择 Go 以作为您的平台。	系统管理员、应用程序开发人员
将源捆绑包 .zip 文件上传到 Elastic Beanstalk。	1. 对于 Application code(应用程序代码)，选择 Upload	系统管理员、应用程序开发人员

Task	描述	所需技能
	your code(上传您的代码)，然后选择 Local file(本地文件)。 2. 选择包含源捆绑包的 .zip 文件。 3. 在版本标签，为文件指定唯一的名称，然后选择创建环境。	
测试已部署的 Go Web 应用程序。	您将被重定向至 Elastic Beanstalk 应用程序概述页面。在概述顶部的环境 ID 旁边，选择结尾为 `elasticbeanstalk.com` 的 URL，以导航到您的应用程序。您的应用程序必须在其配置文件中使用此名称作为环境变量，并将其显示在网页上。	系统管理员、应用程序开发人员

故障排除

事务	解决方案
无法通过应用程序负载均衡器访问应用程序	检查包含 Elastic Beanstalk 应用程序的目标群体。如果运行状况不佳，请登录您的 Elastic Beanstalk 实例并检查 `nginx.conf` 文件配置，以验证其路由到的运行状况网址是否正确。您可能需要更改目标群组的运行状况检查 URL。

相关资源

- Elastic Beanstalk 支持的 Go 平台版本
- 在 Elastic Beanstalk 中使用配置文件

- 在 Elastic Beanstalk 中创建示例应用程序

将本地 SFTP 服务器迁移到使用 AWSAWS Transfer for SFTP

由 Akash Kumar (AWS) 创建

摘要

此模式描述了如何使用该 AWS Transfer for SFTP 服务将使用安全外壳 (SSH) 文件传输协议 (SFTP) 的本地文件传输解决方案迁移到。 AWS Cloud 用户通常通过其域名或固定 IP 连接至 SFTP 服务器。此模式涵盖了两种情况。

AWS Transfer for SFTP 是... 的成员 AWS Transfer Family。它是一种安全的传输服务，可用于通过 SFTP 将文件传入和传出 AWS 存储服务。您可以使用亚马逊简单存储服务 (AWS Transfer for SFTP Amazon S3) Service 或亚马逊弹性文件系统 (Amazon EFS)。此模式使用了 Amazon S3 存储。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- 现有 SFTP 域名或固定 SFTP IP。

限制

- 当前，您可在一个请求中传输的最大对象为 5 GiB。对于大于 100 MiB 的文件，可以考虑使用 Amazon S3 分段上传。

架构

源技术堆栈

- 本地平面文件或数据库转储文件。

目标技术堆栈

- AWS Transfer for SFTP
- Amazon S3
- Amazon Virtual Private Cloud(Amazon VPC)
- AWS Identity and Access Management (IAM) 角色和策略
- 弹性 IP 地址

- 安全组
- Amazon CloudWatch 日志（可选）

目标架构

自动化和扩缩

要自动执行此模式的目标架构，请使用随附的 AWS CloudFormation 模板：

- `amazon-vpc-subnets.yml` 预置具有两个公有子网和两个私有子网的虚拟私有云（VPC）。
- `amazon-sftp-server.yml` 预置 SFTP 服务器。
- `amazon-sftp-customer.yml` 添加用户。

工具

AWS 服务

- Amazon CloudWatch Lo gs 可帮助您集中所有系统和应用程序的日志， AWS 服务 这样您就可以监控它们并安全地将其存档。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。此模式使用了 Amazon S3 作为文件传输存储系统。
- AWS Transfer for SFTP帮助您通过 SFTP 协议将文件传入和传出 AWS 存储服务。
- Amazon Virtual Private Cloud（亚马逊 VPC）可帮助您将 AWS 资源启动到您定义的虚拟网络中。该虚拟网络类似于您在数据中心中运行的传统网络，并具有使用 AWS的可扩展基础设施的优势。

操作说明

创建 VPC

Task	描述	所需技能
创建带有子网的 VPC。	打开 Amazon VPC 控制台。创建具有两个公共子网的虚拟私	开发人员、系统管理员

Task	描述	所需技能
	有云（VPC）。（第二个子网具有高可用性。） —或者— 您可以在CloudFormation 控制台中部署附加的 CloudFormation 模板`amazon-vpc-subnets.yml` ，以自动执行此长篇故事中的任务。	
添加互联网网关。	预置互联网网关，并将其连接至 VPC。	开发人员、系统管理员
迁移现有 IP。	将现有 IP 连接至弹性 IP 地址。您可从地址池创建弹性 IP 地址，并使用该地址。	开发人员、系统管理员

预置 SFTP 服务器

Task	描述	所需技能
创建 SFTP 服务器。	打开 AWS Transfer Family 管理控制台。按照 AWS Transfer Family 文档中为服务器创建面向 Internet 的终端节点中的说明，创建带有面向互联网的终端节点的 SFTP 服务器。对于端点类型，请选择 VPC 托管。对于访问，请选择面向 Internet。对于 VPC，请选择您在上一操作中创建的 VPC。 —或者—	开发人员、系统管理员

Task	描述	所需技能
	您可以在CloudFormation 控制台中部署附加的 CloudFormation 模板`amazon-sftp-server.yml` ，以自动执行此长篇故事中的任务。	
迁移域名。	将现有域名附加至自定义主机名。如果正在使用新域名，请使用 Amazon Route 53 DNS 别名。对于现有域名，请选择其他 DNS。有关更多信息，请参阅 AWS Transfer Family 文档中的使用自定义主机名。	开发人员、系统管理员
添加 CloudWatch 日志角色。	（可选）如果要启用 CloudWatch 日志记录，请使用 CloudWatch 日志 API 操作创建一个`Transfer`角色`logs:CreateLogGroup` `logs:CreateLogStream` 、 `logs:DescribeLogStreams` 、和`logs:PutLogEvents` 。有关更多信息，请参阅 AWS Transfer Family 文档 CloudWatch中的使用记录活动。	开发人员、系统管理员
保存并提交。	选择保存。对于操作，请选择启动，然后等待 SFTP 服务器的创建状态为联机。	开发人员、系统管理员

将弹性 IP 地址映射至 SFTP 服务器

Task	描述	所需技能
停止服务器，以便修改设置。	在AWS Transfer Family 控制台上，选择 “服务器”，然后选择您创建的 SFTP 服务器。对于操作，请选择停止。当服务器处于离线状态时，选择编辑以修改其设置。	开发人员、系统管理员
选择可用区和子网。	在可用区部分中，为您的 VPC 选择可用区和子网。	开发人员、系统管理员
添加弹性 IP 地址。	对于IPv4 地址，为每个子网选择一个弹性 IP 地址，然后选择保存。	开发人员、系统管理员

添加用户

Task	描述	所需技能
为用户访问 S3 存储桶创建 IAM 角色。	为 `Transfer` 创建 IAM 角色，并添加将 S3 存储桶名称作为资源的 `s3:ListBucket` 、 `s3:GetBucketLocation` 和 `s3:PutObject` 。有关更多信息，请参阅 AWS Transfer Family 文档中的创建 IAM 角色和策略。 —或者— 您可以在CloudFormation 控制台中部署附加的 CloudFormation 模板`amazon-sftp-`	开发人员、系统管理员

Task	描述	所需技能
	`customer.yml` ，以自动执行此长篇故事中的任务。	
创建 S3 存储桶。	为应用程序创建 S3 存储桶。	开发人员、系统管理员
创建可选文件夹。	（可选）如果您想将用户的文件单独存储在特定的 Amazon S3 文件夹中，则根据需要添加文件夹。	开发人员、系统管理员
创建 SSH 公有密钥。	要创建 SSH 密钥对，请参阅 AWS Transfer Family 文档中的生成 SSH 密钥。	开发人员、系统管理员
添加用户。	在AWS Transfer Family 控制台上，选择服务器，选择您创建的 SFTP 服务器，然后选择添加用户。对于主目录，请选择您创建的 S3 存储桶。对于SSH 公有密钥，请输入 SSH 密钥对的 SSH 公有密钥部分。为 SFTP 服务器添加用户，然后选择添加。	开发人员、系统管理员

测试 SFTP 服务器。

Task	描述	所需技能
更新安全组。	在 SFTP 服务器的安全组部分中，添加测试计算机的 IP 以获得 SFTP 访问权限。	开发人员
使用 SFTP 客户端实用程序测试服务器。	使用任何 SFTP 客户端实用程序测试文件传输功能。有关客户端和说明的列表，请参阅文	开发人员

Task	描述	所需技能
	档中的使用客户端传输 AWS Transfer Family 文件。	

相关资源

- AWS Transfer Family 用户指南
- Amazon S3 用户指南
- 亚马逊 EC2 文档中的@@ 弹性 IP 地址

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS 应用程序迁移服务将本地虚拟机迁移到 Amazon EC2

由 Thanh Nguyen (AWS) 创建

摘要

就应用程序迁移而言，组织可以采取不同的方法将应用程序服务器从本地环境重新托管（直接迁移）至 Amazon Web Services (AWS) Cloud。一种方法是配置新的亚马逊弹性计算云 (Amazon EC2) 实例，然后从头开始安装和配置应用程序。另一种方法是通过第三方或 AWS 原生迁移服务同时迁移多台服务器。

此模式概述了使用 AWS 应用程序迁移服务将支持的虚拟机 (VM) 迁移到 AWS 云上的 Amazon EC2 实例的步骤。您可以使用此模式中的方法，手动迁移一个或多个虚拟机（逐一），还可以根据概述的步骤创建适当的自动化脚本执行自动迁移。

先决条件和限制

先决条件

- 支持应用程序迁移服务的 AWS 区域的有效 Amazon Web Services account
- 通过使用 AWS Direct Connect 或虚拟专用网络 (VPN) 的私有网络，或者通过互联网实现源 EC2 服务器和目标服务器之间的网络连接

限制

- 有关支持区域的最新列表，请参阅支持的 AWS 区域。
- 有关支持的操作系统的列表，请参阅 Amazon 的 “支持的操作系统” 和 “常规” 部分 EC2 FAQs。

架构

源技术堆栈

- 运行 Amazon 支持的操作系统的物理、虚拟或云托管服务器 EC2

目标技术堆栈

- 与源虚拟机运行相同操作系统的 Amazon EC2 实例
- Amazon Elastic Block Store（Amazon EBS）

源架构和目标架构

下图显示了解决方案的高级架构以及主要组件。本地数据中心中存在带有本地磁盘的虚拟机。在 AWS 上，有一个包含复制服务器的暂存区和一个带有用于测试和切换的 EC2 实例的迁移资源区域。两个子网都包含 EBS 卷。

1. 初始化 AWS Application Migration Service。
2. 设置暂存区域服务器配置和报告，包括暂存区域资源。
3. 在源服务器上安装代理，并使用连续块级数据复制（压缩和加密）。
4. 自动编排和系统转换，以缩短割接窗口。

网络架构

下图从联网角度显示了此解决方案的高级架构和主要组件，包括本地数据中心和 AWS 主要组件之间通信所需协议和端口。

工具

- AWS Application Migration Service 可帮助您将应用程序更换主机（直接迁移）到 Amazon Web Services Cloud 中，无需更改且停机时间最短。

最佳实践

- 在向目标 EC2 实例的直接转换完成之前，请勿使源服务器离线或重新启动。
- 为用户提供充足的机会，使其在目标服务器上执行用户验收测试 (UAT)，以识别和解决任何问题。理想情况下，此测试应在转换前至少两周开始。
- 经常在应用程序迁移服务控制台上监控服务器复制状态，以便尽早发现问题。
- 使用临时的 AWS Identity and Access Management (IAM) 凭证安装代理，而非永久 IAM 用户凭证。

操作说明

生成 AWS 凭证

Task	描述	所需技能
创建 AWS Replication Agent IAM 角色。	使用 Amazon Web Services Account 的管理权限登录。 在 AWS Identity and Access Management (IAM) 控制台，创建 IAM 角色： 1. 在 IAM 控制台上，选择角色。 2. 选择 Create role(创建角色)。 3. 在选择可信实体页面的 可信实体类型部分中，选择Amazon Web Services Account。 4. 在 Amazon Web Services account部分中，选择此账户 (<account-id>)。 5. 选择 Next(下一步)。 6. 在添加权限页面上，搜索 `AWSApplicationMigrationAgentInstallationPolicy` 策略，然后选中策略名称旁边的复选框。 7. 选择 Next(下一步)。 8. 在角色详细信息页面上，输入 MGN_Agent_Installation_Role 作为角色名称。	AWS 管理员、迁移工程师

Task	描述	所需技能
	9. 验证字段是否正确，然后选择创建角色。	

Task	描述	所需技能
生成临时安全凭证。	在安装了 AWS 命令行界面（AWS CLI）的计算机上，使用管理权限登录。或者（在支持的 AWS 区域内），在 AWS 管理控制台上，使用 AWS 账户的管理权限登录，然后打开 AWS CloudShell。 使用以下命令生成临时凭证，将 `<account-id>` 替换为 Amazon Web Services account ID。 `aws sts assume-role --role-arn arn:aws:iam::<account-id>:role/MGN_Agent_Installation_Role --role-session-name mgn_installation_session_role` 从命令输出，复制 `AccessKeyId` 、 `SecretAccessKey` 以及 `SessionToken` 的值。将其存储于安全的地方，以备后用。 **Important** 这些临时证书将在一小时后过期。如果一小时后需要凭证，请重复前面的步骤。	AWS 管理员、迁移工程师

初始化应用程序迁移服务，并创建复制设置模板

Task	描述	所需技能
初始化此服务。	在控制台上，使用 Amazon Web Services Account 的管理权限登录。 选择应用程序迁移服务，然后选择 开始使用。	AWS 管理员、迁移工程师
创建并配置复制设置模板。	1. 提供以下配置详细信息： a. 选择暂存区域子网。 b. 选择复制服务器实例类型（默认为 `t3.small`）。 c. 选择 EBS 卷类型 (默认为 gp3)。 d. 选择 EBS 加密选项。 e. 确保选中始终使用应用程序迁移服务安全组复选框。 f. 如果您在本地环境和 AWS 之间使用私有网络连接 DirectConnect，请选中使用私有 IP 进行数据复制（VPN、、VPC 对等）复选框。 g. 如果要限制应用程序迁移服务的网络带宽，请选中限制网络带宽（每台服务器-以 Mbps 为单位）复选框。 2. 选择创建模板。	AWS 管理员、迁移工程师

Task	描述	所需技能
	Application Migration Service 将自动创建促进数据复制和启动迁移服务器所需所有 IAM 角色。	

在源计算机上安装 AWS Replication Agents

Task	描述	所需技能
准备好所需 AWS 凭证。	在源服务器上运行安装程序文件时，需要输入之前生成的临时凭证，包括 `AccessKeyId` 、`SecretAccessKey` 和 `SessionToken` 。	迁移工程师、AWS 管理员
对于 Linux 服务器，请安装代理。	复制安装程序命令，登录至源服务器，并运行安装程序。有关详细说明，请参阅 AWS 文档。	AWS 管理员、迁移工程师
对于 Windows 服务器，请安装代理。	将安装程序文件下载至每台服务器，然后运行安装程序命令。有关详细说明，请参阅 AWS 文档。	AWS 管理员、迁移工程师
等待初始数据复制完成。	代理完成安装后，源服务器将出现在 Application Migration Service 控制台的源服务器部分。服务器正在执行初始数据复制，请稍候。	AWS 管理员、迁移工程师

配置启动设置

Task	描述	所需技能
指定服务器的详细信息。	在 Application Migration Service 控制台上，选择源服务器部分，然后从列表中选择服务器名称，以访问服务器详细信息。	AWS 管理员、迁移工程师
配置启动设置。	选择启动设置选项卡。您可以配置各种设置，包括常规启动设置和 EC2 启动模板设置。有关详细说明，请参阅 AWS 文档。	AWS 管理员、迁移工程师

执行测试

Task	描述	所需技能
测试源服务器。	1. 在 Application Migration Service 控制台的源服务器部分中，确保源服务器的迁移周期已准确好进行测试且数据复制状态为正常。 2. 选中每台源服务器左侧的复选框。 3. 选择测试并割接，然后选择启动测试实例。 4. 系统提示时，请选择启动。 将启动服务器。	AWS 管理员、迁移工程师

Task	描述	所需技能
验证测试是否成功完成。	测试服务器完全启动后，页面上的警报状态将显示每台服务器已启动。	AWS 管理员、迁移工程师
测试服务器。	执行测试服务器测试，以确保其按预期运行。	AWS 管理员、迁移工程师

计划并执行割接

Task	描述	所需技能
计划割接时段。	与相关团队计划适当的割接时间表。	AWS 管理员、迁移工程师
执行割接。	1. 在 Application Migration 控制台的源服务器页面上，选中每台源服务器左侧的复选框。 2. 选择测试并割接，然后选择标记为“准备割接”。 3. 验证每台源服务器的迁移生命周期是否已准备好割接。 4. 选择测试并割接，然后选择启动割接实例。 5. 系统提示时，请选择启动。将启动服务器。 源服务器的迁移生命周期将更改为正在进行割接。	AWS 管理员、迁移工程师
验证割接是否成功完成。	割接服务器完全启动后，源服务器页面上的警报状态将显示每台服务器已启动。	AWS 管理员、迁移工程师

Task	描述	所需技能
测试服务器。	执行割接服务器测试，以确保其按预期运行。	AWS 管理员、迁移工程师
完成割接。	选择测试并割接，然后选择完成割接以完成迁移过程。	AWS 管理员、迁移工程师

相关资源

- AWS Application Migration Service
- AWS Application Migration Service 用户指南

使用 AWS SFTP 将小型数据集从本地迁移至 Amazon S3

由查尔斯·吉布森 (AWS) 和塞尔吉·舍甫琴科 (AWS) 创作

摘要

此模式描述了如何使用 AWS Transfer for SFTP (AWS SFTP)将小型数据集 (5 TB 或更少) 从本地数据中心迁移至 Amazon Simple Storage Service (Amazon S3)。数据可以是数据库转储文件，也可以是平面文件。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 在您的数据中心与 AWS 之间建立 AWS Direct Connect 链接

限制

- 数据文件必须小于 5 TB 对于超过 5 TB 的文件，您可以分段上传至 Amazon S3 或选择其他数据传输方法。

架构

源技术堆栈

- 本地平面文件或数据库转储文件。

目标技术堆栈

- Amazon S3

源架构和目标架构

工具

- AWS SFTP — 允许通过安全文件传输协议 (SFTP) 将文件直接传入和传出 Amazon S3。

- AWS Direct Connect — 建立从您的本地数据中心至 AWS 的专用网络连接。
- VPC 终端节点 — PrivateLink 无需互联网网关、网络地址转换 (NAT) 设备、VPN 连接或 AWS Direct Connect 连接，即可将 VPC 与支持的 AWS 服务和由 AWS 提供支持的 VPC 终端节点服务进行私密连接。VPC 中的实例无需公有 IP 地址便可与服务中的资源进行通信。

操作说明

准备迁移

Task	描述	所需技能
记录当前 SFTP 要求。		应用程序所有者，系统管理员
识别身份验证要求。	要求可能包括基于密钥的身份验证、用户名或密码或身份提供者（IdP）。	应用程序所有者，系统管理员
确定应用程序集成要求。		应用程序所有者
确定需要服务的用户。		应用程序所有者
确定 SFTP 服务器端点 DNS 名称。		网络连接
确定备份策略。		系统管理员、数据库管理员（如果传输了数据）
确定应用程序迁移或割接策略。		应用程序所有者、系统管理员、数据库管理员

配置基础设施

Task	描述	所需技能
在您的 AWS 账户中创建一个或多个虚拟私有云 (VPCs) 和子网。		应用程序所有者，AMS

Task	描述	所需技能
创建安全组和网络访问控制列表（ACL）。		安全，联网，AMS
创建 S3 存储桶。		应用程序所有者，AMS
创建身份和访问管理（IAM）角色。	创建 IAM policy，该策略包含让 AWS SFTP 能够访问您的 S3 存储桶的权限。此 IAM policy 确定您为 SFTP 用户提供的访问级别。创建另一个 IAM policy，以与 AWS SFTP 建立信任关系。	安全、AMS
关联已注册的域名（可选）。	如果您拥有自己的注册域，您可以将其与 SFTP 服务器关联。您可以将 SFTP 流量从域或子域路由到 SFTP 服务器端点。	联网、AMS
创建 SFTP 服务器。	指定服务用于对用户进行身份验证的身份提供程序类型。	应用程序所有者，AMS
打开 SFTP 客户端。	打开 SFTP 客户端并配置连接以使用 SFTP 端点主机。AWS SFTP 支持任何标准 SFTP 客户端。常用的 SFTP 客户端包括 OpenSSH、WinSCP、Cyberduck 和。 FileZilla您可从 AWS SFTP 控制台获取 SFTP 服务器的主机名。	应用程序所有者，AMS

计划和测试

Task	描述	所需技能
计划应用程序迁移。	规划所需任何应用程序配置更改，设置迁移日期并确定测试安排。	应用程序所有者，AMS
测试基础设施。	在非生产环境中测试。	应用程序所有者，AMS

相关资源

参考

- AWS Transfer For SFTP 用户指南
- AWS Direct Connect 资源
- VPC 端点

教程和视频

- AWS Transfer for SFTP（视频）
- AWS Transfer For SFTP 用户指南
- AWS SA Whiteboarding - Direct Connect（视频）

从 Oracle 迁移 GlassFish 到 AWS Elastic Beanstalk

由 Sandeep Bondugula (AWS) 编写

摘要

此模式描述了如何将在本地 Oracle GlassFish 服务器上运行的 Java 应用程序迁移到 AWS 云中的 AWS Elastic Beanstalk。

在 AWS 上，Java 应用程序使用 AWS Elastic Beanstalk 部署在 Docker GlassFish 服务器上，该服务器在亚马逊弹性计算云 (亚马逊) Auto Scaling 组中运行。 EC2

其他功能

- Amazon Elastic Beanstalk 充当多个底层资源的包装器。它设置 Elastic Load Balancing（处理来自 Amazon Route 53 的传入流量），将流量分散到一个或多个 EC2 实例，还可以用作部署工具。
- 要将本地数据库迁移至 Amazon Relational Database Service (Amazon RDS)，请更新数据库连接详细信息。在后端数据库中，您可配置 Amazon RDS 多可用区部署并选择数据库引擎类型。
- 您可使用多可用区部署来实现高可用性，并使用自动扩缩组和扩展策略来提高弹性。
- 您可以基于 Amazon CloudWatch 指标设置扩展策略。
- 在 AWS Elastic Beanstalk 中，你可以配置底层的 Elastic Load Balancing 设置和 Amazon Auto Scaling。 EC2

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 在上运行的本地 Java 应用程序 GlassFish
- Java Web 应用程序资源 (WAR) 文件

产品版本

- Oracle Glassfish 4.1.2 和 5.0
- Java 7 GlassFish 4.0
- Java 8 GlassFish 4.1 或更高版本

架构

源技术堆栈

- 开发的应用程序 GlassFish

目标技术堆栈

- Elastic Beanstalk

目标架构

部署工作流

工具

- Amazon Elastic Beanstalk — 一项服务，用于在包括 Apache、NGINX、Passenger 和 IIS 在内的服务器上部署和扩展使用 Java、.NET、PHP、Node.js、Python、Ruby、GO 和 Docker 开发的 Web 应用程序和服务。
- Amazon CloudWatch — 提供数据和可操作的见解来监控应用程序，响应系统范围的性能变化，优化资源利用率，并提供统一的运营状况视图。
- Docker — 一款将软件打包成标准化单元的平台，用于快速构建、测试和部署应用程序。
- Java — 一种通用的计算机编程语言。Java 是基于类的、面向对象，旨在减少实现依赖性。

操作说明

设置 VPC

Task	描述	所需技能
创建包含所需信息的虚拟私有云（VPC）实例。		SysAdmin
在 VPC 中创建至少两个子网。		SysAdmin

Task	描述	所需技能
根据要求创建路由表。		SysAdmin

设置 Amazon S3

Task	描述	所需技能
创建 Amazon Simple Storage Service (Amazon S3) 存储桶		SysAdmin
将 WAR 文件复制至 S3 存储桶，然后上载应用程序代码。		SysAdmin

创建 IAM 角色

Task	描述	所需技能
创建 AWS Identity and Access Management (IAM) 角色。	你可以使用默认的 “aws-elasticbeanstalk-ec双角色” 配置文件，也可以让 Elastic Beanstalk 自动创建它。	SysAdmin

设置 Elastic Beanstalk

Task	描述	所需技能
打开 Elastic Beanstalk 控制面板。		SysAdmin
创建新应用程序，并选择 Web 服务器环境。		SysAdmin
选择 GlassFish Docker 作为预配置平台。		SysAdmin

Task	描述	所需技能
上传代码。	提供本地系统文件中的 S3 存储桶文件 URL 或者 ZIP 文件。	SysAdmin
选择环境类型。	在配置容量设置中，选择单个实例或负载均衡器。	SysAdmin
配置负载均衡器。	如果您在上一步选择了负载均衡器，请配置多可用区部署。	SysAdmin
在配置安全设置，选择之前创建的 IAM 角色。		SysAdmin
在配置安全设置中，如果您已有密钥对，请使用它或创建新的 Amazon EC2 密钥对。		SysAdmin
在配置监控设置中，配置 Amazon CloudWatch。		SysAdmin
在配置安全设置中，选择之前创建的 VPC。		SysAdmin
选择 Create environment（创建环境）。		SysAdmin

测试应用程序

Task	描述	所需技能
使用创建环境中提供的 URL 测试应用程序。		
在 Amazon Route 53 中应用域名服务 (DNS) 更改。		

相关资源

- 甲骨 GlassFish 文文档
- GlassFish 开源 Java EE 参考实现
- AWS Elastic Beanstalk 文档
- 在亚马逊上使用 Elastic Beanstalk CloudWatch
- AWS Elastic Beanstalk 定价
- EC2 Auto Scaling 小组
- 扩展自动扩缩组的大小
- Amazon RDS 多可用区部署

将本地 Oracle 数据库迁移到亚马逊上的 Oracle EC2

创建者：Baji Shaik (AWS) 和 Pankaj Choudhary (AWS)

摘要

此模式将引导您完成在亚马逊弹性计算云 (Amazon EC2) 实例上将本地 Oracle 数据库迁移到 Oracle 的步骤。它描述了两种迁移选项：使用 AWS 数据迁移服务 (AWS DMS) 或使用原生 Oracle 工具，例如 RMAN、数据泵导入/导出、可传输表空间和 Oracle。 GoldenGate

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 本地数据中心中的 Oracle 源数据库

限制

- Amazon 必须支持目标操作系统 (OS) EC2。有关支持系统的完整列表，请参阅 Amazon EC2 FAQs。

产品版本

- Oracle 10.2 和更高版本（对于版本 10.x）、11g 直至 12.2 版本以及 18c 版本（Enterprise、Standard、Standard One 和 Standard Two 版）。有关 AWS DMS 支持的最新版本列表，请参阅 AWS DMS 文档中数据迁移来源中的 “本地和 Amazon EC2 实例数据库”。

架构

源技术堆栈

- 本地 Oracle 数据库

目标技术堆栈

- 亚马逊上的 Oracle 数据库实例 EC2

目标架构

数据迁移架构

使用 AWS DMS：

使用原生 Oracle 工具：

工具

- AWS DMS – AWS Database Migration Service (AWS DMS) 支持多种不同的源数据库和目标数据库。有关支持的数据库版本和版本的信息，请参阅使用 Oracle 数据库作为 AWS DMS 的源。建议使用最新版本的 AWS DMS，以获得最全面的版本和功能支持。
- Oracle 原生工具- RMAN、数据泵导入/导出、可传输表空间、Oracle GoldenGate

操作说明

计划迁移

Task	描述	所需技能
验证源数据库和目标数据库的版本。		数据库管理员
识别目标操作系统的版本。		数据库管理员，SysAdmin
根据 Oracle 兼容性列表和容量要求，确定目标服务器实例的硬件要求。		数据库管理员，SysAdmin
识别存储需求（存储类型和容量）。		数据库管理员，SysAdmin
确定网络要求（延迟与带宽）。		数据库管理员，SysAdmin

Task	描述	所需技能
根据容量、存储功能和网络功能选择正确的实例类型。		数据库管理员，SysAdmin
确定源数据库和目标数据库的网络/主机访问安全要求。		数据库管理员，SysAdmin
确定安装 Oracle 软件所需操作系统用户列表。		数据库管理员，SysAdmin
下载 AWS Schema Conversion Tool（AWS SCT）和驱动程序。		数据库管理员
为工作负载创建 AWS SCT 项目，然后连接至源数据库。		数据库管理员
生成用于创建对象（表、索引、序列等）的 SQL 文件。		数据库管理员
确定备份策略。		数据库管理员，SysAdmin
确定可用性要求。		数据库管理员
确定应用程序迁移/切换策略。		DBA、SysAdmin、应用程序所有者

配置基础设施

Task	描述	所需技能
在 Amazon Web Services account 中创建虚拟私有云（VPC）和子网。		SysAdmin
创建安全组和网络访问控制列表 (ACLs)。		SysAdmin

Task	描述	所需技能
配置并启动实 EC2 例。		SysAdmin

安装 Oracle 软件

Task	描述	所需技能
创建运行 Oracle 软件所需操作系统用户和组。		数据库管理员，SysAdmin
下载必要版本的 Oracle 软件。		
在 EC2 实例上安装 Oracle 软件。		数据库管理员，SysAdmin
使用 AWS SCT 生成的脚本创建表、主键、视图和序列等对象。		数据库管理员

迁移数据 - 选项 1

Task	描述	所需技能
使用原生 Oracle 工具或第三方工具迁移数据库对象和数据。	Oracle 工具包括数据泵导入/导出、RMAN、可传输表空间和。 GoldenGate	数据库管理员

迁移数据 - 选项 2

Task	描述	所需技能
确定迁移方法。		数据库管理员
使用 AWS DMS 控制台创建复制实例。		数据库管理员

Task	描述	所需技能
创建源和目标端点。		数据库管理员
创建复制任务。		数据库管理员
启用变更数据捕获 (CDC) 以捕获变更，从而进行持续复制。		数据库管理员
运行复制任务和监控日志。		数据库管理员
完全加载完成后，创建索引和外键等辅助对象。		数据库管理员

迁移应用程序

Task	描述	所需技能
遵循应用程序迁移策略。		DBA、 SysAdmin、应用程序所有者

割接

Task	描述	所需技能
遵循应用程序割接/切换策略。		DBA、 SysAdmin、应用程序所有者

关闭项目

Task	描述	所需技能
关闭 AWS Secrets Manager 的临时资源。		数据库管理员 ， SysAdmin

Task	描述	所需技能
查看和验证项目文档。		DBA、 SysAdmin、应用程序所有者
收集与迁移时间、手动与工具各自的百分比、成本节约等相关的指标。		DBA、 SysAdmin、应用程序所有者
关闭项目并提供反馈。		

相关资源

参考

- 将 Oracle 数据库迁移至 AWS 的策略
- 将 Oracle 数据库迁移至 AWS Cloud
- 亚马逊 EC2 网站
- AWS DMS 网站
- AWS DMS 博客文章
- 亚马逊 EC2 定价
- 在云计算环境内许可 Oracle 软件

教程和视频

- 亚马逊入门 EC2
- AWS DMS 入门
- Amazon 简介 EC2 -使用 AWS 实现弹性云服务器和托管（视频）

使用 Oracle 数据泵将本地 Oracle 数据库迁移到亚马逊 EC2

由 Navakanth Talluri (AWS) 编写

摘要

迁移数据库时，必须考虑源数据库和目标数据库引擎与版本、迁移工具和服务以及可接受的停机时间等因素。如果您要将本地 Oracle 数据库迁移到亚马逊弹性计算云 (Amazon EC2)，则可以使用 Oracle 工具，例如 Oracle 数据泵和 Oracle Recovery Manager (RMAN)。有关策略的更多信息，请参阅将 Oracle 数据库迁移至 AWS Cloud。

Oracle Data Pump 可帮助您提取数据库的逻辑、一致性备份并将其恢复到目标 EC2 实例。此模式描述了如何使用 Oracle Data Pump 和NETWORK_LINK参数将本地 Oracle 数据库迁移到 EC2 实例，同时最大限度地减少停机时间。NETWORK_LINK 参数通过数据库链接开始导入。目标 EC2 实例上的 Oracle Data Pump Import (impdp) 客户端连接到源数据库，从中检索数据，然后将数据直接写入目标实例上的数据库。此解决方案中没有使用备份或转储文件。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 本地 Oracle 数据库可以：
  - 不是 Oracle Real Application Clusters (RAC) 数据库
  - 不是 Oracle Automatic Storage Management (Oracle ASM) 数据库
  - 处于读写模式。
- 您已在本地数据中心和 AWS 间创建了 AWS Direct Connect 链接。有关更多信息，请参阅创建连接（Direct Connect 文档）。

产品版本

- Oracle Database 10g 版本 1 (10.1)和以上版本

架构

源技术堆栈

- 本地数据中心中的独立（非 RAC 和非 ASM）Oracle 数据库服务器

目标技术堆栈

- 在亚马逊上运行的 Oracle 数据库 EC2

目标架构

AWS Well-Architected Framework 的可靠性支柱建议创建数据备份，以帮助提供高可用性和弹性。有关更多信息，请参阅 AWS 上运行 Oracle 数据库的最佳实践中的高可用性架构。这种模式使用 Oracle Active Data Guard 在 EC2 实例上设置主数据库和备用数据库。为了获得高可用性， EC2 实例应位于不同的可用区。但是，可用区可位于同一 AWS 区域或其他 AWS 区域。

Active Data Guard 提供对物理备用数据库的只读访问权限，并从主数据库持续应用重做更改。根据您的恢复点目标（RPO）和恢复时间目标（RTO），您可在同步重做传输选项和异步重做传输选项之间进行选择。

如果主实例和备用 EC2 实例位于不同的 AWS 区域，则下图显示了目标架构。

数据迁移架构

设置完目标架构后，您可以使用 Oracle Data Pump 将本地数据和架构迁移到主 EC2 实例。在割接期间，应用程序无法访问本地数据库或目标数据库。在这些应用程序可以连接到主 EC2 实例上的新目标数据库之前，请将其关闭。

下图展示了数据迁移过程中的架构。在此示例架构中，主实例和备用 EC2 实例位于不同的 AWS 区域。

工具

Amazon Web Services

- AWS Direct Connect 通过标准的以太网光纤电缆将内部网络链接到 Direct Connect 位置。通过此连接，您可以直接创建连接到公有 Amazon Web Services 的虚拟接口，同时绕过网络路径中的互联网服务提供商。
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。

其他工具和服务

- Oracle Active Data Guard 可帮助您创建、维护、管理和监控备用数据库。
- Oracle 数据泵可帮助您将数据和元数据从一个数据库高速移动至另一个数据库。

最佳实践

- 在 AWS 上运行 Oracle 数据库的最佳实践
- 使用 NETWORK_LINK 导入数据

操作说明

在 AWS 上设置 EC2 实例

Task	描述	所需技能
确定本地主机的源硬件配置和内核参数。	验证本地配置，包括存储大小、每秒进行读写操作的次数（IOPS）和 CPU。这对基于 CPU 内核的 Oracle 许可非常重要。	数据库管理员，SysAdmin
在 AWS 上创建基础设施。	创建虚拟私有云 (VPCs)、私有子网、安全组、网络访问控制列表 (ACLs)、路由表和 Internet 网关。有关更多信息，请参阅下列内容： • VPCs 和子网 • 教程：创建 VPC 以结合数据库实例使用	数据库管理员、AWS 系统管理员
使用 Active Data Guard 设置 EC2 实例。	按照 AWS W EC2 ell-Ar chitected Framework 中所述，使用主动数据保护配置配置 AWS 实例。 EC2 实例上的 Oracle 数据库版本可能与本地版本不同，因为这种模式使用逻辑备份。请注意以下几点：	数据库管理员、AWS 系统管理员

Task	描述	所需技能
	• 将目标数据库置于读写模式。 • 在目标数据库上，提供源数据库的 Transparent Network Substrate (TNS) 详细信息。 有关更多信息，请参阅： • 启动数据库（Oracle 文档） • 创建和配置 Oracle 数据库（Oracle 文档）	

将数据库迁移到 Amazon EC2

Task	描述	所需技能
从 EC2 实例创建指向本地数据库的 dblink。	在 EC2 实例上的 Oracle 数据库和本地 Oracle 数据库之间创建数据库链接 (dblink)。有关更多信息，请参阅使用网络链接导入移动数据（Oracle 文档）。	数据库管理员
验证 EC2 实例与本地主机之间的连接。	使用 dblink 确认 EC2 实例和本地数据库之间的连接是否正常运行。有关说明，请参阅创建数据库链接（Oracle 文档）。	数据库管理员
停止连接到本地数据库的所有应用程序。	批准数据库停机时间后，关闭连接至本地数据库的所有应用程序和相关作业。您可直接从应用程序执行此操作，也可以使用 cron 从数据库执行此操作。有关更多信息，请参阅使	数据库管理员，应用程序开发人员

Task	描述	所需技能
	用 Crontab 实用程序在 Oracle Linux 上计划任务。	
安排数据迁移任务。	在目标主机上，使用命令 `impdb` 安排 Data Pump 导入。这会将目标数据库连接至本地主机并开始数据迁移。有关更多信息，请参阅 Data Pump 导入和 NETWORK_LINK（Oracle 文档）。	数据库管理员
验证数据迁移。	数据验证是关键步骤。对于数据验证，您可使用自定义工具或 Oracle 工具，例如 dblink 和 SQL 查询的组合。	数据库管理员

割接

Task	描述	所需技能
将源数据库置于只读模式。	确认应用程序已关闭并且未对源数据库进行任何更改。以只读模式打开源数据库。这可帮助您避免任何未结事务。有关更多信息，请参阅 SQL 语句中的`ALTER DATABASE`（Oracle 文档）。	DBA、 DevOps 工程师、应用程序开发人员
验证对象数量与数据。	若要验证数据和对象，请使用自定义工具或 Oracle 工具，例如 dblink 和 SQL 查询的组合。	数据库管理员，应用程序开发人员

Task	描述	所需技能
将应用程序连接到主 EC2 实例上的数据库。	更改应用程序的连接属性，使其指向您在主 EC2 实例上创建的新数据库。	数据库管理员，应用程序开发人员
验证应用程序性能。	启动应用程序。使用自动工作负载存储库验证应用程序的功能和性能（Oracle 文档）。	应用程序开发人员、 DevOps 工程师、数据库管理员

相关资源

AWS 参考

- 将 Oracle 数据库迁移至 AWS Cloud
- EC2 适用于甲骨文的亚马逊
- 将庞大的 Oracle 数据库迁移至 AWS 以适应跨平台环境
- VPCs 和子网
- 教程：创建 VPC 以结合数据库实例使用

Oracle 参考

- Oracle Data Guard 配置
- Data Pump 导入

使用 AWS MGN 将 RHEL BYOL 系统迁移至 AWS License-Included 实例

由 Mike Kuznetsov (AWS) 编写

摘要

当您使用 AWS Application Migration Service (AWS MGN) 将工作负载迁移至 AWS 时，可能需要在迁移期间直接迁移（更换主机）您的 Red Hat Enterprise Linux (RHEL) 实例，并将许可从默认的自带许可（BYOL）模型更改为 AWS License Included (LI) 模型。AWS MGN 支持使用亚马逊系统映像 (AMI) IDs 的可扩展方法。本示例介绍了在大规模更换主机迁移期间，如何在 RHEL 服务器上完成许可证变更。它还解释了如何更改已在亚马逊弹性计算云（亚马逊 EC2）上运行的 RHEL 系统的许可。

先决条件和限制

先决条件

- 访问目标 Amazon Web Services account
- AWS MGN 已在目标 Amazon Web Services account 和区域中初始化，以进行迁移（如果您已经从本地系统迁移至 AWS，则无需这样做）
- 具有有效 RHEL 许可的源 RHEL 服务器

架构

此示例介绍了两种场景：

- 使用 AWS MGN 将系统从本地直接迁移至 AWS LI 实例。对于这种情况，请按照第一篇操作说明（迁移至 LI 实例 - 选项 1）和第三篇操作说明中的说明进行操作。
- 将之前已在亚马逊上运行的 RHEL 系统的许可模式从 BYOL 更改为 LI。 EC2对于这种情况，请按照第二篇操作说明（迁移至 LI 实例-选项 2）和第三篇操作说明中进行操作。

> **Note**
>
> 第三篇故事涉及重新配置新的 RHEL 实例，使其使用 AWS 提供的红帽更新基础设施 (RHUI) 服务器。这两个场景的进程相同。

工具

Amazon Web Services

- AWS Application Migration Service (AWS MGN) 可帮助您将应用程序更换主机（直接迁移）到 Amazon Web Services Cloud 中，无需更改且停机时间最短。

操作说明

迁移到 LI 实例 - 选项 1（适用于本地 RHEL 系统）

Task	描述	所需技能
在目标区域中查找 RHEL AWS LI 实例 AMI ID。	访问 AWS Marketp lace 或使用亚马逊 EC2 控制台查找与 RHEL 源系统版本（例如 RHEL-7.7）匹配的 RHEL AMI ID，然后写下 AMI ID。在 Amazon EC2 控制台上 AMIs ，您可以使用以下搜索词之一进行筛选： • 描述 = 由 Red Hat, Inc. 提供 • AMI 名称 = RHEL-7.7	云管理员
配置 AWS MGN 启动设置。	1. 在 AWS MGN 控制台，添加源 RHEL 系统：安装 AWS Replication Agent 并按照 AWS MGN 文档中的说明添加源服务器。 2. 在源服务器页面，选择源 RHEL 系统，然后选择启动设置选项卡。 3. 在 Settings（设置）部分中，选择 Edit（编辑）。若要禁用自动选择并手动指定目标实例类型，请将实例类型大小调整为无，然后选择保存设置。这允许您使用在 Amazon EC2 启动模板中配置的实例类型。有关更多	云管理员

Task	描述	所需技能
	信息，请参阅 AWS MGN 文档。 4. 在 “EC2 启动模板” 部分，选择修改。在关于修改 EC2 启动模板对话框中，再次选择修改。这将打开 Amazon EC2 控制台，因此您可以更改此实例的模板。 5. 查看 AWS MGN 文档中的主要注意事项。 **Note** 您可以忽略警告，不要选择自己的 AMI。 6. 在 Amazon EC2 控制台的新启动模板中，修改以下内容： • 对于 AMI，请指定您之前识别的 AMI ID，或者搜索 RHEL-x 并指定所需版本（例如，RHEL-7.7）。 • 在实例类型，设置所需目标实例类型。 • 保持以下部分不变：密钥对（登录）、网络设置（除非您要指定目标子网和安全组）、存储、资源标签（除非您想添加或修改任何标签）。 • （可选）在高级详细信息部分，如果需要 AWS	

Task	描述	所需技能
	Systems Manager 将来进行管理，请指定 IAM 实例配置文件角色。 7. 选择创建模板版本，然后选择成功消息中的链接以查看启动模板。 8. 选择操作、设置默认版本。对于模板版本，选择最新版本（新系统为版本 2），然后选择设置为默认版本。 AWS MGN 现在将使用此版本启动模板来启动测试或直接割接实例。有关更多信息，请参阅 AWS MGN 文档。	
验证设置。	1. 在 AWS MGN 控制台的源服务器页面，选择您的源服务器，然后选择启动设置选项卡。 2. 在EC2 启动模板部分，验证实例类型、子网和安全组参数设置是否正确。 **Note** 此部分不显示您选择的 AMI ID。要查看 ID，您可以打开 Amazon EC2 控制台、“启动模板”视图，然后搜索本节中显示的模板 ID。	云管理员

Task	描述	所需技能
启动新的 LI 实例。	1. 初始同步完成后，将 AWS MGN 控制台源服务器页面的服务器迁移生命周期列更改为准备测试。若要启动新的测试实例，请选择您的源服务器，打开测试并割接菜单，然后选择启动测试实例。选择查看任务详细信息以监控启动任务的状态。有关更多信息，请参阅 AWS MGN 文档。 2. 等待启动任务完成，然后打开已启动的 EC2 实例详细信息页面。选择详细信息选项卡，并验证实例详细信息部分是否包含以下内容： • 平台详情：“Red Hat Enterprise Linux” • AMI 名称：您在 EC2 启动模板中指定的 AMI 名称 3. 按 AWS MGN 文档中的说明割接到新的 LI 实例。 4. 按照上一篇操作说明中的步骤，重新配置新实例以使用 AWS 提供的 RHUI 服务器。	云管理员

迁移到 LI 实例-选项 2（对于 RHEL BYO EC2 L 实例）

Task	描述	所需技能
将您的 RHEL 自带 EC2 实例迁移到 AWS LI 实例。	您可以将之前作为 BYOL 迁移至 AWS 的 RHEL 系统切换到	云管理员

Task	描述	所需技能
	AWS LI 实例，方法是移动其磁盘（Amazon Elastic Block Store 卷）并将其连接到新的 LI 实例。若要进行此切换，请按照以下步骤操作： 1. 从 RHEL LI AMI 启动新的目标 RHEL 实例。确保您选择的 AMI： • 使用与当前 RHEL 实例相同的 RHEL 版本。 • 启动过程（BIOS 或 UEFI）与您当前的 RHEL 实例相同。例如，如果源服务器基于 BIOS，则使用同样基于 BIOS 的 AWS Marketplace RHEL AMI；对于基于 UEFI 的系统，请选择基于 UEFI 的 AMI。 2. 停止两个实例：新 LI 实例与原始源实例。 3. 将所有 EBS 卷（包括根磁盘）与新 LI 实例分离，然后将其删除。 4. 将所有 EBS 卷（包括根磁盘）与旧源实例分离，然后将其连接到新的 LI 实例。保持卷至设备的映射相同。（例如，以前连接到 /dev/sda 驱动程序的 EBS 卷必须以 /dev/sda 连接至新实例。） 5. 删除源（现为无盘）实例。	

Task	描述	所需技能
	6. 启动新 LI 实例。按下一个操作说明中的步骤，登录实例并对其进行重新配置以使用 AWS 提供的 RHUI 服务器。	

将 RHEL 操作系统重新配置：为使用 AWS 提供的 RHUI — 这两个选项都是

Task	描述	所需技能
从 Red Hat 订阅和许可中注销操作系统。	迁移并成功割接后，必须从 Red Hat 订阅中删除 RHEL 系统，以停止使用 Red Hat 许可证并避免双重计费。 要从红帽订阅中删除 RHEL 操作系统，请按照红帽订阅管理 (RHSM) 文档中描述的流程进行操作。使用 CLI 命令： `subscription-manager unregister` 您也可禁用订阅管理器插件，停止在每次 yum 调用时检查订阅状态。为此，请编辑配置文件 `/etc/yum/pluginconf.d/subscription-manager.conf` ，并将参数 `enabled=1` 更改为 `enabled=0` 。	Linux 或系统管理员
用 AWS 提供的 RHUI 替换旧更新配置（RHUI、Red	您必须重新配置迁移的 RHEL 系统才可使用 AWS 提供的 RHUI 服务器。这样，您无需	Linux 或系统管理员

Task	描述	所需技能

Hat Satellite 网络、yum 存储库）。	外部更新基础设施即可访问 AWS 区域 内的 RHUI 服务器。该更改涉及到以下过程：
	
	1. 备份现有 yum 配置。
	2. 删除旧的 RHUI（yum 存储库）配置和软件包。
	3. 添加 AWS 提供的新 RHUI 配置与证书包。您必须从 AWS 的其他 RHEL 实例检索这些配置包，因为这些配置包仅在 AWS 提供的 RHUI 服务器上可用。
	
	以下是详细步骤和命令：
	
	1. 通过将所有 `/etc/yum*` 和 `/etc/pki/*` 文件夹复制到备份位置，备份现有 yum 配置和证书。例如：
	
	`mkdir yum-backup`
	`cp -ra /etc/yum* /etc/pki ./yum-backup`
	`tar czf yum-backu p.tgz ./yum-backup`
	
	2. 删除旧 RHUI 配置和软件包：
	a. 查找所有已安装 RHUI 软件包：
	
	`sudo rpm -qa | grep rhui`
	
	b. 删除以下软件包：	

Task	描述	所需技能

	`sudo yum remove $(rpm -qa | grep rhui)`
	c. 如果 `/etc/yum/vars/releasever` 文件存在，请将其删除。
	3. 添加 AWS 提供的新 RHUI 与证书包。您必须从 AWS 的另一个 RHEL 实例中检索这些内容。我们可以通过多种方式来实现这一目的。例如，您可以按 Red Hat 知识库文章中提供的说明进行操作：
	a. 从 AWS Marketplace 启动另一个 RHEL (RHEL-EC2) 实例。
	b. 从此实例下载两个软件包：最新 RHUI 客户端配置包和证书颁发机构 (CA) 证书。例如，在桌面运行以下命令：
	`ssh RHEL-EC2 "sudo yumdownloader ca-certificates rh-amazon-rhui-client"`
	c. 将软件包从 RHEL EC2 实例复制到新迁移的系统。例如：	

Task	描述	所需技能

	`scp RHEL-EC2:rh-amazon-rhui-client\* RHEL-EC2:ca-certificates\* .`
	`ssh <migrated-instance> "mkdir /tmp/amazon"`
	`scp rh-amazon-rhui-client* ca-certificates* <migrated-instance>:/tmp/amazon`
	d. 在迁移实例上安装新的 RHUI 和 CA 配置包：
	`ssh <migrated-instance> "sudo rpm -Uhv /tmp/amazon/*"`	

验证配置。	在目标迁移实例上，验证新配置是否正确：
	`sudo yum clean all`
	`sudo yum repolist`
		Linux 或系统管理员

相关资源

- AWS Application Migration Service (AWS MGN) 用户指南
- 获取支持的 AWS RHUI 客户端软件包 IMDSv2（红帽知识库文章）
- 亚马逊 EC2 发布模板（亚马逊 EC2 文档）

将本地 SAP ASE 数据库迁移到亚马逊 EC2

由谢尔盖·德米特里耶夫 (AWS) 和 Gergely Cserdi (AWS) 创作

摘要

此模式描述了如何将 SAP 自适应服务器企业版 (ASE) 数据库从本地主机迁移到亚马逊弹性计算云 (Amazon EC2) 实例。此模式涵盖了使用 AWS Database Migration Service (AWS DMS) 或 ASE Cockpit、Sybase Central for ASE 以及数据库管理员 Cockpit 等 SAP ASE 原生工具迁移的步骤。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 本地数据中心的 SAP AS 源数据库

限制

- 源数据库必须小于 64 TB

产品版本

- SAP ASE 版本 15.x 和 16.x 或更高版本

架构

源技术堆栈

- 本地 SAP ASE 数据库

目标技术堆栈

- EC2 实例上的 SAP ASE 数据库

数据库迁移架构

使用 AWS DMS：

使用原生 SAP ASE 工具：

工具

- AWS DMS - AWS Data Migration Service (AWS DMS) 可支持多种不同的源数据库和目标数据库。有关更多信息，请参阅数据迁移源和数据迁移目标。建议使用最新版本的 AWS DMS，以获得最全面的版本和功能支持。
- SAP ASE - 原生工具包括 ASE Cockpit、适用于 ASE 的 Sybase Central 和数据库管理员 Cockpit。

操作说明

分析迁移

Task	描述	所需技能
验证源数据库和目标数据库版本。		数据库管理员
确定目标操作系统的版本。		数据库管理员，SysAdmin
根据 SAP ASE 兼容性列表和容量要求确定目标服务器实例的硬件要求。		数据库管理员，SysAdmin
确定存储类型和容量的要求。		数据库管理员，SysAdmin
确定网络需求，包括延迟和带宽。		数据库管理员，SysAdmin
选择正确的实例类型、容量、存储功能和网络功能。		数据库管理员，SysAdmin
确定源数据库和目标数据库的网络和主机访问安全要求。		数据库管理员，SysAdmin

Task	描述	所需技能
确定安装 SAP ASE 软件所需操作系统用户列表。		数据库管理员，SysAdmin
确定备份策略。		数据库管理员
确定可用性要求。		数据库管理员
确定应用程序迁移和切换策略。		DBA、SysAdmin、应用程序所有者

配置基础设施

Task	描述	所需技能
创建虚拟私有云（VPC）和子网。		SysAdmin
创建安全组和网络访问控制列表（ACL）。		SysAdmin
配置并启动实 EC2 例。		SysAdmin

安装软件

Task	描述	所需技能
创建运行 SAP ASE 软件所需操作系统用户和组。		数据库管理员，SysAdmin
下载必要版本的 SAP ASE 软件。		数据库管理员，SysAdmin
在 EC2 实例上安装 SAP ASE 数据库、备份服务器软件和复		数据库管理员，SysAdmin

Task	描述	所需技能
制服务器软件，然后配置服务器。		

迁移数据 - 选项 1

Task	描述	所需技能
使用原生 SAP ASE 工具或第三方工具迁移数据库对象和数据。	请参阅 SAP ASE 或第三方工具文档。其中包括 ASE Cockpit、适用于 ASE 的 Sybase Central 和 数据库管理员 Cockpit。	数据库管理员

迁移数据 - 选项 2

Task	描述	所需技能
使用 AWS DMS 迁移数据。		数据库管理员

迁移应用程序

Task	描述	所需技能
遵循应用程序迁移策略。		DBA、 SysAdmin、应用程序所有者

割接

Task	描述	所需技能
遵循应用程序割接或切换策略。		DBA、 SysAdmin、应用程序所有者

关闭项目

Task	描述	所需技能
关闭临时 AWS 资源。		数据库管理员，SysAdmin
验证和查看项目文档。		DBA、SysAdmin、应用程序所有者
收集与迁移时间、手动与工具成本节约比等相关的指标。		DBA、SysAdmin、应用程序所有者
关闭项目并提供任何反馈。		DBA、SysAdmin、应用程序所有者

相关资源

参考

- Amazon EC2
- AWS DMS
- 亚马逊 EC2 定价

教程和视频

- 亚马逊入门 EC2
- AWS Database Migration Service 入门
- AWS Data Migration Service（视频）
- Amazon 简介 EC2 -使用 AWS 实现弹性云服务器和托管（视频）

将本地微软 SQL Server 数据库迁移到亚马逊 EC2

由 Senthil Ramasamy (AWS) 创建

摘要

此模式描述了如何将本地微软 SQL Server 数据库迁移到亚马逊弹性计算云 (亚马逊 EC2) 实例上的微软 SQL Server。它涵盖了两个迁移选项：使用 AWS Database Migration Service (AWS DMS) 或使用本机 Microsoft SQL Server 工具，例如备份和恢复、复制数据库向导或复制并附加数据库。

先决条件和限制

先决条件

- 一个活跃的 AWS 账户
- 亚马逊支持的操作系统 EC2 （有关支持的操作系统版本的完整列表，请参阅亚马逊 EC2 FAQs）
- 本地数据中心的 Microsoft SQL Server 源数据库

产品版本

- 对于本地和 Amazon EC2 实例数据库， AWS DMS 支持：
  - SQL Server 版本 2005、2008、2008R2、2012、2014、2016、2017 和 2019
  - 企业版、标准版、工作组版、开发版和 Web 版
- 有关支持版本的最新列表，请参阅使用 Microsoft SQL Server 数据库作为目标 AWS DMS。

架构

源技术堆栈

- 本地 Microsoft SQL Server 数据库

目标技术堆栈

- EC2 实例上的微软 SQL Server 数据库

目标架构

数据迁移架构

- 使用 AWS DMS

- 使用原生 SQL Server 工具

工具

- AWS Database Migration Service (AWS DMS) 可帮助您将数据迁移到广泛使用的商业和开源数据库，包括 Oracle、SQL Server、MySQL 和 PostgreSQL。您可以使用将数据迁移 AWS DMS 到 AWS Cloud、本地实例之间（通过 AWS Cloud 设置），或者在云和本地设置的组合之间迁移数据。
- AWS Schema Conversion Tool (AWS SCT) 通过自动将源数据库架构和大部分自定义代码转换为与目标数据库兼容的格式来支持异构数据库迁移。
- Microsoft SQL Server 原生工具包括备份和恢复、复制数据库向导以及复制和附加数据库。

操作说明

计划迁移

Task	描述	所需技能
验证源数据库和目标数据库版本。		数据库管理员
确定目标操作系统版本。		数据库管理员、系统管理员
根据 Microsoft SQL Server 兼容性列表和容量要求确定目标服务器实例的硬件要求。		数据库管理员、系统管理员
确定类型和容量的存储要求。		数据库管理员、系统管理员
确定网络需求，包括延迟和带宽。		数据库管理员、系统管理员

Task	描述	所需技能
根据容量、存储功能和网络功能选择 EC2 实例类型。		数据库管理员、系统管理员
确定源数据库和目标数据库的网络和主机访问安全要求。		数据库管理员、系统管理员
确定安装 Microsoft SQL 软件所需用户列表。		数据库管理员、系统管理员
确定备份策略。		数据库管理员
确定可用性要求。		数据库管理员
确定应用程序迁移和割接策略。		数据库管理员、系统管理员

配置基础设施

Task	描述	所需技能
创建虚拟私有云（VPC）和子网。		系统管理员
创建安全组和网络访问控制列表（ACL）。		系统管理员
配置并启动实 EC2 例。		系统管理员

安装软件

Task	描述	所需技能
为 Microsoft SQL Server 软件创建所需用户和组。		数据库管理员、系统管理员

Task	描述	所需技能
下载 Microsoft SQL Server 软件。		数据库管理员、系统管理员
在 EC2 实例上安装 Microsoft SQL Server 软件并配置服务器。		数据库管理员、系统管理员

迁移数据 - 选项 1

Task	描述	所需技能
使用原生 Microsoft SQL Server 工具或第三方工具迁移数据库对象和数据。	工具包括备份和恢复、Copy Database Wizard 以及复制与附加数据库。有关更多信息，请参阅指南将 Microsoft SQL Server 数据库迁移到 AWS Cloud。	数据库管理员

迁移数据 - 选项 2

Task	描述	所需技能
使用 AWS DMS 迁移数据。	有关使用的更多信息 AWS DMS，请参阅 “相关资源” 部分中的链接。	数据库管理员

迁移应用程序

Task	描述	所需技能
遵循应用程序迁移策略。	使用 AWS Schema Conversion Tool (AWS SCT) 分析和修	数据库管理员、应用程序所有者

Task	描述	所需技能
	改嵌入在应用程序源代码中的 SQL 代码。	

割接

Task	描述	所需技能
遵循应用程序切换策略。		数据库管理员、应用程序所有者、系统管理员

关闭项目

Task	描述	所需技能
关闭所有临时 AWS 资源。	临时资源包括 AWS DMS 复制实例和的 EC2 实例 AWS SCT。	数据库管理员、系统管理员
查看和验证项目文档。		数据库管理员、应用程序所有者、系统管理员
收集与迁移时间、手动与工具占比、成本节约等相关的指标。		数据库管理员、应用程序所有者、系统管理员
关闭项目并提供反馈。		数据库管理员、应用程序所有者、系统管理员

相关资源

参考

- 将微软 SQL Server 数据库迁移到 AWS Cloud
- Amazon EC2

- Amazon EC2 FAQs
- 亚马逊 EC2 定价
- AWS Database Migration Service
- 微软产品开启 AWS
- 微软许可开启 AWS
- 微软 SQL Server 开启 AWS

教程和视频

- 亚马逊@@ 入门 EC2
- https://aws.amazon.com/dms/getting-started/ 入门AWS Database Migration Service
- 将亚马逊 EC2 实例加入您的 Simple AD 活动目录
- 将 Amazon EC2 实例加入您的 AWS Managed Microsoft AD 活动目录
- AWS Database Migration Service（视频）
- Amazon 简介 EC2 — 弹性云服务器和托管 AWS（视频）

将本地 MySQL 数据库迁移到亚马逊 EC2

由 Lorenzo Mota (AWS) 编写

摘要

此模式为将本地 MySQL 数据库迁移到亚马逊弹性计算云 (亚马逊 EC2) 实例上的 MySQL 数据库提供了指导。该模式讨论了如何使用 AWS Database Migration Service (AWS DMS) 或原生 MySQL 工具（例如 my sqldump）进行迁移。它侧重于将数据库完整迁移到 MySQL 数据库实例。

该模式主要面向 DBAs 解决方案架构师。它可用于小型或大型项目、测试或最终迁移阶段。我们建议您在生产环境中使用此模式之前，至少运行一个测试周期。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 本地数据中心的 MySQL 源数据库

产品版本

- MySQL 版本 5.5 及更高版本
- 亚马逊支持的目标操作系统 EC2；参见 Ama z on EC2 FAQs

架构

源技术堆栈

- 本地 MySQL 数据库

目标技术堆栈

- 亚马逊上的 MySQL 数据库实例 EC2

AWS 数据迁移方法

- AWS DMS
- 像 mysq ldump 这样的本机 MySQ L 工具，或者像 Percona 这样的第三方工具 XtraBackup

目标架构

下图说明了转换后的目标 Amazon EC2 实施。

AWS 数据迁移架构

使用 AWS DMS：

下图说明了在切换之前向目标 MySQL 数据库发送完整和增量更改所基于 AWS DMS 的数据迁移工作流程。从本地到的网络连接 AWS 取决于 SQL 客户端的要求，不在此模式的范围之内。

使用其他 MySQL 工具：

下图说明了基于使用 MySQL 工具从本地数据库生成导出转储文件的数据迁移工作流程。在转换之前，这些文件将移至亚马逊简单存储服务 (Amazon S3)，并导入到目标 MySQL 数据库。从本地到的网络连接 AWS 取决于 SQL 客户端的要求，不在此模式的范围之内。

注意:

- 根据停机时间考虑因素和最终转换的数据库大小，您可以使用 AWS DMS 或其他变更数据捕获 (CDC) 工具来最大限度地缩短切换时间。当您使用诸如之类的 CDC 工具时 AWS DMS，可以在几分钟内迁移到目标数据库。
- 如果数据库的大小和网络延迟允许较短的直接转换迁移窗口，那么使用 mysqldump 的离线策略就足够了。（我们建议您进行测试以获得大致时间。）
- 通常，CDC策略比离线选项 AWS DMS 需要更多的监控和复杂性。

工具

AWS 服务

- AWS Database Migration Service (AWS DMS) 支持多个源数据库和目标数据库。有关支持的 MySQL 源数据库和目标数据库的信息 AWS DMS，请参阅使用与 MySQL 兼容的数据库作为源 AWS DMS和使用兼容 MySQL 的数据库作为目标。 AWS DMS如果您的源数据库不受支持 AWS DMS，则必须选择其他方法来迁移数据。

其他工具

- my@@ sqldump 是一个 MySQL 实用工具，它从 MySQL 数据库创建转储文件用于备份或迁移。
- Percona XtraBackup 是一个开源实用程序，用于在 MySQL 数据库上执行非阻塞备份。

操作说明

计划迁移

Task	描述	所需技能
验证数据库版本。	验证源数据库和目标数据库的版本。有关支持的 MySQL 版本的信息 AWS DMS，请参阅 AWS DMS 文档 AWS DMS中的源 AWS DMS和目标。	数据库管理员
确定目标操作系统。	确定目标操作系统的版本。有关亚马逊支持的目标操作系统的列表 EC2，请参阅亚马逊 EC2 FAQs。	数据库管理员、系统管理员
确定硬件要求。	根据 MySQL 兼容性列表和容量要求确定目标服务器实例的硬件要求。	数据库管理员、系统管理员
确定存储要求。	确定目标数据库的存储类型和容量。	数据库管理员、系统管理员
识别网络要求。	确定网络要求，例如延迟和带宽。	数据库管理员、系统管理员
选择目标实例类型。	根据容量、存储功能和网络功能选择目标实例类型。	数据库管理员、系统管理员
确定安全要求。	确定源数据库和目标数据库的网络或主机访问安全要求。	数据库管理员、系统管理员

Task	描述	所需技能
识别用户。	确定安装 MySQL 软件的操作系统用户列表。有关更多信息，请参阅 MySQL 文档。	数据库管理员、系统管理员
确定备份策略。		数据库管理员
确定可用性要求。		数据库管理员
确定应用程序迁移或切换策略。		数据库管理员、系统管理员

配置基础设施

Task	描述	所需技能
创建虚拟私有云（VPC）和子网。	配置路由表、互联网网关、NAT 网关和子网。有关更多信息，请参阅 Amazon VPC 文档中的 VPC 配置选项。	系统管理员
创建安全组和网络访问控制列表 (ACLs)。	根据您的要求配置端口（MySQL 的默认值为 3306）和 CIDR 范围或特 IPs 定。	系统管理员
配置并启动实 EC2 例。	有关说明，请参阅 Amazon EC2 文档中的启动 EC2 实例。	系统管理员

安装 MySQL 软件

Task	描述	所需技能
创建用户和群组。	创建需要访问服务器和数据库的操作系统用户和组。有关更	数据库管理员、系统管理员

Task	描述	所需技能
	多信息，请参阅 MySQL 文档中的访问控制和账户管理。	
下载 MySQL。	下载 MySQL 软件。有关说明和二进制文件，请参阅 MySQL 文档中的安装 M ySQL。	数据库管理员、系统管理员
在 EC2 实例上安装 MySQL 并配置服务器。	连接到您的 EC2 实例并安装 MySQL 软件。有关更多信息，请参阅 Amazon EC2 文档中的 Connect 到您的 EC2 实例。	数据库管理员、系统管理员

迁移数据-选项 1

Task	描述	所需技能
使用原生 MySQL 或第三方工具迁移数据。	此选项使用本机 MySQL 工具或第三方工具来迁移数据库对象和数据。有关说明，请参阅 mysqldump 或 Percona XtraBackup（用于物理迁移）的文档。有关使用这些工具的更多信息，请参阅 AWS 博客文章 MySQL 到 Amazon RDS for MySQL 或 Amazon Aurora MySQL 的迁移选项。	数据库管理员

迁移数据-选项 2

Task	描述	所需技能
使用迁移数据 AWS DMS。	有关更多信息，请参阅 AWS DMS 文档AWS DMS中的高级视图。	数据库管理员

为转换做好准备

Task	描述	所需技能
收集物体数量。	从源数据库和新的目标数据库收集对象计数。修复目标数据库中的任何差异。	数据库管理员
检查依赖项。	确认与其他数据库的依赖关系（链接）仍然有效且工作正常。	数据库管理员
测试。	如果这是测试周期，请执行查询测试，收集指标并修复所有问题。	数据库管理员

割接

Task	描述	所需技能
移动客户。	将应用程序客户端切换至新基础设施。	数据库管理员、应用程序所有者、系统管理员
提供支持。	在功能应用程序测试期间提供支持。	数据库管理员

关闭项目

Task	描述	所需技能
关闭资源。	关闭 AWS DMS 复制实例和其他临时 AWS 资源。	数据库管理员、系统管理员
审查和项目文件。	查看和验证项目文档。	数据库管理员、应用程序所有者、系统管理员

Task	描述	所需技能
收集指标。	收集指标，例如迁移时间、手动更改与工具辅助更改相比的百分比以及成本节约情况。	数据库管理员、应用程序所有者、系统管理员
结束项目。	结束迁移项目并提供反馈。	数据库管理员、应用程序所有者、系统管理员
停用源数据库。	停用本地 MySQL 数据库。	数据库管理员、系统管理员

相关资源

参考

- 亚马逊 EC2 文档
- AWS DMS 文档
- 亚马逊 EC2 定价
- AWS DMS Step-by-Step 演练
- mysqldump
- Percona XtraBackup

教程和视频

- 入门 AWS DMS
- Amazon 简介 EC2 — 弹性云服务器和托管 AWS（视频）

使用 Application Migration Service 缩短同构 SAP 迁移割接时间

由 Pavel Rubin (AWS)、Diego Valverde (AWS) 以及 Sunil Yadav (AWS) 编写

摘要

此模式概述了使用 AWS Application Migration Service 迁移 SAP 工作负载的步骤。Application Migration Service 通过使用块级复制来维护从源持续同步的复制卷，从而简化了割接。

SAP 工作负载包括 SAP Customer Relationship Management (SAP CRM)、SAP Enterprise Resource Planning (ERP) 以及 SAP Business Warehouse (SAP BW) 应用程序。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account 在 AWS 上的源 SAP 服务器和 AWS 上的目标虚拟私有云（VPC）之间具有稳定的网络连接
- 本地数据中心中适用于 Linux 或 Windows 的 SAP 自适应服务器企业版 (ASE) 源数据库

限制

- 目标操作系统必须由亚马逊弹性计算云 (Amazon EC2) 支持。有关更多信息，请参阅 Amazon EC2 FAQs。

架构

源技术堆栈

- SAP ASE 数据库

目标技术堆栈

- Amazon EC2
- Amazon Elastic Block Store（Amazon EBS）

源架构和目标架构

下图显示了通过复制代理从本地服务器迁移至 Application Migration Service 端点。Amazon Simple Storage Service (Amazon S3) 端点用于访问安装和配置文件。暂存区和迁移资源的子网包含 EC2 实

例，数据存储在 EBS 卷上。端口 TCP 443 用于将源计算机网络连接到应用程序迁移服务，以及将暂存区域子网连接到应用程序迁移服务、Amazon 和 Amazon S3 区域 EC2终端节点。端口 TCP 1500 用于本地网络和暂存区域之间的数据复制。

工具

- AWS 应用程序迁移服务可帮助您将应用程序重新托管 (lift-and-shift) 到 AWS 云中，无需更改且停机时间最短。
- 亚马逊弹性区块存储 (Amazon EBS) E lastic Block Store 提供块级存储卷，用于亚马逊弹性计算云 (Amazon) 实例。 EC2
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Security Token Service (AWS STS) 可帮助您为用户申请临时、权限有限的凭证。

操作说明

初始化 Application Migration Service

Task	描述	所需技能
初始化 Application Migration Service。	在要部署 SAP ASE 数据库的 AWS 区域中初始化 Applicati on Migration Service。当您首次导航到每个区域的 Applicati on Migration Service 页面时，AWS 会提供自动设置。	AWS 管理员
手动创建服务角色。	（可选）如果您想使用自动化 (例如 AWS Control Tower) 来设置账户，则可以手动创建安装、复制和启动所需六个 AWS 身份和访问管理 (IAM) 角色。	AWS 管理员

Task	描述	所需技能
	有关说明，请参阅 AWS 文档。	
创建复制设置模板。	复制设置模板定义子网、实例类型、Amazon EBS 加密及数据路由方式。有关详细设置信息，请参阅 AWS 文档。	常规 AWS

为代理安装生成凭证

Task	描述	所需技能
创建新 IAM 角色。	在 IAM 控制台中，导航至 Roles（角色），然后选择 Create Role（创建角色）。 对于可信实体类型，选择 Amazon Web Services account，然后选择下一步。	AWS 系统管理员
附加 AWSApplicationMigrationAgentPolicy 到 IAM 角色。	AWS 托管式 `AWSApplicationMigrationAgentPolicy` 策略包含执行 Application Migration Service 代理安装所需权限。 附加策略后，选择下一步。	AWS 系统管理员
完成角色创建。	指定一个友好名称，然后选择创建角色。	AWS 系统管理员
生成临时凭证。	若要生成访问密钥 ID、秘密访问密钥和会话令牌，请按照 AWS STS 文档中的说明进行操作。这些凭证在代理安装期间使用。	AWS 系统管理员

在 SAP 源机上安装 Application Migration Service Agent

Task	描述	所需技能
在 SAP 源机下载 Agent 安装程序。	下载适用于您的源操作系统（Windows 或 Linux）的代理安装程序。	应用程序所有者
安装 AWS Replication Agent。	当您在源机上运行 Agent 安装程序文件时，系统首先会要求您输入访问密钥、秘密访问密钥、会话令牌以及要复制到的区域。使用您之前创建的 IAM 角色中的临时凭证，以及您在初始化期间配置的同一区域。	应用程序所有者
等待初始数据复制。	安装代理后，源机将出现在 Application Migration Service 控制台的计算机选项卡。	应用程序所有者

配置目标计算机启动模板

Task	描述	所需技能
更新源服务器启动模板。	每台源服务器都使用唯一的 EC2 Launch 模板来通知目标 EC2 服务器的配置。如果您想自定义迁移服务器的 Amazon EC2 配置，则可以编辑此模板。	常规 AWS
设置默认启动模板版本。	对启动模板进行所需更改后，指定使用此更新版本作为默认启动模板。有关更多信息，请参阅 AWS 文档。	常规 AWS

Task	描述	所需技能
关闭实例类型，正确调整大小。	（可选）正确调整实例类型会根据源 SAP 服务器的配置自动提供实例类型建议。我们建议关闭此设置，以便您可在 Launch 模板中指定自定义的实例类型。	常规 AWS

执行测试

Task	描述	所需技能
启动测试启动程序。	在 Application Migration Service 控制台，选择一台或多台服务器，然后在测试和割接下选择启动测试实例。	常规 AWS，迁移工程师，迁移主管
等待转换和启动过程完成。	您可在启动历史记录选项卡上查看启动过程。计算机作为 EC2 实例成功启动后，“警报”选项卡将更新为 “已启动”。	
验证测试是否成功完成。	通过远程桌面协议 (RDP) 或 SSH (Secure Shell) 连接到启动的实例，然后执行相应的应用程序检查。例如登录 SAP 界面并验证功能。	迁移工程师、应用程序所有者
更新源生命周期。	如果测试成功，请在测试和割接选项卡上将源计算机生命周期更新标记为“准备割接”。	迁移工程师，迁移主管

安排并执行与 Amazon 目标的切换 EC2

Task	描述	所需技能
计划割接时段。		割接负责人，迁移主管，应用程序所有者
发起割接启动。	选择一个或多个服务器。在 Application Migration Service 控制台的测试和割接选项卡下的测试和割接下选择启动割接实例。	迁移工程师
等待割接和启动过程完成。	您可在启动历史记录选项卡上查看启动过程。计算机作为 EC2 实例成功启动后，“警报”选项卡将更新为 “已启动”。	
验证割接是否成功完成。	通过 RDP 或 SSH 连接至启动的实例，然后执行相应的应用程序检查。	应用程序所有者、迁移工程师
更新源生命周期。	如果割接成功，请在测试和割接选项卡上选择完成割接，更新源计算机生命周期。	迁移工程师

相关资源

参考

- AWS Application Migration Service
- AWS Application Migration 常见问题解答

视频

- AWS Application Migration Service 架构

在 Amazon Web Services Cloud 中重新托管本地工作负载：迁移核对清单

由 Srikanth Rangavajhala (AWS) 编写

摘要

在 Amazon Web Services (AWS) Cloud 中重新托管本地工作负载涉及以下迁移阶段：规划、预发现、发现、构建、测试和割接。该模式概述了各个阶段及其相关任务。这些任务进行了高层描述，支持大约 75% 的应用程序工作负载。您可以在敏捷冲刺周期中，用两到三周的时间实施这些任务。

您应该与迁移团队和顾问一起审查这些任务。审核后，您可收集意见，根据需要删除或重新评估任务以满足您的要求，还可以修改其他任务以支持您的投资组合中至少 75% 的应用程序工作负载。然后，您可使用 Atlassian Jira 或 Rally Software 等敏捷项目管理工具导入任务，将其分配至资源并跟踪您的迁移活动。

该模式假设您正在使用 AWS Cloud Migration Factory 来重新托管工作负载，但您可使用自己选择的迁移工具。

Amazon Macie 可以帮助识别知识库中的敏感数据，这些数据存储为数据源、模型调用日志，并提示存储在亚马逊简单存储服务 (Amazon S3) 存储桶中。 有关更多信息，请参阅 Macie 文档。

先决条件和限制

先决条件

- 用于跟踪迁移任务的项目管理工具（例如 Atlassian Jira 或 Rally Software）
- 用于在 AWS 上重新托管工作负载的迁移工具（例如 Cloud Migration Factory）

架构

源平台

- 本地源堆栈（包括技术、应用程序、数据库和基础设施）

目标平台

- Amazon Web Services Cloud 目标堆栈（包括技术、应用程序、数据库和基础设施）

架构

下图说明了使用 Cloud Migration Factory 和 AWS Application Migration Service 进行重新托管（发现服务器并将其从本地源环境迁移到 AWS）。

工具

- 您可使用您选择的迁移和项目管理工具。

操作说明

规划阶段

Task	描述	所需技能
整理发现前的待办事项。	与部门领导和应用程序所有者一起召开发现前待办事项整理工作会议。	项目经理、敏捷 Scrum 领导者
召开冲刺计划工作会话。	作为范围界定练习，在冲刺和波次之间分发您想要迁移的应用程序。	项目经理、敏捷 Scrum 领导者

发现前阶段

Task	描述	所需技能
确认应用程序知识。	确认并记录应用程序所有者及其对应用程序的了解。确定是否还有其他负责技术问题关键人物。	迁移专家（面试官）
确定应用程序合规要求。	与应用程序所有者确认该应用程序不必遵守支付卡行业数据安全标准 (PCI DSS)、萨班斯-奥克斯利法案 (SOX)、个人身份信息（PII）或其他标准的要求。如果存在合规性要求，团	迁移专家（面试官）

Task	描述	所需技能
	队必须完成对将迁移的服务器的合规性检查。	
确认生产版本的要求。	向应用程序所有者或技术联系人确认将迁移的应用程序发布到生产环境的要求（例如发布日期和停机时间）。	迁移专家（面试官）
获取服务器列表。	获取与此目标应用程序关联的服务器列表。	迁移专家（面试官）
获取显示当前状态的逻辑图。	从企业架构师或应用程序所有者处获取应用程序的当前状态图。	迁移专家（面试官）
创建显示目标状态的逻辑图。	创建应用程序的逻辑图，显示 AWS 上的目标架构。该图应说明服务器、连接性和映射因素。	企业架构师、企业主
获取服务器信息。	收集与应用程序关联的服务器相关信息，包括其配置详细信息。	迁移专家（面试官）
将服务器信息添加至发现模板中。	将详细的服务器信息添加到应用程序发现模板 `mobilize-application-questionnaire.xlsx` 中（有关此模式，请参阅附件中的）。此模板包括所有与应用程序相关的安全、基础设施、操作系统和联网详细信息。	迁移专家（面试官）
发布应用程序发现模板。	与应用程序所有者和迁移团队共享应用程序发现模板，以便共同访问和使用。	迁移专家（面试官）

发现阶段

Task	描述	所需技能
确认服务器列表。	与应用程序所有者或技术主管确认服务器列表以及每个服务器的用途。	迁移核对清单
识别和添加服务器组。	识别服务器组（例如 Web 服务器或应用程序服务器），并将此信息添加到应用程序发现模板中。选择每台服务器应属于的应用程序层（Web、应用程序、数据库）。	迁移核对清单
填写应用程序发现模板。	在迁移团队、应用程序团队和 AWS 的帮助下，完成应用程序发现模板详细信息。	迁移核对清单
添加缺少的服务器详细信息（中间件和操作系统团队）。	要求中间件和操作系统 (OS) 团队检查应用程序发现模板并添加任何缺少的服务器详细信息，包括数据库信息。	迁移核对清单
获取入站/出站流量规则（网络小组）。	要求网络团队获取源服务器和目标服务器的入站/出站流量规则。网络团队还应添加现有的防火墙规则，将其导出为安全组格式，并将现有的负载均衡器添加到应用程序发现模板中。	迁移核对清单
确定所需标记。	确定应用程序标签要求。	迁移核对清单
创建防火墙请求详细信息。	捕获并过滤与应用程序通信所需防火墙规则。	迁移专家、解决方案架构师、网络主管
更新 EC2 实例类型。	根据基础设施和服务器要求，更新要在目标环境中使用的	迁移专家、解决方案架构师、网络主管

Task	描述	所需技能
	亚马逊弹性计算云 (Amazon EC2) 实例类型。	
确定当前状态图。	识别或创建显示应用程序当前状态图表。此图表将用于信息安全 (InfoSec) 请求。	迁移专家、解决方案架构师
完成未来状态图。	完成显示应用程序未来（目标）状态的示意图。此图表也将在 InfoSec 请求中使用。	迁移专家、解决方案架构师
创建防火墙或安全组服务请求。	创建防火墙或安全组服务请求（用于开发/QA、预生产和生产）。如果您使用的是 Cloud Migration Factory，请包括特定于复制的端口（如果它们尚未打开）。	迁移专家、解决方案架构师、网络主管
查看防火墙或安全组请求（InfoSec 团队）。	在此步骤中， InfoSec 团队将审查并批准在上一步中创建的防火墙或安全组请求。	InfoSec 工程师，迁移专家
实施防火墙安全组请求（网络小组）。	在 InfoSec 团队批准防火墙请求后，网络团队将实施所需的入站/出站防火墙规则。	迁移专家、解决方案架构师、网络主管

构建阶段（在开发/QA、预生产和生产环境中重复执行）

Task	描述	所需技能
导入应用程序与服务器数据。	1. 验证您是否以对范围内源服务器具有本地管理员权限的域用户身份登录到迁移执行服务器。 2. 使用迁移导入表单导入范围内源服务器的属性。有	迁移专家、云管理员

Task	描述	所需技能
	关更多信息，请参阅 Cloud Migration Factory 实施指南。 如果您没有使用 Cloud Migration Factory，请按照说明设置迁移工具。	
检查源服务器先决条件。	连接范围内的源服务器以验证先决条件，例如 TCP 端口 1500、TCP 端口 443、根卷可用空间、.NET Framework 版本以及其他参数。这些是复制要求。有关更多信息，请参阅 Cloud Migration Factory 实施指南。	迁移专家、云管理员
创建安装复制代理服务请求。	创建服务请求，以在范围内的服务器上安装复制代理以进行开发/QA、预生产或生产。	迁移专家、云管理员
安装复制代理。	在开发/QA、预生产或生产计算机范围内源服务器上安装复制代理。有关更多信息，请参阅 Cloud Migration Factory 实施指南。	迁移专家、云管理员

Task	描述	所需技能
推送启动后脚本。	Application Migration Service 支持启动后脚本，以帮助您在启动目标实例后自动执行操作系统级别的活动，例如安装或卸载软件。此步骤将启动后的脚本推送至 Windows 或 Linux 计算机，具体取决于确定要迁移的服务器。有关说明，请参阅 Cloud Migration Factory 实施指南。	迁移专家、云管理员
验证复制状态。	使用提供的脚本自动确认范围内源服务器的复制状态。该脚本每五分钟重复一次，直到给定波次中所有源服务器的状态更改为健康。有关说明，请参阅 Cloud Migration Factory 实施指南。	迁移专家、云管理员
创建管理员用户。	从范围内的源服务器迁移割接至 AWS 后，可能需要源计算机上的本地管理员或 sudo 用户来故障排除任何问题。当身份验证服务器（例如 DC 或 LDAP 服务器）无法访问时，迁移团队会使用此用户登录目标服务器。有关此步骤的说明，请参阅 Cloud Migration Factory 实施指南。	迁移专家、云管理员
验证启动模板。	验证服务器元数据，以确保其成功运行且没有无效数据。此步骤将验证测试与割接元数据。有关说明，请参阅 Cloud Migration Factory 实施指南。	迁移专家、云管理员

测试阶段（在开发/QA、预生产和生产环境中重复测试）

Task	描述	所需技能
创建服务请求。	为基础设施团队和其他团队创建服务请求，以将应用程序割接到开发/QA、预生产或生产实例。	迁移专家、云管理员
配置负载均衡器（可选）。	使用 iRules 配置所需负载均衡器，例如应用程序负载均衡器或 F5 负载均衡器。	迁移专家、云管理员
启动实例进行测试。	在测试模式下，在 Application Migration Service 中启动给定波次的所有目标计算机。有关更多信息，请参阅 Cloud Migration Factory 实施指南。	迁移专家、云管理员
验证目标实例状态。	通过检查同一波中所有范围内源服务器的启动过程，来验证目标实例的状态。目标实例启动可能需要多达 30 分钟。您可以通过登录 Amazon EC2 控制台、搜索源服务器名称并查看 “状态检查” 列来手动检查状态。通过状态 2/2 检查表明该实例从基础设施的角度来看是健康的。	迁移专家、云管理员
修改 DNS 条目。	修改域名系统（DNS）条目。(对于 Microsoft Windows 环境，请使用 `resolv.conf` 或 `host.conf` 。) 将每个 EC2 实例配置为指向该主机的新 IP 地址。	迁移专家、云管理员

Task	描述	所需技能
	Note 确保本地服务器和 AWS 云服务器之间没有 DNS 冲突。此步骤和以下步骤是可选的，具体取决于托管服务器环境。	
测试 EC2 实例与后端主机的连接。	使用已迁移服务器的域凭证查验登录信息。	迁移专家、云管理员
更新 DNS A 记录。	更新每台主机的 DNS A 记录，使其指向新的亚马逊 EC2 私有 IP 地址。	迁移专家、云管理员
更新 DNS CNAME 记录。	更新虚拟 IPs（负载均衡器名称）的 DNS CNAME 记录，使其指向 Web 和应用程序服务器的集群。	迁移专家、云管理员
在适用环境中测试应用程序。	登录新 EC2 实例，在开发/QA、预生产和生产环境中测试应用程序。	迁移专家、云管理员
标记为已准备好进行割接。	测试完成后，更改源服务器的状态以表明它已准备好进行割接，这样用户就可以启动割接实例。有关说明，请参阅 Cloud Migration Factory 实施指南。	迁移专家、云管理员

割接阶段

Task	描述	所需技能
创建生产部署计划。	制定生产部署计划（包括回退计划）。	迁移专家、云管理员
将停机时间告知运营团队。	将服务器的停机时间表告知运营团队。有些团队可能需要变更请求或服务请求 (CR/SR) 票证才能收到此通知。	迁移专家、云管理员
复制生产机器。	使用 Application Migration Service 或其他迁移工具复制生产计算机。	迁移专家、云管理员
关闭范围内源服务器。	验证源服务器的复制状态后，您可以关闭源服务器以停止从客户端应用程序到服务器的事务。可在割接窗口中关闭源服务器。有关更多说明，请参阅 Cloud Migration Factory 实施指南。	云管理员
启动割接实例。	在应用程序迁移服务中以割接模式启动给定波次的所有目标计算机。有关更多说明，请参阅 Cloud Migration Factory 实施指南。	迁移专家、云管理员
检索目标实例 IPs。	检索 IPs 目标实例的。如果 DNS 更新在您的环境中是手动过程，则需要获取所有目标实例的新 IP 地址。有关更多说明，请参阅 Cloud Migration Factory 实施指南。	迁移专家、云管理员

Task	描述	所需技能
验证目标服务器连接。	更新 DNS 记录后，使用主机名连接到目标实例以验证连接。有关更多说明，请参阅 Cloud Migration Factory 实施指南。	迁移专家、云管理员

相关资源

- 如何迁移
- AWS Cloud Migration Factory 实施指南
- 使用 Cloud Migration Factory 自动化大规模服务器迁移
- AWS Application Migration Service 用户指南
- AWS 迁移加速计划

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 Amazon 为 SQL Server Always On FCI 设置多可用区基础架构 FSx

由 Manish Garg (AWS)、T.V.R.L.Phani Kumar Dadi (AWS)、Nishad Mankar (AWS) 和 RAJNEESH TYAGI (AWS) 编写

摘要

如果你需要快速迁移大量 Microsoft SQL Server Always On 故障转移群集实例 (FCIs)，这种模式可以帮助你最大限度地缩短配置时间。通过使用自动化和 Amazon FSx for Windows 文件服务器，它可以减少手动工作、人为错误和部署大量集群所需的时间。

这种模式 FCIs 在 Amazon Web Services (AWS) 上的多可用区（多可用区）部署中为 SQL Server 设置基础设施。使用 AWS CloudFormation 模板可以自动配置该基础设施所需的 AWS 服务。使用 PowerShell 命令在亚马逊弹性计算云 (Amazon EC2) 实例上安装 SQL Server 和创建集群节点。

该解决方案使用高度可用的多可用区 A mazon f FSx or Windows 文件系统作为存储 SQL Server 数据库文件的共享见证。亚马逊 FSx 文件系统和托管 SQL Server 的 EC2 Windows 实例加入了适用于微软 Active Directory 的同一 AWS 目录服务（AWS 托管的微软 AD）域。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 具有足够权限的 AWS 用户，可以使用 AWS CloudFormation 模板配置资源
- AWS Directory Service for Microsoft Active Directory
- AWS Secrets Manager 中用于向键值对中的 AWS Managed Microsoft AD 进行身份验证的凭证：
  - `ADDomainName`: <域名>
  - `ADDomainJoinUserName`: <域用户名>
  - `ADDomainJoinPassword`:<域用户密码>
  - `TargetOU`: <目标 OU 值>

> **Note**
>
> 在 AWS Systems Manager 自动化中，您将在 AWS 托管的 Microsoft AD 加入活动中使用相同的密钥名称。

- 用于创建 SQL Server 安装和 Windows 服务或域账户的 SQL Server 媒体文件，这些文件将在集群创建期间使用

- 虚拟私有云（VPC），具有位于不同可用区的两个公有子网、可用区中的两个私有子网、一个互联网网关、NAT 网关、路由表关联和一个跳转服务器

产品版本

- Windows Server 2012 R2 和 Microsoft SQL Server 2016

架构

源技术堆栈

- FCIs 使用共享云端硬盘的本地 SQL Server

目标技术堆栈

- AWS EC2 实例
- FSx 适用于 Windows 文件服务器的亚马逊
- AWS Systems Manager Automation 运行手册
- 网络配置（VPC、子网、互联网网关、NAT 网关、跳转服务器、安全组）
- AWS Secrets Manager
- AWS 托管的 Microsoft AD
- Amazon EventBridge
- AWS Identity and Access Management (IAM)

目标架构

下图显示了单个 AWS 区域中的一个 AWS 账户，其中一个 VPC 包括两个可用区、两个带有 NAT 网关的公有子网、第一个公有子网中的跳转服务器、两个私有子网（每个子网都有一个用于节点安全组中的 SQL Server 节点的 EC2 实例）以及一个连接到每个 SQL Server 节点的 Amazon FSx 文件系统。还包括 AWS Directory Service EventBridge、亚马逊、AWS Secrets Manager 和 AWS Systems Manager。

自动化和扩缩

- 您可以使用 AWS Systems Manager 加入 AWS 托管 Microsoft AD 并执行 SQL Server 安装。

工具

Amazon Web Services

- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。
- AWS Di rectory Service 提供了多种使用微软活动目录 (AD) 与其他AWS服务的方式，例如亚马逊弹性计算云 (亚马逊 EC2)、适用于SQL Server的亚马逊关系数据库服务 (Amazon RDS) 和 FSx 适用于Windows文件服务器的亚马逊。
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如，AWS Lambda 函数、使用 API 目标的 HTTP 调用端点或其他 Amazon Web Services account 中的事件总线。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Secrets Manager 帮助您将代码中的硬编码凭证（包括密码）替换为对 Secrets Manager 的 API 调用，以便以编程方式检索密钥。
- AWS Systems Manager 可帮助您管理在 Amazon Web Services Cloud 中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。

其他工具

- PowerShell是一款在 Windows、Linux 和 macOS 上运行的微软自动化和配置管理程序。此模式使用 PowerShell 脚本。

代码存储库

此模式的代码可在 GitHub aws-windows-failover-cluster-automat ion存储库中找到。

最佳实践

- 用于部署此解决方案的 IAM 角色应遵守最低权限原则。有关更多信息，请参阅 IAM 文档。
- 遵循 A WS CloudFormation 最佳实践。

操作说明

部署基础设施

Task	描述	所需技能
部署 Systems Manager CloudFormation 堆栈。	1. 登陆您的 Amazon Web Services account，然后打开 AWS 管理控制台。 2. 导航到 CloudFormation 控制台，然后通过上传`ssm.yaml`模板创建 System CloudFormation s Manager 堆栈。为以下参数提供值： • StateUnJoinAssociationLoggingBucketName— 提供模板将为记录目的而创建的 S3 存储桶的名称。 • SSMAssociationADUnjoin名称-提供`AWS::SSM::Association` 资源的名称。 • SSMAutomationDocumentName— 为 Systems Manager 自动化运行手册提供一个名称。 • EventBridgeName— 提供 EventBridge 活动总线的名称。 3. 通过启动`ssm.yaml` CloudFormation 模板来部署 Systems Manager CloudFormation 堆栈。该模板将创建 Systems	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	Manager Atomation 运行手册，该运行手册将在带有标签的新 EC2 实例启动时启动。`ADJoined:FSXADD`自动化运行手册将在 AWS Managed Microsoft AD 目录中添加此实例。	

Task	描述	所需技能
部署基础设施堆栈。	成功部署 Systems Manager 堆`infra`栈后，创建堆栈，其中包括 EC2 实例节点、安全组、Amazon FSx for Windows 文件服务器文件系统和 IAM 角色。 1. 导航到 CloudFormation 控制台并启动`infra-cf.yaml` 模板。若要部署此堆栈，必需参数： • `ActiveDirectoryId` — AWS 托管的 Microsoft AD ID • `ADDnsIpAddresses1` — AWS 托管的 Microsoft AD 的主要 DNS IP 地址 • `ADDnsIpAddresses2` — AWS 托管的 Microsoft AD 的辅助 DNS IP 地址 • `FSxSecurityGroupName` — Amazon FSx 安全组的名称 • `FSxWindowsFileSystemName` — Amazon FSx 硬盘的名称 • `ImageID` — 用于创建 SQL Server 实例节点的基本 Windows 2012 R2 镜像或亚马逊机器映像（AMI）的 ID	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	• KeyPairName — 要附加到 EC2 实例节点以供访问的键值对 • Node1SecurityGroupName — 第一个节点安全组的名称 • Node2SecurityGroupName — 第二个节点安全组的名称 • OUSecretName — 包含 AWS 托管的 Microsoft AD 信息的机密名称 • PrivateSubnet1 — 第一个私有子网的 ID • PrivateSubnet2 — 第二个私有子网的 ID • SqlFSxFCIName — 应用于主节点和辅助节点以及 Amazon 的标签的名称 FSx。 • SqlFSxServerNetBIOSName1 — 主 EC2 实例节点的名称（最多 15 个字符） • SqlFSxServerNetBIOSName2 — 辅助 EC2 实例节点的名称（最多 15 个字符） • VPC — VPC ID • WorkloadInstanceType — EC2 实例类型	

Task	描述	所需技能
	部署 `infra` 堆栈。该堆栈将创建设置 Windows SQL Server FCI 所需所有基础结构组件。 2. EC2 实例节点启动后，系统将调用 Systems Manager 自动化文档，将这些实例加入到 AWS 托管的 Microsoft AD。您可以在 Systems Manager 控制台的自动化页面上跟踪进度。	

设置 Windows SQL Server Always On FCI

Task	描述	所需技能

安装 Windows 工具。	1. 登录主 EC2 实例，即节点 1。要安装 Windows 功能（Active Directory 和 FCI 工具），请运行以下 PowerShell 脚本。
	`Install-WindowsFeature -Name RSAT-AD-Powershell,Failover-Clustering -IncludeManagementTools`
	`Install-WindowsFeature -Name RSAT-Clustering,RSAT-ADDS-Tools,RSAT-AD-Powershell,RSAT-DHCP,RSAT-DNS-Server`
		AWS DevOps， DevOps 工程师，数据库管理员

Task	描述	所需技能
	2. 登录到辅助 EC2 实例（节点 2），然后运行相同的脚本以启用节点 2 上的功能。	
在 Active Directory 域服务中预存集群计算机的对象。	要在 Active Directory 域服务 (AD DS) 中预存集群名称对象 (CNO) 并为集群角色预留虚拟计算机对象 (VCO)，请按照 Windows Server 文档中的说明进行操作。	AWS DevOps、数据库管理员、工程师 DevOps

Task	描述	所需技能
创建 WSFC。	要创建 Windows Server Failover Clustering (WSFC) 集群，请执行以下操作： 1. 登录主 EC2 实例，即节点 1。要创建 Amazon FSx 文件共享并授予对所列广告服务账户的完全访问权限，请运行以下代码。 `Invoke-Command -ComputerName "<FSx Windows Remote PowerShell Endpoint>" -ConfigurationName FSxRemoteAdmin -scriptblock { New-FSxSmbShare -Name "SQLDB" -Path "D:\share" -Description "SQL Databases Share" -ContinuouslyAvailable $true -FolderEnumerationMode AccessBased -EncryptData $true grant-fsxsmbshareaccess -name SQLDB -AccountName "<domain\user>" -accessRight Full }` 此命令还将创建持续可用 (CA) 文件共享，该共享已针对 Microsoft SQL Server 使用进行优化。	AWS DevOps、数据库管理员、工程师 DevOps

Task	描述	所需技能
	2. 要在主实例（节点 1）上创建失效转移群集，请运行以下命令。 `New-Cluster -Name <CNO Name> -Node <Node1 Name>, <Node2 Name> -StaticAddress <Node1 Secondary Private IP>, <Node2 Secondary Private IP>` 命令需要使用以下参数： • Name — 集群 (CNO) 的名称。 • Node — 分别是主节点和辅助节点的名称 • `StaticAddress` — 分别是主节点和辅助节点的辅助 IP 地址 **Important** 域管理员或普通用户必须对两个节点都具有管理员权限才能创建 Windows Server 故障转移群集 (WSFC) 群集。否则，前面的命令将失败并返回消息 `You do not have administrator privilege on servers`。	

Task	描述	所需技能
	3. 集群创建后，运行以下命令，以附加文件共享见证。 `Set-ClusterQuorum -FileShareWitness \ \<FSx Windows Remote PowerShell Endpoint> \share\witness`	

Task	描述	所需技能
安装 SQL Server 失效转移群集。	设置 WSFC 集群后，在主实例 (节点 1) 上安装 SQL Server 集群。 1. 在两个节点上的 T 盘中，创建 `tempdb` 和 `log` 文件夹。PowerShell 命令中使用文件夹。 2. 在两个节点上复制用于安装 SQL Server 的 SQL Server 媒体文件后，在节点 1 上运行以下 PowerShell 命令，在节点 1 上安装 SQL Server。 `D:\setup.exe /Q ` ` /ACTION=InstallFailoverCluster ` /IACCEPTSQLSERVERLICENSETERMS ` /FEATURES="SQL,IS,BC,Conn" ` /INSTALLSHAREDDIR="C:\Program Files\Microsoft SQL Server” ` /INSTALLSHAREDWOWDIR="C:\Program Files (x86)\Microsoft SQL Server" ` /RSINSTALLMODE="FilesOnlyMode" ` /INSTANCEID="MSSQLSERVER" ` /INSTANCENAME="MSSQLSERVER" ` /FAILOVERCLUSTERGROUP="SQL Server (MSSQLSERVER)" `	AWS DevOps、数据库管理员、工程师 DevOps

Task	描述	所需技能
	/FAILOVERCLUSTER IPADDRESSES="IPv4; <2nd Sec Private Ip node1>;Cluster Network 1;<subnet mask>" ` /FAILOVERCLUSTER NETWORKNAME="<Fail over cluster Network Name>" ` /INSTANCEDIR="C: \Program Files\Mic rosoft SQL Server" ` /ENU="True" ` /ERRORREPORTING=0 ` /SQMREPORTING=0 ` /SAPWD="<Domain User password>" ` /SQLCOLLATION="S QL_Latin1_General_ CP1_CI_AS" ` /SQLSYSADMINACCO UNTS="<domain\user name>" ` /SQLSVCACCOUNT=" <domain\username>" /SQLSVCPASSWORD="< Domain User password>" ` /AGTSVCACCOUNT=" <domain\username>" /AGTSVCPASSWORD="< Domain User password>" ` /ISSVCACCOUNT="<domain \username>" /ISSVCPAS SWORD="<Domain User password>" ` /FTSVCACCOUNT="NT Service\MSSQLFDLau ncher" ` /INSTALLSQLDATADIR="\ \<FSX DNS name>\sha	

Task	描述	所需技能
	re\Program Files\Mic rosoft SQL Server" ` /SQLUSERDBDIR="\\<FSX DNS name>\share\data" ` /SQLUSERDBLOGDIR="\ \<FSX DNS name>\share \log" ` /SQLTEMPDBDIR="T: \tempdb" ` /SQLTEMPDBLOGDIR="T: \log" ` /SQLBACKUPDIR="\\<FSX DNS name>\share\SQLBac kup" ` /SkipRules=Clust er_VerifyForErrors ` /INDICATEPROGRESS	

Task	描述	所需技能
向集群添加一个辅助节点。	要将 SQL Server 添加到辅助节点（节点 2），请运行以下PowerShell 命令。 `D:\setup.exe /Q ` `/ACTION=AddNode ` `/IACCEPTSQLSERVERLICENSETERMS ` `/INSTANCENAME="MSSQLSERVER" ` `/FAILOVERCLUSTERGROUP="SQL Server (MSSQLSERVER)" ` `/FAILOVERCLUSTERIPADDRESSES="IPv4;<2nd Sec Private Ip node2>;Cluster Network 2;<subnet mask>" ` `/FAILOVERCLUSTERNETWORKNAME="<Fail over cluster Network Name>" ` `/CONFIRMIPDEPENDENCYCHANGE=1 ` `/SQLSVCACCOUNT="<domain\username>" /SQLSVCPASSWORD="<Domain User password>" ` `/AGTSVCACCOUNT="domain\username>" /AGTSVCPASSWORD="<Domain User password>" ` `/FTSVCACCOUNT="NT Service\MSSQLFDLauncher" ` `/SkipRules=Cluster_VerifyForErrors ` `/INDICATEPROGRESS`	AWS DevOps、数据库管理员、工程师 DevOps

Task	描述	所需技能
测试 SQL Server FCI。	1. 在其中一个节点的 Windows 实例上，在管理工具中启动失效转移群集管理器。 2. 导航到节点，确认节点状态为运行状态。 3. 选择角色，打开 SQL Server (MSSQLSERVER) 的上下文（右键单击）菜单，然后选择移动和选择节点。 4. 选择节点后，SQL Server 应在另一节点上运行。	数据库管理员、工程师 DevOps

清理资源

Task	描述	所需技能
清理资源。	要清理资源，请使用 AWS CloudFormation 堆栈删除流程： 1. 打开 AWS CloudFormation 控制台。 2. 在堆栈页面，选择 `infra` 堆栈。该堆栈当前必须处于运行状态。 3. 在堆栈详细信息窗格中，选择删除。 4. 在系统提示时，选择删除堆栈。 5. 对 `ssm` 堆栈重复步骤 2-4。	AWS DevOps、数据库管理员、工程师 DevOps

Task	描述	所需技能
	堆栈删除过程完成之后，堆栈将处于 DELETE_COMPLETE 状态。默认情况下，处于该DELETE_COMPLETE 状态的堆栈不会显示在 CloudFormation 控制台中。要显示已删除的堆栈，您必须按照在 AWS CloudFormation 控制台上查看已删除堆栈中所述更改堆栈视图筛选条件。 如果删除失败，则堆栈将处于 DELETE_FAILED 状态。有关解决方案，请参阅 CloudFormation 文档中的删除堆栈失败。	

故障排除

事务	解决方案
AWS CloudFormation 模板失败	如果 CloudFormation 模板在部署过程中出现故障，请执行以下操作： 1. 打开 AWS CloudFormation 控制台。 2. 在 CloudFormation 控制台的堆栈页面上，选择堆栈。 3. 选择事件，然后检查堆栈状态。
AWS Managed Microsoft AD	若要解决加入问题，请执行以下步骤： 1. 打开 Systems Manager 控制台。 2. 选择部署区域。 3. 在左侧窗格中，选择自动化，然后找到失败的自动化运行手册。

事务	解决方案
	4. 打开自动化运行手册，检查执行状态和执行步骤。 5. 调查失败步骤的详细信息，以查看确切的错误或失败。

相关资源

- 使用 FSx 适用于 Windows 的亚马逊文件服务器简化微软 SQL Server 高可用性部署
- FSx 用于搭载微软 SQL Server 的 Windows 文件服务器

使用 BMC Discovery 查询提取迁移数据以进行迁移规划

由 Ben Tailor-Hamblin (AWS)、Simon Cunningham (AWS)、Emma Baldry (AWS) 和 Shabnam Khan (AWS) 创建

摘要

本指南提供查询示例和步骤，帮助您使用 BMC Discovery 从本地基础架构和应用程序中提取数据。该模式向您展示了如何使用 BMC Discovery 查询来扫描您的基础架构并提取软件、服务和依赖项信息。提取的数据是大规模迁移到 Amazon Web Services（AWS）云的评测和动员阶段所必需的。您可以使用此数据来做出关键决策，决定将哪些应用程序一起迁移作为迁移计划的一部分。

先决条件和限制

先决条件

- BMC Discovery（以前称为 BMC ADDM）或 BMC Helix Discovery 的软件即服务（SaaS）版本的许可证
- 已安装本地或 SaaS 版本的 BMC Discovery

> **Note**
>
> 对于 BMC Discovery 的本地版本，您必须将应用程序安装在能够访问跨多个数据中心迁移的所有网络和服务器设备的客户端网络上。必须根据应用程序安装说明提供对客户端网络的访问权限。如果需要扫描 Windows 服务器信息，则必须在网络中设置 Windows 代理管理器设备。

- 如果您使用的是 BMC Helix Discovery，则网络访问允许应用程序跨数据中心扫描设备

产品版本

- BMC Discovery 22.2 (12.5)
- BMC Discovery 22.1 (12.4)
- BMC Discovery 21.3 (12.3)
- BMC Discovery 21.05 (12.2)
- BMC Discovery 20.08 (12.1)
- BMC Discovery 20.02 (12.0)
- BMC Discovery 11.3

- BMC Discovery 11.2
- BMC Discovery 11.1
- BMC Discovery 11.0
- BMC Atrium Discovery 10.2
- BMC Atrium Discovery 10.1
- BMC Atrium Discovery 10.0

架构

下图显示了资产经理如何使用 BMC Discovery 查询扫描 SaaS 和本地环境中的 BMC 建模应用程序。

该图显示了以下工作流程：资产管理器使用 BMC Discovery 或 BMC Helix Discovery 扫描在多个物理服务器上托管的虚拟服务器上运行的数据库和软件实例。该工具可以使用跨多个虚拟和物理服务器的组件对应用程序进行建模。

技术堆栈

- BMC Discovery
- BMC Helix Discovery

工具

- BMC Discovery 是一款数据中心发现工具，可帮助您自动发现数据中心。
- BMC Helix Discovery 是一个基于 SaaS 的发现和依赖关系建模系统，可帮助您对数据资产及其依赖关系进行动态建模。

最佳实践

最佳做法是在迁移到云时映射应用程序、依赖项和基础结构数据。映射可帮助您了解当前环境的复杂性以及各种组件之间的依赖关系。

这些查询提供的资产信息很重要，原因如下：

1. 规划 – 了解组件之间的依赖关系有助于更有效地规划迁移过程。例如，您可能需要先迁移某些组件，以确保可以成功迁移其他组件。

2. 风险评测 – 映射组件之间的依赖关系可以帮助您识别迁移过程中可能出现的任何潜在风险或问题。例如，你可能会发现某些组件依赖于过时或不受支持的技术，这些技术可能会导致云中出现问题。
3. 云架构 – 映射应用程序和基础架构数据还可以帮助您设计合适的云架构，以满足您的组织需求。例如，您可能需要设计一个多层体系结构来支持高可用性或可扩展性要求。

总体而言，映射应用程序、依赖项和基础架构数据是云迁移过程中的关键步骤。映射练习可以帮助您更好地了解当前环境，识别任何潜在问题或风险，并设计合适的云体系结构。

操作说明

识别和评估发现工具

Task	描述	所需技能
确定 ITSM 所有者。	确定 IT 服务管理 (ITSM) 所有者（通常通过联系运营支持团队）。	迁移主管
检查 CMDB。	确定包含资产信息的配置管理数据库 (CMDBs) 的数量，然后确定该信息的来源。	迁移主管
识别发现工具并检查是否使用了 BMC Discovery。	如果您的组织正在使用 BMC Discovery 将有关您的环境的数据发送到 CMDB 工具，请检查其扫描的范围和覆盖范围。例如，检查 BMC Discovery 是否正在扫描所有数据中心，以及访问服务器是否位于外围区域中。	迁移主管
检查应用程序建模的级别。	检查应用程序是否在 BMC Discovery 中建模。如果没有，建议使用 BMC Discovery 工具对哪些正在运行的软件实例提供应用程序和业务服务进行建模。	迁移工程师，迁移主管

提取基础架构数据

Task	描述	所需技能
在物理和虚拟服务器上提取数据。	要提取 BMC Discovery 扫描的物理服务器和虚拟服务器上的数据，请使用查询生成器运行以下查询： `search Host show key as 'Serverid', virtual, name as 'HOSTNAME', os_type as 'osName', os_version as 'OS Version', num_logical_processors as 'Logical Processor Counts', cores_per_processor as 'Cores per Processor', logical_ram as 'Logical RAM', #Consumer:StorageUse:Provider:DiskDrive.size as 'Size'` **Note** 您可以使用提取的数据来确定适合迁移的实例大小。	迁移工程师，迁移主管
在建模的应用程序上提取数据。	如果您的应用程序是在 BMC Discovery 中建模的，则可以提取有关运行应用程序软件的服务器的数据。若要获取服务器名称，请使用查询生成器运行以下查询：	BMC Discovery 应用程序所有者

Task	描述	所需技能
	`search SoftwareInstance show key as 'ApplicationID', #RunningSoftware:HostedSoftware:Host:Host.key as 'ReferenceID', type, name` **Note** 在 BMC Discovery 中，应用程序由一组正在运行的软件实例进行建模。应用程序依赖于运行应用程序软件的所有服务器。	

Task	描述	所需技能
提取数据库上的数据。	若要获取所有已扫描数据库以及运行这些数据库的服务器的列表，请使用查询生成器运行以下查询： `search Database show key as 'Key', name, type as 'Source Engine Type', #Detail:Detail:ElementWithDetail:SoftwareInstance.name as 'Software Instance', #Detail:Detail:ElementWithDetail:SoftwareInstance.product_version as 'Product Version', #Detail:Detail:ElementWithDetail:SoftwareInstance.edition as 'Edition', #Detail:Detail:ElementWithDetail:SoftwareInstance.#RunningSoftware:HostedSoftware:Host:Host.key as 'ServerID'`	应用程序所有者

Task	描述	所需技能
提取服务器通信数据。	若要从历史网络通信日志中获取有关 BMC Discovery 收集的服务器之间的所有网络通信的信息，请使用查询生成器运行以下查询： `search Host TRAVERSE InferredElement:Inference:Associate:DiscoveryAccess TRAVERSE DiscoveryAccess:DiscoveryAccessResult:DiscoveryResult:NetworkConnectionList TRAVERSE List:List:Member:DiscoveredNetworkConnection PROCESS WITH networkConnectionInfo`	BMC Discovery 应用程序所有者
提取有关应用程序发现的数据。	若要获取有关应用程序依赖项的信息，请使用查询生成器运行以下查询： `search SoftwareInstance show key as 'SRC App ID', #Dependant:Dependency:DependedUpon:SoftwareInstance.key as 'DEST App ID'`	BMC Discovery 应用程序所有者

Task	描述	所需技能
提取有关业务服务的数据。	若要提取有关主机提供的业务服务的数据，请使用查询生成器运行以下查询： `search Host show name, #Host:HostedSoftware:AggregateSoftware:BusinessService.name as 'Name'`	BMC Discovery 应用程序所有者

故障排除

事务	解决方案
查询无法运行或包含未填充的列。	查看 BMC Discovery 中的资产记录，并确定所需字段。然后，使用查询生成器替换查询中的这些字段。
未填充从属资产的详细信息。	这可能是由于访问权限或网络连接造成的。发现工具可能没有访问某些资产所需权限，尤其是当它们位于不同的网络或不同的环境中时。 我们建议您与发现主题专家密切合作，以确保识别所有相关资产。

相关资源

参考

- BMC Discovery 许可授权（BMC 文档）
- BMC Discovery 功能和组件（BMC 文档）
- BMC Discovery 用户指南（BMC 文档）
- 搜索数据（在 BMC Discovery 上）（BMC 文档）
- 迁移的产品组合发现和分析 (AWS Prescriptive Guidance)

教程和视频

- BMC Discovery：网络研讨会-报告查询最佳实践（第 1 部分）(YouTube)

重新定位

主题

- 将 Amazon RDS for Oracle 数据库迁移到另一个 AWS 区域 数据库 AWS 账户 并 AWS DMS 用于持续复制
- 使用 H VMware CX 将 SDDC 迁移到 AWS VMware 上的 VMware 云端
- 将 Amazon RDS 数据库实例迁移到另一个 VPC 或账户
- 将 Amazon RDS for Oracle 数据库实例迁移至另一个 VPC
- 将 Amazon Redshift 集群迁移至中国的 AWS 区域
- 使用 VMware HCX 将工作负载迁移到 AWS 上的 VMware 云端
- 使用 pg_transport 在两个 Amazon RDS 数据库实例之间传输 PostgreSQL 数据库

将 Amazon RDS for Oracle 数据库迁移到另一个 AWS 区域 数据库 AWS 账户 并 AWS DMS 用于持续复制

由 Durga Prasad Cheepuri (AWS) 和 Eduardo Valentim (AWS) 创作

摘要

> **Warning**
>
> IAM 用户具有长期凭证，这会带来安全风险。为帮助减轻这种风险，我们建议仅向这些用户提供执行任务所需的权限，并在不再需要这些用户时将其移除。

此模式将引导您完成将适用于 Oracle 的亚马逊关系数据库服务 (Amazon RDS) 源数据库迁移到其他 AWS 账户 和的步骤 AWS 区域。该模式使用数据库快照进行一次性完整数据加载，并启用 AWS Database Migration Service (AWS DMS) 进行持续复制。

先决条件和限制

先决条件

- AWS 账户 包含源 Amazon RDS for Oracle 数据库的活动，该数据库已使用非默认 AWS Key Management Service (AWS KMS) 密钥进行加密
- 在与源数据库不同的 AWS 区域 数据库 AWS 账户 中处于活动状态，用于目标 Amazon RDS for Oracle 数据库
- 源和目标之间的虚拟私有云 (VPC) 对等 VPCs
- 熟悉使用 Oracle 数据库作为数据源 AWS DMS
- 熟悉使用 Oracle 数据库作为目标 AWS DMS

产品版本

- Oracle 版本 11g（版本 11.2.0.3.v1 及更高版本）以及最高版本 12.2 和 18c。有关支持的版本和版本的最新列表，请参阅 AWS 文档 AWS DMS中的 “使用 Oracle 数据库作为源 AWS DMS以及使用 Oracle 数据库作为目标”。有关 Amazon RDS 支持的 Oracle 版本，请参阅 Oracle on Amazon RDS。

架构

源和目标技术堆栈

- Amazon RDS for Oracle 数据库实例

持续复制架构

工具

用于一次性完整数据加载的工具

- Amazon Relational Database Service (Amazon RDS) 创建数据库实例的存储卷快照，备份整个数据库实例，而不仅仅是单个数据库。创建数据库快照时，需要识别出将要备份的数据库实例，然后为数据库快照命名，以便稍后从此快照还原。创建快照所用时间因数据库大小而异。由于快照包含整个存储卷，因此，文件（如临时文件）的大小也会影响创建快照所需时间。有关创建数据库快照的信息，请参阅 Amazon RDS 文档中的创建数据库快照。
- AWS Key Management Service (AWS KMS) 为 A mazon RDS 加密创建密钥。 在创建加密数据库实例时，您还可以提供加密AWS KMS密钥的密钥标识符。如果您未指定AWS KMS密钥标识符，Amazon RDS 会将您的默认加密密钥用于您的新数据库实例。 AWS KMS为您创建默认加密密钥 AWS 账户。每个 AWS 账户 都有不同的默认加密密钥 AWS 区域。对于这种模式，应使用非默认AWS KMS密钥对 Amazon RDS 数据库实例进行加密。有关使用AWS KMS密钥进行 Amazon RDS 加密的更多信息，请参阅 A mazon RDS 文档中的加密 Amazon RDS 资源。

用于持续复制的工具

- AWS Database Migration Service (AWS DMS) 用于复制正在进行的更改以及使源数据库和目标数据库保持同步。有关使用进行持续复制 AWS DMS 的更多信息，请参阅 AWS DMS 文档中的使用 AWS DMS 复制实例。

操作说明

配置您的来源 AWS 账户

Task	描述	所需技能
准备源 Oracle 数据库实例。	使 Amazon RDS for Oracle 数据库实例在 ARCHIVELOG 模式下运行，然后设置保留期。有关详细信息，请参阅使用 AWS 托管 Oracle 数据库作为源 AWS DMS。	数据库管理员
设置源 Oracle 数据库实例的补充日志记录。	为 Amazon RDS for Oracle 数据库实例设置数据库级和表级补充日志。有关详细信息，请参阅使用 AWS 托管 Oracle 数据库作为源 AWS DMS。	数据库管理员
更新来源账户中的 AWS KMS 密钥政策。	更新源代码中的 AWS KMS 密钥策略 AWS 账户 以允许目标 AWS 账户 使用加密的 Amazon RDS AWS KMS 密钥。有关详细信息，请参阅AWS KMS 文档。	SysAdmin
创建手动拍摄的源数据库实例的 Amazon RDS 数据库快照。		AWS IAM 用户
与目标共享手动加密的 Amazon RDS 快照 AWS 账户。	有关详细信息，请参阅共享数据库快照。	AWS IAM 用户

配置你的目标 AWS 账户

Task	描述	所需技能
附加策略。	在目标中 AWS 账户，将 AWS Identity and Access Management (IAM) 策略附加到根 IAM 用户，以允许 IAM 用户使用共享 AWS KMS 密钥复制加密的数据库快照。	SysAdmin
切换到源 AWS 区域。		AWS IAM 用户
复制共享快照。	在 Amazon RDS 控制台的 “快照” 窗格中，选择 “与我共享”，然后选择共享快照。使用源数据库使用的 AWS KMS 密钥的 Amazon 资源名称 (ARN)，将快照复制到与源数据库相同的 AWS 区域 快照中。有关详细信息，请参阅复制数据库快照。	AWS IAM 用户
切换到目标 AWS 区域，然后创建新 AWS KMS 密钥。		AWS IAM 用户
复制快照。	切换到源 AWS 区域。在 Amazon RDS 控制台的快照窗格中，选择 Own ed by Me，然后选择复制的快照。使用新目标 AWS 区域 的 AWS KMS 密钥将快照复制到目标 AWS 区域。	AWS IAM 用户
还原快照。	切换到目标 AWS 区域。在 Amazon RDS 控制台的 “快照” 窗格中，选择 “我所有”。选择复制的快照并将其还原至	AWS IAM 用户

Task	描述	所需技能
	Amazon RDS for Oracle 数据库实例。有关详细信息，请参阅从数据库快照恢复。	

为持续进行的复制准备源数据库

Task	描述	所需技能
创建具有适当权限的 Oracle 用户。	创建一个具有 Oracle 所需权限的 Oracle 用户作为来源 AWS DMS。有关详细信息，请参阅AWS DMS 文档。	数据库管理员
为 Oracle LogMiner 或 Oracle 二进制读取器配置源数据库。		数据库管理员

为持续复制准备目标数据库

Task	描述	所需技能
创建具有适当权限的 Oracle 用户。	创建一个具有 Oracle 所需权限的 Oracle 用户作为目标 AWS DMS。有关详细信息，请参阅AWS DMS 文档。	数据库管理员

创建 AWS DMS 组件

Task	描述	所需技能
在目标中创建复制实例 AWS 区域。	在目标的 VPC 中创建复制实例 AWS 区域。有关详细信息，请参阅AWS DMS 文档。	AWS IAM 用户

Task	描述	所需技能
创建具有所需加密的源端点和目标端点并测试连接。	有关详细信息，请参阅AWS DMS 文档。	数据库管理员
创建复制任务。	1. 对于迁移类型，请选择持续复制。 2. 对于变更数据捕获 (CDC) 起点，请使用 Amazon RDS 快照进行满载时的 Oracle 系统更改编号 (SCN)，或使用全负荷时的时间戳。 3. 对于`TargetTablePrepMode`，选择 DO_NOTHING。如果任务有大型二进制对象 (LOB) 数据表，请选择受限 LOB 模式，并将最大 LOB 大小设置为表中 LOB 数据的最大大小。 4. 启用日志记录。 5. 将通过键关联的表分组到单个任务中。如果存在包含大量 LOB 数据的表，且该表与其他表没有关系，请使用前面描述的 LOB 设置为其创建单独的任务。 有关详细信息，请参阅AWS DMS 文档。	IAM 用户
启动和监测任务。	有关详细信息，请参阅AWS DMS 文档。	AWS IAM 用户

Task	描述	所需技能
如果需要，可以对任务启用验证。	注意，启用验证确实会影响复制的性能。有关详细信息，请参阅AWS DMS 文档。	AWS IAM 用户

相关资源

- 更改密钥策略
- 创建手动 Amazon RDS 数据库快照
- 共享手动 Amazon RDS 数据库快照
- 复制快照
- 从 Amazon RDS 数据库快照还原
- 入门 AWS DMS
- 使用 Oracle 数据库作为来源 AWS DMS
- 使用 Oracle 数据库作为目标 AWS DMS
- AWS DMS 使用 VPC 对等互连进行设置
- 如何与其他人共享手动 Amazon RDS 数据库快照或数据库集群快照 AWS 账户？（AWS Knowledge Center 文章）

使用 H VMware CX 将 SDDC 迁移到 AWS VMware 上的 VMware 云端

由 Deepak Kumar (AWS) 编写

摘要

注意：自 2024 年 4 月 30 日起，VMware Cloud on AWS 不再由其渠道 AWS 合作伙伴转售。该服务将继续通过博通提供。我们鼓励您联系您的 AWS 代表了解详情。

此模式描述了如何使用 VMware 混合云扩展 (HCX) 将您的本地虚拟机 (VMs) 和应用程序迁移到 Amazon Web Services (AWS) 上的 VMware 云端。迁移使用 AWS 云上的 VMware 企业级软件定义数据中心 (SDDC) 软件来优化对 AWS 服务的访问。

VMware Cloud on AWS 将计算、存储和网络虚拟化产品（vSphere、vSAN 和 NS VMware X）与 VMware vCenter 服务器管理集成在一起，后者经过优化，可在专用、弹性、裸机 AWS 基础设施上运行。由此产生的基础设施维护成本低、简化且超融合。

借助这项服务，IT 团队可以使用熟悉的 VMware 工具管理其基于云的资源。有关更多信息，请参阅 VMware 网站上的 VMware Cloud on AWS。

VMware HCX 支持三种类型的云迁移：

- Hybridity（数据中心扩展）：将现有的本地 VMware SDDC 扩展到 AWS，以提供占地面积扩展、按需容量、测试/开发环境和虚拟桌面。
- 云撤离（数据中心基础设施更新）：整合数据中心并完全迁移到 Amazon Web Services Cloud（包括处理数据中心主机托管或租赁结束）。
- 应用程序特定迁移：将单个应用程序移至 Amazon Web Services Cloud 以满足特定的业务需求。

先决条件和限制

先决条件

- 注册 AWS 账户（创建 C VMware loud SDDC 所必需的）。
- 注册 “我的 VMware 账户”。在 https://my.vmware.com/web/vmware/ 上注册并填写所有字段。
- 检查 vCenter 和主机的版本，并收集其数量。 VMs如果可能，请要求RVTools导出以显示有关您的虚拟环境的信息。建议 vCenter 版本 6.0 或更高版本。

- 如果要扩展数据中心网络 (L2)、使用 HCX 测试 vMotion 或使用 vRealize Network Insight 分析应用程序依赖项，则必须部署分布式虚拟交换机。
- 选择一个不冲突的本地当前管理子网网络，在 AWS 上创建 C VMware loud 上的 SDDC。
- 通过查看 HCX 用户指南中提供的先决条件来验证 VMware HCX 要求。
- 识别移民浪潮并 VMs 进行分组。检查 VMs 一下你是否可以用来测试。
- 收集有关相对带宽消耗、广域网压缩以及数据传输速度的所有数据。

备注

- 无需在本地部署 VMware NSX-V 或 NSX-T。
- HCX 无需支付额外费用（已包含在 C VMware loud on AWS 中）。

架构

以下图表介绍了基于多个组件服务的 HCX 解决方案。每个组件均支持 HCX 解决方案中的特定功能。有关每个 HCX 组件的更多信息，请参阅博客文章使用混合 VMware 云扩展 (HCX) 在 AWS 上将工作负载迁移到云。

源技术堆栈

- 本地部署 VMs 和由 VMware vSphere 管理的应用程序

目标技术堆栈

- VMware AWS 上的云

工具

- VMware HCX — VMware HCX 是一款可用于跨数据中心和云环境迁移应用程序和工作负载的工具。它包含在 VMware Cloud on AWS 中。

操作说明

计划迁移

Task	描述	所需技能
选择迁移策略。	决定是要扩展数据中心（混合型）、移动所有数据中心（云撤离），还是要将特定应用程序迁移至 AWS。	SysAdmin，应用程序所有者
验证 HCX 要求。	有关迁移信息，请查看 VMware HCX 用户指南。	SysAdmin，应用程序所有者

在 AWS 上迁移到 VMware 云端

Task	描述	所需技能
迁移您的 VMs 或应用程序。	有关更多信息，请参阅 VMware 文档中的使用 VMware HCX 进行混合迁移。	SysAdmin，应用程序所有者

相关资源

- VMware AWS 上的 Cloud：入门
- 使用 H VMware CX 进行混合迁移
- VMware HCX 用户指南
- VMware AWS 上的云定价
- VMware AWS 上的云路线图

将 Amazon RDS 数据库实例迁移到另一个 VPC 或账户

由 Dhrubajyoti Mukherjee (AWS) 创建

摘要

此示例介绍了以下操作指南：将 Amazon Relational Database Service (Amazon RDS) 数据库实例从虚拟私有云（VPC）迁移至同一 Amazon Web Services account 的其他项，或从 Amazon Web Services account 迁移至其他 Amazon Web Services account。

如果您出于分离或安全原因想要将 Amazon RDS 数据库实例迁移到另一个 VPC 或账户（例如，当您想将应用程序堆栈和数据库放在不同的 VPC 或账户中时 VPCs），这种模式非常有用。

将数据库实例迁移至其他 Amazon Web Services account 涉及的步骤，包括拍摄手动快照、共享快照以及在目标账户中还原快照。此过程可能很耗时，具体取决于数据库更改和事务速率。其还会导致数据库停机，因此请提前做好迁移计划。考虑蓝绿部署策略，以最大限度减少停机时间。或者您可以评估 AWS Data Migration Service (AWS DMS)，以最大限度减少变更造成的停机时间。但是，此模式不包含此选项。要了解更多信息，请参阅 AWS DMS 文档。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- VPC、子网和 Amazon RDS 控制台所需 AWS Identity and Access Management (IAM) 权限

限制

- 对 VPC 的更改将会导致数据库重启，从而导致应用程序中断。建议您在低峰时段迁移。
- 将 Amazon RDS 迁移至其他 VPC 时的限制：
  - 您要迁移的数据库实例必须为没有备用实例的单个实例。它不能是集群成员。
  - Amazon RDS 不得位于多个可用区。
  - Amazon RDS 不得包含任何只读副本。
  - 在目标 VPC 中创建的子网组必须包含来自源数据库运行的可用区的子网。
- 将 Amazon RDS 迁移至其他 Amazon Web Services account 时的限制：
  - 目前不支持共享使用 Amazon RDS 默认服务密钥加密的快照。

架构

在同一 Amazon Web Services account 中迁移至 VPC

下图显示了将 Amazon RDS 数据库实例迁移至同一 Amazon Web Services account 中其他 VPC 的工作流。

包含以下步骤。有关详细说明，请参阅操作部分。

1. 在目标 VPC 创建数据库子网组。数据库子网组是您在创建数据库实例时用于指定特定 VPC 的子网集合。
2. 将源 VPC 中的 Amazon RDS 数据库实例配置为使用新数据库子网组。
3. 应用更改将 Amazon RDS 数据库迁移至目标 VPC。

迁移到不同的 Amazon Web Services account

下图显示了将 Amazon RDS 数据库实例迁移至其他 Amazon Web Services account 的工作流。

包含以下步骤。有关详细说明，请参阅操作部分。

1. 在源 Amazon Web Services account 中访问 Amazon RDS 数据库实例。
2. 在源 Amazon Web Services account 中创建 Amazon RDS 快照。
3. 与目标 Amazon Web Services account 共享 Amazon RDS 快照。
4. 访问目标 Amazon Web Services account 中的 Amazon RDS 快照。
5. 在目标 Amazon Web Services account 中创建 Amazon RDS 数据库实例。

工具

Amazon Web Services

- Amazon Relational Database Service (Amazon RDS) 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展关系数据库。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。

最佳实践

- 如果将 Amazon RDS 数据库实例迁移至另一个账户时担心数据库停机问题，我们建议您使用 AWS DMS。该服务提供数据复制，导致停机时间少于五分钟。

操作说明

使用同一 Amazon Web Services account 迁移至其他 VPC

Task	描述	所需技能
创建新的 VPC。	在 Amazon VPC 控制台，创建新的 VPC 和具有所需属性和 IP 地址范围的子网。有关详细说明，请参阅 Amazon VPC 文档。	管理员
创建数据库子网组。	打开 Amazon RDS 控制台： 1. 选择子网组，创建数据库子网组。 2. 输入子网组的名称、描述和 VPC ID。 3. 添加子网组的子网。添加至少来自两个可用区的子网。 4. 选择创建。 有关更多信息，请参阅 Amazon RDS 文档。	管理员
修改 Amazon RDS 数据库实例，以使用新的子网组。	打开 Amazon RDS 控制台： 1. 在导航窗格中，选择数据库，然后选择要迁移的 Amazon RDS 数据库实例。 2. 在连接部分，选择与目标 VPC 关联的子网组。	管理员

Task	描述	所需技能
	3. 在计划修改部分，选择立即应用。 向目标 VPC 的迁移完成后，目标 VPC 默认安全组将分配至 Amazon RDS 数据库实例。您可以为该 VPC 配置新的安全组，其中包含数据库实例所需入站和出站规则。 或者，使用 AWS 命令行界面（AWS CLI），通过明确提供新的 VPC 安全组 ID 以迁移至目标 VPC。例如： `aws rds modify-db-instance \ --db-instance-identifier testrds \ --db-subnet-group-name new-vpc-subnet-group \ --vpc-security-group-ids sg-idxxxx \ --apply-immediately`	

迁移至不同的 Amazon Web Services account

Task	描述	所需技能
在目标 Amazon Web Services account 中创建新的 VPC 和子网组。	1. 在 Amazon VPC 控制台，创建一个具有所需属性和 IP 地址范围的新 VPC。有关详细说明，请参阅 Amazon VPC 文档。	管理员

Task	描述	所需技能
	2. 按照 Amazon VPC 文档中的说明为新 VPC 创建子网。 3. 在 Amazon RDS 控制台，创建数据库子网组。有关说明，请参阅 Amazon RDS 文档。	
共享数据库手动快照并与目标账户共享。	1. 按 Amazon RDS 文档说明手动拍摄源数据库快照。 2. 通过提供目标账户 ID，与目标 Amazon Web Services account 共享快照。有关说明，请参阅 re:Post article 中关于与其他账户共享数据库快照的文章。	管理员
启动新的 Amazon RDS 数据库实例。	使用目标 Amazon Web Services account 中的共享快照，启动新的 Amazon RDS 数据库实例。有关说明，请参阅 Amazon RDS 文档。	管理员

相关资源

- Amazon VPC 文档
- Amazon RDS 文档
- 如何更改 RDS 数据库实例的 VPC？（AWS re: Post 文章）
- 如何将 Amazon RDS 资源的所有权转移至不同 Amazon Web Services account？（AWS re: Post 文章）
- 如何与其他 Amazon Web Services account 共享手动 Amazon RDS 数据库快照或者 Aurora 数据库集群快照？（AWS re: Post 文章）
- AWS DMS 文档

将 Amazon RDS for Oracle 数据库实例迁移至另一个 VPC

由 Pinesh Singal (AWS) 编写

摘要

此迁移模式为将适用于 Oracle 的亚马逊关系数据库服务 (Amazon RDS) 数据库 (DB) 实例从一个虚拟私有云 (VPC) 迁移到同一个亚马逊网络服务 (AWS) 账户中的另一个 VPC 提供了 step-by-step指导。例如，如果您的业务要求数据库和亚马逊弹性计算云 (Amazon EC2) 应用程序服务器位于同一 VPC 中，则可以使用此模式。

该模式描述了一种在线迁移策略，对于具有大量事务的多 TB Oracle 源数据库，几乎没有停机时间。

要将 Amazon RDS for Oracle 数据库实例移动至另一个 VPC，您必须更改 Amazon RDS 子网组。该子网组需要使用新 VPC 和所需子网进行预配置。在 VPC 从一个网络割接至另一个网络期间，Amazon RDS 实例会重新启动，因此在移动过程中将无法访问数据库。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 两个 VPCs 带有私有子网
- 一个 Amazon RDS for Oracle 数据库实例（已启动并正在运行），配置了入站和出站安全组

限制

- 不支持跨多个可用区（多可用区）的数据库实例。但是，这种模式提供了解决此限制的方法。
- 只读副本处于开启状态时，无法迁移数据库实例。
- 新 VPC 中的子网组应与数据库位于同一可用区。
- 迁移应在计划维护期或流量较低时段进行，因为将数据库移至其他 VPC 会导致数据库重启，从而导致应用程序中断几分钟。

产品版本

- Amazon RDS for Oracle 数据库实例，12.1.0.2 及更高版本

架构

源技术堆栈

- VPC 中的 Amazon RDS for Oracle 12.1.0.2.v22 数据库实例
- 在单独路由表中配置的 VPC
- 在 VPC 中配置 Amazon RDS 子网组
- Amazon RDS 选项组（如需要）

目标技术堆栈

- Amazon RDS for Oracle 数据库实例，其他 VPC 版本为 12.1.0.2.v22 的 Amazon RDS for Oracle 数据库实例
- 在单独的路由中配置 Amazon VPC
- 在新的 VPC 中配置的 Amazon RDS 子网组
- Amazon RDS 选项组（如需要）

源架构和目标架构

下图显示了如何使用控制台将 Amazon RDS for Oracle 数据库从一个 VPC 中的私有子网移动至另一个 VPC 中的私有子网。

1. 使用控制台修改源 Amazon RDS for Oracle 数据库实例。
2. 在目标 VPC，修改子网组，并修改选项组（如果使用）。

工具

- Amazon RDS - Amazon Relational Database Service (Amazon RDS) 是一项 Web 服务，让用户能够在 Amazon Web Services Cloud 中轻松设置、操作和扩展关系数据库。它为行业标准的关系数据库提供了经济高效、可调整大小的容量，并管理常见的数据库管理任务。此模式使用了 Amazon RDS for Oracle。

操作说明

更改现有 VPC 中的 Amazon RDS for Oracle 数据库配置

Task	描述	所需技能
创建子网组。	在 Amazon RDS 中配置子网组。	常规 AWS
创建选项组。	（可选）在 Amazon RDS 中配置选项组。	常规 AWS
修改 Amazon RDS for Oracle 实例。	通过子网组和选项组修改数据库。	常规 AWS、数据库管理员
如有必要，更新 Oracle 数据库。	要迁移源 Amazon RDS for Oracle 数据库，请进行以下更改： • 删除只读副本（如果存在）。 • 如多可用区功能已开启，请将其关闭。	常规 AWS

将 Amazon RDS for Oracle 数据库配置至目标 VPC

Task	描述	所需技能
创建子网组。	在 Amazon RDS，使用新 VPC 的子网和数据库的可用区配置子网组。	常规 AWS
创建选项组。	（可选）在 Amazon RDS 中配置选项组。	常规 AWS
修改 Amazon RDS for Oracle 数据库。	使用新 VPC 新子网组和选项组修改数据库。您可以立即应用	常规 AWS、数据库管理员

Task	描述	所需技能
	此更改，也可以在维护时段内应用这些更改。 此修改可能需要几分钟才能完成。在修改过程中，您将发现以下状态变化： • moving-to-vpc • C onfiguring-enhanced-monitoring • 修改 • Available 修改后将附加新 VPC 默认安全组。按 Amazon RDS for Oracle 需要附加新的安全组。	
如有必要，更新 Amazon RDS for Oracle 数据库。	迁移至新 VPC 中的目标 Amazon RDS for Oracle 数据库后，根据需要进行以下修改： • 如果源数据库中包含只读副本，请将其打开。 • 如果源数据库已开启多可用区功能，请将其打开。	常规 AWS
测试应用程序连接性。	通过任何应用程序执行数据库连接测试。确认新 VPC 中修改后的 Amazon RDS for Oracle 数据库已连接，并且可以从应用程序进行访问。	应用程序所有者

相关资源

- Amazon VPC 文档
- VPCs 和子网
- 在 VPC 中使用数据库实例
- Amazon RDS 文档
- Amazon RDS 上的 Oracle
- Amazon RDS 控制台
- 如何更改 Amazon RDS 数据库实例的 VPC？

将 Amazon Redshift 集群迁移至中国的 AWS 区域

由 Jing Yan (AWS) 创建

摘要

这种模式提供了 step-by-step一种将 Amazon Redshift 集群从另一个 AWS 区域迁移到中国的 AWS 区域的方法。

此模式使用 SQL 命令重新创建所有数据库对象，并使用 UNLOAD 命令将这些数据从 Amazon Redshift 迁移至 Amazon Simple Storage Service (Amazon S3) 存储桶。然后将数据迁移至中国 AWS 区域的 S3 存储桶。COPY 命令用于从 S3 存储桶加载数据，并将其传输到目标 Amazon Redshift 集群。

Amazon Redshift 目前不支持跨区域功能，例如将快照复制到中国 AWS 区域。这种模式提供了解决此限制的方法。您也可以逆转此模式的步骤，将数据从中国 AWS 区域迁移至另一个 AWS 区域。

先决条件和限制

先决条件

- 中国地区和中国以外的 AWS 区域的活跃 Amazon Web Services account
- 中国地区和中国以外的 AWS 区域中的现有 Amazon Redshift 集群

限制

- 这是离线迁移，这意味着源 Amazon Redshift 集群在迁移期间无法执行写入操作。

架构

源技术堆栈

- 位于中国以外的 AWS 区域的 Amazon Redshift 集群

目标技术堆栈

- 位于中国 AWS 区域的 Amazon Redshift 集群

目标架构

工具

工具

- Amazon S3 — Amazon Simple Storage Service（Amazon S3）是一种对象存储服务，提供可扩展性、数据可用性、安全性和性能。你可以使用 Amazon S3 存储来自 Amazon Redshift 的数据，也可以将数据从 S3 存储桶复制到 Amazon Redshift。
- Amazon Redshift – Amazon Redshift 是一种完全托管的 PB 级云中数据仓库服务。
- psql – psql 是 PostgreSQL 中基于终端的前端。

操作说明

为在来源 Region 进行迁移做准备

Task	描述	所需技能
在源区域启动并配置 EC2 实例。	登录 AWS 管理控制台并打开亚马逊弹性计算云 (Amazon EC2) 控制台。您当前的区域显示在屏幕顶部的导航栏中。此区域不能是中国 AWS 区域。从 Amazon EC2 控制台控制面板中选择 “启动实例”，然后创建和配置 EC2 实例。重要：确保入站规则 EC2 的安全组允许从源计算机不受限制地访问 TCP 端口 22。有关如何启动和配置 EC2 实例的说明，请参阅 “相关资源” 部分。	数据库管理员、开发人员
安装 psql 工具。	下载和安装 PostgreSQL。Amazon Redshift 不提供 psql 工具，它随 PostgreSQL 一起安装。有关使用 psql 和安装 PostgreSQL 工具的更多信息，请参阅“相关资源”部分。	数据库管理员

Task	描述	所需技能
记录 Amazon Redshift 集群的详细信息。	打开 Amazon Redshift 控制台，在导航窗格中，选择“集群”。然后从列表中选择 Amazon Redshift 集群名称。在“属性”选项卡上，在“数据库配置”部分，记录“数据库名称”和“端口”。打开“连接详细信息” 部分并录制 endpoint:<port>/<databasename> 格式端点。重要提示：确保您的入站规则的 Amazon Redshift 安全组允许您的实例不受限制地访问 TCP 端口 5439。 EC2	数据库管理员
连接 psql 到 Amazon Redshift 集群。	<databasename><port>在命令提示符处，通过运行 psql-h-<endpoint>U-<userid>d-p 命令来指定连接信息。在 psql 密码提示符处，输入“<userid>”用户的密码。您已连接到 Amazon Redshift 集群，并能以交互方式输入命令。	数据库管理员
创建 S3 存储桶。	打开 Amazon S3 控制台，创建 S3 存储桶以存放从 Amazon Redshift 导出的文件。有关如何创建 S3 存储桶的说明，请参阅”相关资源“ 部分。	数据库管理员、AWS General

Task	描述	所需技能
创建支持数据卸载的 IAM policy。	打开 AWS Identity and Access Management (IAM) 控制台并选择 “策略”。选择 “创建策略”，然后选择 “JSON” 选项卡。从“其他信息”部分复制并粘贴用于卸载数据的 IAM policy。重要提示：将“s3_bucket_name”替换为您的 S3 存储桶的名称。选择“审核策略”并输入策略的名称和描述。选择“创建策略”。	数据库管理员
创建 IAM 角色，以允许 Amazon Redshift 执行卸载操作。	打开 IAM 控制台，选择“角色”。选择创建角色”，然后在“选择可信实体类型” 中选择“Amazon Web Services”。为服务选择“Redshift”，选择“Redshift-可自定义”，然后选择 “下一步”。选择您之前创建的“卸载”策略，然后选择“下一步”。输入“角色名称”，然后选择“创建角色”。	数据库管理员
将 IAM 角色与 Amazon Redshift 集群关联。	打开 Amazon Redshift 控制台，然后选择“管理 IAM 角色”。从下拉菜单中选择“可用角色”，然后选择您之前创建的角色。选择“应用更改”。当“管理 IAM 角色”上的“IAM 角色的状态”显示为“同步”时，您可以运行 UNLOAD 命令。	数据库管理员
停止 Amazon Redshift 集群的写入操作。	迁移完成之前，您必须记得停止对源 Amazon Redshift 集群的所有写入操作。	数据库管理员

为目标地区迁移做好准备

Task	描述	所需技能
在目标区域启动并配置 EC2 实例。	登录中国某个区域（北京或宁夏）的 AWS 管理控制台。在 Amazon EC2 控制台中，选择“启动实例”，然后创建和配置 EC2 实例。重要：请确保您的入站规则的 Amazon EC2 安全组允许从您的源计算机不受限制地访问 TCP 端口 22。有关如何启动和配置 EC2 实例的更多说明，请参阅 “相关资源” 部分。	数据库管理员
记录 Amazon Redshift 集群的详细信息。	打开 Amazon Redshift 控制台，在导航窗格中，选择“集群”。然后从列表中选择 Amazon Redshift 集群名称。在“属性”选项卡上，在“数据库配置”部分，记录“数据库名称”和“端口”。打开“连接详细信息” 部分并录制 endpoint:<port>/<databasename> 格式端点。重要提示：请确保您的入站规则的 Amazon Redshift 安全组允许从您的实例不受限制地访问 TCP 端口 5439。EC2	数据库管理员
连接 psql 到 Amazon Redshift 集群。	<databasename><port>在命令提示符处，通过运行 psql-h-<endpoint>U-<userid>d-p 命令来指定连接信息。在 psql 密码提示符处，输入“<userid>”用户的密码。您已连接到 Amazon	数据库管理员

Task	描述	所需技能
	Redshift 集群，并能以交互方式输入命令。	
创建 S3 存储桶。	打开 Amazon S3 控制台，然后创建 S3 存储桶，以存放从 Amazon Redshift 导出的文件。有关此操作和其他操作的帮助，请参阅 “相关资源” 部分。	数据库管理员
创建支持复制数据的 IAM policy。	打开 IAM 控制台，然后选择“策略”。选择 “创建策略”，然后选择 “JSON” 选项卡。从“其他信息”部分复制并粘贴用于复制数据的 IAM policy。重要提示：将“s3_bucket_name”替换为您的 S3 存储桶的名称。选择“审核策略”并输入策略的名称和描述。选择“创建策略”。	数据库管理员
创建一个 IAM 角色，允许 Amazon Redshift 进行复制操作。	打开 IAM 控制台，选择“角色”。选择创建角色”，然后在“选择可信实体类型” 中选择“Amazon Web Services”。为服务选择“Redshift”，选择“Redshift-可自定义”，然后选择 “下一步”。选择您之前创建的“复制”策略，然后选择“下一步”。输入“角色名称”，然后选择“创建角色”。	数据库管理员

Task	描述	所需技能
将 IAM 角色与 Amazon Redshift 集群关联。	打开 Amazon Redshift 控制台，然后选择“管理 IAM 角色”。从下拉菜单中选择“可用角色”，然后选择您之前创建的角色。选择“应用更改”。当“管理 IAM 角色”上的 IAM 角色的“状态”显示为“同步”时，您可以运行“复制”命令。	数据库管理员

在开始迁移前验证源数据和对象信息

Task	描述	所需技能
验证源 Amazon Redshift 表行。	使用“其他信息”部分中的脚本验证和记录源 Amazon Redshift 表中的行数。切记均匀分割卸载和复制脚本的数据。这将提高数据卸载和加载效率，使每个脚本所涵盖的数据量将保持平衡。	数据库管理员
验证源 Amazon Redshift 集群的数据库对象数量。	使用 “其他信息” 部分中的脚本来验证和记录源 Amazon Redshift 集群中数据库、用户、架构、表、视图和用户定义函数 (UDFs) 的数量。	数据库管理员
迁移之前验证 SQL 语句的结果。	部分用于数据验证的 SQL 语句应根据实际业务和数据情况进行排序。这是为了验证导入的数据，以确保其一致性并正确显示。	数据库管理员

将数据与对象迁移至目标 Regions

Task	描述	所需技能
生成 Amazon Redshift DDL 脚本。	使用“其他信息”部分中用于查询 Amazon Redshift 的 SQL 语句”部分中的链接生成数据定义语言 (DDL) 脚本。这些 DDL 脚本应包括“创建用户”、“创建架构”、“用户对架构的权限”、“创建表/视图”、“用户对对象的权限”和创建函数”查询。	数据库管理员
在 Amazon Redshift 集群中为目标区域创建对象。	在 AWS 区域使用 AWS 命令行界面（AWS CLI）来运行 DDL 脚本。这些脚本将在 Amazon Redshift 集群中为目标 Region 创建对象。	数据库管理员
将源 Amazon Redshift 集群数据卸载至 S3 存储桶。	运行 UNLOAD 命令将数据从源 Regions Amazon Redshift 集群卸载至 S3 存储桶。	数据库管理员、开发人员
将源 Region S3 存储桶数据传输至目标 Region S3 存储桶。	将数据从您的源 Region S3 存储桶传输至目标 S3 存储桶。由于无法使用$ aws s3 sync”命令，因此请务必使用“相关资源”部分的“将 Amazon S3 数据从 AWS 区域传输至中国 AWS 区域”一文中概述的流程。	开发人员
将数据加载至目标 Amazon Redshift 集群。	在目标 Regions的 psql 工具，运行 COPY 命令，将数据从 S3 存储桶加载至目标 Amazon Redshift 集群。	数据库管理员

迁移后验证源 Regions 和目标 Regions 中的数据

Task	描述	所需技能
验证并比较源表和目标表中行数。	验证并比较源区域和目标区域中的表行数，以确保全部迁移。	数据库管理员
验证并比较源数据库和目标数据库对象数量。	验证并比较源 Regions 和目标 Regions 中的所有数据库对象，以确保所有数据库对象都已迁移。	数据库管理员
验证并比较源 Regions 和目标 Regions 中的 SQL 脚本结果。	运行在迁移前准备的 SQL 脚本。验证并比较数据，以确保 SQL 结果正确无误。	数据库管理员
重置目标 Amazon Redshift 集群中的所有用户密码。	迁移完成并验证所有数据后，您应重置中国 AWS 区域的 Amazon Redshift 集群的所有用户密码。	数据库管理员

相关的资源

- 将 Amazon S3 数据从 AWS 区域传输至中国 AWS 区域
- 创建 S3 存储桶
- 重置 Amazon Redshift 用户密码
- psql 文档

其他信息

用于卸载数据的 IAM policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::s3_bucket_name"]
    },
    {
      "Effect": "Allow",
      "Action": ["s3:GetObject", "s3:DeleteObject"],
      "Resource": ["arn:aws:s3:::s3_bucket_name/*"]
    }
  ]
}
```

用于复制数据的 IAM policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::s3_bucket_name"]
    },
    {
      "Effect": "Allow",
      "Action": ["s3:GetObject"],
      "Resource": ["arn:aws:s3:::s3_bucket_name/*"]
    }
  ]
}
```

用于查询 Amazon Redshift 的 SQL 语句

```
##Database

select * from pg_database where datdba>1;

##User

select * from pg_user where usesysid>1;

##Schema

SELECT n.nspname AS "Name",
```

```
  pg_catalog.pg_get_userbyid(n.nspowner) AS "Owner"

FROM pg_catalog.pg_namespace n

WHERE n.nspname !~ '^pg_' AND n.nspname <> 'information_schema'

ORDER BY 1;

##Table

select count(*) from pg_tables where schemaname not in
 ('pg_catalog','information_schema');

select schemaname,count(*) from pg_tables where schemaname not in
 ('pg_catalog','information_schema') group by schemaname order by 1;

##View

SELECT

    n.nspname AS schemaname,c.relname AS
 viewname,pg_catalog.pg_get_userbyid(c.relowner) as "Owner"

FROM

    pg_catalog.pg_class AS c

INNER JOIN

    pg_catalog.pg_namespace AS n

    ON c.relnamespace = n.oid

WHERE relkind = 'v' and n.nspname not in ('information_schema','pg_catalog');

##UDF

SELECT

   n.nspname AS schemaname,

   p.proname AS proname,

   pg_catalog.pg_get_userbyid(p.proowner) as "Owner"
```

```
FROM pg_proc p

LEFT JOIN pg_namespace n on n.oid = p.pronamespace

WHERE p.proowner != 1;
```

用于生成 DDL 语句的 SQL 脚本

- get_schema_priv_by_user 脚本
- Generate_tbl_ddl 脚本
- Generate_view_ddl
- Generate_user_grant_revoke_ddl
- Generate_udf_ddl

使用 VMware HCX 将工作负载迁移到 AWS 上的 VMware 云端

由 Deepak Kumar (AWS)、Derek Cox (AWS) 和 Himanshu Gupta (AWS) 创作

摘要

注意：自 2024 年 4 月 30 日起， VMware Cloud on AWS 不再由其渠道 AWS 合作伙伴转售。该服务将继续通过博通提供。我们鼓励您联系您的 AWS 代表了解详情。

此模式说明了如何在不更改底层平台的情况下使用 VMware 混合云扩展 (HCX) 将工作负载从本地 VMware 环境迁移到 VMware Cloud on AWS。 VMware HCX 可以简化迁移，帮助重新平衡工作负载，帮助保护数据，并优化本地数据中心和云服务器的灾难恢复流程。该模式介绍了安装、配置、升级和卸载 HCX 的步骤。

HCX 支持以下各项：

- 旧版本的 VMware vSphere — HCX 可帮助您将虚拟机 (VMs) 从旧版本的 vSphere 迁移到 AWS 上的云。 VMware 主机会自动更新和修复，避免在准备迁移时进行耗时的更新。
- 批量迁移 — 您可以将 HCX 与 WAN 优化服务结合使用， VMs 在不停机的情况下一次性迁移大量迁移，从而将本地网络扩展到云端。
- 异构网络环境 — 您当前的网络 (例如 vSphere、NSX、VXLAN 或 NSX-T) 决定了迁移的复杂性。HCX 提取网络应用程序基础知识，无需任何复杂的程序即可将您当前的网络扩展到云端。
- 网络速度慢 — 迁移通常需要 250 Mbps 以上的连接速度。HCX 可以较低的速度迁移您的工作负载，大约为 100 Mbps。

HCX 支持三类云迁移：

- Hybridity（数据中心扩展）— 将现有的本地 VMware 软件定义数据中心 (SDDC) 扩展到 AWS，以提供占地面积扩展、按需容量、测试/开发环境和虚拟桌面。
- 云撤离（数据中心基础设施更新）– 整合数据中心并完全迁移到 Amazon Web Services Cloud（包括处理数据中心主机托管或租赁结束）。
- 应用程序特定迁移 – 将单个应用程序移至 Amazon Web Services Cloud 以满足特定的业务需求。

您可以使用 HCX 在本地环境和 Cloud VMware on AWS 之间双向迁移工作负载。HCX 提供了多种源位置和目标位置间的工作负载迁移方法：

- HCX 冷迁移会迁移处于离 VMs 线状态。此方法适用于已关闭电源的情况 VMs ，因为它需要很长的停机时间。
- HCX vMotion 使用 vMoti VMware on 协议进行移动。 VMsHCX vMotion 提供零停机迁移，但每次只能迁移一个虚拟机。
- HCX 批量迁移使用 v VMware Sphere 复制协议迁 VMs 移到目标。您可以并行迁移多个 VMs 并行迁移并安排切换。停机时间等同于服务器重启，所有切换都是并行进行 VMs 的。
- HCX Replication Assisted vMotion (RAV) 是 HCX 批量迁移和 HCX vMotion 的组合。它提供并行迁移、调度以及零停机时间。
- 当您在本地使用多个 VMs 虚拟机管理程序和非 VMs vSphere 时，HCX 操作系统辅助迁移可帮助您批量迁移多个虚拟机管理程序。HCX 操作系统辅助迁移在您使用它从本地迁移到 C VMware loud on AWS 时是免费的，但是如果您想在两个本地环境之间迁移或从本地迁移到其他云提供商，则需要额外的许可。

先决条件和限制

先决条件

- 在 vmwar e.com 上访问 VMware 控制台的 VMware 帐户。
  - HCX 需要以下防火墙端口。

来源	目标	端口
本地 HCX Manager 和设备 IP	HCX Manager 和设备 AW VMware S 云上的 IP	UDP 500、UDP 4500 和 ICMP
本地 HCX Manager 和设备 IP	connect.hcx.vmware.com | hbridity-depot.vmware.com	TCP 443
本地 HCX Manager 和设备 IP	HCX 云端 URL	TCP 443

    如果本地网络内部有防火墙，则必须允许在数据中心内增加几个本地端口。有关 HCX 端口要求的完整列表，请参阅 H VMware CX 文档。
- 若要配置 HCX，您需要域名系统（DNS）IP、vCenter 完全限定域名 (FQDN)、NTP 服务器 FQDN、单点登录（SSO）用户以及类似信息。请提前收集这些详细信息，以避免部署出现的任何延迟。

限制

您可以使用网络扩展设备在本地环境和 VMware Cloud on AWS 之间扩展最多八个网络。有关 HCX 服务限制的完整列表，请参阅 VMware HCX 文档。

架构

源技术堆栈

- 本地 VMware 工作负载

目标技术堆栈

- VMware AWS 上的云

工具

工具

- VMware Cloud on AWS 是一项由 AWS 联合设计的服务 VMware ，旨在帮助您将 VMware 基于 vSphere 的本地环境迁移和扩展到 AWS 云。
- VMware 混合云扩展 (HCX) 是一种 VMware 实用工具，用于在不更改底层平台的情况下将工作负载从本地 VMware 环境迁移到 VMware Cloud on AWS。

操作说明

部署 HCX

Task	描述	所需技能
在 AW VMware S 云端启用 HCX 服务	1. 在 AWS 控制台上登录VMware 云端。 2. 导航至 SDCC，然后选择查看详细信息。 3. 选择 Add Ons（附加组件）选项卡。 4. 选择打开 HCX。	云管理员、系统管理员

Task	描述	所需技能
	5. 选择部署 HCX 并确认。HCX 部署将开始。	
生成 HCX 激活密钥。	1. 在 AWS 控制台上的 VMware Cloud 上。 2. 导航至 SDCC，然后选择查看详细信息。 3. 选择 Add Ons（附加组件）选项卡。 4. 选择打开 HCX，然后选择激活密钥。 5. 选择创建激活密钥并复制密钥。	云管理员、系统管理员

Task	描述	所需技能
为云 SDDC 上的 HCX 添加防火墙规则。	部署 HCX Manager 后，您需要配置防火墙规则，以启用本地环境与 SDDC 之间的通信。您需要创建两条防火墙规则：一条用于入站通信，另一条则用于出站通信。 1. 在 VMware Cloud on AWS 控制台上，选择您的 SDDC，然后导航到“网络与安全”。 2. 选择网关防火墙，然后选择管理网关选项卡。 3. 选择添加规则，创建出站规则： a. 提供规则名称。 b. 编辑源，选择 HCX。 c. 编辑目标，并提供可以访问 HCX 的本地 IP 和子网。 d. 对于 Services（服务），选择 Any（任何）。 e. 对于 Action（操作），选择 Allow（允许）。 f. 选择 Publish（发布）。 4. 选择添加规则，并创建入站规则： a. 提供规则名称。 b. 编辑源，并提供可以访问 HCX 的本地 IP 和子网。 c. 编辑目标并选择 HCX。	云管理员、系统管理员

Task	描述	所需技能
	d. 对于服务，请选择 SSH、HTTPS、TCP (9443) 以及 ICMP。 e. 对于 Action（操作），选择 Allow（允许）。 f. 选择 Publish（发布）。	
在本地安装 HCX Manager。	1. 登录至云端 vCenter，然后从菜单中导航到 HCX。 2. 在 HCX 控制面板，选择管理、系统更新。 3. 申请 VMware HCX Connector 的下载链接，然后下载本地 OVA 文件。 4. 登录您的本地 vCenter，并使用下载的 OVA 文件部署 OVF 模板。 5. 在模板部署期间，根据提示提供静态 IP、NTP、DNS、DNS 搜索列表以及其他详细信息。 6. 验证所有详细信息，以完成 HCX Manager 部署。	云管理员、系统管理员

Task	描述	所需技能
在本地配置 HCX Manager。	1. 在浏览器中打开 HCX Manager：`https://<HCX_Manager_IP>:9433` 。 2. 使用部署期间提供的用户名与密码登录。 3. 输入您之前创建的激活密钥，然后选择激活以激活您的 HCX 实例。 4. 选择下一步转至下一步。 5. 选择本地数据中心位置，然后选择继续。 6. 在系统名称中，输入主机名，然后选择继续以完成激活。 7. 输入信息以配置 vCenter 连接。 8. 输入信息以配置 SSO/PSC 详细信息。 9. 选择重启以使您的更改生效。	云管理员、系统管理员

Task	描述	所需技能
配置站点配对。	在云端与本地配置 HCX 后，请按以下步骤配置它们之间的站点配对。 1. 登录本地 vCenter，然后导航到 HCX 控制面板。 2. 在左侧导航窗格中，选择站点配对，然后选择连接至远程站点。 3. 在连接至远程站点对话框，添加 HCX 云 URL 和凭证，然后选择连接。 站点配对完成后，站点配对控制面板将显示本地和云端 SDDC 已连接。	云管理员、系统管理员

Task	描述	所需技能
创建网络配置文件。	网络配置文件是对网络第 3 层组件的抽象。此配置文件是创建计算配置文件的先决条件。 1. 登录云 vCenter，然后导航至 HCX 控制面板。 2. 选择互连，选择网络配置文件选项卡，然后选择创建网络配置文件。 3. 配置网络配置文件： a. 选择 vCenter 服务器。 b. 选择网络。 c. 为配置文件添加名称。 d. 提供 IP 池、前缀长度、网关、DND 以及 MTU。 e. 选择 Create(创建)。 4. 按照同样的流程，在本地创建网络配置文件。	云管理员、系统管理员

Task	描述	所需技能
创建计算配置文件。	计算配置文件包含 HCX 的网络、存储以及计算详细信息。 在创建服务网格期间，HCX 在创建 HCX 设备时使用这些设置。 1. 登录本地 vCenter，然后导航到 HCX 控制面板。 2. 选择互连，选择计算配置文件选项卡，然后选择创建计算配置文件。 3. 指定计算配置文件名称。 4. 选择要启用的 HCX 服务，然后选择继续。 5. 选择服务资源。如果包含多个集群，请选择要为其激活 HCX 服务的每个集群，然后选择继续。 6. 选择用于部署 HCX 设备的计算与存储资源，然后选择继续。 7. 选择可用于访问 vCenter 和 ESXi 主机的管理界面的管理网络配置文件，然后选择继续。 8. 选择上行链路网络配置文件，该配置文件可用于访问远程站点上的互连设备，并且远程站点设备可以使用该配置文件访问本地互连设备，然后选择继续。 9. 选择 vMotion 网络配置文件，然后选择继续。	云管理员、系统管理员

Task	描述	所需技能
	10选择 vSphere 复制网络配置文件，然后选择继续 11为网络扩展选择相应的分布式交换机，然后选择继续。 12查看在 WAN 和 LAN 连接中需打开的所有端口，然后选择继续。 13要创建计算配置文件，请选择 Finish（完成）。 14按同样的步骤在云端创建计算配置文件。	

Task	描述	所需技能
创建服务网格。	服务网格为本地站点与云站点提供 HCX 服务配置。创建服务网格时，会启动在两个站点上部署 HCX 互连虚拟设备。互连服务必须在源站点创建。 1. 登录本地 vCenter，然后导航到 HCX 控制面板。 2. 选择互连，选择服务网格选项卡，然后选择创建服务网格。 3. 选择将在其间创建服务网格的源站点和目标站点，然后选择继续 4. 选择您之前创建的源站点和目标站点的计算配置文件，然后选择继续。 5. 选择要启用的 HCX 服务，然后选择继续。 6. 选择源站点和目标站点的上行链路配置文件，然后选择继续。 7. 查看资源和网络，然后选择继续。 8. 为服务网格提供一个名称，然后选择完成。 开始部署服务网格。您可以在服务网格的任务选项卡关注进度。部署完成后，将显示您为服务网格启用的所有 HCX 服务的状态。	云管理员、系统管理员

使用 HCX 扩展网络

Task	描述	所需技能
创建网络扩展。	您可使用 HCX 网络扩展功能在云 SDDC HCX 站点创建 L2 网络扩展，并桥接远程和源网络。 这使您可以将服务器从本地迁移到 AWS 上的 VMware Cloud，同时保留相同的 IP 地址。 1. 登录本地 vCenter，然后导航到 HCX 控制面板。 2. 选择服务、网络扩展。 3. 选择扩展网络 或创建网络扩展。 4. 选择相应的服务网格、分布式端口组或者 NSX 逻辑交换机。 5. 提供网关的 IP 地址，然后选择提交。 网络扩展完成后，系统会显示扩展已完成。	云管理员、系统管理员

使用 HCX 配置复制作业

Task	描述	所需技能
配置复制。	要使用 HCX VMs 进行复制，请执行以下操作：	云管理员、系统管理员

Task	描述	所需技能
	1. 登录本地 vCenter，然后导航到 HCX 控制面板。 2. 选择迁移，然后选择迁移 选项卡。 3. 提供移动组名称，选择要迁移的虚拟机，然后选择添加。 4. 选择目标计算容器、存储文件夹、迁移类型 (冷迁移、批量迁移、RAV、vMotion) 和切换计划。 5. 选择验证，等待验证完成，然后选择开始以开始复制。	

升级 HCX

Task	描述	所需技能
查看建议与步骤。	大型迁移项目可能持续六到八个月，有时甚至更长，并且会 VMware 定期发布包含软件修复、安全更新和错误修复的 HCX 更新。我们建议您使 HCX 和设备保持最新状态，以消除任何安全漏洞并利用新功能。 **Note** 如果您当前的 HCX 版本比最新版本或更早版本落后三个版本，则无	云管理员、系统管理员

Task	描述	所需技能
	法升级 HCX，必须重新部署它。 HCX 升级包括三个步骤： 1. 在本地和云端备份 HCX Manager。 2. 在本地和云端升级 HCX Manager。 3. 在本地和云端升级服务网格设备。 以下故事将更详细讨论这些步骤。	

Task	描述	所需技能
备份 HCX Cloud Manager。	适用于 AWS 云的 HCX C VMware loud Manager 由管理 VMware，因此您无法拍摄快照。若要备份 HCX Cloud Manager，您必须从 HCX 控制台下载备份并使用此备份还原 HCX 配置，以防升级失败或必须回滚到上一阶段。 1. 登录 HCX Cloud Manager，网址为 `https://<HCX_cloudmanager_ip_or_fqdn>:9433` 。 2. 导航到 管理、故障排除、备份和还原。 3. 在备份部分，选择生成以创建备份文件。 4. 选择下载以保存备份文件。 HCX-IX、HCX-NE 和 HCX-WO 等 HCX 服务设备不需单独备份。	云管理员、系统管理员

Task	描述	所需技能
在本地备份 HCX Manager。	您可通过两种方式在本地备份 HCX Manager：拍摄虚拟机快照或备份配置文件。 若要拍摄虚拟机快照，请执行以下操作： 1. 登录本地 vCenter。 2. 转到虚拟机和模板，然后导航至 HCX Manager 虚拟机。 3. 选择操作、快照、拍摄快照。 若要备份配置文件，请执行以下操作： 1. 登录 HCX Cloud Manager，网址为 `https://<HCX_cloudmanager_ip_or_fqdn>:9433` 。 2. 导航到 管理、故障排除、备份和还原。 3. 在备份部分，选择生成以创建备份文件。 4. 选择下载以保存备份文件。 HCX-IX、HCX-NE 和 HCX-WO 等 HCX 服务设备不需单独备份。	云管理员、系统管理员

Task	描述	所需技能
在本地和云端升级 HCX Manager。	您必须先在本地升级 HCX Manager，然后再升级 HCX Cloud Manager。 要在本地升级 HCX Manager，请执行以下操作： 1. 登录 vCenter，然后导航至 HCX 控制面板。 2. 选择系统、管理。 3. 在管理页面上，选择系统更新选项卡。可用服务更新版本列显示待处理的更新。 4. 选择选择服务更新，选择下载以下载更新供日后升级，或者选择下载和升级立即下载并部署更新。如果您选择下载，请选择升级并确认以在准备就绪后启动升级。 5. 升级完成后： • 在 HCX Manager 管理页面，验证是否显示了最新的 HCX 版本。 • 在 HCX 控制面板，检查网站配对是否已启动。 • 选择基础设施、服务网格，并确认所有 HCX 服务都运行正常。 按同样的步骤升级 HCX Cloud Manager。	云管理员、系统管理员

Task	描述	所需技能
升级服务网格设备。	服务网格的更新独立于源站点上的 HCX Manager。目标站点的服务网格设备会自动更新。 若要升级源站点的服务网格设备，请执行以下操作： 1. 登录 vCenter，然后导航至 HCX 控制面板。 2. 选择基础设施，然后选择服务网格选项卡。 3. 如果看到横幅 “服务网格设备的新版本可用，点击更新设备以升级到最新版本”，选择更新设备。 4. 在显示装置的对话框中，选择一个或多个装置，然后选择确定开始升级过程。(建议您更新所有服务网格设备。) 5. 为每个服务网格选择查看任务，以监控升级。 6. 升级完成后，将显示每台设备和服务的横幅，以确认成功完成。 7. 升级后验证隧道状态： • 选择基础设施、服务网格、查看设备。 • 隧道状态栏应显示为开启，屏幕不应显示设备的任何其他可用版本。	云管理员、系统管理员

移除 HCX 网络扩展

Task	描述	所需技能
取消网络扩展。	前面的步骤解释了如何使用 HCX 网络扩展功能创建 L2 网络扩展，并在 AWS 上从本地迁移到 VMware 云端 IPs 期间保持现有网络扩展。当 VMs 来自特定 VLAN 的所有内容都移至 AWS 上的 VMware 云端后，您必须取消本地站点和云 SDDC 之间的网络扩展，并在 SDDC 中使网络可路由。 我们建议您在所有 VMs 扩展网络从本地迁移到 VMware Cloud on AWS 后立即移除扩展网络，以避免延迟。 1. 登录本地 vCenter，然后导航到 HCX 控制面板。 2. 在 HCX 控制面板，选择服务、网络扩展。 3. 选择要取消扩展的网络，然后选择取消扩展网络。 4. 选择取消扩展后将云网络连接至云边缘网关。这将激活云端网络。	云管理员、系统管理员
在云 SDDC 中路由移动的网络。	1. 登录至 VMC 门户。 2. 导航到 SDCC，然后选择查看详细信息。 3. 选择网络和安全选项卡。 4. 在网络与安全页面上：	云管理员、系统管理员

Task	描述	所需技能
	• 选择网络、分段，然后确认最近未扩展的子网显示为可路由。 • 选择清单、组，然后将该子网添加到组中。 • 选择安全、分布式防火墙，然后确认该组是预期防火墙规则的一部分。	

卸载 HCX

Task	描述	所需技能
检查先决条件。	如果退出数据中心，我们建议您在迁移项目结束时卸载 HCX 并移除其组件。但是，如果您仍保留本地占用空间，则可能需要保持 HCX 运行。 在卸载 HCX 前，请确保： • 没有活跃迁移。 • 所有网络扩展都已删除。	云管理员、系统管理员
本地卸载 HCX。	1. 登录本地 vCenter，然后导航到 HCX 控制台。 2. 选择服务、迁移，然后确认您没有有效的迁移。 3. 选择服务、网络扩展，然后确认没有扩展的网络。 4. 选择基础设施、站点配对、服务网格。 5. 识别服务网格，然后选择删除。	云管理员、系统管理员

Task	描述	所需技能
	6. 出现确认提示时，再次选择 Delete（删除）。服务网格屏幕上会显示 “移除服务网格” 的横幅。 7. 对您拥有的任何其他服务网格重复第 5-6 步。 8. 要移除站点配对，请选择基础设施、站点配对，然后断开所有已配对站点的连接。 9. 移除 HCX Manager 设备： a. 登录本地 vCenter，然后导航到 HCX Manager 设备。 b. 选择操作、电源、关机。 c. 选择操作、从磁盘中删除。	

Task	描述	所需技能
从本地 vCenter 服务器取消注册 HCX 插件。	1. 登录 vCenter MOB 用户界面，网址为`https://<vc_fqdn>/mob` 。 2. 在属性部分，选择值列中的内容。 3. 在内容页面上，选择 ExtensionManager查看所有已注册的插件。 4. 记下以 `com.vmware.hybridity` 、`com.vmware.hcsp.alarm` 和 `com.vmware.vca.marketing.ngc.ui` 开头的扩展名。 5. 移除扩展： • 在 “方法” 部分中，选择UnregisterExtension。 • 输入步骤 4 中记下的扩展密钥，然后选择调用方法以移除该扩展名。 移除所有扩展后，HCX 插件将会从 vSphere Web Client 中消失。	云管理员、系统管理员

Task	描述	所需技能
在云端卸载 HCX。	若要移除云中的 HCX 服务网格和站点配对，请请重复前述卸载本地 HCX中的步骤。在 VMware Cloud on AWS 中，HCX 管理器由管理。VMware您无法将其从 vCenter 删除，但可以从 VMC 管理界面取消部署。 若要取消部署 HCX Manager，请执行以下操作： 1. 登录 VMC 管理界面。 2. 选择您的组织和 SDDC。 3. 选择 “附加组件” 可显示所有 SDDCs 已部署 HCX 的内容。 4. 选择取消部署 HCX。	云管理员、系统管理员

故障排除

事务	解决方案
配置 HCX 批量迁移时，您无法选择所要迁移的服务器。	原因：这些服务器迁移已取消，但清理期间未更新 HCX 数据库。HCX 认为数据库迁移仍在进行中，因此已将状态锁定为 “正在进行切换”。 解决方案：联系 VMware 支持团队清理 HCX 数据库。
切换失败，但可以使用强制关机选项。	原因：T VMware ools 版本不满足 HCX 批量迁移的先决条件，因此 HCX 无法关闭源虚拟机。 解决方案：将该 VMware 工具更新到适用于您的迁移类型的推荐版本。

事务	解决方案
迁移过程中，HCX 站点配对设备升级失败，并显示错误消息 “不允许正在进行的批量迁移操作”。	原因：切换后 HCX 数据库未更新。 解决方案：确保没有正在进行的迁移。升级站点配对设备时，选择强制升级。
割接失败，并显示错误 “资源可用性较低”。	原因：主机 VM 的存储空间不足。 解决方案：迁移前检查存储和计算资源。

相关资源

参考

- VMware AWS 上的 Cloud 功能
- VMware Cloud on AWS 概述和运营模式（AWS Prescriptive Guidence）
- 使用 VMware HCX 将 VMware SDDC 迁移到 AWS 上的 VMware 云端（AWS Prescriptive Guidence）
- VMware AWS VMware 云端的 HCX（VMware 文档）
- HCX HCX 发行说明（文档）VMware
- AWS 上的 SDDC 部署和最佳实践指南（AWS 白皮书）

工具

- VMware 使用 PowerCLI（VMware 云技术专区）在 AWS 上实现自动化

合作伙伴

- VMware AWS 上的云合作伙伴计划

视频

- VMware AWS 上的云（YouTube 视频）

使用 pg_transport 在两个 Amazon RDS 数据库实例之间传输 PostgreSQL 数据库

创建者：Raunak Rishabh（AWS）和 Jitender Kumar（AWS）

摘要

此模式描述了使用 pg_transport 扩展程序在适用于 PostgreSQL 数据库实例的两个 Amazon Relational Database Service（Amazon RDS）之间迁移超大型数据库的步骤。该扩展提供了物理传输机制以移动每个数据库。通过以最少的处理流式传输数据库文件，它为在数据库实例之间迁移大型数据库提供了一种极快的方法，将停机时间降到最低。此扩展程序使用拉取模式，其中，目标数据库实例从源数据库实例导入数据库。

先决条件和限制

先决条件

- 两个数据库实例必须运行相同的 PostgreSQL 主要版本。
- 数据库不得存在于目标上。否则，传输将失败。
- 在源数据库中，除了 pg_transport 之外的任何扩展程序都不能启用。
- 所有源数据库对象必须位于默认 pg_default 表空间中。
- 源数据库实例的安全组应允许来自目标数据库实例的流量。
- 安装像 psql 这样的 PostgreSQL 客户端，PgAdmin或者使用 Amazon RDS PostgreSQL 数据库实例。您可以在本地系统中安装客户端，也可以使用亚马逊弹性计算云 (Amazon EC2) 实例。在这种模式中，我们在 EC2 实例上使用 psql。

限制

- 您无法在 Amazon RDS for PostgreSQL 的不同主要版本之间传输数据库。
- 源数据库的访问权限和所有权不会转移到目标数据库。
- 您不能在只读副本或只读副本的父实例上传输数据库。
- 您不能在打算使用该方法传输的任何数据库表中使用 reg 数据类型。
- 您总共可以在数据库实例上同时运行 32 个传输（包括导入和导出）。
- 您不能重命名或包含/排除表。所有内容都按原样迁移。

小心

- 在移除扩展程序之前先进行备份，因为移除扩展程序还会移除依赖对象和一些对数据库运行至关重要的数据。
- 在确定 pg_transport 的 Worker 数量和 `work_mem` 值时，请考虑在源实例上的其他数据库上运行的实例类和进程。
- 传输开始时，源数据库上的所有连接都将终止，数据库将进入只读模式。

> **Note**
>
> 在一个数据库上运行传输时，它不会影响同一服务器上的其他数据库。

产品版本

- Amazon RDS for PostgreSQL 10.10 和更高版本，Amazon RDS for PostgreSQL 11.5 和更高版本。有关最新版本的信息，请参阅 Amazon RDS 文档中的在数据库实例之间传输 PostgreSQL 数据库。

架构

工具

- pg_transport 提供了物理传输机制以移动每个数据库。通过以最少的处理流式传输数据库文件，物理传输移动数据的速度比传统的转储和加载过程快得多，并且需要最少的停机时间。PostgreSQL 可传输数据库使用拉取模式，其中，目标数据库实例从源数据库实例导入数据库。在准备源环境和目标环境时，您可以在数据库实例上安装此扩展程序，如本模式中所述。
- psql 使您能够连接和使用 PostgreSQL 数据库实例。要在系统上安装 psql，请参阅 PostgreSQL 下载量页面。

操作说明

创建目标参数组

Task	描述	所需技能
创建目标系统的参数组。	指定一个将其标识为目标参数组的组名；例如，`pgtarget-`	数据库管理员

Task	描述	所需技能
	`param-group` 。有关说明，请参阅 Amazon RDS 文档。	

Task	描述	所需技能
修改参数组中的参数。	设置以下参数： 1. 将 pg_transport 添加到 shared_preload_libraries 参数。 `shared_preload_libraries = pg_stat_statements, pg_transport` 2. 设置 pg_transport.num_workers 参数。选择要与之一起运行传输的 Worker 数量。您设置的值决定了将在源中创建的 transport.send_file Worker 数量。 3. 将 max_worker_processes 的值增加到 pg_transport.num_workers 的值的三倍以上。例如，如果将 pg_transport.num_workers 的值设置为 4，则该 max_worker_processes 值应至少为 13。如果失败，pg_transport 建议使用最小值。 4. 将 pg_transport.timing 设置为 1。此设置允许在传输期间报告计时信息。	数据库管理员

Task	描述	所需技能
	5. 设置 pg_transport.work_mem 参数。此参数指定分配给每个 Worker 的最大内存。默认值为 128 MB。 有关参数组的更多信息，请参阅 Amazon RDS 文档。	

创建源参数组

Task	描述	所需技能
创建源系统的参数组。	指定一个将其标识为源参数组的组名；例如，pgsource-param-group 。有关说明，请参阅 Amazon RDS 文档。	数据库管理员
修改参数组中的参数。	设置以下参数： 1. 将 pg_transport 添加到 shared_preload_libraries 参数。 ``` shared_preload_libraries = pg_stat_statements, pg_transport ``` 2. 设置 pg_transport.num_workers 参数。目标中定义的此参数的值决定了要使用的 transport.send_fil	数据库管理员

Task	描述	所需技能
	e Worker 数量。如果您正在此实例上运行导入，请增加此值，但要考虑已在运行的 Worker 数量。 3. 在目标上将 `max_worker_processes` 的值增加到 `pg_transport.num_workers` 的值的三倍以上。例如，如果您在目标上将 `pg_transport.num_workers` 的值设置为 4，则在源上的该 `max_worker_processes` 值应至少为 13。如果失败，pg_transport 建议使用最小值。 4. 设置 `pg_transport.work_mem` 参数。此参数指定分配给每个 Worker 的最大内存。默认值为 128 MB。 有关参数组的更多信息，请参阅 Amazon RDS 文档。	

准备目标环境

Task	描述	所需技能
创建一个新的 Amazon RDS for PostgreSQL 数据库实例，将源数据库传输到该数据库中。	根据业务需求确定实例类和 PostgreSQL 版本。	数据库管理员、系统管理员、数据库架构师

Task	描述	所需技能
修改目标的安全组以允许从实例通过数据库实例端口进行 EC2 连接。	PostgreSQL 实例的默认端口为 5432。如果您使用的是其他端口，则必须为该 EC2 实例打开与该端口的连接。	数据库管理员、系统管理员
修改实例，然后分配新的目标参数组。	例如，`pgtarget-param-group` 。	数据库管理员
重启 Amazon RDS 目标数据库实例。	参数 `shared_preload_libraries` 和 `max_worker_processes` 是静态参数，需要重启实例。	数据库管理员、系统管理员
使用 psql 从 EC2 实例连接到数据库。	使用 命令： `psql -h <rds_end_point> -p PORT -U username -d database -W`	数据库管理员
创建 pg_transport 扩展程序。	以具有该 `rds_superuser` 角色的用户身份运行以下查询： `create extension pg_transport;`	数据库管理员

准备源环境

Task	描述	所需技能
修改源的安全组以允许通过数据库实例端口从 Amazon EC2 实例和目标数据库实例进行连接	默认情况下，PostgreSQL 实例的端口为 5432。如果您使用的是其他端口，则必须为该 EC2 实例打开与该端口的连接。	数据库管理员、系统管理员

Task	描述	所需技能
修改实例，然后分配新的源参数组。	例如，pgsource-param-group 。	数据库管理员
重启 Amazon RDS 源数据库实例。	参数 shared_preload_libraries 和 max_worker_processes 是静态参数，需要重启实例。	数据库管理员
使用 psql 从 EC2 实例连接到数据库。	使用 命令： `psql -h <rds_end_point> -p PORT -U username -d database -W`	数据库管理员
创建 pg_transport 扩展程序并从要传输的数据库中删除所有其他扩展程序。	如果源数据库上安装了 pg_transport 以外的任何扩展程序，则传输将失败。此命令必须由具有该 rds_superuser 角色的用户运行。	数据库管理员

执行传输

Task	描述	所需技能
执行试运行。	使用该 transport.import_from_server 函数先执行试运行： `SELECT transport.import_from_server( 'source-db-instance-endpoint', source-db-instance-port, 'source-db-instance-user', 'source-user-password', 'source-`	数据库管理员

Task	描述	所需技能
	`database-name', 'destination-user-password', 'true');` 此函数的最后一个参数（设置为 `true`）定义了试运行。 此函数显示您在运行主传输时会看到的任何错误。在运行主传输之前解决错误。	
如果试运行成功，则启动数据库传输。	运行该 `transport.import_from_server` 函数以执行传输。它连接到源并导入数据。 `SELECT transport.import_from_server( 'source-db-instance-endpoint', source-db-instance-port, 'source-db-instance-user', 'source-user-password', 'source-database-name', 'destination-user-password', false);` 此函数的最后一个参数（设置为 `false`）表示这不是试运行。	数据库管理员

Task	描述	所需技能
执行传输后步骤。	数据库传输完成后： • 在目标环境中验证数据。 • 向目标添加所有角色和权限。 • 如果需要，在目标和源中启用所有必需的扩展程序。 • 恢复 `max_worker_processes` 参数的值。	数据库管理员

相关的资源

- Amazon RDS 文档
- pg_transport 文档
- 使用 RDS PostgreSQL 可传输数据库迁移数据库（博客文章）
- PostgreSQL 下载量
- psql 实用程序
- 创建数据库参数组
- 修改数据库参数组中的参数
- PostgreSQL 下载量

更换平台

主题

- 使用 AWS DMS 将 Microsoft SQL Server 数据库导出至 Amazon S3
- SageMaker 使用 AWS 开发人员工具将 ML 构建、训练和部署工作负载迁移到 Amazon
- 将 OpenText TeamSite 工作负载迁移到 AWS 云
- 将 Oracle CLOB 值迁移到 AWS 上 PostgreSQL 中的单独的行
- 使用通过数据库链接直接导入 Oracle Data Pump 将本地 Oracle 数据库迁移到 Amazon RDS for Oracle
- 将 Oracle 电子商务套件迁移到 Amazon RDS Custom
- 将 Oracle 迁移 PeopleSoft 到亚马逊 RDS 定制版
- 将 Oracle ROWID 功能迁移到 AWS 上的 PostgreSQL
- 将 Oracle 数据库错误代码迁移到 Amazon Aurora PostgreSQL-Compatible 数据库
- 将 Redis 工作负载迁移至 Redis Enterprise Cloud on AWS
- 使用 AWS SCT 和 AWS DMS EC2 将亚马逊上的 SAP ASE 迁移到兼容 Aurora PostgreSQL 的亚马逊 Aurora
- 使用 ACM 将 Windows SSL 凭证迁移到应用程序负载均衡器
- 将消息队列从 Microsoft Azure 服务总线迁移到 Amazon SQS
- 将关系数据库迁移到 MongoDB Atlas AWS
- 将自托管 MongoDB 环境迁移到 MongoDB Atlas AWS
- 使用 Oracle 数据泵和 AWS DMS 将 Oracle JD Edwards EnterpriseOne 数据库迁移到 AWS
- 使用 AWS DMS 将 Oracle PeopleSoft 数据库迁移到 AWS
- 将本地 MySQL 数据库迁移至 Amazon RDS for MySQL
- 将本地 Microsoft SQL Server 数据库迁移至 Amazon RDS for SQL Server
- 使用 Rclone 将数据从 Microsoft Azure Blob 迁移至 Amazon S3
- 从 Couchbase Server 迁移至 AWS 上的 Couchbase Capella
- 在 Amazon 上从 IBM WebSphere 应用程序服务器迁移到 Apache Tomcat EC2
- 使用 Auto Scaling 从 IBM WebSphere 应用程序服务器迁移到亚马逊 EC2 上的 Apache Tomcat
- 将 .NET 应用程序从 Microsoft Azure 应用服务迁移到 AWS Elastic Beanstalk
- 在 Amazon ECS 上从 Oracle 迁移 WebLogic 到 Apache Tomcat (ToMee)
- 使用 AWS DMS 将 Oracle 数据库从亚马逊迁移 EC2 到 Amazon RDS for Oracle

- 使用 Logstash 将本地 Oracle 数据库迁移到亚马逊 OpenSearch 服务
- 将本地 Oracle 数据库迁移到 Amazon RDS for Oracle
- 使用 Oracle 数据泵将本地 Oracle 数据库迁移到 Amazon RDS for Oracle
- 使用 pglogical 从 EC2 亚马逊上的 PostgreSQL 迁移到适用于 PostgreSQL 的亚马逊 RDS
- 将本地 PostgreSQL 数据库迁移到 Aurora PostgreSQL
- 将本地微软 SQL Server 数据库迁移到 EC2 运行 Linux 的亚马逊上的微软 SQL Server
- 使用链接服务器将本地 Microsoft SQL Server 数据库迁移至 Amazon RDS for SQL Server
- 使用本机备份和还原将本地 Microsoft SQL Server 数据库迁移到 Amazon RDS for SQL Server。
- 使用 AWS DMS 和 AWS SCT 将 Microsoft SQL Server 数据库迁移到 Aurora MySQL
- 使用原生工具将本地 MariaDB 数据库迁移至 Amazon RDS for MariaDB
- 将本地 MySQL 数据库迁移至 Aurora MySQL
- 使用 Percona、A XtraBackup mazon EFS 和 Amazon S3 将本地 MySQL 数据库迁移到 Aurora MySQL
- 使用 AWS App2Container 将本地 Java 应用程序迁移到 AWS
- 在 AWS 大规模迁移中迁移共享文件系统
- 使用 Oracle GoldenGate 平面文件适配器将 Oracle 数据库迁移到 Amazon RDS for Oracle
- 更改 Python 和 Perl 应用程序以支持数据库从Microsoft SQL Server 迁移至兼容 Amazon Aurora PostgreSQL 的版本
- 将数据从 IBM Db2、SAP、Sybase 和其他数据库流式传输到 MongoDB Atlas AWS

使用 AWS DMS 将 Microsoft SQL Server 数据库导出至 Amazon S3

由 Sweta Krishna (AWS) 编写

摘要

组织通常需要将数据库复制到 Amazon Simple Storage Service (Amazon S3) 以进行数据库迁移、备份和恢复、数据归档和数据分析。此模式描述了如何将 Microsoft SQL Server 数据库导出到Amazon S3。源数据库可以托管在本地，也可以托管在亚马逊弹性计算云 (Amazon EC2) 上，也可以托管在亚马逊网络服务 (AWS) 云上的微软 SQL Server 的亚马逊关系数据库服务 (Amazon RDS) 上。

使用 AWS Database Migration Service (AWS DMS) 导出数据。默认情况下，AWS DMS 以逗号分隔值 (.csv) 格式写入完全加载和更改数据捕获（CDC）数据。为了获得更紧凑的存储和更快的查询选项，此模式使用 Apache Parquet (.parquet) 格式选项。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 账户的 AWS Identity and Access Management (IAM) 角色，具有对目标 S3 存储桶的写入、删除和标记访问权限，并且 AWS DMS (`dms.amazonaws.com`) 作为可信实体添加到此 IAM 角色
- 本地微软 SQL Server 数据库（或 EC2 实例上的微软 SQL Server 或 Amazon RDS for SQL Server 数据库）
- AWS 上的虚拟私有云（VPC）与 AWS Direct Connect 或虚拟专用网络（VPN）提供的本地网络之间的网络连接

限制

- 3.4.7 之前的 AWS DMS 版本目前不支持启用了 vPC（网关 VPC）的 S3 存储桶。
- 不支持在完全加载期间对源表结构进行更改。
- 不支持 AWS DMS 完整大型二进制对象 (LOB) 模式。

产品版本

- Microsoft SQL Server 版本 2005 或更高版本 (Enterprise、Standard、Workgroup 和 Developer 版)。
- AWS DMS 版本 3.3.2 和更高版本中支持将 Microsoft SQL Server 版本 2019 作为源。

架构

源技术堆栈

- 本地微软 SQL Server 数据库（或 EC2 实例上的微软 SQL Server 或 Amazon RDS for SQL Server 数据库）

目标技术堆栈

- AWS Direct Connect
- AWS DMS
- Amazon S3

目标架构

工具

- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到 Amazon Web Services Cloud，或者在云和本地设置的组合之间迁移。
- AWS Direct Connect 通过标准的以太网光纤电缆将内部网络链接到 Direct Connect 位置。通过此连接，您可以直接创建连接到公有 Amazon Web Services 的虚拟接口，同时绕过网络路径中的互联网服务提供商。
- Amazon Simple Storage Service(Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

操作说明

准备迁移

Task	描述	所需技能
验证数据库版本。	验证源数据库版本，并确保 AWS DMS 支持该版本。有关支持的 SQL Server 数据库版本的信息，请参阅使用	数据库管理员

Task	描述	所需技能
	Microsoft SQL Server 数据库作为 AWS DMS 的源。	
创建 VPC 和安全组。	在您的 Amazon Web Services account 中，创建 VPC 和安全组。有关更多信息，请参阅 Amazon VPC 文档。	系统管理员
为 AWS DMS 任务创建用户。	在源数据库中创建 AWS DMS 用户，并授予其读权限。AWS DMS 将使用该用户。	数据库管理员
测试数据库连接。	测试 AWS DMS 用户与 SQL Server 数据库实例的连接。	数据库管理员
创建 S3 存储桶。	创建目标 S3 存储桶。此存储桶将存放至迁移的表数据。	系统管理员
创建 IAM policy 和角色。	1. 若要创建具有存储桶权限的 IAM policy，请使用其他信息部分中的代码。 2. 为 AWS DMS 创建角色，并将策略附加到该角色。	系统管理员

使用 AWS DMS 迁移数据

Task	描述	所需技能
创建 AWS DMS 复制实例。	登录 AWS 管理控制台，并打开 AWS DMS 控制台。在导航窗格中，选择复制实例，选择创建复制实例。有关说明，请参阅 AWS DMS 文档中的步骤 1。	数据库管理员

Task	描述	所需技能
创建源和目标端点。	创建源和目标端点。测试从复制实例至源端点和目标端点的连接。有关说明，请参阅 AWS DMS 文档中的步骤 2。	数据库管理员
创建复制任务。	创建复制任务，然后使用更改数据捕获 (CDC) 选择满载或满载，将数据从 SQL Server 迁移至 S3 存储桶。有关说明，请参阅 AWS DMS 文档中的步骤 3。	数据库管理员
启动数据复制。	启动复制任务并监控日志中是否存在错误。	数据库管理员

验证数据

Task	描述	所需技能
验证迁移数据。	在控制台中，导航到您的目标 S3 存储桶。打开与源数据库同名子文件夹。确认该文件夹包含从源数据库迁移的所有表格。	数据库管理员

清理资源

Task	描述	所需技能
关闭并删除临时 AWS 资源。	关闭您为数据迁移创建的临时 AWS 资源（例如 AWS DMS 复制实例），并在验证导出后将其删除。	数据库管理员

相关的资源

- AWS Database Migration Service 用户指南
- 使用 Microsoft SQL Server 数据库作为 AWS DMS 的源
- 使用 Amazon S3 作为 AWS Database Migration Service 的目标
- 使用 S3 存储桶作为 AWS DMS 目标 (AWS re：Post)

其他信息

使用以下代码，为 AWS DMS 角色添加具有 S3 存储桶权限的 IAM policy。将 `bucketname`替换为您的桶的名称。

```
{
     "Version": "2012-10-17",
     "Statement": [
         {
             "Effect": "Allow",
             "Action": [
                 "s3:PutObject",
                 "s3:DeleteObject"
             ],
             "Resource": [
                 "arn:aws:s3:::bucketname*"
             ]
         },
         {
             "Effect": "Allow",
             "Action": [
                 "s3:ListBucket"
             ],
             "Resource": [
                 "arn:aws:s3:::bucketname*"
             ]
         }
     ]
 }
```

SageMaker 使用 AWS 开发人员工具将 ML 构建、训练和部署工作负载迁移到 Amazon

由 Mustafa Waheed (AWS) 创建

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

此模式为使用 Amazon 迁移在 Unix 或 Linux 服务器上运行的本地机器学习 (ML) 应用程序以在 AWS 上进行训练和部署提供了指导 SageMaker。此部署使用了连续集成和连续部署 (CI/CD) 管线。迁移模式是使用 AWS CloudFormation 堆栈部署的。

先决条件和限制

先决条件

- 使用 AWS 登录区的有效 Amazon Web Services account
- AWS 命令行界面（AWS CLI）已在您的 Unix 或 Linux 服务器上安装并配置
- 在 AWS 中预置的 ML 源代码存储库 CodeCommit

限制

- 一个 AWS 区域 中只能部署 300 个单独的管线。
- 此模式适用于带有 Python train-and-deploy 代码的受监管机器学习工作负载。

产品版本

- Docker 版本 19.03.5，内部版本 633a0ea，使用 Python 3.6x

架构

源技术堆栈

- 本地 Linux 计算实例，数据位于本地文件系统或关系数据库中

源架构

目标技术堆栈

- AWS 与 Amazon S3 一起 CodePipeline 部署用于数据存储，将 Amazon DynamoDB 部署为元数据存储，用于跟踪或记录管道运行情况

目标架构

应用程序迁移架构

- 原生 Python 包和 AWS CodeCommit 存储库（以及用于数据库实例上的本地数据集的 SQL 客户端）

工具

- Python3
- Git
- AWS CLI — AWS CLI 部署 AWS CloudFormation 堆栈并将数据移至 S3 存储桶。反过来，S3 存储桶会指向目标。

操作说明

计划迁移

Task	描述	所需技能
验证源代码与数据集。		数据科学家
识别目标构建、训练和部署实例类型和大小。		数据工程师、数据科学家
创建功能列表和容量要求。		
识别网络要求。		数据库管理员、系统管理员

Task	描述	所需技能
识别源应用程序和目标应用程序的网络或主机访问安全要求。		数据工程师、机器学习工程师、系统管理员
确定备份策略。		机器学习工程师、系统管理员
确定可用性要求。		机器学习工程师、系统管理员
识别应用程序迁移或切换策略。		数据科学家、机器学习工程师

配置基础设施

Task	描述	所需技能
创建虚拟私有云（VPC）。		机器学习工程师、系统管理员
创建安全组。		机器学习工程师、系统管理员
为机器学习代码设置 Amazon S3 存储桶和 AWS CodeCommit 存储库分支。		机器学习工程师

上传数据和代码

Task	描述	所需技能
使用原生 MySQL 工具或第三方工具将训练、验证和测试数据集迁移到预调配的 S3 存储桶。	这是部署 AWS CloudFormation 堆栈所必需的。	数据工程师、机器学习工程师
Package 将 ML 训练和托管代码打包为 Python 包，然后推	您需要存储库的分支名称才能部署 AWS CloudFormation 模板进行迁移。	数据科学家、机器学习工程师

Task	描述	所需技能
送到 AWS 中的预配置存储库 CodeCommit 或 GitHub。		

迁移应用程序

Task	描述	所需技能
遵循机器学习工作负载迁移策略。		应用程序所有者、机器学习工程师
部署 AWS CloudFormation 堆栈。	使用 AWS CLI 创建此解决方案提供的 YAML 模板中声明的堆栈。	数据科学家、机器学习工程师

割接

Task	描述	所需技能
将应用程序客户端切换至新基础设施。		应用程序所有者、数据科学家、机器学习工程师

关闭项目

Task	描述	所需技能
关闭临时 AWS 资源。	关闭 AWS CloudFormation 模板中的所有自定义资源（例如，任何未使用的 AWS Lambda 函数）。	数据科学家、机器学习工程师
查看和验证项目文档。		应用程序所有者、数据科学家

Task	描述	所需技能
使用运算符验证结果和机器学习模型评估指标。	确保模型性能符合应用程序用户的期望，并且与本地状态相当。	应用程序所有者、数据科学家
关闭项目并提供反馈。		应用程序所有者、机器学习工程师

相关资源

- AWSCodePipeline
- AWS CodeCommit
- AWSCodeBuild
- AmazonSageMaker
- Amazon S3
- Amazon DynamoDB
- AWS Lambda

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

将 OpenText TeamSite 工作负载迁移到 AWS 云

创建者：Battulga Purevragchaa（AWS）、Michael Stewart 和 Carlos Marruenda Molina

摘要

> **Warning**
>
> 此场景需要 IAM 用户具有编程访问权限和长期凭证，这会带来安全风险。为帮助减轻这种风险，我们建议仅向这些用户提供执行任务所需的权限，并在不再需要这些用户时将其移除。必要时可以更新访问密钥。有关更多信息，请参阅 IAM 用户指南中的更新访问密钥。

许多 E OpenText xperience Platform 实例都托管在本地或具有固定容量和传统成本模式的传统托管解决方案上。将您的 OpenText 体验平台工作负载迁移到 Amazon Web Services (AWS) 云除了降低总体拥有成本外，还可以提高业务灵活性和集成机会，从而提供额外的功能和价值。

此模式提供了将OpenText TeamSite工作负载迁移到 AWS 云的步骤和模板。该模式通过提供详细的长篇故事部分来指导您完成迁移过程，从而帮助您了解如何确定 OpenText TeamSite 迁移项目的范围和预算。

该模式由 AWS 和 AWS 合作伙伴 TBSCG 开发，并附有 AWS Prescriptive Guidence 网站上的《将媒体管理工作负载迁移到 OpenText TeamSite AWS 云》指南。

先决条件和限制

先决条件

- 至少一个有效的 Amazon Web Services account
- 托管在本地数据中心或其他云提供商上的 OpenText 工作负载
- 有效 OpenText 许可证

迁移过程还需要下表中描述的角色和责任。

角色	责任
赞助商	内部赞助

传送经理	迁移交付
解决方案架构师	定义当前架构和新架构
DevOps 工程师	DevOps 活动
QA 测试员	系统级测试
产品所有者	根据业务需求确定任务优先级
TeamSite 作者	迁移用户验收测试 (UAT)
TeamSite 管理员	迁移 UAT
OpenText 铅	OpenText 产品专家
OpenText 开发者	OpenText 产品专家
定价专家	AWS 和 OpenText 许可
IT 安全	IT 安全基准
第三方集成开发人员	重做现有集成
前端开发人员	对迁移的前端代码进行更改
数据库管理员	数据库配置

限制

- 确保与目标操作系统的兼容性 (OSs)。您可以使用正在迁移的产品版本 OpenText 的产品发行说明中的兼容性列表。

架构

源技术堆栈

- OpenText 托管在本地或其他云提供商上的客户体验解决方案：
  - OpenText TeamSite
  - OpenText LiveSite

- OpenText 媒体管理
- OpenText MediaBin

目标技术堆栈

- 托管在 AWS 云上且使用以下 AWS 服务的 OpenText 客户体验平台：
  - 亚马逊弹性计算云（亚马逊 EC2）
  - Amazon Elastic Container Service (Amazon ECS)
  - 亚马逊 OpenSearch 服务
  - Elastic Load Balancing
  - AWS Lambda
  - Amazon API Gateway
  - Amazon Relational Database Service (Amazon RDS)
  - Amazon Elastic Block Store（Amazon EBS）
  - Amazon Simple Storage Service（Amazon S3）

目标架构

工具

- AWS Database Migration Service（AWS DMS）是一项云服务，可轻松迁移关系数据库、数据仓库、NoSQL 数据库及其他类型的数据存储。
- AWS 应用程序迁移服务可自动将源服务器转换为在 AWS 上本地运行。它还通过内置和自定义优化选项简化了应用程序现代化。

操作说明

发现与评测

Task	描述	所需技能
举办关于发现要求的研讨会。	与业务和技术团队举行研讨会，了解当前形势，收集要求并验证迁移策略。根据迁移的复杂性和范围，贵组织可能需要举办几次研讨会。 时长：两周	赞助商（可选）、交付经理、解决方案架构师、 OpenText 负责人、产品负责人
分析解决方案和迁移要求。	分析并记录影响计划解决方案设计和迁移过程的业务、功能和技术要求。 时长：一周	解决方案架构师、 OpenText 负责人、产品负责人
记录您的现有 OpenText 架构。	记录您的现有 OpenText 架构，包括核心组件和所有相关的应用程序和服务。 时长：一周	解决方案架构师、 OpenText 负责人、产品负责人
定义计划中的 AWS 架构。	根据已确定的组件、要求并使用 OpenText 兼容性矩阵定义计划中的 AWS 架构。你可以在 OpenText TeamSite 版本的发行说明中找到 OpenText 兼容性表。 时长：一周	解决方案架构师、 OpenText 负责人、产品负责人、IT 安全
评测您计划的 AWS 架构的规模。	不同架构组件的大小要求因工作负载和其他非功能要求而异。 时长：两天	解决方案架构师、 OpenText 主管

Task	描述	所需技能
计算 TCO。	计算您建议的解决方案的总拥有成本（TCO）。 时长：两天	解决方案架构师、定价专家
定义每个组件的迁移策略。	对于必须迁移到 Amazon Web Services Cloud 的每个核心或其他组件，定义并记录使用七种常见迁移策略（7 R）中的哪一种。 时长：一周	解决方案架构师、 OpenText 负责人、产品负责人
定义组件的迁移过程。	为每个工作负载组件定义详细的迁移过程。 时长：一周	解决方案架构师、 OpenText 负责人、产品负责人、IT 安全
定义全球迁移过程和依赖项。	创建全球迁移过程和日历，其中包括组件、依赖项和业务连续性的迁移详细信息。 时长：三天	解决方案架构师、 OpenText 负责人、产品负责人、IT 安全

安全与合规活动

Task	描述	所需技能
创建安全策略。	在 Amazon Web Services account 中配置客户托管的安全策略。除了自动关闭未使用的账户外，还应包括密码的复杂性和轮换。 有关客户管理型策略的更多信息，请参阅在 AWS Identity	解决方案架构师

Task	描述	所需技能
	and Access Managemen t（IAM）文档中的客户管理型策略。	
创建 IAM 用户。	创建需要访问 AWS 管理控制台、AWS 命令行界面（AWS CLI）和 AWS 软件开发工具包的 IAM 用户。 有关创建 IAM 用户的更多信息，请参阅 IAM 文档中的在您的 Amazon Web Services account 中创建 IAM 用户。	解决方案架构师
创建 IAM 组。	创建所需 IAM 用户组（例如，管理员或开发人员组），并将 IAM 用户添加到这些组。 有关 IAM 用户组的更多信息，请参阅 IAM 文档中的 IAM 用户组。	解决方案架构师
附加安全策略。	向 IAM 群组或角色附加安全策略。 有关这方面的更多信息，请参阅 IAM 文档中的将策略附加到 IAM 用户组。	解决方案架构师
开启详细账单。	有关账单的更多信息，请参阅 AWS 账单与成本管理文档中的监控使用量和成本。	解决方案架构师

Task	描述	所需技能
查看账户的联系方式。	确保账户的联系方式是最新的，并与贵组织中的多个个人对应。 有关更多信息，请参阅 AWS 账单与成本管理文档中的管理 Amazon Web Services account。	解决方案架构师、产品负责人
添加安全联系人信息。	使用安全联系人信息配置联系信息。 有关这方面的更多信息，请参阅 AWS 账单与成本管理文档中的管理 Amazon Web Services account。	解决方案架构师、IT 安全
为 EC2 实例设置 IAM 角色。	为 EC2 实例配置 IAM 角色。 有关这 EC2方面的更多信息，请参阅 Amazon EC2 文档中的 Amazon 的 IAM 角色。	解决方案架构师
配置对 Amazon Web Services Support 的访问权限。	向需要访问 Amazon Web Services Support 进入支持中心和创建支持案例的 IAM 用户附上 IAM policy。 有关这方面的更多信息，请参阅 Amazon Web Services Support 文档中的 Amazon Web Services Support 访问权限。	解决方案架构师

Task	描述	所需技能
启用 CloudTrail。	CloudTrail 在您的所有 AWS 区域自动启用 AWS。 有关这方面的更多信息，请参阅 AWS CloudTrail 文档`create-trail` 中的使用。	解决方案架构师
启用 CloudTrail 日志文件验证。	启用 CloudTrail 日志文件验证。 有关这方面的更多信息，请参阅 AWS CloudTrail 文档 CloudTrail中的启用日志文件完整性验证。	解决方案架构师
限制对包含 CloudTrail 日志的任何 S3 存储桶的访问权限。	应用存储桶策略，限制对包含 CloudTrail 日志文件的 S3 存储桶的访问权限。 有关这 CloudTrail方面的更多信息，请参阅 AWS CloudTrail 文档中的 Amazon S3 存储桶策略。	解决方案架构师
CloudTrail 与 CloudWatch 日志集成	将生成的跟踪 CloudTrail 与 Amazon CloudWatch 日志集成。 有关这方面的更多信息，请参阅 AWS CloudTrail 文档中的向 CloudWatch 日志发送事件	解决方案架构师

Task	描述	所需技能
在所有必需的区域启用 AWS Config。	在所有必需的区域自动启用 AWS Config。 您可以使用 AWS CLI 设置 AWS Config。有关更多信息，请参阅 AWS Config 文档中的使用 AWS CLI 设置 AWS Config。	解决方案架构师
启用 S3 存储桶访问日志记录。	使用自动记录 S3 存储桶访问权限 CloudTrail。 有关这方面的更多信息，请参阅 Amazon S3 文档中的“为 S3 存储桶和对象启用 CloudTrail 事件记录”。	解决方案架构师
为配置 AWS KMS 密钥策略 CloudTrail。	自动配置 AWS 密钥管理服务 (AWS KMS) 的密钥策略 CloudTrail。 有关这方面的更多信息，请参阅 AWS CloudTrail 文档 CloudTrail中的配置 AWS KMS 密钥策略。	解决方案架构师
加密静态 CloudTrail 日志。	使用 AWS KMS 中 CloudTrail 保存的客户托管密钥配置日志的服务器端加密。 有关这方面的更多信息，请参阅 AWS 文档中的使用 AWS KMS 托管密钥 (SSE-KMS) 加密 CloudTrail 日志文件。CloudTrail	解决方案架构师

Task	描述	所需技能
自动轮换 KMS 密钥。	配置 AWS KMS 密钥的轮换。 有关这方面的更多信息，请参阅 AWS KMS 文档中的如何启用和禁用自动密钥轮换。	解决方案架构师
配置 CloudWatch 警报。	配置由特定事件启动的 Amazon CloudWatch 警报。例如，对根账户的未经授权的请求 APIs 或使用。 有关这方面的更多信息，请参阅 AWS 安全博客中的如何在使用 Amazon Web Services account 的根访问密钥时接收通知。	解决方案架构师
配置安全组。	配置安全组以确保端口 22 和 3389 不允许不受限制的入站流量。	解决方案架构师
打开 VPC 流量日志记录。	捕获进出虚拟私有云 (VPC) 网络接口的被拒绝的 IP 流量，并进行配置 CloudWatch 以捕获该流量。 有关这方面的更多信息，请参阅 Amazon VPC 文档中的创建流量日志。	解决方案架构师

Task	描述	所需技能
修改默认安全组以限制所有流量。	修改每个 VPC 的默认安全组，以便在默认情况下拒绝流量，并通过安全组明确授予访问权限。 有关这方面的更多信息，请参阅 Amazon VPC 文档中的 VPC 的安全组。	解决方案架构师
在之间配置路由表 VPCs。	使用所需最低访问权限配置 VPC 对等互连的路由表。 有关这方面的更多信息，请参阅 Amazon VPC 文档中的为 VPC 对等连接更新路由表。	解决方案架构师

新 AWS 基础设施的设置活动

Task	描述	所需技能
预调配 AWS 基础设施。	创建 Amazon Web Services account 和资源。 时长：两周	DevOps 工程师，解决方案架构师
设置 DevOps 工具和流程。	设置 DevOps 工具和程序，例如持续集成和持续交付 (CI/CD) 管道以及自动化测试框架。	DevOps 工程师，解决方案架构师
自动迁移核心组件。	使用现有模板或脚本自动安装和配置 OpenText 产品 TeamSite，包括 LiveSite、OpenDeploy 和 MediaBin。 时长：一周	DevOps 工程师、解决方案架构师、 OpenText 主管

Task	描述	所需技能
自动迁移其他组件。	分析并自动迁移与 OpenText 核心组件集成的其他应用程序（例如，其他数据库、通信、监控或缓存组件）。 时长：两周	DevOps 工程师、解决方案架构师、 OpenText 主管
调整核心组件。	对 OpenText 核心组件的自定义（例如集成）进行任何必要的更改。	解决方案架构师、 OpenText 主管、 OpenText 开发人员、第三方集成开发人员、前端开发人员
实施和配置其他服务。	预调配、配置和实施任何新的 Amazon Web Services，例如 AWS Lambda 函数或 Amazon API Gateway。	DevOps 工程师、解决方案架构师、第三方集成开发人员、前端开发人员
迁移或重构其他组件。	迁移其他组件，包括任何必需的重构。这包括外部应用程序，例如定制的报告门户或现有的 API 集成层。	DevOps 工程师、解决方案架构师、第三方集成开发人员、前端开发人员
在开发环境中进行迁移。	开发环境的自动迁移活动，包括系统预调配、数据迁移、应用程序迁移、安装和配置。	DevOps 工程师
在生产环境中执行迁移。	生产环境的自动迁移活动，包括系统预调配、数据迁移、应用程序迁移、安装和配置。	DevOps 工程师

联网活动

Task	描述	所需技能
定义每个 VPC 的 CIDR 块。	定义每个非默认 VPC 的无类别域间路由 (CIDR) 块（IP 范围和掩码）。 时长：不到一周	DevOps 工程师，解决方案架构师
定义子网和可用区。	定义每个非默认 VPC 中使用的子网和可用区。 时长：不到一周	DevOps 工程师，解决方案架构师
定义安全组。	定义用于控制 AWS 资源安全的安全组和安全组规则。 时长：不到一周	DevOps 工程师，解决方案架构师
定义网络 ACLs。	定义网络访问控制列表 (ACLs) 以控制子网边界的安全。 时长：不到一周	DevOps 工程师，解决方案架构师

迁移数据库

Task	描述	所需技能
准备源数据库。	使用 AWS DMS 为每个源数据库做好准备，以便持续复制到 Amazon Web Services Cloud 中。	DevOps 工程师，解决方案架构师
为 OpenText 核心组件创建数据库。	创建 Opentext TeamSite LiveSite、和 MediaBin 组件所需的数据库。确保根据	解决方案架构师、 OpenText 负责人、 OpenText 开发人员

Task	描述	所需技能
	OpenText 安装文档正确配置了用户和访问权限。	
从源数据库服务器复制数据。	自动执行将 OpenText 核心组件的数据从源数据库服务器复制到目标数据库服务器的过程。	解决方案架构师、 OpenText 负责人、 OpenText 开发人员
同步来自数据库服务器的数据。	自动执行从源数据库到目标数据库的定期数据同步的过程。	OpenText 开发者

内容迁移活动

Task	描述	所需技能
复制 OpenText TeamSite 内容存储。	自动执行将内容存储从源 OpenText TeamSite 服务器复制到目标 OpenText TeamSite 服务器的过程。	解决方案架构师、 OpenText 负责人、 OpenText 开发人员
映射用户和组。	内部 OpenText TeamSite 用户 IDs 到目标系统的内部映射 IDs。	OpenText 铅
同步 OpenText TeamSite 内容存储。	自动执行源内容存储和目标内容存储的定期同步过程。这是作为迁移和 QA 过程的一部分实施的。	OpenText 开发者
从 Web 服务器复制数据。	自动执行将数据从 Web 源服务器复制到 Web 目标服务器的过程。	解决方案架构师、 OpenText 负责人、 OpenText 开发人员
同步 Web 服务器数据。	自动执行 Web 源服务器数据和 Web 目标服务器数据的定期同步过程。	OpenText 开发者

Task	描述	所需技能
从 Web 服务器文件系统复制数据。	自动执行将内容和其他 Web 资产从 Web 源服务器文件系统复制到 Web 目标服务器的过程。	解决方案架构师、 OpenText 负责人、 OpenText 开发人员
同步 Web 服务器文件系统。	自动执行将内容和其他 Web 资产从 Web 源服务器文件系统定期同步到 Web 目标服务器的过程。	OpenText 开发者
生成数据源和索引。	自动运行任何生成使用 OpenText TeamSite 或 Web 服务器内容作为数据源的源或其他索引（例如 Web 搜索）的进程。	解决方案架构师、 OpenText 负责人、 OpenText 开发人员
同步数据源和索引的生成。	自动执行数据同步后定期重新生成数据源和索引的过程。	OpenText 开发者

测试和质量保证活动

Task	描述	所需技能
执行迁移 QA。	测试目标 AWS 环境、应用程序和服务，确保正确构建和配置自动迁移过程。	DevOps 工程师、 OpenText 主管、QA 测试员
进行性能测试。	测试在特定工作负载下的响应能力和稳定性方面的性能。调查、测量、验证或验证目标系统的其他质量属性，例如可扩展性和可靠性。 要使此测试发挥作用，测试环境必须与生产环境大小相同。	DevOps 工程师， OpenText 主管

Task	描述	所需技能
	时长：一到两周之间	
安全测试。	漏洞扫描和渗透测试，以揭示应用程序安全机制中的潜在缺陷，从而根据需要保护数据和维护功能。 要使此测试发挥作用，就联网和安全性而言，您必须拥有与生产环境相当的测试环境。 时长：一到两周之间	DevOps 工程师， OpenText 主管

操作整合活动

Task	描述	所需技能
检查操作准备情况。	了解您当前如何执行 IT 操作以及您将如何在 Amazon Web Services Cloud 中操作。您可以通过定义云操作模型来实现此业务成果。 时长：一周	DevOps 工程师、 OpenText 主管、服务交付经理
投资操作自动化。	投资自动化，以交付 AWS 操作模式。	DevOps 工程师、 OpenText 主管、服务交付经理
整合操作。	继续使用当前的 IT 工具，并通过集成将其扩展到 AWS Cloud。	DevOps 工程师、 OpenText 主管、服务交付经理

割接活动

Task	描述	所需技能
切换 DNS。	手动将域名系统（DNS）从现有主机切换到基于 Amazon Web Services Cloud 的主机。 时长：一小时	DevOps 工程师，OpenText 主管
测试灾难恢复。	测试灾难恢复、备份恢复并运行自动测试。 时长：一天	DevOps 工程师、OpenText 主管、QA 测试员
验证监控和分析。	验证监控和分析是否正常运行。 时长：两小时	DevOps 工程师，OpenText 主管
关闭旧环境并请求关闭服务器。	时长：三天	DevOps 工程师，OpenText 主管

相关资源

- 客户管理型策略
- 在您的 Amazon Web Services account 中创建 IAM 用户
- IAM 用户组
- 将策略附加到 IAM 用户组
- 监控使用率和成本
- 管理 Amazon Web Services account
- Amazon 的 IAM 角色 EC2
- Amazon Web Services Support 的访问权限
- 使用 create-trail
- 为启用日志文件完整性验证 CloudTrail
- 适用于 Amazon S3 存储桶政策 CloudTrail

- 将事件发送到 CloudWatch 日志
- 通过 AWS CLI 设置 AWS Config
- 为 S3 存储桶和对象启用 CloudTrail 事件记录
- 配置 AWS KMS 密钥策略 CloudTrail
- 使用 AWS KMS 托管密钥 (SSE-KMS) 加密 CloudTrail 日志文件
- 如何启用和禁用自动密钥轮换
- 如何在使用 Amazon Web Services account 的根访问密钥时接收通知
- 创建流日志
- 您的 VPC 的安全组
- 为 VPC 对等连接更新路由表

将 Oracle CLOB 值迁移到 AWS 上 PostgreSQL 中的单独的行

创建者：Sai Krishna Namburu (AWS) 和 Sindhusha Paturu (AWS)

摘要

此模式描述了如何在 Amazon Aurora PostgreSQL-Compatible Edition 和 Amazon Relational Database Service（Amazon RDS）for PostgreSQL 中将 Oracle 字符大型对象（CLOB）值拆分为单独的行。PostgreSQL 不支持 CLOB 数据类型。

在 Oracle 源数据库中标识具有间隔分区的表，并捕获表名、分区类型、分区间隔和其他元数据并将其加载到目标数据库中。您可以使用 AWS Database Migration Service（AWS DMS）将大小小于 1 GB 的 CLOB 数据作为文本加载到目标表中，也可以用 CSV 格式导出数据，将其加载到 Amazon Simple Storage Service（Amazon S3）存储桶中，然后将其迁移到目标 PostgreSQL 数据库。

迁移后，您可以使用此模式提供的自定义 PostgreSQL 代码，根据换行符标识符（`CHR(10)`）将 CLOB 数据拆分为单独的行，然后填充目标表。

先决条件和限制

先决条件

- Oracle 数据库表，具有间隔分区和具有 CLOB 数据类型的记录。
- Aurora PostgreSQL-Compatible 或 Amazon RDS for PostgreSQL 数据库，其表结构与源表相似（列和数据类型相同）。

限制

- CLOB 值不能超过 1 GB。
- 目标表中的每一行都必须有一个换行符字符标识符。

产品版本

- Oracle 12c
- Aurora PostgreSQL 11.6

架构

下图显示了包含 CLOB 数据的 Oracle 源表，以及 Aurora PostgreSQL-Compatible 版本 11.6 中的等效 PostgreSQL 表。

工具

Amazon Web Services

- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩展 PostgreSQL 部署。
- Amazon Relational Database Service（Amazon RDS）for PostgreSQL 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展 PostgreSQL 关系数据库。
- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到 Amazon Web Services Cloud，或者在云和本地设置的组合之间迁移。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

其他工具

您可以使用以下客户端工具连接、访问和管理 Aurora PostgreSQL-Compatible 数据库和 Amazon RDS for PostgreSQL 数据库。（此模式中不使用这些工具。）

- pgAdmin 是一种适用于 PostgreSQL 的开源管理工具。它提供了一个图形界面，可帮助您创建、维护和使用数据库对象。
- DBeaver是面向开发人员和数据库管理员的开源数据库工具。您可以使用该工具来操作、监控、分析、管理和迁移数据。

最佳实践

有关将数据库从 Oracle 迁移到 PostgreSQL 的最佳实践，请参阅 AWS Blog 文章将 Oracle 数据库迁移至 Amazon RDS PostgreSQL 或 Amazon Aurora PostgreSQL 的最佳实践：迁移过程和基础设施注意事项。

有关配置 AWS DMS 任务以迁移大型二进制对象的最佳实践，请参阅 AWS DMS 文档中的迁移大型二进制对象 (LOBs)。

操作说明

识别 CLOB 数据

Task	描述	所需技能
分析 CLOB 数据。	在 Oracle 源数据库中，分析 CLOB 数据以查看其是否包含列标题，这样您就可以确定将数据加载到目标表中的方法。 要分析输入数据，请使用以下查询。 `SELECT * FROM clobdata_or;`	开发人员
将 CLOB 数据加载到目标数据库。	将包含 CLOB 数据的表迁移到 Aurora 或 Amazon RDS 目标数据库中的临时（暂存）表。您可以使用 AWS DMS，也可以将数据作为 CSV 文件上传到 Amazon S3 存储桶。 有关使用 AWS DMS 完成此任务的信息，请参阅 AWS DMS 文档中的使用 Oracle 数据库作为源以及使用 PostgreSQL 作为目标。 有关使用 Amazon S3 完成此任务的信息，请参阅 AWS DMS 文档中的使用Amazon S3作为目标。	迁移工程师、数据库管理员
验证目标 PostgreSQL 表。	在目标数据库中使用以下查询，根据源数据验证目标数据（包括标头）。	开发人员

Task	描述	所需技能

	`SELECT * FROM clobdata_pg;`
	`SELECT * FROM clobdatatarget;`
	
	将结果与源数据库的查询结果（从第一步开始）进行比较。	

将 CLOB 数据拆分为单独的行。	运行其他信息部分中提供的自定义 PostgreSQL 代码，拆分 CLOB 数据并将其插入目标 PostgreSQL 表中的单独行中。	开发人员

验证数据。

Task	描述	所需技能

验证目标表中的数据。	使用以下查询验证插入到目标表中的数据。
	
	`SELECT * FROM clobdata_pg;`
	`SELECT * FROM clobdatatarget;`
		开发人员

相关资源

- CLOB 数据类型（Oracle 文档）
- 数据类型（PostgreSQL 文档）

其他信息

用于拆分 CLOB 数据的 PostgreSQL 函数

```
do
$$
declare
totalstr varchar;
str1 varchar;
str2 varchar;
pos1 integer := 1;
pos2 integer ;
len integer;


begin
        select rawdata||chr(10) into totalstr from clobdata_pg;
        len :=  length(totalstr) ;
        raise notice 'Total length : %',len;
        raise notice 'totalstr : %',totalstr;
        raise notice 'Before while loop';


        while pos1 < len  loop


                  select position (chr(10) in totalstr) into pos2;
                 raise notice '1st position of new line : %',pos2;


                str1 := substring (totalstr,pos1,pos2-1);
                raise notice 'str1 : %',str1;


                   insert into clobdatatarget(data) values (str1);
                   totalstr := substring(totalstr,pos2+1,len);
                   raise notice 'new totalstr :%',totalstr;
                len :=  length(totalstr) ;


        end loop;
end
$$
LANGUAGE 'plpgsql' ;
```

输入和输出示例

在迁移数据之前，您可以使用以下示例试用 PostgreSQL 代码。

创建一个包含三行输入的 Oracle 数据库。

```
CREATE TABLE clobdata_or (
id INTEGER GENERATED ALWAYS AS IDENTITY,
rawdata clob  );


insert into clobdata_or(rawdata) values (to_clob('test line 1') || chr(10) ||
 to_clob('test line 2') || chr(10) || to_clob('test line 3') || chr(10));
COMMIT;


SELECT * FROM clobdata_or;
```

这会显示以下输出。

id	rawdata
1	测试行 1 测试行 2 测试行 3

将源数据加载到 PostgreSQL 暂存表（`clobdata_pg`）中进行处理。

```
SELECT * FROM clobdata_pg;

CREATE TEMP TABLE clobdatatarget (id1 SERIAL,data VARCHAR );

<Run the code in the additional information section.>

SELECT * FROM clobdatatarget;
```

这会显示以下输出。

id1	数据
1	测试行 1
2	测试行 2

3	测试行 3

使用通过数据库链接直接导入 Oracle Data Pump 将本地 Oracle 数据库迁移到 Amazon RDS for Oracle

由 Rizwan Wangde (AWS) 编写

摘要

许多模式包括使用 Oracle Data Pump 将本地 Oracle 数据库迁移到适用于 Oracle 的亚马逊关系数据库服务 (Amazon RDS)，Oracle 数据泵是一种原生 Oracle 实用程序，是迁移大型 Oracle 工作负载的首选方式。这些模式通常涉及将应用程序架构或表导出到转储文件中，将转储文件传输到 Amazon RDS for Oracle 上的数据库目录，然后从转储文件中导入应用程序架构和数据。

使用这种方法，迁移可能需要更长时间，具体取决于数据的大小以及将转储文件传输到 Amazon RDS 实例所需时间。此外，转储文件存储在 Amazon RDS 实例的 Amazon Elastic Block Store (Amazon EBS) 卷，该卷必须足够大，可以存放数据库和转储文件。导入后删除转储文件后，空余空间将无法恢复，因此您需要继续为未使用的空间付费。

这种模式通过数据库链接使用 Oracle Data Pump API (DBMS_DATAPUMP) 在 Amazon RDS 实例执行直接导入，从而缓解了这些问题。该模式在源数据库和目标数据库之间启动同步导出和导入管道。这种模式不需要为转储文件调整 EBS 卷大小，因为该卷上不会创建或存储任何转储文件。这种方法可以节省每月未使用磁盘空间的成本。

先决条件和限制

先决条件

- 一个有效的亚马逊 Web Services (AWS) 账户。
- 一种虚拟私有云 (VPC)，配置了跨越至少两个可用区的私有子网，用于为 Amazon RDS 实例提供网络基础设施。
- 位于本地数据中心的 Oracle 数据库，或者在亚马逊弹性计算云 (Amazon EC2) 上自行管理。
- 单个可用区中的现有 Amazon RDS for Oracle 实例。使用单个可用区可提高迁移期间的写入性能。可以在割接前 24-48 小时启用多可用区部署。

  该解决方案还可以使用适用于 Oracle 的 Amazon RDS 定制版作为目标。
- AWS Direct Connect （建议用于大型数据库）。
- 本地网络连接和防火墙规则配置为允许从 Amazon RDS 实例至本地 Oracle 数据库的入站连接。

限制

- 截至2022年12月，Amazon RDS for Oracle 的数据库大小限制为 64 太字节 (TiB)。
- 适用于 Oracle 的 Amazon RDS 数据库实例上单个文件的最大大小为 16 TiB。了解这一点很重要，因为您可能需要将表分布在多个表空间中。

产品版本

- 源数据库：Oracle 数据库 10g 版本 1 及以上版本。
- 目标数据库：有关 Amazon RDS 上支持的版本和版本的最新列表，请参阅 Amazon RDS 文档中的 Amazon RDS for Oracle。

架构

源技术堆栈

- 在本地或云自托管式 Oracle 数据库

目标技术堆栈

- 适用于 Oracle 的 Amazon RDS 或适用于 Oracle 的 Amazon RDS 定制

目标架构

下图显示了在单可用区环境中从本地 Oracle 数据库迁移至 Amazon RDS for Oracle 的架构。箭头方向描绘了架构数据流。该图没有显示哪个组件正在启动连接。

1. Amazon RDS for Oracle 实例连接至本地源 Oracle 数据库，通过数据库链接执行满负荷迁移。
2. AWS Database Migration Service (AWS DMS) 连接到本地源 Oracle 数据库，使用变更数据捕获 (CDC) 执行持续复制。
3. CDC 更改将应用于 Amazon RDS for Oracle 数据库。

工具

Amazon Web Services

- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到云和本地设置的组合中，AWS Cloud 或者迁移到云端和本地设置的组合之间。此模式使用 CDC 和仅复制数据更改设置。

- AWS Direct Connect通过标准的以太网光纤电缆将您的内部网络链接到某个 AWS Direct Connect 位置。通过此连接，您可以直接创建面向公众的虚拟接口， AWS 服务 同时绕过网络路径中的互联网服务提供商。
- Amazon Relational Database Servic e 可帮助您在 AWS 云中设置、操作和扩展 Oracle 关系数据库。

其他工具

- Oracle 数据泵可帮助您将数据和元数据从一个数据库高速移动至另一个数据库。
- Oracle Instant Client 或 SQL Developer 等客户端工具用于连接数据库并在数据库上运行 SQL 查询。

最佳实践

尽管在本地网络和之间 AWS Direct Connect 使用专用的私有网络连接 AWS，但要为传输中的数据提供更高的安全性和数据加密，请考虑以下选项：

- 使用虚拟专用网络 (VPN) AWS Site-to-Site VPN 或从本地网络到网络的 IPsec VPN 连接 AWS
- 在本地 Oracle 数据库上配置的 Oracle 数据库本机网络加密
- 使用 TLS 的加密

操作说明

准备本地源 Oracle 数据库

Task	描述	所需技能
设置从目标数据库到源数据库的网络连接。	配置本地网络和防火墙以允许从目标 Amazon RDS 实例到本地源 Oracle 数据库的传入连接。	网络管理员、安全工程师
创建具有相应权限的数据库用户。	在本地源 Oracle 数据库中创建具有使用 Oracle 数据泵在源和目标之间迁移数据的权限的数据库用户：	数据库管理员

Task	描述	所需技能

	`GRANT CONNECT to <migration_user>;`
	`GRANT DATAPUMP_EXP_FULL_DATABASE to <migration_user>;`
	`GRANT SELECT ANY TABLE to <migration_user>;`	

为 AWS DMS CDC 迁移准备本地源数据库。	（可选）在 Oracle 数据泵满负荷运行完成后，准备本地源 Oracle 数据库以进行 AWS DMS CDC 迁移：
	1. 配置在 Oracle 数据泵迁移期间管理 FLASHBACK 所需的其他权限：
	`GRANT FLASHBACK ANY TABLE to <migration_user>;`
	`GRANT FLASHBACK ARCHIVE ADMINISTER to <migration_user>;`
	2. 要配置自行管理的 Oracle 源所需的用户帐户权限 AWS DMS，请参阅AWS DMS 文档。
	3. 要使用 CDC 准备 Oracle 自行管理的源数据库 AWS DMS，请参阅AWS DMS 文档。
		数据库管理员

安装和配置 SQL Developer。	安装和配置 SQL Developer 以连接源数据库和目标数据库并运行 SQL 查询。	数据库管理员，迁移工程师

Task	描述	所需技能
生成脚本来创建表空间。	使用以下示例 SQL 查询在源数据库上生成脚本： `SELECT 'CREATE TABLESPACE ' tablespace_name ' DATAFILE SIZE 1G AUTOEXTEND ON MAXSIZE UNLIMITED;' from dba_tablespaces where tablespace_name not in ('SYSTEM', 'SYSAUX','TEMP','UNDOTBS1') order by 1;` 脚本将应用至目标数据库。	数据库管理员
生成用于创建用户、配置文件、角色和权限脚本。	要生成用于创建数据库用户、配置文件、角色和权限的脚本，请使用 Oracle Support 文档 如何使用 dbms_metadata.get_ddl 提取用户的 DDL，包括权限和角色（文档 ID 2739952.1）(需要 Oracle 账户)。 脚本将应用至目标数据库。	数据库管理员

创建目标 Amazon RDS for Oracle 实例。

Task	描述	所需技能
创建到源数据库的数据库链接并验证连接性。	要创建指向本地源数据库的数据库链接，可以使用以下示例命令：	数据库管理员

Task	描述	所需技能

	`CREATE DATABASE LINK link2src CONNECT TO <migration_user_account> IDENTIFIED BY <password> USING '(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=<dns or ip address of remote db>) (PORT=<listener port>))(CONNECT_DATA=(SID=<remote SID>)))';`
	
	要验证连接，请运行以下 SQL 命令：
	
	`select * from dual@link2src;`
	
	如果响应为 X，则连接成功。	

运行脚本，以准备目标实例。	运行之前生成的脚本，以准备目标 Amazon RDS for Oracle 实例： 1. Tablespaces 2. 配置文件 3. 角色 这有助于确保 Oracle Data Pump 迁移可以创建模式及其对象。	数据库管理员，迁移工程师

通过数据库链接使用 Oracle Data Pump 导入来执行满负荷迁移

Task	描述	所需技能
迁移所需架构。	要将所需架构从源本地数据库迁移至目标 Amazon RDS 实例，请使用其他信息部分中的代码： • 要迁移单个架构，请运行其他信息部分中的代码 1。 • 要迁移多个架构，请运行其他信息部分中的代码 2。 要调整迁移性能，您可以通过运行以下命令来调整 parallel 进程的数量： `DBMS_DATAPUMP.SET_PARALLEL (handle => v_hdnl, degree => 4);`	数据库管理员
收集架构统计信息以提高性能。	收集架构统计信息命令返回为数据库对象收集的 Oracle 查询优化器统计信息。通过使用这些信息，优化器可以为针对以下对象的任何查询选择最佳执行计划： `EXECUTE DBMS_STATS.GATHER_SCHEMA_STATS(ownname => '<schema_name>');`	数据库管理员

使用 Oracle 数据泵执行满载迁移和 CDC 复制 AWS DMS

Task	描述	所需技能
捕获源本地 Oracle 数据库上的 SCN。	在源本地 Oracle 数据库上捕获系统更改号 (SCN)。您将使用 SCN 进行满载导入，并用作 CDC 复制起点。 要在源数据库上生成当前 SCN，请运行以下 SQL 语句： `SELECT current_scn FROM V$DATABASE;`	数据库管理员
执行架构的满负荷迁移。	要将所需架构 (FULL LOAD) 从源本地数据库迁移至目标 Amazon RDS 实例，请执行以下操作： • 要迁移单个架构，请运行其他信息部分中的代码 3。 • 要迁移多个架构，请运行其他信息部分中的代码 4。 在代码中，<CURRENT_SCN_VALUE_IN_SOURCE_DATABASE> 用您从源数据库捕获的 SCN 替换： `DBMS_DATAPUMP.SET_PARAMETER (handle => v_hdnl, name => 'FLASHBACK_SCN', value => <CURRENT_SCN_VALUE_IN_SOURCE_DATABASE>);`	数据库管理员

Task	描述	所需技能
	要调整迁移性能，可以调整 parallel 进程的数量： `DBMS_DATAPUMP.SET_ PARALLEL (handle => v_hdnl, degree => 4);`	
在迁移架构下禁用触发器。	在开始 AWS DMS 仅限 CDC 的任务之前，请禁用已迁移TRIGGERS架构下的。	数据库管理员
收集架构统计信息以提高性能。	“收集架构统计信息” 命令返回为数据库对象收集的 Oracle 查询优化器统计信息： `EXECUTE DBMS_STAT S.GATHER_SCHEMA_ST ATS(ownname => '<schema_name>');` 通过使用此信息，优化器可以为针对这些对象的任何查询选择最佳执行计划。	数据库管理员
用于 AWS DMS 执行从源到目标的持续复制。	用于 AWS DMS 执行从源 Oracle 数据库到目标 Amazon RDS for Oracle 实例的持续复制。 有关更多信息，请参阅使用创建用于持续复制的任务，AWS DMS以及中的博客文章《如何使用本地 CDC 支持》 AWS DMS。	数据库管理员，迁移工程师

割接 Amazon RDS for Oracle

Task	描述	所需技能
在割接前 48 小时在实例上启用多可用区以在切换之前启用。	如果这是生产实例，我们建议在 Amazon RDS 实例上启用多可用区部署，以提供高可用性 (HA) 以及灾难恢复 (DR) 的优势。	数据库管理员，迁移工程师
停止 AWS DMS 仅限CDC的任务（如果CDC已开启）。	1. 确保 AWS DMS 任务的 Amazon CloudWatch 指标的源延迟和目标延迟显示 0 秒。 2. 停止 AWS DMS 仅限 CDC 的任务。	数据库管理员
启用触发器。	启用您在创建 CDC 任务之前禁用的。`TRIGGERS`	数据库管理员

相关的资源

AWS

- 使用为 CDC 准备 Oracle 自行管理的源数据库 AWS DMS
- 使用为正在进行的复制创建任务 AWS DMS
- 多可用区部署，可实现高可用性
- 如何在 AWS DMS（博客文章）中使用本地 CDC 支持

Oracle 文档

- DBMS_DATAPUMP

其他信息

代码 1：仅限满载迁移，单一应用程序架构

```
DECLARE
    v_hdnl NUMBER;
BEGIN
    v_hdnl := DBMS_DATAPUMP.OPEN(operation => 'IMPORT', job_mode  => 'SCHEMA',
 remote_link => '<DB LINK Name to Source Database>', job_name  => null);
    DBMS_DATAPUMP.ADD_FILE( handle => v_hdnl, filename  => 'import_01.log',  directory
 => 'DATA_PUMP_DIR', filetype  => dbms_datapump.ku$_file_type_log_file);
    DBMS_DATAPUMP.METADATA_FILTER(v_hdnl,'SCHEMA_EXPR','IN (''<schema_name>'')');  --
 To migrate one selected schema
    DBMS_DATAPUMP.METADATA_FILTER (hdnl, 'EXCLUDE_PATH_EXPR','IN (''STATISTICS'')'); --
 To prevent gathering Statistics during the import
    DBMS_DATAPUMP.SET_PARALLEL (handle => v_hdnl, degree => 4);  -- Number of parallel
 processes performing export and import
    DBMS_DATAPUMP.START_JOB(v_hdnl);
END;
/
```

代码 2：仅限满载迁移，多个应用程序架构

```
DECLARE
    v_hdnl NUMBER;
BEGIN
    v_hdnl := DBMS_DATAPUMP.OPEN(operation => 'IMPORT', job_mode  => 'SCHEMA',
 remote_link => '<DB LINK Name to Source Database>', job_name  => null);
    DBMS_DATAPUMP.ADD_FILE( handle => v_hdnl, filename  => 'import_01.log',  directory
 => 'DATA_PUMP_DIR', filetype  => dbms_datapump.ku$_file_type_log_file);
    DBMS_DATAPUMP.METADATA_FILTER (v_hdnl, 'SCHEMA_LIST',
 '''<SCHEMA_1>'',''<SCHEMA_2>'', ''<SCHEMA_3>'''); -- To migrate multiple schemas
    DBMS_DATAPUMP.METADATA_FILTER (v_hdnl, 'EXCLUDE_PATH_EXPR','IN (''STATISTICS'')');
 -- To prevent gathering Statistics during the import
    DBMS_DATAPUMP.SET_PARALLEL (handle => v_hdnl, degree => 4);  -- Number of parallel
 processes performing export and import
    DBMS_DATAPUMP.START_JOB(v_hdnl);
END;
/
```

代码 3：仅限 CDC 的任务之前的满载迁移，单个应用程序架构

```
DECLARE
    v_hdnl NUMBER;
BEGIN
```

```
    v_hdnl := DBMS_DATAPUMP.OPEN(operation => 'IMPORT', job_mode  => 'SCHEMA',
 remote_link => '<DB LINK Name to Source Database>', job_name  => null);
    DBMS_DATAPUMP.ADD_FILE( handle => v_hdnl, filename  => 'import_01.log',  directory
 => 'DATA_PUMP_DIR', filetype  => dbms_datapump.ku$_file_type_log_file);
    DBMS_DATAPUMP.METADATA_FILTER(v_hdnl,'SCHEMA_EXPR','IN (''<schema_name>'')');  --
 To migrate one selected schema
    DBMS_DATAPUMP.METADATA_FILTER (v_hdnl, 'EXCLUDE_PATH_EXPR','IN (''STATISTICS'')');
 -- To prevent gathering Statistics during the import
    DBMS_DATAPUMP.SET_PARAMETER (handle => v_hdnl, name => 'FLASHBACK_SCN', value =>
 <CURRENT_SCN_VALUE_IN_SOURCE_DATABASE>); -- SCN required for AWS DMS CDC only task.
    DBMS_DATAPUMP.SET_PARALLEL (handle => v_hdnl, degree => 4);  -- Number of parallel
 processes performing export and import
    DBMS_DATAPUMP.START_JOB(v_hdnl);
END;
/
```

代码 4：仅限 CDC 任务之前的满载迁移，多个应用程序架构

```
DECLARE
    v_hdnl NUMBER;
BEGIN
    v_hdnl := DBMS_DATAPUMP.OPEN (operation => 'IMPORT', job_mode  => 'SCHEMA',
 remote_link => '<DB LINK Name to Source Database>', job_name  => null);
    DBMS_DATAPUMP.ADD_FILE (handle => v_hdnl, filename  => 'import_01.log',  directory
 => 'DATA_PUMP_DIR', filetype  => dbms_datapump.ku$_file_type_log_file);
    DBMS_DATAPUMP.METADATA_FILTER (v_hdnl, 'SCHEMA_LIST',
 '''<SCHEMA_1>'',''<SCHEMA_2>'', ''<SCHEMA_3>'''); -- To migrate multiple schemas
    DBMS_DATAPUMP.METADATA_FILTER (v_hdnl, 'EXCLUDE_PATH_EXPR','IN (''STATISTICS'')');
 -- To prevent gathering Statistics during the import
    DBMS_DATAPUMP.SET_PARAMETER (handle => v_hdnl, name => 'FLASHBACK_SCN', value =>
 <CURRENT_SCN_VALUE_IN_SOURCE_DATABASE>); -- SCN required for AWS DMS CDC only task.
    DBMS_DATAPUMP.SET_PARALLEL (handle => v_hdnl, degree => 4);  -- Number of parallel
 processes performing export and import
    DBMS_DATAPUMP.START_JOB(v_hdnl);
END;
/
```

混合迁移方法可更好地发挥作用的场景

在极少数情况下，源数据库包含具有数百万行和非常大的 LOBSEGMENT 列的表，此模式会减慢迁移速度。Oracle 通过网络链路逐一迁 LOBSEGMENTs 移。它从源表中提取单行（以及 LOB 列数据），然后将该行插入目标表，重复该过程，直到所有行都迁移完毕。通过数据库链接进行的 Oracle Data Pump 不支持批量加载或直接路径加载机制 LOBSEGMENTs。

在这种情况下，我们建议采取以下：

- 通过添加以下元数据筛选器，在 Oracle Data Pump 迁移期间跳过已识别的表：

```
dbms_datapump.metadata_filter(handle =>h1, name=>'NAME_EXPR', value => 'NOT IN
 (''TABLE_1'',''TABLE_2'')');
```

- 使用 AWS DMS 任务（满载迁移，必要时可复制 CDC）迁移已识别的表。 AWS DMS 将从源 Oracle 数据库中提取多行，然后将它们成批插入目标 Amazon RDS 实例，这样可以提高性能。

将 Oracle 电子商务套件迁移到 Amazon RDS Custom

创建者：Simon Cunningham（AWS）、Jaydeep Nandy（AWS）、Nitin Saxena（AWS）和 Vishnu Vinnakota（AWS）

摘要

Oracle 电子商务套件是一种企业资源规划 (ERP) 解决方案，用于自动化企业范围内的流程，例如财务、人力资源、供应链和制造。它具有三层架构：客户端、应用程序和数据库。以前，您必须在自行管理的亚马逊弹性计算云 (Amazon EC2) 实例上运行 Oracle 电子商务套件数据库，但现在您可以从亚马逊关系数据库服务 (Amazon RDS) 定制版中受益。

适用于 Oracle 的 Amazon RDS Custom 是一项托管式数据库服务，适用于需要访问底层操作系统和数据库环境的旧版、自定义和打包应用程序。它可以自动执行数据库管理任务和操作，并使您能够作为数据库管理员访问和自定义数据库环境和操作系统。当您将 Oracle 数据库迁移至 Amazon RDS Custom 时，Amazon Web Services 会处理诸如备份任务和确保高可用性之类的繁重工作，同时您可以专注于维护 Oracle 电子商务套件的应用程序和功能。有关迁移时需要考虑的关键因素，请参阅 AWS Prescriptive Guidance 中的 Oracle 数据库迁移策略。

此模式侧重于使用 Oracle Recovery Manager (RMAN) 备份以及在 EC2 实例和 A EC2 mazon RDS Custom 之间使用亚马逊弹性文件系统 (Amazon EFS) 共享文件系统 (Amazon EFS) 共享文件系统将亚马逊上的独立 Oracle 数据库迁移到 Amazon RDS Custom 的步骤。该模式使用 RMAN 完整备份（有时也称为 0 级备份）。 为简单起见，它使用冷备份，即关闭应用程序，装入数据库而不打开。（您也可以使用 Oracle Data Guard 或 RMAN 复制进行备份。但是，此模式不包括这些选项。）

有关在 AWS 上架构 Oracle 电子商务套件以实现高可用性和灾难恢复的信息，请参阅模式使用有效备用数据库在 Amazon RDS Custom 上为 Oracle 电子商务套件设置 HA/DR 架构。

> **Note**
>
> 此模式提供指向 Oracle 支持说明的链接。您需要 Oracle Support 账户才能访问这些文档。

先决条件和限制

先决条件

- EC2 使用 Oracle Linux 7 或红帽企业 Linux (RHEL) 版本 7.x 在亚马逊上运行的 Oracle 版本 12.1.0.2 或 19c（最低 19.3）源数据库。此模式假设源数据库名称为 `VIS`，Oracle 19c 的其他容器数据库名称为 `VISCDB`，但您可以使用其他名称。

> **Note**
>
> 您也可以将此模式用于本地 Oracle 源数据库，前提是您在本地网络和亚马逊虚拟私有云 (Amazon VPC) 之间有适当的网络连接。

- Oracle 电子商务套件 12.2.x 版应用程序（视觉实例）。此过程已在 12.2.11 版本上进行了测试。
- 单个 Oracle 电子商务套件应用程序层。但是，您可以调整此模式以使用多个应用程序层。
- 对于 Oracle 12.1.0.2，配有至少 16 GB 交换空间的 Amazon RDS Custom。否则，12c 示例 CD 将显示一条警告。（如本文档后面所述，Oracle 19c 不需要示例 CD。）

开始迁移前，请完成以下步骤：

1. 在 Amazon RDS 控制台上，使用数据库名称 `VIS`（或源数据库名称）创建适用于 Oracle 数据库实例的 Amazon RDS Custom。有关说明，请参阅 AWS 文档中的使用 Amazon RDS Custom 和博客文章Amazon RDS Custom for Oracle – 数据库环境中的新控制功能。这样可以确保将数据库名称设置为与源数据库相同的名称。（如果留空，则 EC2 实例和数据库名称将设置为`ORCL`。） 确保至少使用已应用于源代码的补丁来创建自定义引擎版本（CEV）。有关更多信息，请参阅 Amazon RDS 文档中的准备创建 CEV。

   Oracle 19c 注意事项：目前，对于 Oracle 19c，可以自定义 Amazon RDS 容器数据库的名称。默认为 `RDSCDB`。请务必使用与源实例相同的系统 ID (SID) 创建 RDS 自定义 Oracle EC2 实例。例如，在这种模式中，假定 Oracle 19c SID 是源实例上的 `VISCDB`。因此，Amazon RDS Custom 上的目标 Oracle 19c SID 也应该是 `VISCDB`。
2. 为 Amazon RDS 自定义数据库实例配置足够的存储空间、vCPU 和内存，以匹配亚马逊 EC2 源数据库。 为此，您可以根据 vCPU 和内存来匹配 Amazon EC2 实例类型。
3. 创建 Amazon EFS 文件系统并将其挂载到亚马逊 EC2 和亚马逊 RDS 自定义实例上。 有关说明，请参阅博客文章将适用于 Oracle 的 Amazon RDS Custom 与 Amazon EFS 集成。此模式假设您已在源 Amazon EC2 和目标 Amazon RDS 自定义数据库实例/`RMAN`上安装了 Amazon EFS 卷，并且源和目标之间可以进行网络连接。您也可以通过使用 Amazon FSx 或任何共享云端硬盘来使用相同的方法。

假设

此模式假设应用程序和数据库使用的是逻辑主机名，从而减少了迁移步骤的数量。您可以调整这些步骤以使用物理主机名，但是逻辑主机名可以降低迁移过程的复杂性。有关使用逻辑主机名的优势的信息，请参阅以下支持说明：

- 对于 12c，Oracle Support Note 2246690.1
- 对于 19c，Oracle Support Note 2617788.1

这种模式不包括 Oracle 12c 到 19c 的升级场景，而是侧重于将亚马逊上运行的相同版本的 Oracle 数据库迁移到适用于 Oracle 的 A EC2 mazon RDS Custom。

适用于 Oracle 的 Amazon RDS Custom 支持 Oracle Home 自定义。（Oracle Home 存储 Oracle 二进制文件。）您可以将 `/rdsdbbin/oracle` 的默认路径更改为您指定的路径，例如 `/d01/oracle/VIS/19c`。为简单起见，此模式中的指令采用默认路径 `/rdsdbbin/oracle`。

限制

此模式不支持以下功能和配置：

- 将数据库 `ARCHIVE_LAG_TARGET` 参数设置为 60–7200 范围之外的值
- 禁用数据库实例日志模式（`NOARCHIVELOG`）
- 关闭 EC2 实例的`EBS-optimized`属性
- 修改附加到实例的原始亚马逊弹性区块存储 (Amazon EBS) 卷 EC2
- 添加新的 EBS 卷或将卷类型从 `gp2` 更改为 `gp3`
- 对 TNS ifile 的支持
- 更改 `control_file` 位置和名称（必须是 `/rdsdbdata/db/VISCDB_A/controlfile/control-01.ctl`，其中 `VISCDB` 是 CDB 名称）

有关这些配置和其他不支持的配置的更多信息，请参阅 Amazon RDS 文档中的修复不支持的配置。

产品版本

有关 Amazon RDS Custom 支持的 Oracle Database 版本和实例类型，请参阅 Amazon RDS Custom for Oracle 的要求和限制。

架构

以下架构图表示在 AWS 的单个可用区中运行的 Oracle 电子商务套件系统。应用程序层可通过应用程序负载均衡器访问，应用程序和数据库均位于私有子网中，Amazon RDS Custom 和 Amazon EC2 数据库层使用 Amazon EFS 共享文件系统来存储和访问 RMAN 备份文件。

工具

Amazon Web Services

- 适用于 Oracle 的 Amazon RDS Custom 是一项托管式数据库服务，适用于需要访问底层操作系统和数据库环境的旧版、自定义和打包应用程序。它可以自动执行数据库管理任务和操作，并使您能够作为数据库管理员访问和自定义数据库环境和操作系统。
- Amazon Elastic File System（Amazon EFS）是一个简单、无服务器的弹性文件系统，无需管理或预调配即可添加和删除文件。此模式使用 Amazon EFS 共享文件系统来存储和访问 RMAN 备份文件。
- AWS Secrets Manager 是一项 AWS 托管服务，可让您轻松轮换、管理和检索数据库凭证、API 密钥和其他机密信息。创建数据库后，Amazon RDS Custom 会将密钥对和数据库用户凭证存储在 Secrets Manager 中。在这种模式中，您可以从 Secrets Manager 检索数据库用户密码来创建 `RDSADMIN` 和 `ADMIN` 用户以及更改 Sys 和系统密码。

其他工具

- RMAN 是一种为 Oracle 数据库提供备份和恢复支持的工具。此模式使用 RMAN 对亚马逊上的源 Oracle 数据库执行冷备份，该数据库在 Amazon EC2 RDS Custom 上还原。

最佳实践

- 使用逻辑主机名。这大大减少了必须运行的克隆后脚本的数量。有关更多信息，请参阅 Oracle Support Note 2246690.1。
- 默认情况下，Amazon RDS Custom 使用 Oracle 自动内存管理 (AMM)。 如果您想使用 Hugemem 内核，可以将 Amazon RDS Custom 配置为改用自动共享内存管理 (ASMM)。
- 默认情况下启用 `memory_max_target` 参数。框架在后台使用此参数来创建只读副本。
- 启用 Oracle 闪回数据库。此功能在失效转移（不是切换）测试场景中非常有用，可以恢复备用状态。
- 对于数据库初始化参数，请自定义 Amazon RDS Custom 数据库实例为 Oracle 电子商务套件提供的标准 PFILE，而不是使用 Oracle 源数据库中的 SPFILE。这是因为在 Amazon RDS Custom 中创建只读副本时，空格和评论会导致问题。有关数据库初始化参数的更多信息，请参阅 Oracle Support Note 396009.1。

在接下来的操作说明部分中，我们为 Oracle 12.1.0.2 和 19c 提供了单独的说明，其中的细节有所不同。

操作说明

关闭源应用程序

Task	描述	所需技能

关闭应用程序。	要关闭源应用程序，请使用以下命令：
	`$ su - applmgr`
	`$ cd $INST_TOP/admin/scripts`
	`$ ./adstpall.sh`
		数据库管理员

创建 .zip 文件。	在源应用程序层上创建 `appsutil.zip` 文件。稍后您将使用此文件来配置 Amazon RDS Custom 数据库节点。 `$ perl $AD_TOP/bin/admkappsutil.pl`	数据库管理员
将 .zip 文件复制到 Amazon EFS。	将 `appsutil.zip` 从 `$INST_TOP/admin/out` 复制到共享 Amazon EFS 卷（`/RMAN/appsutil`）。您可以使用安全复制（SCP）或其他传输机制手动传输文件。	数据库管理员

预克隆源数据库

Task	描述	所需技能
在 Amazon EC2 上预克隆数据库层。	以 Oracle 用户身份登录并运行： 	数据库管理员

Task	描述	所需技能

	`$ cd $ORACLE_HOME/appsutil/scripts/$CONTEXT_NAME`
	`$ perl adpreclone.pl dbTier`
	
	检查生成的日志文件以确认操作成功完成。	

将 appsutil.zip 复制到共享的 Amazon EFS 文件系统。	创建 tar 备份并将 `$ORACLE_HOME/appsutil` 复制到共享的 Amazon EFS 文件系统（例如，`/RMAN/appsutil`）：
	
	`$ cd $ORACLE_HOME`
	`$ tar cvf sourceappsutil.tar appsutil`
	`$ cp sourceappsutil.tar /RMAN/appsutil`
		数据库管理员

对源 Amazon EC2 数据库执行冷 RMAN 完整备份

Task	描述	所需技能
创建备份脚本。	对源数据库执行 RMAN 完整备份到共享的 Amazon EFS 文件系统。 为简单起见，此模式执行 RMAN 冷备份。但是，您可以修改这些步骤，使用 Oracle Data Guard 执行 RMAN 热备份，以缩短停机时间。	数据库管理员

Task	描述	所需技能

	1. 以挂载模式启动源 Amazon EC2 数据库：
	
	`$ sqlplus / as sysdba`
	`$ SQL> shutdown immediate`
	`$ SQL> startup mount`
	
	2. 创建 RMAN 备份脚本（根据 Oracle 版本使用以下示例之一，或运行现有的 RMAN 脚本），将数据库备份到您挂载的 Amazon EFS 文件系统（本示例中的 /RMAN）。
	
	对于 Oracle 12.1.0.2：
	
	`$ vi FullRMANColdBackup.sh`
	`#!/bin/bash`
	`. /home/oracle/.bash_profile`
	
	`export ORACLE_SID=VIS`
	`export ORACLE_HOME=/d01/oracle/VIS/12.1.0`
	`export DATE=$(date +%y-%m-%d_%H%M%S)`
	
	`rman target / log=/RMAN/VISDB_${DATE}.log <<`
	`EOF`
	`run`
	`{`
	`allocate channel ch1`
	`device type disk format`
	`'/RMAN/visdb_full_bkp_%u';`	

Task	描述	所需技能

	allocate channel ch2
	 device type disk format
	 '/RMAN/visdb_full_
	bkp_%u';
	crosscheck backup;
	delete noprompt
	 obsolete;
	BACKUP AS COMPRESSED
	 BACKUPSET DATABASE PLUS
	 ARCHIVELOG;
	backup archivelog all;
	release channel ch1;
	release channel ch2;
	}
	EOF
	
	对于 Oracle 19c：
	
	$ vi FullRMANColdBackup
	.sh
	#!/bin/bash
	. /home/oracle/.bash
	_profile
	
	export ORACLE_SI
	D=VISCDB
	export ORACLE_HOME=/
	d01/oracle/VIS/19c
	export DATE=$(date +
	%y-%m-%d_%H%M%S)
	
	rman target / log=/RMAN
	/VISDB_${DATE}.log <<
	 EOF
	run
	{
	allocate channel ch1
	 device type disk format
	 '/RMAN/visdb_full_
	bkp_%u';	

Task	描述	所需技能
	allocate channel ch2 device type disk format '/RMAN/visdb_full_ bkp_%u'; crosscheck backup; delete noprompt obsolete; BACKUP AS COMPRESSED BACKUPSET DATABASE PLUS ARCHIVELOG; backup archivelog all; backup current controlfile format '/ RMAN/cntrl.bak'; release channel ch1; release channel ch2; } EOF	
运行备份脚本。	更改权限，以 Oracle 用户身份登录，然后运行脚本： $ chmod 755 FullRMANC oldBackup.sh $ ./FullRMANColdBack up.sh	数据库管理员

Task	描述	所需技能

检查是否存在错误，并记下备份文件的名称。	查看 RMAN 日志文件中的错误。如果一切正常，请列出控制文件的备份。记下输出文件的名称。
	
	对于 Oracle 12.1.0.2：
	
	`RMAN> connect target /`
	
	`RMAN> list backup of controlfile;`
	
	`BS Key Type LV Size Device Type Elapsed Time Completion Time`
	`------- ---- -- ---------- --------- ---------- ---------------`
	`9 Full 1.11M DISK 00:00:04 23-APR-22`
	`BP Key: 9 Status: AVAILABLE Compressed: YES Tag: TAG20220423T121011`
	`Piece Name: /RMAN/visdb_full_bkp_100rlsbt`
	`Control File Included: Ckp SCN: 12204595396727 Ckp time: 23-APR-22`
	
	稍后，当您在 Amazon RDS Custom 上恢复数据库时，您将使用备份文件 `/RMAN/visdb_full_bkp_100rlsbt` 。
		数据库管理员

Task	描述	所需技能

	对于 Oracle 19c：
	
	`RMAN> connect target /`
	
	`RMAN> list backup of controlfile;`
	
	`BS Key Type LV Size Device Type Elapsed Time Completion Time`
	`------- ---- -- ---------- --------- -- ------------ ---------------`
	`38 Full 17.92M DISK 00:00:01 25-NOV-22`
	`BP Key: 38 Status: AVAILABLE Compressed: NO Tag: TAG20221125T095014`
	`Piece Name: /RMAN/cntrl.bak`
	`Control File Included: Ckp SCN: 122046201 88873 Ckp time: 23-NOV-22`
	
	稍后，当您在 Amazon RDS Custom 上恢复数据库时，您将使用备份文件 `/RMAN/cntrl.bak` 。	

配置目标 Amazon RDS Custom 数据库

Task	描述	所需技能

更改主机文件并设置主机名。	**Note**
	本节中的命令必须以 root 用户身份运行。
	
	1. 在 Amazon RDS Custom 数据库实例上编辑 `/etc/hosts` 文件。实现此目的的一种简单方法是从源 Amazon 数据库主机文件中复制 EC2 数据库和应用程序主机条目。
	
	`<IP-address> OEBS-app01.localdomain OEBS-app01 OEBS-app01log.localdomain OEBS-app01log`
	`<IP-address> OEBS-db01.localdomain OEBS-db01 OEBS-db01log.localdomain OEBS-db01log`
	
	其中 `<IP-address>` 是数据库节点 IP 地址，您应将其替换为 Amazon RDS Custom IP 地址。逻辑主机名后面附有 `*log`。
	
	2. 运行 `hostnamectl` 命令来更改数据库主机名：
		数据库管理员

Task	描述	所需技能

	`$ sudo hostnamectl set-hostname --static persistent-hostname`
	
	例如：
	
	`$ sudo hostnamectl set-hostname --static OEBS-db01log`
	
	有关更多信息，请参阅知识中心关于分配静态主机名的文章。
	
	3. 重启 Amazon RDS Custom 数据库实例。 不用担心会关闭数据库，因为您将在以后的步骤中将其删除。
	
	`$ reboot`
	
	4. 当 Amazon RDS Custom 数据库实例恢复后，登录并验证主机名是否已更改：
	
	`$ hostname`
	`oebs-db01`	

Task	描述	所需技能

安装 Oracle 电子商务套件软件。	在 Amazon RDS 自定义数据库实例上安装推荐 RPMs 到 Oracle 主目录的 Oracle 电子商务套件。有关详细信息，请参阅 Oracle Support Note #1330701.1。以下是部分列表。每个版本的 RPM 列表都会发生变化，因此请检查并确保已安装所有必 RPMs 需的版本。
	
	以根用户身份运行：
	
	`$ sudo yum -y update`
	`$ sudo yum install -y elfutils-libelf-devel*`
	`$ sudo yum install -y libXp-1.0.2-2.1*.i686`
	`$ sudo yum install -y libXp-1.0.2-2.1*`
	`$ sudo yum install -y compat-libstdc++-*`
	
	在继续下一步之前，请确认所有必需的补丁都已安装。
		数据库管理员

Task	描述	所需技能

安装 VNC 服务器。	**Note**
	对于 Oracle 19c，你可以省略此步骤，因为不再需要示例 CD；请参阅 Oracle Support Note 2782085.1。
	
	对于 Oracle 12.1.0.2：
	
	安装 VNC 服务器及其相关桌面软件包。这是在下一步中安装 12c 示例 CD 的必要条件。
	
	1. 以根用户身份运行：
	
	`$ sudo yum install -y tigervnc-server`
	`$ sudo yum install -y *kde*`
	`$ sudo yum install -y *xorg*`
	
	2. 为 `rdsdb` 用户启动 VNC 服务器，并设置 VNC 的密码：
	
	`$ su - rdsdb`
	`$ vncserver :1`
	`$ vncpassword`
		数据库管理员

Task	描述	所需技能

安装 12c 示例 CD。	**Note**
	对于 Oracle 19c，你可以省略此步骤，因为不再需要示例 CD；请参阅 Oracle Support Note 2782085.1。
	
	对于 Oracle 12.1.0.2：
	
	1. 从 https://edelivery.oracle.com/下载安装文件。 对于 Oracle 电子商务套件 12.2.11 – Oracle 数据库 12c 第 1 版（12.1.0.2），请查看 Linux x86-64 V100102-01.zip 的示例。
	
	2. 创建用于存储示例 CD 的目录：
	
	`$ mkdir /RMAN/12c examples`
	
	3. 使用您选择的传输机制（例如 SCP）将示例 CD .zip 文件复制到此目录：
	
	`V100102-01.zip`
	
	4. 将所有权更改为 `rdsdb`：
	
	`$ chown -R rdsdb:rdsdb /RMAN/12cexamples`
		数据库管理员

Task	描述	所需技能

	5. 以 rdsdb 用户身份解压缩文件：
	`$ unzip V10010201.zip`
	6. 从有权访问 VNC 客户端和 Amazon RDS Custom 的客户端进行连接。请确保打开必要的网络连接和防火墙端口，以允许 VNC 访问。例如，正在运行的 VNC 服务器`display :1`需要在与 Amazon RDS 自定义 EC2 主机关联的安全组上打开端口 5901。
	7. 更改为复制了示例 CD 的目录：
	`$ cd /RMAN/12cexamples/examples`
	8. 运行安装程序。 请务必验证 `oraInst.loc` 文件的位置。
	`./runInstaller -invPtrLoc /rdsdbbin/oracle.12.1.custom.r1.EE.1/oraInst.loc`
	9. 在安装示例 CD 的过程中使用以下参数：
	`Skip Software Update Downloads`	

Task	描述	所需技能
	`Select Oracle Home 12.1.0.2 (Oracle Base = /rdsdbbin) (Software Location = /rdsdbbin/oracle/12.1.custom.r1.EE.1)` 10. 安装程序包括五个带有提示的步骤。 按照这些步骤操作直到安装完成。	

删除起始数据库并创建用于存储数据库文件的目录

Task	描述	所需技能
暂停自动化模式。	在继续执行后续步骤之前，您必须暂停 Amazon RDS Custom 数据库实例的自动化模式，以确保自动化不会干扰 RMAN 活动。 使用以下 AWS 命令行界面（AWS CLI）命令暂停自动化。（首先确保您已配置 AWS CLI。） `aws rds modify-db-instance \ --db-instance-id entifier VIS \ --automation-mode all-paused \ --resume-full-automation-mode-minute 360 \ --region eu-west-1`	数据库管理员

Task	描述	所需技能
	当您指定暂停的持续时间时，请确保为 RMAN 恢复留出足够的时间。这取决于源数据库的大小，因此请相应地修改 360 值。	

删除起始数据库。	删除现有的 Amazon RDS Custom 数据库。
	
	以 Oracle 主用户的身份运行以下命令。（除非您对其进行了自定义，否则默认用户为 `rdsdb`。）
	
	`$ sqlplus / as sysdba`
	`SQL> shutdown immediate ;`
	`SQL> startup nomount restrict;`
	`SQL> alter database mount;`
	`SQL> drop database;`
	`SQL> exit`
		数据库管理员

Task	描述	所需技能

创建用于存储数据库文件的目录。	对于 Oracle 12.1.0.2：
	
	为数据库、控制文件、数据文件和联机日志创建目录。使用上一个命令中 `control_files` 参数的父目录（在本例中为 `VIS_A`）。以 Oracle 主用户（默认为 `rdsdb`）的身份运行以下命令。
	
	`$ mkdir -p /rdsdbdata/db/VIS_A/controlfile`
	`$ mkdir -p /rdsdbdata/db/VIS_A/datafile`
	`$ mkdir -p /rdsdbdata/db/VIS_A/onlinelog`
	
	对于 Oracle 19c：
	
	为数据库、控制文件、数据文件和联机日志创建目录。使用上一个命令中 `control_files` 参数的父目录（在本例中为 `VISCDB_A`）。以 Oracle 主用户（默认为 `rdsdb`）的身份运行以下命令。
	
	`$ mkdir -p /rdsdbdata/db/cdb/VISCDB_A/controlfile`
	`$ mkdir -p /rdsdbdata/db/cdb/VISCDB_A/datafile`
	`$ mkdir -p /rdsdbdata/db/cdb/VISCDB_A/onlinelog`
		数据库管理员

Task	描述	所需技能
	`$ mkdir -p /rdsdbdata/db/cdb/VISCDB_A/onlinelog/arch` `$ mkdir /rdsdbdata/db/pdb/VISCDB_A`	

Task	描述	所需技能
创建和修改 Oracle 电子商务套件的参数文件。	在此步骤中，您不会从源数据库复制服务器参数文件（SPFILE）。相反，您将使用以 Amazon RDS Custom 数据库实例创建的标准参数文件（PFILE），并添加 Oracle 电子商务套件所需参数。 当您删除数据库时，Amazon RDS 自动化会创建 `init.ora` 文件的备份，该备份与 Amazon RDS Custom 数据库相关联。此文件称为 `oracle_pfile` ，位于 `/rdsdbdata/config` 。 对于 Oracle 12.1.0.2： 1. 将 `/rdsdbdata/config/oracle_pfile` 复制到 `$ORACLE_HOME` 。 `$ cp /rdsdbdata/config/oracle_pfile $ORACLE_HOME/dbs/initVIS.ora` 2. 在 Amazon RDS Custom 数据库实例上编辑 `initVIS.ora` 文件。验证源上的所有参数并根据需要添加任何参数。有关详细信息，请参阅 Oracle Support Note 396009.1。	数据库管理员

Task	描述	所需技能

	Important
	确保您添加的参数中没有注释。注释会导致自动化出现问题，例如创建只读副本和发布 point-in-time恢复 () PITRs。
	
	3. 根据要求将与以下内容类似的参数添加到 `initVIS.ora` 文件中：
	
	`*.workarea_size_policy='AUTO'`
	`*.plsql_code_type='INTERPRETED'`
	`*.cursor_sharing='EXACT'`
	`*._b_tree_bitmap_plans=FALSE`
	`*.session_cached_cursors=500`
	`*.optimizer_adaptive_features=false`
	`*.optimizer_secure_view_merging=false`
	`*.SQL92_SECURITY=TRUE`
	`*.temp_undo_enabled=true`
	`_system_trig_enabled = TRUE`
	`nls_language = american`
	`nls_territory = america`
	`nls_numeric_characters = "."`
	`nls_comp = binary`	

Task	描述	所需技能
	nls_sort = binary nls_date_format = DD-MON-RR nls_length_semantics = BYTE aq_tm_processes = 1 _sort_eliminatio n_cost_ratio =5 _like_with_bind _as_equality = TRUE _fast_full_scan_enable d = FALSE _b_tree_bitmap_plans = FALSE optimizer_secure_view _merging = FALSE _optimizer_autostats_ job = FALSE parallel_max_servers = 8 parallel_min_servers = 0 parallel_degree_policy = MANUAL sec_case_sensitive_l ogon = FALSE compatible = 12.1.0 o7_dictionary_access ibility = FALSE utl_file_dir =/tmp 4. 修改如下。这些值将取决于源系统，因此请根据您当前的设置对其进行修改。 *.open_cursors=500 *.undo_tablespace ='APPS_UNDOTS1 5. 移除 SPFILE 引用。	

Task	描述	所需技能

	`*.spfile='/rdsdbbin/oracle/dbs/spfileVIS.ora'`
	
	备注：
	• 请勿更改 Amazon RDS Custom PFILE 为 `control_files` 和 `db_unique_name` 提供的值。Amazon RDS 需要这些值。如果您将来尝试创建只读副本，偏离这些值会导致问题。
	• 默认情况下，Amazon RDS Custom 使用自动内存管理（AMM）。如果您想使用 Hugemem，可以将 Amazon RDS Custom 配置为改用自动共享内存管理（ASMM）。
	• 默认情况下启用 `memory_max_target` 参数。Amazon RDS 框架在后台使用此参数来创建只读副本。
	
	6. 运行以下 `startup nomount` 命令确认 `initVIS.ora` 文件没有问题：
	
	`SQL> startup nomount pfile=/rdsdbbin/oracle/dbs/initVIS.ora;`	

Task	描述	所需技能
	SQL> create spfile='/rdsdbdata/admin/VIS/pfile/spfileVIS.ora' from pfile; SQL> exit 7. 为 SPFILE 创建符号链接。 $ ln -s /rdsdbdata/admin/VIS/pfile/spfileVIS.ora $ORACLE_HOME/dbs/ 对于 Oracle 19c： 1. 将 /rdsdbdata/config/oracle_pfile 复制到 $ORACLE_HOME 。 $ cp /rdsdbdata/config/oracle_pfile $ORACLE_HOME/dbs/initVISCDB.ora 2. 在 Amazon RDS Custom 数据库实例上编辑 initVISCDB.ora 文件。验证源上的所有参数并根据需要添加任何参数。有关详细信息，请参阅 Oracle Support Note 396009.1。 **Important** 确保您添加的参数中没有注释。 如果有评论，它们将导致自动化	

Task	描述	所需技能

	出现问题，例如创建只读副本和发布 point-in-time恢复（ ）PITRs。
	
	3. 根据要求将与以下内容类似的参数添加到 `initVISCDB.ora` 文件中。
	
	`*.instance_name=VISCDB`
	`*.sec_case_sensitive_logon= FALSE`
	`*.result_cache_max_size = 600M`
	`*.optimizer_adaptive_plans =TRUE`
	`*.optimizer_adaptive_statistics = FALSE`
	`*.pga_aggregate_limit = 0`
	`*.temp_undo_enabled = FALSE`
	`*._pdb_name_case_sensitive = TRUE`
	`*.event='10946 trace name context forever, level 8454144'`
	`*.workarea_size_policy='AUTO'`
	`*.plsql_code_type='INTERPRETED'`
	`*.cursor_sharing='EXACT'`
	`*._b_tree_bitmap_plans=FALSE`
	`*.session_cached_cursors=500`
	`*.optimizer_secure_view_merging=false`	

Task	描述	所需技能
	*.SQL92_SECURITY=TRUE _system_trig_enabled = TRUE nls_language = american nls_territory = america nls_numeric_charact ers = "." nls_comp = binary nls_sort = binary nls_date_format = DD- MON-RR nls_length_semantics = BYTE aq_tm_processes = 1 _sort_eliminatio n_cost_ratio =5 _like_with_bind _as_equality = TRUE _fast_full_scan_enable d = FALSE _b_tree_bitmap_plans = FALSE optimizer_secure_view _merging = FALSE _optimizer_autostats_ job = FALSE parallel_max_servers = 8 parallel_min_servers = 0 parallel_degree_policy = MANUAL 4. 修改如下。这些值将取决于源系统，因此请根据您当前的设置对其进行修改。 *.open_cursors=500	

Task	描述	所需技能

	`*.undo_tablespace ='UNDOTBS1'`
	
	5. 移除 SPFILE 引用：
	
	`*.spfile='/rdsdbbin/oracle/dbs/spfileVISCDB.ora'`
	
	备注：
	
	• 请勿更改 Amazon RDS Custom PFILE 为 `control_files` 和 `db_unique_name` 提供的值。Amazon RDS 需要这些值。如果您将来尝试创建只读副本，偏离这些值会导致问题。
	• 默认情况下，Amazon RDS Custom 使用自动内存管理（AMM）。如果您想使用 Hugemem，可以将 Amazon RDS Custom 配置为改用自动共享内存管理（ASMM）。
	• 默认情况下启用 `memory_max_target` 参数。Amazon RDS 框架在后台使用此参数来创建只读副本。
	
	6. 运行以下 `startup nomount` 命令确认 `initVISCDB.ora` 文件没有问题：	

Task	描述	所需技能

	`SQL> startup nomount pfile=/rdsdbbin/oracle/dbs/initVISCDB.ora;`
	`SQL> create spfile='/rdsdbdata/admin/VISCDB/pfile/spfileVISCDB.ora' from pfile;`
	`SQL> exit`
	
	7. 为 SPFILE 创建符号链接。
	
	`$ ln -s /rdsdbdata/admin/VISCDB/pfile/spfileVISCDB.ora $ORACLE_HOME/dbs/`	

Task	描述	所需技能

从备份中恢复 Amazon RDS Custom 数据库。	对于 Oracle 12.1.0.2：
	
	1. 使用之前在源上捕获的备份文件恢复控制文件：
	
	`RMAN> connect target /`
	`RMAN> RESTORE CONTROLFILE FROM '/RMAN/visdb_full_bkp_100rlsbt';`
	
	`Starting restore at 10-APR-22`
	`using target database control file instead of recovery catalog`
	`allocated channel: ORA_DISK_1`
	`channel ORA_DISK_1: SID=201 device type=DISK`
	
	`channel ORA_DISK_1: restoring control file`
	`channel ORA_DISK_1: restore complete, elapsed time: 00:00:01`
	`output file name=/rdsdbdata/db/VIS_A/controlfile/control-01.ctl`
	`Finished restore at 10-APR-22`
	
	2. 对备份片段进行编目，这样您就可以发出 RMAN `restore`：
	
	`RMAN> alter database mount;`
		数据库管理员

Task	描述	所需技能

	RMAN> catalog start
	 with '/RMAN/visdb';
	
	3. 创建用于恢复数据库的脚本：
	
	$ vi restore.sh
	rman target / log=/home
	/rdsdb/rman.log << EOF
	run
	{
	set newname for database
	 to '/rdsdbdata/db/VIS
	_A/datafile/%b';
	restore database;
	switch datafile all;
	switch tempfile all;
	}
	EOF
	
	4. 将源恢复到 Amazon RDS Custom 目标数据库。必须更改脚本的权限才能运行脚本，然后运行 `restore.sh` 脚本以恢复数据库。
	
	$ chmod 755 restore.sh
	$ nohup ./restore.sh &
	
	对于 Oracle 19c：
	
	1. 使用之前在源上捕获的备份文件恢复控制文件：
	
	RMAN> connect target /
	RMAN> RESTORE CONTROLFI
	LE FROM '/RMAN/cn
	trl.bak';	

Task	描述	所需技能
	Starting restore at 07-JUN-23 using target database control file instead of recovery catalog allocated channel: ORA_DISK_1 channel ORA_DISK_1: SID=201 device type=DISK channel ORA_DISK_1: restoring control file channel ORA_DISK_1: restore complete, elapsed time: 00:00:01 output file name=/rdsdbdata/db/cdb/VISCDB_A/controlfile/control-01.ctl Finished restore at 07-JUN-23 2. 对备份片段进行编目，这样您就可以发出 RMAN `restore`： RMAN> alter database mount; RMAN> catalog start with '/RMAN/visdb'; 如果您在使用 `start with` 命令时遇到问题，可以单独添加备份片段；例如： RMAN> catalog backuppiece '/RMAN/visdb_full_bkp_1d1e507m';	

Task	描述	所需技能

	然后对每个备份片段重复该命令。
	
	3. 创建用于恢复数据库的脚本。根据要求修改可插拔数据库的名称。根据CPUs 可用的 v 数分配 parallel 通道，以加快恢复过程。
	
	`$ vi restore.sh`
	`rman target / log=/home`
	`/rdsdb/rmancdb.log <<`
	`EOF`
	`run {`
	`allocate channel c1`
	`type disk;`
	`allocate channel c2`
	`type disk;`
	`.... .... ....`
	`allocate channel c<N>`
	`type disk;`
	`set newname for database`
	`to '/rdsdbdata/db/cdb`
	`/VISCDB_A/datafile/`
	`%b';`
	`set newname for database`
	`root to '/rdsdbda`
	`ta/db/cdb/VISCDB_A/`
	`datafile/%f_%b';`
	`set newname for`
	`database "PDB$SEED" to`
	`'/rdsdbdata/db/cdb/`
	`pdbseed/%f_%b';`
	`set newname for`
	`pluggable database VIS`
	`to '/rdsdbdata/db/pdb`
	`/VISCDB_A/%f_%b';`
	`restore database;`
	`switch datafile all;`
	`switch tempfile all;`	

Task	描述	所需技能
	release channel c1; release channel c2; release channel c3; .... release channel c<N>; } EOF	

	4. 将源恢复到 Amazon RDS Custom 目标数据库。必须更改脚本的权限才能运行脚本，然后运行 `restore.sh` 脚本以恢复数据库。
	
	$ chmod 755 restore.sh
	$ nohup ./restore.sh &	

Task	描述	所需技能

检查日志文件中是否存在问题。	对于 Oracle 12.1.0.2：
	
	1. 通过查看 `rman.log` 文件确认没有问题：
	
	`$ cat /home/rdsdb/rman.log`
	
	2. 确认在控制文件中注册的日志文件的路径：
	
	`SQL> select member from v$logfile;`
	`MEMBER`
	`----------`
	`/d01/oracle/VIS/data/log1.dbf`
	`/d01/oracle/VIS/data/log2.dbf`
	`/d01/oracle/VIS/data/log3.dbf`
	
	3. 重命名日志文件以匹配目标的文件路径。替换路径以匹配上一步的输出：
	
	`SQL> ALTER DATABASE RENAME FILE '/d01/oracle/VIS/data/log1.dbf' TO '/rdsdbdata/db/VIS_A/onlinelog/log1.dbf';`
	`SQL> ALTER DATABASE RENAME FILE '/d01/oracle/VIS/data/log2.`
		数据库管理员

Task	描述	所需技能
	dbf' TO '/rdsdbdata/db/VIS_A/onlinelog/log2.dbf'; SQL> ALTER DATABASE RENAME FILE '/d01/oracle/VIS/data/log3.dbf' TO '/rdsdbdata/db/VIS_A/onlinelog/log3.dbf'; 对于 Oracle 19c： 1. 通过查看 rmancdb.log 文件确认没有问题： $ cat /home/rdsdb/rmancdb.log 2. 确认在控制文件中注册的日志文件的路径： SQL> select member from v$logfile; MEMBER ---------- /d01/oracle/VIS/oradata/VISCDB/redo03.log /d01/oracle/VIS/oradata/VISCDB/redo02.log /d01/oracle/VIS/oradata/VISCDB/redo01.log	

Task	描述	所需技能

	3. 重命名日志文件以匹配目标的文件路径。替换路径以匹配上一步的输出：
	
	`SQL> ALTER DATABASE RENAME FILE '/d01/oracle/VIS/oradata/VISCDB/redo01.log' TO '/rdsdbdata/db/cdb/VISCDB_A/onlinelog/log1.dbf';`
	`SQL> ALTER DATABASE RENAME FILE '/d01/oracle/VIS/oradata/VISCDB/redo02.log' TO '/rdsdbdata/db/cdb/VISCDB_A/onlinelog/log2.dbf';`
	`SQL> ALTER DATABASE RENAME FILE '/d01/oracle/VIS/oradata/VISCDB/redo03.log' TO '/rdsdbdata/db/cdb/VISCDB_A/onlinelog/log3.dbf';`
	
	4. 确认路径、日志文件的状态以及在控制文件中注册的组号：
	
	`SQL> column REDOLOG_FILE_NAME format a50`
	`SQL> SELECT a.GROUP#, a.status, b.MEMBER AS REDOLOG_FILE_NAME, (a.BYTES/1024/1024) AS SIZE_MB`
	`FROM v$log a`
	`JOIN v$logfile b ON a.Group#=b.Group#`	

Task	描述	所需技能
	ORDER BY a.GROUP#; GROUP# STATUS REDOLOG_F ILE_NAME SIZE_MB 1 CURRENT /rdsdbdat a/db/cdb/VISCDB_A/ onlinelog/log1.dbf 512 2 INACTIVE /rdsdbdat a/db/cdb/VISCDB_A/ onlinelog/log2.dbf 512 3 INACTIVE /rdsdbdat a/db/cdb/VISCDB_A/ onlinelog/log3.dbf 512	

Task	描述	所需技能

确认您可以打开 Amazon RDS Custom 数据库并创建 OMF 日志文件。	适用于 Oracle 的 Amazon RDS Custom 使用 Oracle 托管文件 (OMF) 来简化操作。您可以将只读副本提升为独立实例，但必须先使用 OMF 创建日志文件。这是为了确保在提升实例时使用正确的路径。有关如何提升只读副本的更多信息，请参阅 Amazon RDS 文档。尝试提升只读副本时，不使用 OMF 文件可能会导致问题。
	
	1. 使用 `resetlogs` 打开数据库：
	
	`SQL> alter database open resetlogs;`
	
	Note
	如果您收到错误 ORA-00392：线程 1 的日志 xx 正在清除，不允许操作，请按照“疑难解答”部分中针对 ORA-00392 的步骤进行操作。
	
	2. 确认该数据库打开：
	
	`SQL> select open_mode from v$database;`
	`OPEN_MODE`
	`----------`
		数据库管理员

Task	描述	所需技能
	READ WRITE 3. 创建 OMF 日志文件。使用上一个日志文件查询的输出，根据要求更改组号、组数和大小。以下示例从组 4 开始，为简单起见，添加了三个组。 SQL> alter database add logfile group 4 size 512M; Database altered. SQL> alter database add logfile group 5 size 512M; Database altered. SQL> alter database add logfile group 6 size 512M; Database altered. 4. 删除之前的非 OMF 文件。以下是您可以根据自己的要求和前述步骤中查询的输出进行自定义的示例： SQL> alter database drop logfile group 1; System altered. SQL> alter database drop logfile group 2; System altered. SQL> alter database drop logfile group 3; System altered.	

Task	描述	所需技能

	Note
	如果您在尝试删除日志文件时收到 ORA-01624 错误，请参阅故障排除部分。
	
	5. 确认您可以看到已创建的 OMF 文件。（Oracle 12.1.0.2 和 19c 的目录路径各不相同，但概念是一样的。）
	
	`SQL> select member from v$logfile;`
	
	`MEMBER`
	`----------`
	`/rdsdbdata/db/cdb/VISCDB_A/onlinelog/o1_mf_4_ksrbslny_.log`
	`/rdsdbdata/db/cdb/VISCDB_A/onlinelog/o1_mf_5_ksrchw0k_.log`
	`/rdsdbdata/db/cdb/VISCDB_A/onlinelog/o1_mf_6_ksrcnl9v_.log`
	
	6. 重新启动数据库并确认实例正在使用 SPFILE：
	
	`SQL> shutdown immediate`
	`SQL> startup`
	`SQL> show parameter spfile`	

Task	描述	所需技能

	对于 Oracle 12.1.0.2，此查询会返回：
	`spfile /rdsdbbin /oracle/dbs/spfile VIS.ora`
	对于 Oracle 19c，该查询会返回：
	`spfile /rdsdbbin /oracle/dbs/spfile VISCDB.ora`
	7. 仅适用于 Oracle 19c，请检查容器数据库的状态，并在需要时将其打开：
	`SQL> show pdbs CON_ID CON_NAME OPEN MODE RESTRICTED ---------- --------- ---------- --------- - 2 PDB$SEED READ ONLY NO 3 VIS MOUNTED NO`
	`SQL> alter session set container=VIS; Session altered.`
	`SQL> alter database open; Database altered.`	

Task	描述	所需技能

	SQL> alter database save state;
	Database altered.
	
	SQL> show pdbs
	CON_ID CON_NAME OPEN MODE RESTRICTED
	---------- ---------- ---------- ----------
	3 VIS READ WRITE NO
	
	SQL> exit
	
	8. 由于您没有使用 PFILE，请从 $ORACLE_HOME/dbs 中删除该 init.ora 文件：
	
	$ cd $ORACLE_HOME/dbs
	
	对于 Oracle 12.1.0.2，使用命令：
	
	$ pwd
	/rdsdbbin/oracle/dbs
	$ rm initVIS.ora
	
	对于 Oracle 19c，使用命令：
	
	$ pwd
	/rdsdbbin/oracle/dbs
	$ rm initVISCDB.ora	

从 Secrets Manager 中检索密码，创建用户和更改密码

Task	描述	所需技能
从 Secrets Manager 中检索密码。	您可以在控制台中或使用 AWS CLI 执行这些步骤。以下步骤提供了控制台的说明。 1. 登录 AWS 管理控制台并通过以下网址打开 Amazon RDS 控制台https://console.aws.amazon.com/rds/。 2. 在导航窗格中，选择数据库，然后选择 Amazon RDS 数据库。 3. 选择配置，并记下实例的资源 ID（格式为：`db-WZ4WLCK6AOQ6TJGZKMGRCDCI3Y`）。 4. 打开 AWS Secrets Manager 控制台，网址为https://console.aws.amazon.com/secretsmanager/。 5. 选择与 `do-not-delete-custom-<resource_id>` 同名的密钥，其中 `resource-id` 指的是您在步骤 3 中记下的实例的 ID。 6. 选择 Retrieve secret value (检索密钥值)。	数据库管理员
创建 RDSADMIN 用户。	`RDSADMIN` 是 Amazon RDS Custom 数据库实例中用于监控和编排数据库的用户。由于	数据库管理员

Task	描述	所需技能

	新手数据库已删除，目标数据库已使用 RMAN 从源中恢复，因此您必须在恢复操作后重新创建此用户，以确保 Amazon RDS Custom 监控按预期运行。您还必须为 RDSADMIN 用户创建单独的配置文件和表空间。Oracle 12.1.0.2 和 19c 的说明略有不同。
	
	对于 Oracle 12.1.0.2：
	
	1. 在 SQL 提示符中，输入以下命令：
	
	`SQL> set echo on feedback on serverout on`
	`SQL> @?/rdbms/admin/utlpwdmg.sql`
	
	`SQL> ALTER PROFILE DEFAULT LIMIT FAILED_LOGIN_ATTEMPTS UNLIMITED PASSWORD_LIFE_TIME UNLIMITED PASSWORD_VERIFY_FUNCTION NULL;`
	
	2. 创建配置文件 RDSADMIN：
	
	`SQL> create profile RDSADMIN LIMIT COMPOSITE_LIMIT UNLIMITED`	

Task	描述	所需技能
	SESSIONS_PER_USER UNLIMITED CPU_PER_SESSION UNLIMITED CPU_PER_CALL UNLIMITED LOGICAL_READS_PER _SESSION UNLIMITED LOGICAL_READS_PER_CALL UNLIMITED IDLE_TIME UNLIMITED CONNECT_TIME UNLIMITED PRIVATE_SGA UNLIMITED FAILED_LOGIN_ATTEMPTS 10 PASSWORD_LIFE_TIME UNLIMITED PASSWORD_REUSE_TIME UNLIMITED PASSWORD_REUSE_MAX UNLIMITED PASSWORD_VERIFY_F UNCTION NULL PASSWORD_LOCK_TIME 86400/86400 PASSWORD_GRACE_TIME 604800/86400; 3. 将 SYS、SYSTEM 和 DBSNMP 用户配置文件设置为 RDSADMIN： SQL> set echo on feedback on serverout on SQL> alter user SYS profile RDSADMIN; SQL> alter user SYSTEM profile RDSADMIN; SQL> alter user DBSNMP profile RDSADMIN;	

Task	描述	所需技能

	4. 创建 RDSADMIN 表空间：
	
	`SQL> create bigfile tablespace rdsadmin datafile size 7M autoextend on next 1m Logging online permanent blocksize 8192 extent management local autoallocate default nocompress segment space management auto;`
	
	5. 创建 RDSADMIN 用户。将 RDSADMIN 密码替换为您之前从 Secrets Manager 获得的密码：
	
	`SQL> create user rdsadmin identified by xxxxxxxxxx Default tablespace rdsadmin Temporary tablespace temp profile rdsadmin ;`
	
	6. 向 RDSADMIN 授予权限：
	
	`SQL> grant select on sys.v_$instance to rdsadmin; SQL> grant select on sys.v_$archived_log to rdsadmin; SQL> grant select on sys.v_$database to rdsadmin;`	

Task	描述	所需技能
	SQL> grant select on sys.v_$database_in carnation to rdsadmin; SQL> grant select on dba_users to rdsadmin; SQL> grant alter system to rdsadmin; SQL> grant alter database to rdsadmin; SQL> grant connect to rdsadmin with admin option; SQL> grant resource to rdsadmin with admin option; SQL> alter user rdsadmin account unlock identified by xxxxxxxxxxx; SQL> @?/rdbms/admin/userlock.sql SQL> @?/rdbms/admin/utlrp.sql 对于 Oracle 19c： 1. 在 SQL 提示符中，输入以下命令： SQL> set echo on feedback on serverout on SQL> @?/rdbms/admin/utlpwdmg.sql SQL> alter profile default LIMIT FAILED_LOGIN_ATTEMPTS UNLIMITED	

Task	描述	所需技能
	PASSWORD_LIFE_TIME UNLIMITED PASSWORD_VERIFY_F UNCTION NULL; 2. 创建配置文件 RDSADMIN。 Note RDSADMINC##在 Oracle 19c 中有一个前缀。 这是因为数据库参数设置common_user_prefix 为C##。 RDSADMIN在 Oracle 12.1.0.2 中没有前缀。 SQL> create profile C##RDSADMIN LIMIT COMPOSITE_LIMIT UNLIMITED SESSIONS_PER_USER UNLIMITED CPU_PER_SESSION UNLIMITED CPU_PER_CALL UNLIMITED LOGICAL_READS_PER _SESSION UNLIMITED LOGICAL_READS_PER_CALL UNLIMITED IDLE_TIME UNLIMITED CONNECT_TIME UNLIMITED PRIVATE_SGA UNLIMITED FAILED_LOGIN_ATTEMPTS 10	

Task	描述	所需技能
	PASSWORD_LIFE_TIME UNLIMITED PASSWORD_REUSE_TIME UNLIMITED PASSWORD_REUSE_MAX UNLIMITED PASSWORD_VERIFY_F UNCTION NULL PASSWORD_LOCK_TIME 86400/86400 PASSWORD_GRACE_TIME 604800/86400; 3. 将 SYS、SYSTEM 和 DBSNMP 用户配置文件设置为 RDSADMIN： SQL> alter user SYS profile C##RDSADMIN; SQL> alter user SYSTEM profile C##RDSADMIN; SQL> alter user DBSNMP profile C##RDSADMIN; 4. 创建 RDSADMIN 表空间： SQL> create bigfile tablespace rdsadmin datafile size 7M autoextend on next 1m Logging online permanent blocksize 8192 extent managemen t local autoallocate default nocompress segment space managemen t auto;	

Task	描述	所需技能

	5. 创建 RDSADMIN 用户。将 RDSADMIN 密码替换为您之前从 Secrets Manager 获得的密码。
	
	`SQL> create user C##rdsadmin identified by xxxxxxxxxx profile C##rdsadmin container=all;`
	
	6. 向 RDSADMIN 授予权限：
	
	`SQL> grant select on sys.v_$instance to c##rdsadmin;`
	`SQL> grant select on sys.v_$archived_log to c##rdsadmin;`
	`SQL> grant select on sys.v_$database to c##rdsadmin;`
	`SQL> grant select on sys.v_$database_incarnation to c##rdsadmin;`
	`SQL> grant select on dba_users to c##rdsadmin;`
	`SQL> grant alter system to C##rdsadmin;`
	`SQL> grant alter database to C##rdsadmin;`
	`SQL> grant connect to C##rdsadmin with admin option;`
	`SQL> grant resource to C##rdsadmin with admin option;`	

Task	描述	所需技能
	SQL> alter user C##rdsadmin account unlock identified by xxxxxxxxxxx; SQL> @?/rdbms/admin/userlock.sql SQL> @?/rdbms/admin/utlrp.sql	
创建主用户。	由于新手数据库已删除，目标数据库已使用 RMAN 从源中恢复，因此您必须重新创建主用户。在此示例中，主用户名为 admin。 对于 Oracle 12.1.0.2： SQL> create user admin identified by <password>; SQL> grant dba to admin 对于 Oracle 19c： SQL> alter session set container=VIS; Session altered. SQL> create user admin identified by <password>; User created. SQL> grant dba to admin; Grant succeeded.	数据库管理员

Task	描述	所需技能

更改超级用户密码。	1. 使用您从 Secrets Manager 中检索到的密码更改系统密码。
	
	对于 Oracle 12.1.0.2：
	
	`SQL> alter user sys identified by xxxxxxxxxxx;`
	`SQL> alter user system identified by xxxxxxxxxx;`
	
	对于 Oracle 19c：
	
	`SQL> alter user sys identified by xxxxxxxxxxx container =all;`
	`SQL> alter user system identified by xxxxxxxxxx container =all;`
	
	1. 更改 EBS_SYSTEM 密码。
	
	对于 Oracle 12.1.0.2：
	
	`SQL> alter user ebs_system identified by xxxxxxxxxx;`
	
	对于 Oracle 19c：
	
	对于此版本，您还必须连接到容器数据库，以更新那里的 EBS_SYSTEM 密码。
		数据库管理员

Task	描述	所需技能

	`SQL> alter session set container=vis;`
	`SQL> alter user ebs_system identified by xxxxxxxxxx;`
	
	`SQL> exit;`
	
	如果您不更改这些密码，Amazon RDS Custom 会显示错误消息：数据库监控用户或用户凭证已更改。	

为 Oracle 电子商务套件创建目录，安装 ETCC，然后运行 Autoconfig

Task	描述	所需技能
创建 Oracle 电子商务套件所需目录。	1. 在 Amazon RDS Custom Oracle 数据库上，以 Oracle 主用户身份运行以下脚本，在 `$ORACLE_HOME/nls/data/9idata` 中创建 `9idata` 目录。Oracle 电子商务套件需要此目录。 `perl $ORACLE_HOME/nls/data/old/cr9idata.pl` 忽略该 `ORA_NLS10` 消息，因为您将在后续步骤中创建支持上下文的环境。 2. 复制您之前从共享的 Amazon EFS 文件系统中创建的 `appsutil.tar` 文件，然	

Task	描述	所需技能

	后将其解压缩到 Amazon RDS Custom Oracle 主目录中。 这将在 $ORACLE_HOME 目录中创建 appsutil 目录。
	
	`$ cd /RMAN/appsutil`
	`$ cp sourceappsutil.tar $ORACLE_HOME`
	`$ cd $ORACLE_HOME`
	`$ tar xvf sourceappsutil.tar appsutil`
	
	3. 复制您之前保存在 Amazon EFS 共享文件系统上的 appsutil.zip 文件。 这是您在应用程序层创建的文件。
	
	作为 Amazon RDS Custom 数据库实例上的 rdsdb 用户：
	
	`$ cp /RMAN/appsutil/appsutil.zip $ORACLE_HOME`
	`$ cd $ORACLE_HOME`
	
	4. 解压缩 appsutil.zip 文件以在 Oracle 主目录中创建 appsutil 目录和子目录：
	
	`$ unzip -o appsutil.zip`
	
	该 -o 选项意味着将覆盖某些文件。	

Task	描述	所需技能

配置 tsanames.ora 和 sqlnet.ora 文件。	您必须配置 `tnsnames.ora` 文件，这样才能使用 Autoconfig 工具连接到数据库。在以下示例中，您可以看到该 `tnsnames.ora` 文件是软链接的，但默认情况下该文件为空。
	
	`$ cd $ORACLE_HOME/network/admin`
	`$ ls -ltr`
	`-rw-r--r-- 1 rdsdb database 373 Oct 31 2013 shrept.lst`
	`lrwxrwxrwx 1 rdsdb database 30 Feb 9 17:17 listener.ora -> /rdsdbdata/config/listener.ora`
	`lrwxrwxrwx 1 rdsdb database 28 Feb 9 17:17 sqlnet.ora -> /rdsdbdata/config/sqlnet.ora`
	`lrwxrwxrwx 1 rdsdb database 30 Feb 9 17:17 tnsnames.ora -> /rdsdbdata/config/tnsnames.ora`
	
	1. 创建 `tnsnames.ora` 条目。由于 Amazon RDS 自动化解析文件的方式，您必须确保该条目不包含任何空格、评论或多余的行。否则，在使用某些方法（ APIs 例如 create-db-instance-read-replic a ）时可能
		数据库管理员

Task	描述	所需技能

	会遇到问题。使用以下内容作为示例。
	
	2. 根据要求更换端口、主机和 SID：
	
	`$ vi tnsnames.ora`
	`VIS=(DESCRIPTION=`
	`(ADDRESS_LIST=(ADD`
	`RESS=(PROTOCOL=TCP`
	`)(PORT=1521)(HOST=`
	`xx.xx.xx.xx)))(CON`
	`NECT_DATA=(SID=VIS)`
	`(SERVER=DEDICATED)))`
	
	Note
	文件中不应有多余的行。如果不删除这些行，则将来在创建只读副本时可能会遇到问题。创建只读副本可能会失败，并显示错误消息：活动引发异常：HostManagerException：无法在任何主机上成功调用 restrictReplication。
	
	3. 确认可以访问数据库：
	
	`$ tnsping vis`
	`OK (0 msec)`	

Task	描述	所需技能

	4. 仅适用于 Oracle 19c，请更新该 `sqlnet.ora` 文件。不这样做将导致错误 ORA-01017：用户名/密码无效；尝试连接数据库时登录被拒绝。编辑 `$ORACLE_HOME/network/admin` 中的 `sqlnet.ora` 以匹配以下内容：
	`NAMES.DIRECTORY_PATH=(TNSNAMES, ONAMES, HOSTNAME)`
	`SQLNET.EXPIRE_TIME= 10`
	`SQLNET.INBOUND_CONNECT_TIMEOUT =60`
	`SQLNET.ALLOWED_LOGON_VERSION_SERVER=10`
	`HTTPS_SSL_VERSION=undetermined`
	5. 连接测试：
	`$ sqlplus apps/****@vis`	

Task	描述	所需技能

配置数据库。	现在，您已经测试了与数据库的连接，可以使用 appsutil 实用程序配置数据库，以创建支持上下文的环境。
	
	对于 Oracle 12.1.0.2：
	
	1. 运行以下 命令：
	
	`$ cd $ORACLE_HOME/appsutil/bin`
	`$ perl adbldxml.pl appsuser=apps`
	`Enter Hostname of Database server: oebs-db01`
	`Enter Port of Database server: 1521`
	`Enter SID of Database server: VIS`
	`Enter Database Service Name: VIS`
	`Enter the value for Display Variable: :1`
	`The context file has been created at: /rdsdbbin/oracle/appsutil/VIS_oebs-db01.xml`
	
	2. 从根用户创建 `oraInst.loc` ：
	
	`$ vi /etc/oraInst.loc`
	`inventory_loc=/rdsdbbin/oracle.12.1.custom.r1.EE.1/oraInventory`
	`inst_group=database`
		数据库管理员

Task	描述	所需技能

	3. 使用上一步中创建的上下文文件克隆上下文文件以设置逻辑主机名。以 rdsdb 用户身份运行：
	
	`$ cd $ORACLE_HOME/appsutil/clone/bin`
	`$ perl adclonectx.pl \`
	`contextfile=[ORACLE_HOME]/appsutil/[current context file] \`
	`template=[ORACLE_HOME]/appsutil/template/adxdbctx.tmp`
	
	其中，oebs-db01log 指的是逻辑主机名。例如：
	
	`$ perl adclonectx.pl \`
	`contextfile=/rdsdbbin/oracle.12.1.custom.r1.EE.1/appsutil/VIS_oebs-db01.xml \`
	`template=/rdsdbbin/oracle/appsutil/template/adxdbctx.tmp`
	`Target System Hostname (virtual or normal) [oebs-db01] : oebs-db01log`
	`Target System Base Directory : /rdsdbbin/oracle`
	`Target Instance is RAC (y/n) [n] : n`
	
	`Target System Database SID : VIS`	

Task	描述	所需技能
	Oracle OS User [rdsdb] : Oracle OS Group [rdsdb] : database Role separation is supported y/n [n] ? : n Target System utl_file_ dir Directory List : / tmp Number of DATA_TOP's on the Target System [1] : Target System DATA_TOP Directory 1 [/rdsdbbi n/oracle/data] : / rdsdbdata/db/VIS_A/ datafile/ Target System RDBMS ORACLE_HOME Directory [/rdsdbbin/oracle/ 12.1.0] : /rdsdbbin/ oracle Do you want to preserve the Display [:1] (y/n) : y Do you want the target system to have the same port values as the source system (y/n) [y] ? : y The new database context file has been created : /rdsdbbin/oracle.1 2.1.custom.r1.EE.1/ appsutil/clone/bin/ VIS_oebs-db01log.xml contextfile=/rdsdbbin/ oracle.12.1.custom	

Task	描述	所需技能

	`.r1.EE.1/appsutil/clone/bin/VIS_oebs-db01log.xml`
	
	对于 Oracle 19c：
	
	1. 运行以下 命令：
	
	`$ cd $ORACLE_HOME/appsutil/bin`
	`$ perl adbldxml.pl appsuser=apps`
	`Enter Hostname of Database server: oebs-db01`
	`Enter Port of Database server: 1521`
	`Enter SID of Database server: VIS`
	`Enter the database listener name:L_VISCDB_001`
	`Enter the value for Display Variable: :1`
	`The context file has been created at: /rdsdbbin/oracle/appsutil/VIS_oebs-db01.xml`
	
	2. 从根用户创建 `oraInst.loc` ：
	
	`$ vi /etc/oraInst.loc`
	`inventory_loc=/rdsdbbin/oracle/oraInventory`
	`inst_group=database`	

Task	描述	所需技能

	3. 使用上一步中创建的上下文文件克隆上下文文件以设置逻辑主机名。以 rdsdb 用户身份运行：
	
	`$ cd $ORACLE_HOME/appsutil/clone/bin`
	`$ perl adclonectx.pl \`
	`contextfile=[ORACLE_HOME]/appsutil/[current context file] \`
	`template=[ORACLE_HOME]/appsutil/template/adxdbctx.tmp`
	
	其中，oebs-db01log 指的是逻辑主机名。例如：
	
	`$ perl adclonectx.pl \`
	`contextfile=/rdsdbbin/oracle/appsutil/VIS_oebs-db01.xml \`
	`template=/rdsdbbin/oracle/appsutil/template/adxdbctx.tmp`
	`Target System Hostname (virtual or normal) [oebs-db01] : oebs-db01log`
	`Target System Base Directory : /rdsdbbin/oracle`
	`Target Instance is RAC (y/n) [n] : n`
	`Target System CDB Name : VISCDB`
	`Target System PDB Name : VIS`
	`Oracle OS User [oracle] : rdsdb`	

Task	描述	所需技能
	Oracle OS Group [dba] : database Role separation is supported y/n [n] ? : n Number of DATA_TOP's on the Target System [2] : Target System DATA_TOP Directory 1 [/d01/oracle/VISCDB] : /rdsdbdata/db/pdb/VISCDB_A Target System DATA_TOP Directory 2 [/d01/oracle/data] : /rdsdbdata/db/pdb/VISCDB_A/datafile Specify value for OSBACKUPDBA group [database] : Specify value for OSDGDBA group [database] : Specify value for OSKMDBA group [database] : Specify value for OSRACDBA group [database] : Target System RDBMS ORACLE_HOME Directory [/d01/oracle/19.0.0] : /rdsdbbin/oracle Do you want to preserve the Display [:1] (y/n) : y Do you want the target system to have the same port values as the source system (y/n) [y] ? : y	

Task	描述	所需技能
	Validating if the source port numbers are available on the target system.. Complete port information available at /rdsdbbin/oracle/appsutil/clone/bin/out/VIS_oebs-db01log/portpool.lst New context path and file name [VIS_oebs-db01log.xml] : /rdsdbbin/oracle/appsutil/VIS_oebs-db01log.xml Do you want to overwrite it (y/n) [n] ? : y Replacing /rdsdbbin/oracle/appsutil/VIS_oebs-db01log.xml file. The new database context file has been created : contextfile=/rdsdbbin/oracle/appsutil/VIS_oebs-db01log.xml Check Clone Context logfile /rdsdbbin/oracle/appsutil/clone/bin/CloneContext_0609141428.log for details.	

Task	描述	所需技能

安装 ETCC 并运行 Autoconfig。	1. 安装 Oracle 电子商务套件技术代码级别检查器（ETCC）。
	
	从 My Oracle Support 下载补丁 17537119，然后按照 `README.txt` 中的说明进行操作。您将在该 `$ORACLE_HOME` 目录中创建一个名为 `etcc` 的目录，解压缩补丁以创建名为 `checkMTpatch.sh` 的脚本，然后运行该脚本来检查补丁版本。
	
	2. 运行 Autoconfig 实用程序，然后传递新的逻辑主机名上下文文件。
	
	对于 Oracle 12.1.0.2：
	
	`cd $ORACLE_HOME/appsutil/bin`
	`$ ./adconfig.sh contextfile=/rdsdbbin/oracle.12.1.customstom.r1.EE.1/appsutil/clone/bin/VIS_oebs-db01log.xml`
	
	对于 Oracle 19c：
	
	Autoconfig 需要侦听器名称以匹配 CDBNAME。因此，备份的原始侦听器配置文件将暂时使用 `L_<CDBNAME>_001` 。
		数据库管理员

Task	描述	所需技能

	$ lsnrctl stop L_VISCDB_
	001
	$ cp -rp /rdsdbdata/
	config/listener.ora /
	rdsdbdata/config/
	listener.ora_orig
	$ vi /rdsdbdata/
	config/listener.ora
	:%s/L_VISCDB_001/
	VISCDB/g
	$ lsnrctl start VISCDB
	$ cd /rdsdbbin/oracle/a
	ppsutil
	$ . ./txkSetCfgCDB.env
	dboraclehome=/rdsd
	bbin/oracle.19.cus
	tom.r1.EE-CDB.1
	
	Oracle Home being
	passed: /rdsdbbin/
	oracle
	
	$ echo $ORACLE_HOME
	/rdsdbbin/orac
	le.19.custom.r1.EE-
	CDB.1
	$ export ORACLE_SI
	D=VISCDB
	$ cd $ORACLE_HOME/
	appsutil/bin
	$ perl $ORACLE_H
	OME/appsutil/bin/t
	xkPostPDBCreationT
	asks.pl -dboraclehome=
	$ORACLE_HOME -outdir=
	$ORACLE_HOME/appsut
	il/log -cdbsid=VISCDB
	-pdbsid=VIS -appsuser
	=apps -dbport=1521 -
	servicetype=onpremise	

Task	描述	所需技能

	`Enter the APPS Password: <apps password>`
	
	`Enter the CDB SYSTEM Password:<password from secrets manager>`
	
	Note
	如果您的数据库目录已更改，请按照 Oracle Support Note 2525754.1 中的说明进行操作。	

为 Amazon RDS Custom 和 Oracle 电子商务套件配置 TNS 条目

Task	描述	所需技能
为 Amazon RDS Custom 和 Oracle 电子商务套件配置 TNS 条目。	Autoconfig 会在默认位置生成 TNS ifile。对于 Oracle 12.1.0.2（非 CDB）和 Oracle19C PDB，默认位置为 `$ORACLE_HOME/network/admin/$<CONTEXT_NAME>` 。适用于 Oracle 19c 的 CDB 使用默认 `$ORACLE_HOME/network/admin/` ，如在前述步骤中运行 Autoconfig 时生成的环境文件中 `$TNS_ADMIN` 所定义的那样。	数据库管理员

Task	描述	所需技能

	对于 Oracle 12.1.0.2 和 19c CDB，您不会使用这些默认值，因为 Autoconfig 生成的 `tnsnames.ora` 和 `listener.ora` 文件不符合 Amazon RDS 的要求，例如没有空格或评论。相反，您可以使用 Amazon RDS Custom 数据库提供的通用文件来确保符合系统的期望并降低出错的可能性。
	
	例如，Amazon RDS Custom 需要以下命名格式：
	
	`L_<INSTANCE_NAME>_001`
	
	对于 Oracle 12.1.0.2，这将是：
	
	`L_VIS_001`
	
	对于 Oracle 19c，这将是：
	
	`L_VISCDB_001`
	
	以下是您将要使用的 `listener.ora` 文件示例。这是在您创建 Amazon RDS Custom 数据库时生成的。此时，您尚未对此文件进行任何更改，而是将其保留为默认值。
	
	对于 Oracle 12.1.0.2：	

Task	描述	所需技能

	`$ cd $ORACLE_HOME/network/admin`
	`$ cat listener.ora`
	`ADR_BASE_L_VIS_001=/rdsdbdata/log/`
	`SID_LIST_L_VIS_001=(SID_LIST =`
	`(SID_DESC = (SID_NAME = VIS)(GLOBAL_DBNAME = VIS) (ORACLE_HOME = /rdsdbbin/oracle)))`
	`L_VIS_001=(DESCRIPTION_LIST =`
	`(DESCRIPTION =`
	`(ADDRESS = (PROTOCOL = TCP)(PORT = 1521)(HOST = xx.xx.xx.xx))) (DESCRIPTION =`
	`(ADDRESS = (PROTOCOL = TCP)(PORT = 1521)(HOST = 127.0.0.1))))`
	`SUBSCRIBE_FOR_NODE_DOWN_EVENT_L_VIS_001=OFF`
	
	对于 Oracle 19c：使用侦听器名称 `L_<INSTANCE_NAME>_001` 恢复原始 `listener.ora` 文件。
	
	`$ cd $ORACLE_HOME/network/admin`
	
	`$ cp -rp /rdsdbdata/config/listener.ora /rdsdbdata/config/listener.ora_autoconfig`
	`$ cp -rp /rdsdbdata/config/listener.`	

Task	描述	所需技能
	ora_orig /rdsdbdata/config/listener.ora $ cat listener.ora SUBSCRIBE_FOR_NODE_DOWN_EVENT_L_VISCDB_001=OFF ADR_BASE_L_VISCDB_001=/rdsdbdata/log/ USE_SID_AS_SERVICE_L_VISCDB_001=ON L_VISCDB_001=(DESCRIPTION_LIST = (DESCRIPTION = (ADDRESS = (PROTOCOL = TCP)(PORT = 1521)(HOST = xx.xx.xx.xx))) (DESCRIPTION = (ADDRESS = (PROTOCOL = TCP)(PORT = 1521)(HOST = 127.0.0.1)))) SID_LIST_L_VISCDB_001=(SID_LIST = (SID_DESC = (SID_NAME = VISCDB)(GLOBAL_DBNAME = VISCDB) (ORACLE_HOME = /rdsdbbin/oracle)))	

	启动标准 Amazon RDS 操作的侦听器 L_<INSTANCE_NAME>_001 ：
	
	$ lsnrctl stop
	$ lsnrctl start L_VISCDB_001	

	对于 Oracle 12.1.0.2： 编辑 Oracle 电子商务套件环境文件以更改 $TNS_ADMIN 路径，从而使用 Amazon	

Task	描述	所需技能

	RDS Custom 通用 TNS ifile。环境文件是在您之前运行 Autoconfig 时创建的。通过删除 <CONTEXT_NAME> 后缀来编辑 TNS_ADMIN 变量。
	
	Note
	您只能在 Oracle 12.1.0.2 中编辑环境文件，因为 19c 的默认主目录是$ORACLE_HOME/network/admin，这与 Amazon RDS Custom 的默认主目录相同。
	
	例如，在 Oracle 12.1.0.2 中，编辑文件：
	
	`$ vi $ORACLE_HOME/VIS_oebs-db01log.env`
	
	将路径从 更改为：
	
	`TNS_ADMIN=”/rdsdbbin/oracle/network/admin/VIS_oebs-db01log”`
	`export TNS_ADMIN`
	
	更改为：
	
	`TNS_ADMIN=”/rdsdbbin/oracle/network/admin”`
	`export TNS_ADMIN`	

Task	描述	所需技能

	Note
	每次运行 Autoconfig 时，都必须重复此步骤，以确保使用正确的 TNS ifile。（仅限 12.1.0.2）。
	
	对于 Oracle 19c：
	
	1. 将数据库层上下文变量 `s_cdb_tnsadmin` 的值更改为 `<ORACLE_HOME>/network/admin` 而不是 `<ORACLE_HOME>/network/admin/<CONTEXT_NAME>` 。
	
	Note
	不要更新`s_db_tnsadmin` 上下文变量。使其仍为 `<ORACLE_HOME>/network/admin/<CONTEXT_NAME>` 。
	
	`$ . $ORACLE_HOME/VIS_oebs-db01log.env`
	`$ vi $CONTEXT_FILE`
	
	2. 保存您对 `s_cdb_tnsadmin` 的值所做的更改。	

Task	描述	所需技能

	s_db_tnsadmin 和 s_cdb_tnsadmin 的值应类似于以下内容，PDB 名称为 VIS，数据库节点逻辑名称为 oebs-db01log 。
	
	`$ grep -i tns_admin $CONTEXT_FILE <TNS_ADMIN oa_var="s_db_tnsadmin">/rdsdbbin/oracle/network/admin/VIS_oebs-db01log</TNS_ADMIN> <CDB_TNS_ADMIN oa_var="s_cdb_tnsadmin">/rdsdbbin/oracle/network/admin</CDB_TNS_ADMIN>`
	
	3. 在数据库层运行 Autoconfig：
	
	`$ . $ORACLE_HOME/VISCDB_oebs-db01log.env $ export ORACLE_PDB_SID=VIS $ sqlplus "/ as sysdba" @$ORACLE_HOME/appsutil/admin/adgrants.sql APPS $ sqlplus "/ as sysdba" @$ORACLE_HOME/rdbms/admin/utlrp.sql`
	
	`$ . $ORACLE_HOME/VIS_oebs-db01log.env $ echo $ORACLE_SID VIS`	

Task	描述	所需技能
	`$ cd $ORACLE_HOME/appsutil/scripts/$CONTEXT_NAME` `$ ./adautocfg.sh`	

Task	描述	所需技能

为 rdsdb 用户设置环境。	对于 Oracle 19c，请跳过此步骤。
	
	对于 Oracle 12.1.0.2：
	
	现在您已经完成了 Autoconfig 和 TNS 条目，您需要通过在 `rdsdb` 用户配置文件中设置环境文件来加载环境文件。
	
	更新 `.bash_profile` 以调用 Oracle 电子商务套件数据库 `.env` 文件。您需要更新配置文件以确保环境已加载。此环境文件是在您之前运行 Autoconfig 时创建的。
	
	以下示例环境文件是在您运行 Autoconfig 时创建的：
	
	`. /rdsdbbin/oracle/VIS_oebs-db01log.env`
	
	作为 `rdsdb` 用户：
	
	`cd $HOME`
	`vi .bash_profile`
	`export LD_LIBRARY_PATH=${ORACLE_HOME}/lib:${ORACLE_HOME}/ctx/lib`
	`export SHLIB_PATH=${ORACLE_HOME}/lib`
	`export PATH=$PATH:${ORACLE_HOME}/bin`
	`alias sql='rlwrap -c sqlplus / as sysdba'`
	`. ${ORACLE_HOME}/VIS_oebs-db01log.env`
		数据库管理员

Task	描述	所需技能
	Note 对于 Oracle 19c，您无需将 CDB 环境加载到其中。`.bash_profile` 这是因为默认路径 `ORACLE_HOME` 设置为默认路径 `$ORACLE_HOME/network/admin` ，即 `rdsdb`（Oracle 主目录）用户的默认主目录。	

Task	描述	所需技能

为 Amazon RDS Custom 配置应用程序和数据库。	完成 Oracle 12.1.0.2 和 19c 的前两个步骤。每个版本的后续步骤都不同。
	
	1. 在应用程序层上，编辑 `/etc/hosts` 并将数据库的 IP 地址更改为 Amazon RDS Custom IP 地址：
	
	`xx.xx.xx.xx OEBS-db01.localdomain OEBS-db01 OEBS-db01log.localdomain OEBS-db01log`
	
	由于您使用的是逻辑主机名，因此几乎可以无缝地替换数据库节点。
	
	2. 在 Amazon RDS 自定义数据库实例上，添加或修改分配给源 EC2 实例的安全组以反映 Amazon RDS 自定义数据库实例，以确保应用程序可以访问该节点。
	
	对于 Oracle 12.1.0.2：
	
	3. 运行 Autoconfig。以应用程序所有者（例如 `applmgr`）的身份运行：
	
	`$ cd $INST_TOP/admin/scripts`
	`$ ./adautocfg.sh`
	`AutoConfig completed successfully.`
		数据库管理员

Task	描述	所需技能

	4. 验证 fnd_nodes 条目：
	
	`SQL> select node_name from apps.fnd_nodes`
	`NODE_NAME`
	`----------`
	`AUTHENTICATION`
	`OEBS-APP01LOG`
	`OEBS-DB01LOG`
	
	5. 确认您可以登录，然后启动应用程序：
	
	`$ ./adstrtal.sh`
	
	对于 Oracle 19c：
	
	1. 检查 PDB 是否已打开，必要时将其打开：
	
	`SQL> show pdbs`
	
	`CON_ID CON_NAME OPEN MODE RESTRICTED`
	`---------- ---------- ----------`
	`2 PDB$SEED READ ONLY NO`
	`3 VIS MOUNTED`
	
	`SQL> alter session set container=vis;`	

Task	描述	所需技能
	SQL> alter database open; SQL> alter database save state; 2. 测试 apps 的连接： SQL> sqlplus apps/**** @vis 3. 在数据库层运行 Autoconfig： $. $ORACLE_HOME/VIS_oebs-db01log.env $ echo $ORACLE_SID VIS $ cd $ORACLE_HOME/appsutil/scripts/$CONTEXT_NAME $ ./adautocfg.sh 4. 以应用程序所有者的身份在应用程序层上运行 Autoconfig（例如，applmgr）： $ cd $INST_TOP/admin/scripts $ ./adautocfg.sh AutoConfig completed successfully. 5. 验证 fnd_nodes 条目： SQL> select node_name from apps.fnd_nodes	

Task	描述	所需技能
	NODE_NAME ---------- ---------- ---------- ---------- ------ AUTHENTICATION OEBS-APP01LOG OEBS-DB01LOG 6. 启动应用程序： $ ./adstrtal.sh	

执行迁移后步骤

Task	描述	所需技能
恢复自动化以确认其正常工作。	使用以下 AWS CLI 命令恢复自动化： aws rds modify-db-instance \ --db-instance-identifier vis \ --automation-mode full \ 该数据库现在由 Amazon RDS Custom 管理。例如，如果侦听器或数据库出现故障，Amazon RDS Custom 代理将重新启动它们。请运行以下命令进行测试。 停止侦听器示例：	数据库管理员

Task	描述	所需技能

	`-bash-4.2$ lsnrctl stop vis`
	
	关闭数据库示例：
	
	`SQL> shutdown immediate ;`	

验证架构、连接和维护任务。	要完成迁移，您必须至少执行以下任务。 • 运行 FS_CLONE 以同步补丁文件系统。 • 收集架构统计信息。 • 确保外部接口和系统可以连接到新的 Amazon RDS Custom 数据库。 • 设置备份和维护计划。 • 通过发出割接来切换文件系统，验证 AD Online Patching（ADOP）是否按预期运行。	数据库管理员

故障排除

事务	解决方案
当您尝试删除日志文件时，您会收到 ORA-01624 错误。	如果您在尝试删除日志文件时会收到 ORA-01624 错误，按照这些步骤进行操作。 发出以下命令并等到要删除的日志文件的状态变为 INACTIVE。有关 V$log 中状态代码的更多信息，请参阅 Oracle 文档。以下是示例命令及其输出：

事务	解决方案
	SQL> select group#, status from v$log; GROUP# STATUS ---------- ---------- 1 ACTIVE 2 CURRENT 3 UNUSED 4 UNUSED 5 UNUSED 6 UNUSED 6 rows selected. 在此示例中，日志文件 1 是 ACTIVE，因此您必须强制切换日志文件三次，以确保之前添加的第一个新日志文件的状态为 CURRENT： SQL> alter system switch logfile; System altered. SQL> alter system switch logfile; System altered. SQL> alter system switch logfile; System altered. 等到要删除的所有日志文件均变为 INACTIVE（如以下示例所示），然后运行该 DROP LOGFILE 命令。 SQL> select group#, status from v$log; GROUP# STATUS ---------- ---------- 1 INACTIVE 2 INACTIVE 3 INACTIVE 4 CURRENT 5 UNUSED 6 UNUSED 6 rows selected.

事务	解决方案

使用 `resetlogs` 打开数据库时，您会收到 ORA-00392 错误。	如果您收到错误 ORA-00392：正在清除线程 1 的日志 xx，不允许操作，请运行以下命令（xx 替换为日志文件号），然后重新运行 open `resetlogs` 命令：
	`SQL> alter database clear logfile group xx;`
	`SQL> alter database open resetlogs;`

事务	解决方案

您在使用 系统管理员 或应用程序用户连接到应用程序时遇到问题。	要确认问题，请运行以下 SQL 查询：
	
	`SQL> select dbms_java.get_jdk_ version() from dual; select dbms_java.get_jdk_version() from dual ERROR at line 1: ORA-29548: Java system class reported: release of Java system classes in the database (19.0.0.0.220719 1.8) does not match that of the oracle executabl e (19.0.0.0.0 1.8)`
	
	根本原因：源数据库应用了多个补丁，但是 Amazon RDS Custom DB_HOME 是新安装的，或者 CEV 未包含所有补丁，因为您在创建 CEV 时没有使用必要的 RSU 补丁，例如 OJVM。要验证这一点，请检查 `$ORACLE_HOME/sqlpa th` 、`$ORACLE_HOME/.patch_storage` 和 `opatch – lsinventory` 中是否列出了源补丁的详细信息。
	
	参考：datapatch -verbose 失败并出现错误：“ Patch xxxxx：存档的补丁目录为空”（文档 ID 2235541.1 ）
	
	修复：将缺失的补丁相关文件从源代码（`$ORACLE_HOME/sqlpatch/` ）复制到 Amazon RDS Custom（`$ORACLE_HOME/ sqlpatch/` ），然后重新运行 `./datapat ch -verbose` 。
	
	例如：
	
	`-bash-4.2$ cp -rp 18793246 20204035 20887355 22098146 22731026 $ORACLE_H OME/sqlpatch/`

事务	解决方案

	或者，您可以在 CDB 和 PDB 上运行以下命令来使用变通方法：
	`@?/javavm/install/update_javavm_db.sql`
	然后在 PDB 上运行以下命令：
	`sql> alter session set container=vis;`
	`@?/javavm/install/update_javavm_db.sql`
	现在再次运行测试：
	`SQL> select dbms_java.get_jdk_version() from dual;`

相关资源

- 使用 Amazon RDS Custom（Amazon RDS 文档）
- 适用于 Oracle 的 Amazon RDS Custom – 数据库环境中的新控制功能（AWS 新闻博客）
- 将适用于 Oracle 的 Amazon RDS Custom 与 Amazon EFS 集成（AWS 数据库博客）
- 在 AWS 上迁移 Oracle 电子商务套件（AWS 白皮书）
- AWS 上的 Oracle 电子商务套件架构（AWS 白皮书）
- 使用有效备用数据库为 Amazon RDS Custom 上的 Oracle 电子商务套件设置 HA/DR 架构（AWS Prescriptive Guidance）

其他信息

维护操作

使用新补丁修补 Oracle 电子商务套件数据库主页

由于 bin volume (`/rdsdbbin`) 是 out-of-place升级版，因此在 CEV 升级期间，bin 卷中的内容会被丢弃。因此，在使用 CEV 执行任何升级之前，必须创建该 `appsutil` 目录的副本。

在 Amazon RDS Custom 源实例上，在升级 CEV 之前，请备份 `$ORACLE_HOME/appsutil`。

> **Note**
>
> 此示例使用 NFS 卷。但是，您可以改为使用副本 Amazon Simple Storage Service（Amazon S3）。

1. 创建用于在 Amazon RDS Custom 源实例上存储 appsutil 的目录：

```
$ mkdir /RMAN/appsutil.preupgrade
```

2. 压缩并复制到 Amazon EFS 卷：

```
$ tar cvf /RMAN/appsutil.preupgrade appsutil
```

3. 验证 tar 文件是否存在：

```
$ bash-4.2$ ls -l /RMAN/appsutil.preupgrade
-rw-rw-r-- 1 rdsdb rdsdb 622981120 Feb  8 20:16 appsutil.tar
```

4. 按照 Amazon RDS 文档中升级 RDS Custom 数据库实例中的说明升级到最新的 CEV（已创建必备的 CEV）。

您也可以通过使用 OPATCH 直接进行修补。请参阅 Amazon RDS 文档的 适用于 Oracle 的 RDS Custom 的升级要求和注意事项部分。

> **Note**
>
> 在 CEV 修补过程中，主机的 IP 地址不会更改。此过程执行 out-of-place升级，并在启动期间将新的 bin 卷附加到同一个实例上。

将 Oracle 迁移 PeopleSoft 到亚马逊 RDS 定制版

由 Gaurav Gupta (AWS) 编写

摘要

Oracle PeopleSoft 是一款适用于企业级流程的企业资源规划 (ERP) 解决方案。 PeopleSoft 具有三层架构：客户端、应用程序和数据库。 PeopleSoft 可以在亚马逊关系数据库服务 (Amazon RDS) 上运行。现在，您还可以 PeopleSoft 在 Amazon RDS Custom 上运行，它提供对底层操作系统的访问权限。

适用于 Oracle 的 Amazon RDS Custom 是一项托管式数据库服务，适用于需要访问底层操作系统和数据库环境的旧版、自定义和打包应用程序。当您将 Oracle 数据库迁移到 Amazon RDS Custom 时，Amazon Web Services (AWS) 可以管理备份任务和高可用性，而您可以专注于维护 PeopleSoft 应用程序和功能。有关迁移时需要考虑的关键因素，请参阅 AWS Prescriptive Guidance 中的 Oracle 数据库迁移策略。

此模式侧重于使用 Oracle Recovery Manager (RMAN) 备份将亚马逊弹性计算云 (Amazon EC2) 上的 PeopleSoft 数据库迁移到 Amazon RDS Custom 的步骤。它在 EC2 实例和 Amazon RDS Custom 之间使用亚马逊弹性文件系统 (Amazon EFS) 共享文件系统，但您也可以使用亚马逊 FSx 或任何共享云端硬盘。该模式使用 RMAN 完整备份（有时也称为 0 级备份）。

先决条件和限制

先决条件

- 在亚马逊上运行的 Oracle 19C 版本源数据库， EC2 搭载 Oracle Linux 7、Oracle Linux 8、红帽企业 Linux (RHEL) 7 或 RHEL 8。在此模式的示例中，源数据库名称为 `FSDM092`，但这不是必需的。

> **Note**
>
> 您也可以将此模式用于本地 Oracle 源数据库。您必须在本地网络和虚拟私有云（VPC）之间建立适当的网络连接。

- 一个 PeopleSoft 9.2 的演示实例。
- 单个 PeopleSoft 应用程序层。但是，您可以调整此模式以使用多个应用程序层。
- Amazon RDS Custom 配置了至少 8 GB 的交换空间。

限制

此模式不支持以下配置：

- 将数据库 ARCHIVE_LAG_TARGET 参数设置为 60–7200 范围之外的值
- 禁用数据库实例日志模式 (NOARCHIVELOG)
- 关闭实例的亚马逊 Elastic Block Store (Amazon EBS) 优化属性 EC2
- 修改连接到实例的原始 EBS 卷 EC2
- 添加新的 EBS 卷或将卷类型从 gp2 更改为 gp3
- 更改 LOG_ARCHIVE_FORMAT 参数的扩展名格式（需要 *.arc）
- 多路复用或更改控制文件位置和名称（必须为 /rdsdbdata/db/*DBNAME*/controlfile/control-01.ctl）

有关这些配置和其他不支持的配置的更多信息，请参阅 Amazon RDS 文档。

产品版本

有关 Amazon RDS Custom 支持的 Oracle Database 版本和实例类型，请参阅 Amazon RDS Custom for Oracle 的要求和限制

架构

目标技术堆栈

- 应用程序负载均衡器
- Amazon EFS
- 适用于 Oracle 的 Amazon RDS Custom
- AWS Secrets Manager
- Amazon Simple Storage Service（Amazon S3）

目标架构

以下架构图表示在 AWS 的单个可用区中运行的 PeopleSoft 系统。应用程序层可通过应用程序负载均衡器进行访问。应用程序和数据库都位于私有子网中，Amazon RDS Custom 和 Amazon EC2 数据库实例使用 Amazon EFS 共享文件系统来存储和访问 RMAN 备份文件。Amazon S3 用于创建自定义 RDS Oracle 引擎和存储重做日志元数据。

工具

工具

Amazon Web Services

- 适用于 Oracle 的 Amazon RDS Custom 是一项托管式数据库服务，适用于需要访问底层操作系统和数据库环境的旧版、自定义和打包应用程序。它能自动执行数据库管理任务，如备份和高可用性。
- Amazon Elastic File System (Amazon EFS) 可帮助您在 Amazon Web Services Cloud 中创建和配置共享文件系统。此模式使用 Amazon EFS 共享文件系统来存储和访问 RMAN 备份文件。
- AWS Secrets Manager 帮助您将代码中的硬编码凭证（包括密码）替换为对 Secrets Manager 的 API 调用，以便以编程方式检索密钥。在这种模式中，您可以从 Secrets Manager 检索数据库用户密码来创建 `RDSADMIN` 和 `ADMIN` 用户以及更改 `sys` 和 `system` 密码。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- 弹性负载均衡（ELB）将传入的应用程序或网络流量分配到多个目标。例如，您可以跨亚马逊弹性计算云 (Amazon EC2) 实例、容器以及一个或多个可用区中的 IP 地址分配流量。此模式使用应用程序负载均衡器。

其他工具

- Oracle Recovery Manager (RMAN) 为 Oracle 数据库提供备份和恢复支持。此模式使用 RMAN 对亚马逊上的源 Oracle 数据库执行热备份，该数据库在 Amazon EC2 RDS Custom 上还原。

最佳实践

- 对于数据库初始化参数，请自定义 Amazon RDS 自定义数据库实例为其提供的标准 pfile，PeopleSoft 而不是使用 Oracle 源数据库中的 spfile。这是因为在 Amazon RDS Custom 中创建只读副本时，空格和评论会导致问题。有关数据库初始化参数的更多信息，请参阅 Oracle Support Note 1100831.1（需要 Oracle 支持账户）。
- 默认情况下，Amazon RDS Custom 使用 Oracle 自动内存管理。如果您想使用 Hugemem 内核，可以将 Amazon RDS Custom 配置为改用自动共享内存管理。
- 默认情况下启用 `memory_max_target` 参数。框架会在后台使用它来创建只读副本。
- 启用 Oracle 闪回数据库。此功能在故障转移（而非切换）测试场景中恢复备用数据库时非常有用。

操作说明

设置数据库实例和文件系统

Task	描述	所需技能
创建数据库实例。	在 Amazon RDS 控制台中，使用名为 FSDMO92（或您的源数据库名称）的数据库名称创建适用于 Oracle 的 Amazon RDS 自定义数据库实例。 有关说明，请参阅 AWS 文档中的使用 Amazon RDS Custom 和博客文章Amazon RDS Custom for Oracle – 数据库环境中的新控制功能。这样可以确保将数据库名称设置为与源数据库相同的名称。（如果保留为空，则 EC2 实例和数据库名称将设置为ORCL。）	数据库管理员

对源 Amazon EC2 数据库执行 RMAN 完整备份

Task	描述	所需技能
创建备份脚本。	创建 RMAN 备份脚本以将数据库备份到您安装的 Amazon EFS 文件系统（以下示例中为 `/efs`）。您可以使用示例代码或运行一个现有的 RMAN 脚本。 `#!/bin/bash` `Dt=`date +'%Y%m%d-%H%M'` `BACKUP_LOG="rman-${ORACLE_SID}-$Dt"`	数据库管理员

Task	描述	所需技能
	export TAGDATE=`date + %Y%m%d%H%M`; LOGPATH=/u01/scripts/l ogs rman target / >> $LOGPATH/rman-${OR ACLE_SID}-$Dt << EOF SQL "ALTER SYSTEM SWITCH LOGFILE"; SQL "ALTER SESSION SET NLS_DATE_FORMAT="D D.MM.YYYY HH24:MI:S S""; RUN { ALLOCATE CHANNEL ch11 TYPE DISK MAXPIECESIZE 5G; ALLOCATE CHANNEL ch12 TYPE DISK MAXPIECESIZE 5G; BACKUP AS COMPRESSED BACKUPSET FULL DATABASE FORMAT '/efs/rma n_backup/FSCM/%d_%T_ %s_%p_FULL' ; SQL "ALTER SYSTEM ARCHIVE LOG CURRENT"; BACKUP FORMAT '/efs/ rman_backup/FSCM/%d_ %T_%s_%p_ARCHIVE' ARCHIVELOG ALL DELETE ALL INPUT ; BACKUP CURRENT CONTROLFILE FORMAT '/ efs/rman_backup/FSCM/ %d_%T_%s_%p_CONTROL'; } EXIT; EOF	

Task	描述	所需技能

运行备份脚本。	要运行 RMAN 备份脚本，请以 Oracle 主用户身份登录，然后运行该脚本。
	
	`$  chmod a+x rman_backup.sh`
	`$ ./rman_backup.sh &`
		数据库管理员

Task	描述	所需技能

检查是否存在错误，并记下备份文件的名称。	检查 RMAN 日志文件中的错误。如果一切正常，请运行以下命令列出控制文件的备份。
	`RMAN> list backup of controlfile;`
	`using target database control file instead of recovery catalog`
	记下输出文件的名称。
	`List of Backup Sets`
	`==========`
	`BS Key Type LV Size Device Type Elapsed Time Completion Time`
	`12 Full 21.58M DISK 00:00:01 13-JUL-22`
	`BP Key: 12 Status: AVAILABLE Compressed: NO Tag: TAG20220713T150155`
	`Piece Name: /efs/rman_backup/FSCM/FSDMO92_20220713_12_1_CONTROL`
	`Control File Included: Ckp SCN: 16559159985898 Ckp time: 13-JUL-22`
		数据库管理员

Task	描述	所需技能
	当您在 Amazon RDS Custom 上还原数据库时，您将使用备份控制文件 `/efs/rman_backup/FSCM/FSDMO92_20220713_12_1_CONTROL` 。	

关闭源应用程序层

Task	描述	所需技能

关闭应用程序。	若要关闭源应用程序层，请使用 `psadmin` 实用程序或 `psadmin` 命令行实用程序。
	1. 若要关闭 Web 服务器，请运行以下命令。
	`psadmin -w shutdown -d "webserver domain name"`
	2. 若要关闭应用程序服务器，请运行以下命令。
	`psadmin -c shutdown -d "application server domain name"`
	3. 若要关闭进程调度程序，请运行以下命令。
	`psadmin -p stop -d "process scheduler domain name"`
		数据库管理员、管理员 PeopleSoft

配置目标 Amazon RDS Custom 数据库

Task	描述	所需技能
安装 nfs-utils rpm 包。	要安装 `nfs-utils rpm` 程序包，请运行以下命令。 `$ yum install -y nfs-utils`	数据库管理员
附加 EFS 存储。	从 Amazon EFS 控制台页面获取 Amazon EFS 附加命令。使用网络文件系统 (NFS) 客户端在 Amazon RDS 实例上附加 EFS 文件系统。 `sudo mount -t nfs4 -o nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport fs-xxxxxxxxxx.efs.eu-west-1.amazonaws.com:/ /efs` `sudo mount -t nfs4 -o nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport fs-xxxxxxxxxx.efs.eu-west-1.amazonaws.com:/ /efs`	数据库管理员

删除起始数据库并创建用于存储数据库文件的目录

Task	描述	所需技能
暂停自动化模式。	在继续执行后续步骤之前，您必须暂停 Amazon RDS Custom 数据库实例的自动化模式，以确保自动化不会干扰 RMAN 恢复活动。 您可以使用 AWS 管理控制台或 AWS 命令行界面（AWS CLI）命令暂停自动化（请确保您已先配置 AWS CLI）。 `aws rds modify-db-instance \ --db-instance-id entifier peoplesoft-fscm-92 \ --automation-mode all-paused \ --resume-full-au tomation-mode-minute 360 \ --region eu-west-1` 当您指定暂停的持续时间时，请确保为 RMAN 恢复留出足够的时间。这取决于源数据库的大小，因此请相应地修改 360 值。 另外，请确保暂停自动化的总时间不与数据库的备份或维护时段重叠。	数据库管理员
创建和修改的参数文件 PeopleSoft	要创建和修改的 pfile PeopleSoft，请使用使用	数据库管理员

Task	描述	所需技能

	Amazon RDS 自定义数据库实例创建的标准 pfile。添加你需要的参数 PeopleSoft。
	
	1. 运行以下命令切换到 `rdsuser rdsdb`。
	
	``
	$ sudo su - rdsdb
	``
	
	2. 登录到起始数据库上的 SQL*Plus，然后运行以下命令创建 pfile。
	
	``
	SQL> create pfile
	 from spfile;
	``
	
	这将在 `$ORACLE_HOME/dbs` 中创建 pfile。
	
	3. 备份此 pfile。
	
	4. 编辑 pfile 以添加或更新 PeopleSoft参数。
	
	``
	*._gby_hash_aggreg
	ation_enabled=false
	
	*._unnest_subqu
	ery=false
	
	*.nls_language='
	AMERICAN'
	
	*.nls_length_sem
	antics='CHAR'
	
	*.nls_territ
	ory='AMERICA'
	
	*.open_cursors=1000
	``	

Task	描述	所需技能
	``` *.db_files=1200 *.undo_tablespace=' UNDOTBS1' ``` PeopleSoft 相关参数可以在 Oracle Supp ort Note 1100831.1 中找到。 5. 删除 pfile 中的 spfile 引用。 ``` *.spfile='/rdsdbbi n/oracle/dbs/spfil eFSDMO92.ora' ```	

删除起始数据库。	若要删除现有的 Amazon RDS Custom 数据库，请使用以下代码。
	``
	$ sqlplus / as sysdba
	SQL> shutdown immediate
	;
	SQL> startup mount
	 exclusive restrict;
	SQL> drop database;
	SQL> exit
	``	

Task	描述	所需技能
从备份中恢复 Amazon RDS Custom 数据库。	使用以下脚本恢复数据库。该脚本将首先恢复控制文件，然后从存储在 EFS 附加上的备份片段恢复整个数据库。 `#!/bin/bash` `Dt=`date +'%Y%m%d-%H%M'` `BACKUP_LOG="rman-${ORACLE_SID}-$Dt"` `export TAGDATE=`date +%Y%m%d%H%M`;` `LOGPATH=/rdsdbdata/scripts/logs` `rman target / >> $LOGPATH/rman-${ORACLE_SID}-$Dt << EOF` `restore controlfile from "/efs/rman_backup/FSCM/FSDMO92_20220713_12_1_CONTROL";` `alter database mount;` `run` `{` `set newname for database to '/rdsdbdata/db/FSDMO92_A/datafile/%f_%b';` `SET NEWNAME FOR TEMPFILE 1 TO '/rdsdbdata/db/FSDMO92_A/datafile/%f_%b';` `RESTORE DATABASE;` `SWITCH DATAFILE ALL;` `SWITCH TEMPFILE ALL;` `RECOVER DATABASE;` `}` `EOF`	数据库管理员

Task	描述	所需技能
	sqlplus / as sysdba >> $LOGPATH/rman-${ORACLE_SID}-$Dt<<-EOF ALTER DATABASE RENAME FILE '/u01/psoft/db/oradata/FSDMO92/redo01.log' TO '/rdsdbdata/db/FSDMO92_A/onlinelog/redo01.log'; ALTER DATABASE RENAME FILE '/u01/psoft/db/oradata/FSDMO92/redo02.log' TO '/rdsdbdata/db/FSDMO92_A/onlinelog/redo02.log'; ALTER DATABASE RENAME FILE '/u01/psoft/db/oradata/FSDMO92/redo03.log' TO '/rdsdbdata/db/FSDMO92_A/onlinelog/redo03.log'; alter database clear unarchived logfile group 1; alter database clear unarchived logfile group 2; alter database clear unarchived logfile group 3; alter database open resetlogs; EXIT EOF	

从 Secrets Manager 中检索密码，创建用户和更改密码

Task	描述	所需技能
从 Secrets Manager 中检索密码。	您可以使用 AWS 管理控制台或 AWS CLI 执行此步骤。以下步骤显示了控制台的说明。 1. 登录 AWS 管理控制台并打开 Amazon RDS 控制台。 2. 在导航窗格中，选择数据库，然后选择 Amazon RDS 数据库。 3. 选择配置并记下实例的资源 ID。这将采用 db-<ID> 形式（例如，db-73GJNHLGDNZNDOXNWXSECUW6LE ）。 4. 打开 Secrets Manager 控制台。 5. 选择与 do-not-delete-custom-<resource_id> 同名的密钥，其中 resource-id 是指您在步骤 3 中记下的资源 ID。 6. 选择 Retrieve secret value (检索密钥值)。 此密码对于 sys、system、rdsadmin 和 admin 用户相同。	数据库管理员
创建 RDSADMIN 用户。	RDSADMIN 是用于监控和编排 Amazon RDS Custom 数据库实例的数据库用户。由于起始数据库已被删除，并且目标数据库已使用 RMAN 从源恢复，	数据库管理员

Task	描述	所需技能

	因此您必须在恢复操作后重新创建此用户，以确保 Amazon RDS Custom 监控能按预期运行。您还必须为 RDSADMIN 用户创建单独的配置文件和表空间。
	
	1. 在 SQL 提示符中输入以下命令。
	
	`SQL> set echo on feedback on serverout on`
	`SQL> @?/rdbms/admin/utlpwdmg.sql`
	`SQL> ALTER PROFILE DEFAULT LIMIT FAILED_LOGIN_ ATTEMPTS UNLIMITED PASSWORD_LIFE_TIME UNLIMITED PASSWORD_VERIFY_F UNCTION NULL;`
	
	2. 创建配置文件 RDSADMIN。
	
	`SQL> set echo on feedback on serverout on`
	`SQL> alter session set "_oracle_script"=t rue;`
	`SQL> CREATE PROFILE RDSADMIN LIMIT COMPOSITE_LIMIT UNLIMITED SESSIONS_PER_USER UNLIMITED`	

Task	描述	所需技能
	CPU_PER_SESSION UNLIMITED CPU_PER_CALL UNLIMITED LOGICAL_READS_PER _SESSION UNLIMITED LOGICAL_READS_PER _CALL UNLIMITED IDLE_TIME UNLIMITED CONNECT_TIME UNLIMITED PRIVATE_SGA UNLIMITED FAILED_LOGIN_ATTE MPTS 10 PASSWORD_LIFE_TIME UNLIMITED PASSWORD_REUSE_TIME UNLIMITED PASSWORD_REUSE_MAX UNLIMITED PASSWORD_VERIFY_F UNCTION NULL PASSWORD_LOCK_TIME 86400/86400 PASSWORD_GRACE_TIME 604800/86400; 3. 创建 RDSADMIN 表空间。 SQL> CREATE BIGFILE TABLESPACE rdsadmin '/rdsdbdata/db/FSD MO92_A/datafile/rd sadmin.dbf' DATAFILE SIZE 7M AUTOEXTEND ON NEXT 1m LOGGING ONLINE PERMANENT BLOCKSIZE 8192 EXTENT MANAGEMEN T LOCAL AUTOALLOCATE DEFAULT NOCOMPRES	

Task	描述	所需技能
	S SEGMENT SPACE MANAGEMENT AUTO; 4. 创建 RDSADMIN 用户。将 RDSADMIN 密码替换为您之前从 Secrets Manager 获取的密码。 SQL> CREATE USER rdsadmin IDENTIFIED BY xxxxxxxxxx DEFAULT TABLESPACE rdsadmin TEMPORARY TABLESPACE TEMP profile rdsadmin ; 5. 向 RDSADMIN 授予权限。 SQL> GRANT "CONNECT" TO RDSADMIN WITH ADMIN OPTION; SQL> GRANT "RESOURCE " TO RDSADMIN WITH ADMIN OPTION; SQL> GRANT "DBA" TO RDSADMIN; SQL> GRANT "SELECT_C ATALOG_ROLE" TO RDSADMIN WITH ADMIN OPTION; SQL> GRANT ALTER SYSTEM TO RDSADMIN; SQL> GRANT UNLIMITED TABLESPACE TO RDSADMIN; SQL> GRANT SELECT ANY TABLE TO RDSADMIN; SQL> GRANT ALTER DATABASE TO RDSADMIN;	

Task	描述	所需技能
	SQL> GRANT ADMINISTER DATABASE TRIGGER TO RDSADMIN; SQL> GRANT ANY OBJECT PRIVILEGE TO RDSADMIN WITH ADMIN OPTION; SQL> GRANT INHERIT ANY PRIVILEGES TO RDSADMIN; SQL> ALTER USER RDSADMIN DEFAULT ROLE ALL; 6. Set the SYS, SYSTEM, and DBSNMP user profiles to RDSADMIN. SQL> set echo on feedback on serverout on SQL> alter user SYS profile RDSADMIN; SQL> alter user SYSTEM profile RDSADMIN; SQL> alter user DBSNMP profile RDSADMIN;	
创建主用户。	由于起始数据库已被删除，并且目标数据库已使用 RMAN 从源中恢复，因此您必须重新创建主用户。在此示例中，主用户名为 admin。 SQL> create user admin identified by <password>; SQL> grant dba to admin	数据库管理员

Task	描述	所需技能

更改系统密码。	使用您从 Secrets Manager 中检索到的密码更改系统密码。
	
	`SQL> alter user sys identified by xxxxxxxxxxx;`
	`SQL> alter user system identified by xxxxxxxxxx;`
	
	如果您不更改这些密码，Amazon RDS Custom 会显示错误消息：“数据库监控用户或用户凭证已更改。”
		数据库管理员

配置 Amazon RDS 自定义的 TNS 条目和 PeopleSoft

Task	描述	所需技能

配置 tnsnames 文件。	若要从应用程序层连接到数据库，请配置 `tnsnames.ora` 文件，以便您可以从应用程序层连接到数据库。在以下示例中，您可以看到有一个指向 `tnsnames.ora` 文件的软链接，但该文件默认为空。
	
	`$ cd /rdsdbbin/oracle/network/admin`
	`$ ls -ltr`
	`-rw-r--r-- 1 rdsdb database 1536 Feb 14 2018 shrept.lst`
	`lrwxrwxrwx 1 rdsdb database 30 Apr 5 13:19 listener.ora -`
		数据库管理员

Task	描述	所需技能
	> /rdsdbdata/config/ listener.ora lrwxrwxrwx 1 rdsdb database 28 Apr 5 13:19 sqlnet.ora - > /rdsdbdata/config/ sqlnet.ora lrwxrwxrwx 1 rdsdb database 30 Apr 5 13:19 tnsnames.ora - > /rdsdbdata/config/ tnsnames.ora 1. 创建 tsnames.ora 条目。由于 Amazon RDS 自动化解析文件的方式，您必须确保该条目不包含任何空格、评论或多余的行。否则，在使用其中一些（例如 create-db-instance-read-replic a APIs）时可能会遇到问题。 2. 根据您的 PeopleSoft 数据库要求更换端口、主机和 SID。使用以下代码作为示例。 $ vi tnsnames.ora FSDMO92=(DESCRIP TION = (ADDRESS_ LIST = (ADDRESS = (PROTOCOL = TCP)(HOST = x.x.x.x)(PORT = 1521))) (CONNECT_ DATA = (SERVER = DEDICATED) (SID = FSDMO92)))	

Task	描述	所需技能

	3. 要确认可以 PeopleSoft 访问数据库，请运行以下命令。
	
	`$ tnsping FSDMO92`
	
	`TNS Ping Utility for Linux: Version 19.0.0.0.0 - Production on 14-JUL-2022 10:16:45`
	
	`Copyright (c) 1997, 2021, Oracle. All rights reserved.`
	
	`Used parameter files: /rdsdbbin/oracle/network/admin/sqlnet.ora`
	
	`Used TNSNAMES adapter to resolve the alias Attempting to contact (DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST = x.x.x.x)(PORT = 1521))) (CONNECT_DATA = (SERVER = DEDICATED) (SID = FSDMO92)))`
	`OK (0 msec)`	

创建 spfile 软链接

Task	描述	所需技能

创建 spfile 软链接。	1. 若要在位置 `/rdsdbdata/admin/FSDMO92/pfile` 中创建 spfile，请运行以下命令。
	`SQL> create spfile='/rdsdbdata/admin/FSDMO92/pfile/spfileFSDMO92.ora' from pfile;`
	2. 导航到 `$ORACLE_HOME/dbs`，然后为 spfile 创建软链接。
	`ln -s '/rdsdbdata/admin/FSDMO92/pfile/spfileFSDMO92.ora' spfileFSDMO92.ora`
	3. 创建此文件后，您可以使用 spfile 关闭和启动数据库。
		数据库管理员

执行迁移后步骤

Task	描述	所需技能
验证架构、连接和维护任务。	若要完成迁移，请执行以下任务。 • 收集架构统计信息。 • 确保 PeopleSoft 应用程序层可以连接到新的 Amazon RDS 自定义数据库。	数据库管理员

Task	描述	所需技能
	• 设置备份和维护计划。	

相关的资源

- 使用 Amazon RDS Custom
- 适用于 Oracle 的 Amazon RDS Custom — 数据库环境中的新控制功能（博客文章）
- 将适用于 Oracle 的 Amazon RDS Custom 与 Amazon EFS 集成（博客文章）
- 将 Amazon RDS 配置为 Oracle PeopleSoft 数据库（AWS 白皮书）

将 Oracle ROWID 功能迁移到 AWS 上的 PostgreSQL

由 Rakesh Raghav (AWS) 和 Ramesh Pathuri (AWS) 编写

摘要

此模式描述了将甲骨文数据库中的ROWID伪列功能迁移到适用于 PostgreSQL 的亚马逊关系数据库服务（亚马逊 RDS）、兼容亚马逊 Aurora PostgreSQL 的版本或亚马逊弹性计算云（亚马逊）中的 PostgreSQL 数据库的选项。 EC2

在 Oracle 数据库中，ROWID 伪列是表中某一行的物理地址。即使表中不存在主键，该伪列也用于唯一标识行。PostgreSQL 有一个名为 ctid 的类似伪列，但它不能用作 ROWID。正如 PostgreSQL 文档中所述，ctid 可能会在更新后或每次 VACUUM 进程后发生变化。

您可以通过三种方式在 PostgreSQL 中创建 ROWID 伪列功能：

- 使用主键列代替 ROWID 来标识表中的一行。
- 在表中使用逻辑主键/唯一键（可能是复合键）。
- 添加一个带有自动生成值的列，并使其成为模拟 ROWID 的主键/唯一键。

此模式将引导您完成所有三种实现，并描述每个选项的优缺点。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 程序Language/PostgreSQL (PL/pgSQL）编码专业知识
- 源 Oracle 数据库
- 兼容 Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 的集群，或者托管 PostgreSQL 数据库的实 EC2 例

限制

- 此模式为 ROWID 功能提供了解决方法。PostgreSQL 不提供与 Oracle 数据库中的 ROWID 等效的项。

产品版本

- PostgreSQL 11.9 或更高版本

架构

源技术堆栈

- Oracle Database

目标技术堆栈

- 兼容 Aurora PostgreSQL、适用于 PostgreSQL 的亚马逊 RDS 或者带有 PostgreSQL 数据库的实例 EC2

实施选项

有三个选项可以解决 PostgreSQL 中缺少 ROWID 支持的问题，具体取决于您的表是否具有主键或唯一索引、逻辑主键或标识属性。您的选择取决于您的项目时间表、当前的迁移阶段以及对应用程序和数据库代码的依赖关系。

选项	描述	优点	劣势
主键或唯一索引	如果您的 Oracle 表具有主键，您可以使用该键的属性来唯一标识一行。	• 不依赖于专有数据库功能。 • 对性能的影响最小，因为主键字段已编制索引。	• 需要更改依赖于 ROWID 的应用程序和数据库代码才能切换到主键字段。
逻辑主键/唯一键	如果您的 Oracle 表具有逻辑主键，您可以使用该键的属性来唯一标识一行。逻辑主键由可以唯一标识行的一个属性或一组属性组成，但	• 不依赖于专有数据库功能。	• 需要更改依赖于 ROWID 的应用程序和数据库代码才能切换到主键字段。 • 如果未对逻辑主键的属性编制索引，则会对性能产生重

	不会通过约束在数据库上强制执行。		大影响。不过，您可以添加唯一索引来防止出现性能问题。
标识属性	如果您的 Oracle 表没有主键，您可以创建一个附加字段作为 `GENERATED ALWAYS AS IDENTITY`。每当将数据插入到表中时，此属性都会生成一个唯一值，因此它可用于唯一标识数据操作语言（DML）操作的行。	• 不依赖于专有数据库功能。 • PostgreSQL 数据库会填充该属性并保持其唯一性。	• 需要更改依赖于 `ROWID` 的应用程序和数据库代码才能切换到标识属性。 • 如果未对附加字段编制索引，则会对性能产生重大影响。不过，您可以添加一个索引来防止出现性能问题。

工具

- Amazon Relational Database Service（Amazon RDS）for PostgreSQL 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展 PostgreSQL 关系数据库。
- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩展 PostgreSQL 部署。
- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。在此模式中，您可以使用 AWS CLI 通过 pgAdmin 运行 SQL 命令。
- pgAdmin 是一种适用于 PostgreSQL 的开源管理工具。它提供了一个图形界面，可帮助您创建、维护和使用数据库对象。
- AWS Schema Conversion Tool（AWS SCT）通过自动将源数据库架构和大部分自定义代码转换为与目标数据库兼容的格式来支持异构数据库迁移。

操作说明

标识源表

Task	描述	所需技能
确定使用该ROWID属性的 Oracle 表。	使用 AWS Schema Conversion Tool (AWS SCT) 来识别具有 ROWID 功能的 Oracle 表。有关更多信息，请参阅 AWS SCT 文档。 —或者— 在 Oracle 中，使用 DBA_TAB_COLUMNS 视图来标识具有 ROWID 属性的表。这些字段可用于存储 10 字节的字母数字字符。确定用法，并将其转换为 VARCHAR 字段（如果适用）。	数据库管理员或开发人员
标识引用这些表的代码。	使用 AWS SCT 生成迁移评测报告来识别受 ROWID 影响的程序。有关更多信息，请参阅 AWS SCT 文档。 —或者— 在源 Oracle 数据库中，使用 dba_source 表的文本字段来标识使用 ROWID 功能的对象。	数据库管理员或开发人员

确定主键用法

Task	描述	所需技能
标识没有主键的表。	在源 Oracle 数据库中，使用 DBA_CONSTRAINTS 标识没有主键的表。这些信息将帮助您确定每个表的策略。例如： `select dt.* from dba_tables dt where not exists (select 1 from all_constraints ct where ct.owner = Dt.owner and ct.table_name = Dt.table_name and ct.constraint_type = 'P' ) and dt.owner = '{schema}'`	数据库管理员或开发人员

确定并应用解决方案

Task	描述	所需技能
对具有已定义主键或逻辑主键的表应用更改。	对其他信息部分中显示的应用程序和数据库代码进行更改，以使用唯一主键或逻辑主键来标识表中的行。	数据库管理员或开发人员
向没有定义主键或逻辑主键的表添加附加字段。	添加 GENERATED ALWAYS AS IDENTITY 类型的属性。对其	数据库管理员或开发人员

Task	描述	所需技能
	他信息部分中显示的应用程序和数据库代码进行更改。	
如有必要，请添加索引。	为附加字段或逻辑主键添加索引以提高 SQL 性能。	数据库管理员或开发人员

相关资源

- PostgreSQL CTID（PostgreSQL 文档）
- 生成的列（PostgreSQL 文档）
- ROWID 伪列（Oracle 文档）

其他信息

以下各节提供了 Oracle 和 PostgreSQL 代码示例来说明这三种方法。

场景 1：使用主唯一键

在以下示例中，您将创建表 `testrowid_s1`，并使用 `emp_id` 作为主键。

Oracle 代码：

```
create table testrowid_s1 (emp_id integer, name varchar2(10), CONSTRAINT testrowid_pk
 PRIMARY KEY (emp_id));
INSERT INTO testrowid_s1(emp_id,name) values (1,'empname1');
INSERT INTO testrowid_s1(emp_id,name) values (2,'empname2');
INSERT INTO testrowid_s1(emp_id,name) values (3,'empname3');
INSERT INTO testrowid_s1(emp_id,name) values (4,'empname4');
commit;

SELECT rowid,emp_id,name FROM testrowid_s1;
ROWID                  EMP_ID NAME
------------------ ---------- ----------
AAAF3pAAAAAAAMOAAA          1 empname1
AAAF3pAAAAAAAMOAAB          2 empname2
AAAF3pAAAAAAAMOAAC          3 empname3
AAAF3pAAAAAAAMOAAD          4 empname4

UPDATE testrowid_s1 SET name = 'Ramesh' WHERE rowid = 'AAAF3pAAAAAAAMOAAB' ;
```

```
commit;

SELECT rowid,emp_id,name FROM testrowid_s1;
ROWID                  EMP_ID NAME
------------------ ---------- ----------
AAAF3pAAAAAAAMOAAA          1 empname1
AAAF3pAAAAAAAMOAAB          2 Ramesh
AAAF3pAAAAAAAMOAAC          3 empname3
AAAF3pAAAAAAAMOAAD          4 empname4
```

PostgreSQL 代码：

```
CREATE TABLE public.testrowid_s1
(
    emp_id integer,
    name character varying,
    primary key (emp_id)
);

insert into public.testrowid_s1 (emp_id,name) values
(1,'empname1'),(2,'empname2'),(3,'empname3'),(4,'empname4');

select emp_id,name from testrowid_s1;
 emp_id |   name
--------+----------
      1 | empname1
      2 | empname2
      3 | empname3
      4 | empname4

update testrowid_s1 set name = 'Ramesh' where emp_id = 2 ;

select emp_id,name from testrowid_s1;
 emp_id |   name
--------+----------
      1 | empname1
      3 | empname3
      4 | empname4
      2 | Ramesh
```

场景 2：使用逻辑主键

在以下示例中，您将创建表 `testrowid_s2`，并使用 `emp_id` 作为逻辑主键。

Oracle 代码：

```
create table testrowid_s2 (emp_id integer, name varchar2(10) );
INSERT INTO testrowid_s2(emp_id,name) values (1,'empname1');
INSERT INTO testrowid_s2(emp_id,name) values (2,'empname2');
INSERT INTO testrowid_s2(emp_id,name) values (3,'empname3');
INSERT INTO testrowid_s2(emp_id,name) values (4,'empname4');
commit;

SELECT rowid,emp_id,name FROM testrowid_s2;
ROWID                  EMP_ID NAME
------------------ ---------- ----------
AAAF3rAAAAAAAMeAAA          1 empname1
AAAF3rAAAAAAAMeAAB          2 empname2
AAAF3rAAAAAAAMeAAC          3 empname3
AAAF3rAAAAAAAMeAAD          4 empname4

UPDATE testrowid_s2 SET name = 'Ramesh' WHERE rowid = 'AAAF3rAAAAAAAMeAAB' ;
commit;

SELECT rowid,emp_id,name FROM testrowid_s2;
ROWID                  EMP_ID NAME
------------------ ---------- ----------
AAAF3rAAAAAAAMeAAA          1 empname1
AAAF3rAAAAAAAMeAAB          2 Ramesh
AAAF3rAAAAAAAMeAAC          3 empname3
AAAF3rAAAAAAAMeAAD          4 empname4
```

PostgreSQL 代码：

```
CREATE TABLE public.testrowid_s2
(
    emp_id integer,
    name character varying
);

insert into public.testrowid_s2 (emp_id,name) values
(1,'empname1'),(2,'empname2'),(3,'empname3'),(4,'empname4');

select emp_id,name from testrowid_s2;
 emp_id |   name
--------+----------
      1 | empname1
```

```
      2 | empname2
      3 | empname3
      4 | empname4

update testrowid_s2 set name = 'Ramesh' where emp_id = 2 ;

select emp_id,name from testrowid_s2;
 emp_id |   name
--------+----------
      1 | empname1
      3 | empname3
      4 | empname4
      2 | Ramesh
```

场景 3：使用标识属性

在以下示例中，您将使用标识属性创建不带主键的表 testrowid_s3。

Oracle 代码：

```
create table testrowid_s3 (name varchar2(10));
INSERT INTO testrowid_s3(name) values ('empname1');
INSERT INTO testrowid_s3(name) values ('empname2');
INSERT INTO testrowid_s3(name) values ('empname3');
INSERT INTO testrowid_s3(name) values ('empname4');
commit;

SELECT rowid,name FROM testrowid_s3;
ROWID              NAME
------------------ ----------
AAAF3sAAAAAAAMmAAA empname1
AAAF3sAAAAAAAMmAAB empname2
AAAF3sAAAAAAAMmAAC empname3
AAAF3sAAAAAAAMmAAD empname4

UPDATE testrowid_s3 SET name = 'Ramesh' WHERE rowid = 'AAAF3sAAAAAAAMmAAB' ;
commit;

SELECT rowid,name FROM testrowid_s3;
ROWID              NAME
------------------ ----------
AAAF3sAAAAAAAMmAAA empname1
AAAF3sAAAAAAAMmAAB Ramesh
AAAF3sAAAAAAAMmAAC empname3
```

```
AAAF3sAAAAAAAMmAAD empname4
```

PostgreSQL 代码：

```
CREATE TABLE public.testrowid_s3
(
    rowid_seq bigint generated always as identity,
    name character varying
);

insert into public.testrowid_s3 (name) values
('empname1'),('empname2'),('empname3'),('empname4');

select rowid_seq,name from testrowid_s3;
 rowid_seq |   name
-----------+----------
         1 | empname1
         2 | empname2
         3 | empname3
         4 | empname4

update testrowid_s3 set name = 'Ramesh' where rowid_seq = 2 ;

select rowid_seq,name from testrowid_s3;
 rowid_seq |   name
-----------+----------
         1 | empname1
         3 | empname3
         4 | empname4
         2 | Ramesh
```

将 Oracle 数据库错误代码迁移到 Amazon Aurora PostgreSQL-Compatible 数据库

创建者：Sai Parthasaradhi（AWS）和 Veeranjaneyulu Grandhi（AWS）

摘要

此模式显示如何使用预定义的元数据表将 Oracle 数据库错误代码迁移到 Amazon Aurora PostgreSQL-Compatible Edition 数据库。

Oracle 数据库错误代码并不总是有相应的 PostgreSQL 错误代码。错误代码的这种差异会使您很难在目标 PostgreSQL 架构中配置过程或函数的处理逻辑。

您可以简化流程，方法是将对 PL/pgSQL 程序有意义的源数据库和目标数据库错误代码存储在元数据表中。然后，将该表配置为标记有效的 Oracle 数据库错误代码，并将它们映射到其 PostgreSQL 等效项，然后再继续执行剩余的流程逻辑。如果元数据表中没有 Oracle 数据库错误代码，则流程将退出，但有异常。然后，如果程序需要，您可以手动查看错误详细信息并将新的错误代码添加到表中。

通过使用此配置，Amazon Aurora PostgreSQL-Compatible 数据库可以像 Oracle 源数据库处理错误一样处理错误。

> **Note**
>
> 配置 PostgreSQL 数据库以正确处理 Oracle 数据库错误代码通常需要更改数据库和应用程序代码。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 已启动并正在运行的实例和侦听器服务的 Oracle 源数据库
- 已启动并正在运行的 Amazon Aurora PostgreSQL-Compatible 集群
- 熟悉 Oracle 数据库
- 熟悉 PostgreSQL 数据库

架构

下图显示了用于验证和处理数据错误代码的 Amazon Aurora PostgreSQL-Compatible 数据库的工作流程示例：

图表显示了以下工作流：

1. 表中包含了 Oracle 数据库错误代码和分类及其等效的 PostgreSQL 错误代码和分类。该表包含 valid_error 列，用于对特定的预定义错误代码是否有效进行分类。
2. 当 pl/pgSQL 函数（func_processdata）抛出异常时，它会调用第二个 pl/pgSQL 函数（error_validation）。
3. error_validation 函数接受 Oracle 数据库错误代码作为输入参数。然后，该函数对照表检查传入的错误代码，以查看该错误是否包含在表中。
4. 如果表中包含 Oracle 数据库错误代码，则 error_validation 函数将返回真值并且流程逻辑继续运行。如果表中未包含错误代码，则该函数返回假值，流程逻辑退出，且出现异常。
5. 当函数返回假值时，应用程序的职能主管将手动查看错误详细信息以确定其有效性。
6. 然后，要么手动将新的错误代码添加到表中，要么不手动添加。如果错误代码有效且已添加到表中，则下次发生异常时 error_validation 函数将返回真值。如果错误代码无效，并且异常发生时流程必须失效，则不会将错误代码添加到表中。

技术堆栈

- Amazon Aurora PostgreSQL
- pgAdmin
- Oracle SQL Developer

工具

- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩展 PostgreSQL 部署。
- pgAdmin 是 PostgreSQL 的开源管理和开发工具。它提供了图形界面，可简化数据库对象的创建、维护和使用。
- Oracle SQL Developer 是一个集成的免费开发环境，可简化传统部署和云部署中 Oracle 数据库的开发和管理。

操作说明

将 Oracle 数据库错误代码迁移到 Amazon Aurora PostgreSQL-Compatible 数据库中

Task	描述	所需技能
在 Amazon Aurora PostgreSQL-Compatible 数据库中创建表。	运行以下 PostgreSQL CREATE TABLE 命令： `( source_error_code numeric NOT NULL, target_error_code character varying NOT NULL, valid_error character varying(1) NOT NULL );`	PostgreSQL 开发人员、Oracle、适用于 PostgreSQL 的 RDS/Aurora
将 PostgreSQL 错误代码及其相应的 Oracle 数据库错误代码添加到表中。	运行 PostgreSQL INSERT 命令将所需错误代码值添加到 error_codes 表中。 PostgreSQL 错误代码必须使用字符变化的数据类型（SQLSTATE 值）。Oracle 错误代码必须使用数字数据类型（SQLCODE 值）。 插入语句示例： `insert into error_codes values (-1817,'22007','Y');`	PostgreSQL 开发人员、Oracle、适用于 PostgreSQL 的 RDS/Aurora

Task	描述	所需技能

	`insert into error_codes values (-1816,'22007','Y');`
	`insert into error_codes values (-3114,'08006','N');`
	
	Note
	如果您要捕获特定于 Oracle 的 Java 数据库连接 (JDBC) 异常，则必须将这些异常替换为通用的跨数据库异常或切换到 PostgreSQL 特定的异常。	

创建 PL/pgSQL 函数来验证错误代码。	通过运行 PostgreSQL 创建函数命令来创建 PL/pgSQL 函数。确保该函数执行以下操作： • 接受程序抛出的 Oracle 错误代码。 • 检查 error_codes 表中是否存在错误代码。 • 根据元数据表中是否存在错误代码，返回真或假值。	PostgreSQL 开发人员、Oracle、适用于 PostgreSQL 的 RDS/Aurora

Task	描述	所需技能
手动查看 PL/pgSQL 函数记录的新错误代码。	手动查看新错误代码。 如果新错误代码对用例有效，请运行 PostgreSQL INSERT 命令将其添加到 error_codes 表中。 –或者– 如果新错误代码对用例无效，请不要将其添加到表中。当错误发生时，流程逻辑将继续失效并以异常方式退出。	PostgreSQL 开发人员、Oracle、适用于 PostgreSQL 的 RDS/Aurora

相关资源

附录 A. PostgreSQL 错误代码（PostgreSQL 文档）

数据库错误消息（Oracle 数据库文档）

将 Redis 工作负载迁移至 Redis Enterprise Cloud on AWS

由 Antony Prasad Thevaraj (AWS) 和 Srinivas Pendyala (Redis) 编写

摘要

此模式介绍了在 Amazon Web Services (AWS) 上将 Redis 工作负载迁移至 Redis Enterprise Cloud 的高级流程。它描述了迁移步骤，提供了有关可用工具选择的信息，并介绍了使用每种工具的优缺点和步骤。或者，如果您在从 Redis 迁移工作负载时需要其他帮助，可使用 Redis 专业服务。

如果您在本地运行 Redis OSS 或 Redis Enterprise Software，您就能熟悉在数据中心维护 Redis 数据库所带来的巨大管理开销和操作复杂性。通过将工作负载迁移到云端，您可以显著减轻运营负担，并充分利用 Redis Enterprise Cloud，这是一款完全托管的数据库即服务 (DBaaS) 产品。这种迁移有助于提高业务灵活性、提高应用程序可靠性并降低总体成本，同时您可访问最新的 Redis Enterprise Cloud on AWS 功能，例如 99.999% 的可用性、架构简洁性和可扩展性。

Redis Enterprise Cloud 在金融服务、零售、医疗保健和游戏领域以及需要欺诈检测、实时库存、索赔处理和会话管理解决方案的用例中都有潜在应用。您可以使用 Redis 企业云连接到您的 AWS 资源，例如，连接到在亚马逊弹性计算云 (Amazon EC2) 实例上运行的应用程序服务器，或者连接到作为 AWS Lambda 服务部署的微服务。

先决条件和限制

假设

- 您当前正在操作要迁移至云端的本地数据库系统。
- 您已经确定了工作负载迁移要求，包括：
  - 数据一致性要求。
  - 基础设施和系统环境要求
  - 数据映射与转换要求
  - 功能测试要求
  - 性能测试要求
  - 验证要求
  - 定义的割接策略
- 您已经评测迁移所需时间表和成本估算。
- 您的要求会考虑工作范围以及确定为迁移一部分的系统和数据库。
- 您已经在负责任、负责、咨询、知情 (RACI) 矩阵中确定了利益相关者及其角色和责任。
- 您已获得所有利益相关者的必要同意与批准。

成本

根据现有源数据库的技术规格（例如内存大小、吞吐量和总数据大小），Redis 解决方案架构师可以在 Redis Enterprise Cloud 上调整目标系统的大小。 有关一般定价信息，请参阅 Redis 网站上的 Redis 定价。

人员与技能

迁移过程涉及以下角色与职责。

角色	描述	所需技能
迁移解决方案架构师	在定义、规划和实施迁移策略方面具有专长的技术架构师	对源系统和目标系统的技术和应用程序级理解；具有将工作负载迁移至云端的经验
数据架构师	技术架构师，在为各种数据库定义、实施和提供数据解决方案方面拥有丰富的经验	结构化和非结构化数据建模，在为企业实施数据库方面有深刻的理解和经验
Redis 解决方案架构师	技术架构师，可以根据适当的用例帮助架构大小最优的 Redis 集群	在为各种用例设计和部署 Redis 解决方案方面有专长
云解决方案架构师	对云解决方案（尤其是 AWS 上的解决方案）有更深入了解的技术架构师	云解决方案架构方面的专长；工作负载迁移和应用程序现代化经验
企业架构师	技术架构师，对组织的技术格局有全面的了解，对未来的路线图有共同的愿景，并且在组织中的所有团队中实践和建立标准化架构最佳实践	软件架构认证，如 TOGAF、基础软件工程技能、解决方案架构和企业架构专长
IT 或 DevOps 工程师	负责创建和维护基础设施的工程师，职责包括监控基础设施是否存在问题、执行维护任务以及根据需要进行更新。	对各种技术有深刻了解，包括操作系统、网络和云计算；熟悉 Python、Bash 和 Ruby 等编程语言以及 Docker、Kubernetes 和 Ansible 等工具

架构

迁移选项

下图显示了将您的本地（基于 Redis 或其他）数据来源迁移至 AWS 的选项。它显示了几种可供选择的迁移工具，例如使用 Redis 复制功能或使用 AWS DMS，将 Redis 数据库 (RDB) 文件导出到 Amazon Simple Storage Service (Amazon S3)。

1. 本地数据来源：不基于 Redis 数据库，例如 MySQL、PostgreSQL、Oracle、SQL Server 或 MariaDB。
2. 本地数据来源：基于 Redis 协议数据库，例如 Redis OSS 和 Redis 企业软件。
3. 从基于 Redis 的数据库迁移数据的最简单方法是导出 RDB 文件、并将其导入 AWS 上的目标 Redis Enterprise Cloud。
4. 或者，您可使用 Redis 中的复制功能 (`REPLICAOF`) 将数据从源迁移到目标。
5. 如果您的数据迁移要求包含数据转换，则可使用 Redis Input/Output Tools (RIOT) 迁移数据。
6. 或者，您还可使用 AWS Data Migration Service (AWS DMS) 从基于 SQL 的数据库中迁移数据。
7. 您必须使用 AWS DMS 的虚拟私有云（VPC）对等连接才能将数据成功迁移到 AWS 上的目标 Redis Enterprise Cloud中。

目标架构

下图显示了 Redis Enterprise Cloud on AWS 的典型部署架构，并说明了如何将其用于关键 Amazon Web Services。

1. 您可在 AWS 上连接由 Redis Enterprise Cloud 支持的业务应用程序。
2. 您可在自己的 Amazon Web Services account 中运行业务应用程序，也可以在该账户的 VPC 中运行业务应用程序。
3. 您可使用 Redis Enterprise Cloud 数据库端点连接到您的应用程序。示例包括在 EC2 实例上运行的应用程序服务器、部署为 AWS Lambda 服务的微服务、亚马逊弹性容器服务 (Amazon ECS) 应用程序或亚马逊 Elastic Kubernetes 服务 (Amazon EKS) 应用程序 (Amazon EKS) 应用程序。
4. 在您的 VPC 运行业务应用程序需要与 Redis Enterprise Cloud VPC 建立 VPC 对等连接。这使业务应用程序能够通过私有端点安全连接。

5. AWS 上的 Redis Enterprise Cloud 是一个内存中 NoSQL 数据库平台，作为 S 部署 DBaa在 AWS 上，完全由 Redis 管理。
6. Redis Enterprise Cloud 部署在 VPC 内的 Redis 创建的标准 Amazon Web Services account 中。
7. 出于安全考虑，Redis Enterprise Cloud 部署至私有子网中，私有和公有端点均可访问该子网。我们建议您将客户端应用程序连接到私有端点上的 Redis。如果您计划使用公共端点，我们强烈建议您启用 TLS 来加密您的客户端应用程序和 Redis Enterprise Cloud 之间的数据。

Redis 迁移方法与 AWS 迁移方法一致，AWS Prescriptive Guidance 网站上的动员您的组织以加快大规模迁移对此进行了说明。

自动化和扩缩

迁移的环境设置任务可以通过 AWS 登录区和基础设施即代码（IaC）模板自动完成，以实现自动化和扩缩。这些将在此模式的操作说明部分中介绍。

工具

根据您的数据迁移要求，可从一系列技术选项中进行选择，将您的数据迁移至 Redis Enterprise Cloud on AWS。下表对这些工具进行了描述和比较。

工具	描述	优点	劣势
RDB 导出和导入	您可以 RDB 文件的形式从源（例如 Redis OSS 或 Redis Enterprise Software）数据库中导出数据。如果您的数据库是通过 Redis OSS 集群提供，则可以将每个主分片导出到 RDB。 然后，您可一步导入所有 RDB 文件。如果您的源数据库基于 OSS 集群，但目标数据库未使用 OSS 集群 API，则必须更改应用程序	• 简便。 • 适用于任何可以 RDB 格式导出数据作为源的基于 Redis 的解决方案（包括 Redis OSS 和 Redis Enterprise Software）。 • 通过简单流程实现数据一致性。	• 不满足数据转换要求或者不支持逻辑数据库合并。 • 对于较大数据集非常耗时。 • 不支持 delta 迁移，可能会导致更长的停机时间。

	源代码，以使用标准 Redis 客户端库。 数据转换要求或逻辑数据库合并需要更复杂的过程，本表后面的逻辑数据库合并部分对此进行说明。		
Redis 复制功能（主动-被动）	您可将数据从 Redis OSS、Enterprise Software 或 Enterprise Cloud 数据库持续复制到 Redis Enterprise Cloud 数据库。初始同步后，Redis 复制功能 (`ReplicaOf`) 会执行增量迁移，这就意味着几乎没有观察到应用程序停机时间。 Redis 复制功能旨在以主动-被动方式采用。假定目标处于被动状态，并且已完全重新同步（从源数据库刷新和同步）。因此，在源和目标之间割接更复杂。 通过将 OSS 集群的所有主分片指定为源，可从 Redis OSS 集群复制到标准集群 Redis Enterprise Cloud 数据库。但是，Redis 复制功能最多可允许 32 个源数据库。	• 支持连续复制（初始数据加载后是增量）。 • 几乎没有停机时间（取决于复制延迟）。 • 实现数据一致性。	• 只有一个站点打算处于活动状态，因此在站点之间割接更为复杂。 • 从 OSS 集群迁移时最多支持 32 项主分片。

AWS DMS	您可使用 AWS DMS 将数据从任何支持的源数据库迁移到目标 Redis 数据存储，最大限度地减少停机时间。有关最新信息，请参阅 AWS DMS 文档中的使用 Redis 作为 AWS DMS 的目标。	• 支持 NoSQL 和 SQL 数据来源迁移。 • 与其他 Amazon Web Services 良好地协同工作。 • 支持实时迁移和更改数据捕获 (CDC) 用例。	• Redis 键值不能包含特殊字符，如 %。 • 不支持迁移行或字段名中包含特殊字符。 • 不支持完整大型二进制对象 (LOB) 模式。
逻辑数据库合并	特殊数据库合并要求可能需要自定义的数据迁移解决方案。例如，您在 Redis OSS 中可能有四个逻辑数据库 (`SELECT 0..3`)，但您可能希望使用单个数据库端点，而非将数据移动至多个 Redis Enterprise Cloud 数据库。Redis Enterprise 不支持可选逻辑数据库，因此您必须转换源数据库的物理数据模型。例如，您可将每个数据库索引映射到前缀(`0` 至 `usr`，1 至 cmp等)，然后使用迁移脚本或提取、转换、加载（ETL）工具输出 RDB 文件，然后可以将其导入目标数据库。	• 在迁移到目标系统的进程中，使用自定义脚本对数据进行精细控制。	• 如果您决定不完成迁移，则回退可能非常困难，尤其是在必须将较新的数据回退到源系统时。 • 如果目标是为一次性迁移构建一次性解决方案，则构建成本可能较高。 • 如果迁移要求频繁变化，则代码、基础设施、开发时间和其他方面维护成本可能会很高。

此外，您还可使用 AWS 提供的以下工具和服务。

评测和发现工具：

- AWS Application Discovery Service
- Migration Evaluator

应用程序和服务器迁移工具：

- AWS Application Migration Service

数据库迁移工具：

- AWS Schema Conversion Tool (AWS SCT)
- AWS Database Migration Service (AWS DMS)

数据迁移工具：

- AWS Storage Gateway
- AWS DataSync
- AWS Direct Connect
- AWS Snowball
- Amazon Data Firehose

迁移管理：

- AWS Migration Hub

AWS 合作伙伴解决方案：

- AWS Migration Competency Partners

操作说明

完成发现和评测任务

Task	描述	所需技能
识别工作负载。	确定要迁移的适当候选工作负载。选择迁移工作负载前，请注意以下各项： • 迁移/不迁移此工作负载的业务价值是什么？ • 如果此工作负载未成功迁移至目标系统，是否有应急计划？ 理想情况下，选择对业务影响最大、且风险最小的工作负载。保持整个流程的迭代性，并以较小增量进行迁移。	数据架构师、商业支持者、迁移项目发起人
确定数据来源和需求、设计数据模型。	Redis 举办了研讨会，以加快发现并定义项目的迁移计划。本次研讨会期间，Redis 团队将确定数据来源和源数据模型要求，并分析如何在 Redis Enterprise Cloud 中对其进行改造。 Redis 迁移团队 (Professional Services) 与您的组织一起执行详细的数据模型设计练习。在本次练习中，Redis 团队： • 识别目标 Redis 数据结构。 • 定义数据映射策略。 • 记录迁移方法与建议。	Redis 解决方案架构师

Task	描述	所需技能
	• 与利益相关者一起审查并最终确定数据模型。	
确定源数据库特点。	确定源环境和目标环境中的 Redis 产品。例如： • 源数据库是 OSS 集群数据库、独立的 Redis 数据库、还是 Redis 企业级数据库？ • 目标数据库是 Redis Enterprise 标准数据库还是 OSS 集群兼容数据库？ • 应用程序源代码有何影响？	数据架构师
收集当前的系统 SLA 与其他规模调整指标。	根据吞吐量（每秒操作数 SLAs）、延迟、每个数据库的总体内存大小和高可用性 (HA) 要求来确定当前的服务级别协议 ()。	数据架构师

Task	描述	所需技能
确定目标系统特点。	确定以下回答答案： • 必须迁移的数据量？ • 迁移指定数据量需要的时间？ • 迁移的停机时间要求？ 您的服务或应用程序在特定时间段内不可用。这是否可以接受？ 如果是，则持续多长时间？ • 迁移的数据应保持何种一致性？ 目标数据库能否处于稍微不一致（过时）的状态？ • 在将数据加载至目标数据库之前，是否必须对其进行转换？（例如，您可能希望在迁移之前将可选数据库索引转换为前缀。） • 是否可以从目标数据库的主机（例如从对等 VPC 或使用加密的公共端点）访问源数据库？ • 与 Redis 技术架构师一起完成数据大小调整以及 Redis 集群规模调整练习。 • 确定网络需求、基础设施要求、软件版本以及软件许可，并在迁移之前购买任何组件。 • 传输此类数据是否存在任何安全问题？	数据架构师、Redis 解决方案架构师（可选）

Task	描述	所需技能
确定依赖项。	确定要迁移的当前系统上游和下游依赖项。确保迁移工作与其他依赖系统迁移是否保持一致。例如，如果您计划将其他业务应用程序从本地迁移至 AWS Cloud，请识别这些应用程序并根据项目目标、时间表和利益相关者进行调整。	数据架构师、企业架构师
确定迁移工具。	根据您的数据迁移要求（例如源数据或停机时间要求），您可使用前面工具部分描述的任何工具。此外，您还可使用： • 使用 CRDB 部署进行双向（主动-主动）复制。 • 自定义导出/导入脚本（例如使用DUMP/RESTORE 命令）。 • 其他导出/导入工具和辅助工具，例如 RIOT 或 ET LECstats2 工具。 • IaC 工具，例如 Terraform 或 AWS 模板。 CloudFormation	迁移解决方案架构师、Redis 解决方案架构师
制定应急计划。	制定应急计划进行回退，以防在迁移过程中遇到问题。	项目管理、技术团队，包含架构师

完成安全与合规任务

Task	描述	所需技能
保护 Redis 管理控制台。	要保护管理控制台，请按 Redis 文档中的说明进行操作。	IT 基础设施管理员
保护 Redis 数据库。	有关操作，请参阅 Redis 文档中的以下页面： • 定义基于角色的访问控制。 • 定义网络安全。 • 启用 TLS。	
安全 Redis 云 APIs。	启用 API 后，您可对所有 Redis Cloud 账户拥有者管理 API 密钥。有关 API 安全功能的概述，请参阅 Redis 网站的 API 身份验证文档。	IT 基础设施管理员

设置新环境

Task	描述	所需技能

在 AWS 设置新环境。	此任务包括：
	• AWS 登录区设置活动。登录区支持：
	  • 多账户部署
	  • 最低安全基准
	  • 自动为新账户配置安全基准和 ISV 先决条件（网络、安全配置等）
	  • 通知、集中日志记录和监控
		IT 或 DevOps 工程师

Task	描述	所需技能
	• ISV 软件配置活动。这包含迁移中所需配置，例如产品和工作负载的设置和更改。 • IaC 活动，例如配置或自定义 AWS CloudFormation 或 Terraform 模板。	
部署迁移架构。	1. 安装 Redis Enterprise Cloud on AWS 2. 安装迁移工具，例如 RIOT 或 AWS DMS。有关可用工具列表，请参阅工具部分。 3. 在应用程序、迁移和数据库层间建立连接。 4. 创建可流经每一层的示例工作负载，迁移一小部分样本数据。 现在，您可运行实际数据迁移管道并对其进行测试了。	IT 或 DevOps 工程师

设置联网

Task	描述	所需技能
建立连接。	在本地基础设施和 Amazon Web Services Cloud 资源之间建立连接。使用安全组、AWS Direct Connect 等资源来实现此功能。有关更多信息，请参阅 AWS 网站上的将您的数据中心连接到 AWS。	IT 或 DevOps 工程师

Task	描述	所需技能
设置 VPC 对等连接。	在 VPCs 运行业务应用程序（或运行迁移工具或 AWS DMS 复制服务器的 EC2 实例）和运行 Redis 企业云的 VPC 之间建立 VPC 对等关系。有关说明，请参阅 Amazon VPC 文档中的 Amazon VPC 入门和 Redis 文档中的启用 VPC 对等。	IT 或 DevOps 工程师

迁移数据

Task	描述	所需技能
选择数据迁移工具。	查看工具部分的表格，查看这些工具的描述、优点和缺点： • RDS 导出和导入 • Redis 复制功能 (`ReplicaOf`) • AWS DMS • 逻辑数据库合并 以下各行描述了与各种工具相关的数据迁移任务。	迁移解决方案架构师
选项 1：使用 RDB 导出与导入。	1. 断开源连接：停止源数据库上的流量（例如，断开业务应用程序的连接）。 2. 导出：将源数据库数据导出为 RDB 文件。 3. 阶段：将数据上传至 Redis Enterprise Cloud on AWS 实例可以访问的位置(例如，	迁移解决方案架构师、Redis 解决方案架构师

Task	描述	所需技能
	您可将其上传到 S3 存储桶或 FTP 服务器)。 4. 导入：将 RDB 文件（通过在一个导入步骤中列出所有文件）导入 Redis Enterprise Cloud 目标数据库。 5. 割接：移至目标数据库（例如，通过将应用程序连接到目标数据库）。 有关更多信息，请参阅 Redis 文档。	
选项 2：使用 Redis 复制功能（主动-被动）。	1. 连接数据库：在源数据库和目标数据库之间建立`ReplicaOf` 链接。 2. 运行初始同步：等源数据库和目标数据库之间的初始同步完成。 3. 断开源连接：停止源数据库的流量（例如，断开应用程序的连接）。 4. 运行增量复制：等待 delta 复制至目标数据库。 5. 割接：移至目标数据库（例如，通过将应用程序连接到目标数据库）。 6. 删除：移除源数据库和目标数据库之间的 `ReplicaOf` 链接。 有关更多信息，请参阅 Redis 文档。	迁移解决方案架构师、Redis 解决方案架构师

Task	描述	所需技能
选项 3：使用 AWS DMS。	1. 设置 AWS DMS 复制实例：以此实例执行所有迁移过程。有关说明，请参阅 AWS DMS 文档中的使用 AWS DMS 复制实例。 2. 定义源数据库：定义源端点。测试源端点与 AWS DMS 复制服务器间的连接。有关说明，请参阅 AWS DMS 文档中的创建源和目标端点。 3. 设置目标数据库：在 AWS 上设置 Redis Enterprise Cloud 并设置要迁移到的数据库。 4. 定义目标数据库：定义目标端点。确保在运行 AWS DMS 的 VPC 和托管 Redis Enterprise Cloud on AWS 的 VPC 之间建立 VPC 对等关系。测试 AWS DMS 复制服务器和目标服务器间的连接。 5. 创建 AWS DMS 任务：创建一项或一组任务，定义要用于迁移数据的表和复制过程。有关说明：在 AWS DMS 文档中处理 AWS DMS 任务。 6. 迁移：通过运行 AWS DMS 任务迁移数据。	迁移解决方案架构师、Redis 解决方案架构师

Task	描述	所需技能
	7. 割接：移至目标数据库（例如，通过将应用程序连接到目标数据库）。	
选项 4：采用逻辑数据库合并。	此选项包括使用可以转换源数据库物理数据模型、以及生成 RDB 文件的迁移脚本或 ETL 工具。如有必要，Redis Professional Services 可以帮助完成此步骤。	迁移解决方案架构师、Redis 解决方案架构师

迁移应用程序

Task	描述	所需技能
调整项目管理时间表与目标。	将应用程序层迁移项目目标、里程碑和时间表与 Redis 数据迁移项目的目标、里程碑和时间表保持一致。	项目管理
调整测试活动。	在 Amazon Web Services Cloud 中对应用程序层进行迁移与现代化改造后，将应用程序层指向 AWS 上新迁移的 Redis Enterprise Cloud 进行测试。	测试

测试

Task	描述	所需技能
实施测试计划。	根据测试要求，在您站点的测试环境中运行在实施阶段开发的数据迁移例程和脚本。	测试

Task	描述	所需技能
测试数据质量。	迁移数据后，测试数据质量。	测试
测试功能。	测试数据查询与应用程序层，确保应用程序的性能与源系统中的性能相同。	测试

割接

Task	描述	所需技能
做出割接决定。	在所有应用程序级和数据库级测试完成后，执行领导团队和利益相关者将根据测试团队确认的最终结果，就是否割接到 AWS 上的新环境做出最终决定。	项目管理，商业拥护者
割接至 Amazon Web Services Cloud 中。	确认一切准备就绪后，将应用程序层指向新迁移的数据，将客户端指向新 Redis Enterprise Cloud system on AWS 系统运行的新应用程序层。	IT 或 DevOps 工程师、数据架构师、迁移解决方案架构师、Redis 解决方案架构师

相关资源

Redis 资源

- Redis Enterprise Cloud 文档
- RIOT 工具（GitHub 存储库）
- Terraform 提供程序（下载）

AWS 资源

- 迁移演示

- AWS 合作伙伴解决方案
- 文档
- 博客文章
- 白皮书
- 教程和视频
- AWS 云迁移
- AWS Prescriptive Guidance

其他信息

有关将 Redis 工作负载迁移到 AWS 云的标准安全要求，请参阅 AWS 网站上的安全、身份和合规最佳实践，以及 Redis 网站上的 Redis 信任中心。

使用 AWS SCT 和 AWS DMS EC2 将亚马逊上的 SAP ASE 迁移到兼容 Aurora PostgreSQL 的亚马逊 Aurora

由 Amit Kumar (AWS) 和 Ankit Gupta (AWS) 创作

摘要

此模式描述了如何使用 AWS 架构转换工具 (AWS SCT) 和 AWS 数据库迁移服务 (AWS DMS) 将托管在亚马逊弹性计算云 (亚马逊 EC2) 实例上的 SAP 自适应服务器企业 (SAP ASE) 数据库迁移到兼容亚马逊 Aurora PostgreSQL 的版本。该模式侧重于存储对象的数据定义语言 (DDL) 转换以及数据迁移。

Aurora PostgreSQL-Compatible 支持联机事务处理 (OLTP) 工作负载。此托管服务提供可按需要自动扩缩的配置。它可根据应用程序的需求自动启动、关闭、纵向扩展或缩减数据库。无需管理任何数据库实例，可在云中运行数据库。Aurora PostgreSQL-Compatible 为不频繁、间歇性或不可预测的工作负载提供了一种经济高效的选择。

迁移过程包含两个主要阶段：

- 使用 AWS SCT 转换数据库架构
- 通过 AWS DMS 迁移数据

操作说明部分提供了这两个阶段的详细说明。有关对 SAP ASE 数据库使用 AWS DMS 的特定问题进行故障排除的信息，请参阅 AWS DMS 文档中的 SAP ASE 问题故障排除。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- EC2 实例上的源 SAP ASE 数据库，服务器、数据库和侦听器服务已启动并运行
- 目标 Aurora PostgreSQL-Compatible 数据库

限制

- 连接的端口号必须为 5432。
- huge_pages 功能默认开启，但可以进行修改。
- Point-in-time 恢复 (PITR) 粒度为 5 分钟。
- 跨区域复制当前不可用。

- Aurora 数据库最大存储大小为 128 TiB。
- 最多可以创建 15 个只读副本。
- 表大小限制仅受 Aurora 集群卷大小的限制，因此PostgreSQL-Compatible 的数据库集群的最大表大小为 32 TiB。我们建议您遵循表设计的最佳实践，例如对大型表进行分区。

产品版本

- 源数据库：AWS DMS 当前支持 SAP ASE 15、15.5、15.7 和 16.x。有关 SAP ASE 版本支持的最新信息，请参阅 AWS DMS 用户指南。
- 目标数据库：PostgreSQL 9.4 及以上版本（适用于版本 9.x）、10.x、11.x、12.x、13.x 和 14.x。有关最新支持版本的 PostgreSQL，请参阅 AWS DMS 用户指南。
- Amazon Aurora 1.x 或更高版本。有关最新信息，请参阅 Aurora 文档中的与 Aurora PostgreSQL 兼容的版本和引擎版本。

架构

源技术堆栈

- 在亚马逊上运行的 SAP ASE 数据库 EC2

目标技术堆栈

- Aurora PostgreSQL-Compatible 数据库

迁移架构

工具

- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩展 PostgreSQL 部署。
- AWS Schema Conversion Tool (AWS SCT) 通过以下方法来简化异构数据库的迁移工作：将源数据库架构和大部分的自定义代码自动转换成与目标数据库兼容的格式。
- AWS DMS 支持几种不同的源数据库和目标数据库。有关更多信息，请参阅 AWS DMS 文档中的数据迁移源和数据迁移目标。要获得最全面的版本和功能支持，我们建议您使用最新版本的 AWS DMS。

操作说明

设置环境

Task	描述	所需技能
在源 EC2 实例中配置网络访问权限。	在托管源 SAP ASE 数据库的 EC2 实例中设置安全组。 有关说明，请参阅亚马逊 EC2 文档中的适用于 Linux 实例的亚马逊 EC2 安全组。	系统管理员
创建目标 Aurora PostgreSQL-Compatible DB 集群。	为您的目标数据库安装、配置和启动 Aurora PostgreSQL-Compatible 集群。 有关更多信息，请参阅 Aurora 文档中的创建 Amazon Aurora DB 集群。	数据库管理员
为目标数据库集群设置授权。	为目标数据库设置安全组以及防火墙。 有关说明，请参阅 Aurora 文档中的创建 Amazon Aurora 数据库集群。	数据库管理员、系统管理员

通过 AWS SCT 转换您的数据库架构

Task	描述	所需技能
启动 AWS SCT。	按照 AWS SCT 文档中的说明启动 AWS SCT。 AWS SCT 提供基于项目的用户界面，可以自动将 SAP ASE 源数据库的数据库架构转换为与目标 Aurora PostgreSQL-	数据库管理员

Task	描述	所需技能
	Compatible DB 实例兼容的格式。	
创建 AWS SCT 端点。	为源 SAP ASE 和目标 PostgreSQL 数据库创建端点。 有关说明，请参阅 AWS SCT 文档。	数据库管理员
创建评测报告。	创建数据库迁移评估报告以评测迁移情况，并检测任何不兼容的对象和函数。 有关说明，请参阅 AWS SCT 文档。	数据库管理员
转换架构。	按照 AWS SCT 文档的说明转换数据库架构。	数据库管理员
验证数据库对象。	如果 AWS SCT 无法转换数据库对象，它将识别其名称与其他细节。您必须手动转换这些对象。 要识别这些不匹配项，请按照 AWS Blog 文章从 SAP ASE 迁移到 Amazon RDS for PostgreSQL 或 Amazon Aurora PostgreSQL 后验证数据库对象	数据库管理员

分析 AWS DMS 迁移

Task	描述	所需技能
验证源数据库和目标数据库的版本。	检查 SAP ASE 的数据库版本是否与 AWS DMS 兼容。 有关更多信息，请参阅 AWS DMS 文档中的 AWS DMS 来源和 AWS DMS 目标。	数据库管理员
确定存储类型和容量的要求。	根据源数据库的大小，为目标数据库选择适当的存储容量。	数据库管理员、系统管理员
选择复制实例的实例类型、容量以及其他功能。	选择满足需求的实例类型、容量、存储特性和网络特性。 有关指导，请参阅 AWS DMS 文档中的为迁移选择正确的 AWS DMS 复制实例。	数据库管理员、系统管理员
识别网络访问安全要求。	确定源数据库和目标数据库的网络访问安全要求。 按照 AWS DMS 文档中的为复制实例设置网络中的指导进行操作。	数据库管理员、系统管理员

迁移数据

Task	描述	所需技能
通过在 AWS DMS 中创建迁移任务以迁移数据。	要迁移数据，请创建任务并按照 AWS DMS 文档说明进行操作。	数据库管理员

Task	描述	所需技能
	建议使用最新版本的 AWS DMS，以获得最全面的版本和功能支持。	
验证数据。	要验证数据是否准确地从源数据库迁移到目标数据库，请遵循AWS DMS文档中提供的数据验证指南。	数据库管理员

迁移应用程序

Task	描述	所需技能
确定应用程序迁移策略。	从将应用程序迁移至云端的七种策略 (7R) 中选择一种。	数据库管理员、应用程序所有者、系统管理员
遵循应用程序迁移策略。	完成应用程序团队确定的数据库任务，包括更新目标数据库的 DNS 连接详细信息，以及更新动态查询。	数据库管理员、应用程序所有者、系统管理员

割接至目标数据库

Task	描述	所需技能
将应用程序客户端切换至新基础设施。	将连接从源数据库切换到目标数据库。 有关更多信息，请参阅关系数据库迁移策略的割接部分	数据库管理员、应用程序所有者、系统管理员

关闭项目

Task	描述	所需技能
关闭临时 AWS 资源。	终止所有迁移任务、复制实例、端点以及其他 AWS SCT 和 AWS DMS 资源。 有关更多信息，请参阅 AWS DMS 文档。	数据库管理员、系统管理员
审核和验证项目文档。	验证项目文档中的所有步骤，确保所有任务已成功完成。	数据库管理员、应用程序所有者、系统管理员
关闭项目。	关闭迁移项目并提供任何反馈。	数据库管理员、应用程序所有者、系统管理员

相关资源

参考

- 在 Amazon RDS 中为 PostgreSQL 数据库实例启用加密连接 (AWS Prescriptive Guidance)
- 使用 pg_transport 在两个 Amazon RDS 数据库实例之间传输 PostgreSQL 数据库 (AWS Prescriptive Guidance)
- Amazon Aurora 定价
- Amazon Aurora PostgreSQL-Compatible Edition 的最佳实践 (Amazon Aurora 文档)
- AWS SCT 文档
- AWS DMS 文档
- 使用 SAP ASE 数据库作为 AWS DMS 源

教程和视频

- AWS Database Migration Service 入门
- AWS Database Migration Service（视频）

使用 ACM 将 Windows SSL 凭证迁移到应用程序负载均衡器

由 Chandra Sekhar Yaratha (AWS) 和 Igor Kovalchuk (AWS) 编写

摘要

该模式为使用 AWS Certifice Manager (ACM) 从托管在本地服务器上的网站或微软互联网信息服务 (IIS) 上的亚马逊弹性计算云 (Amazon EC2) 实例迁移现有安全套接字层 (SSL) 证书提供了指导。然后，SSL 凭证可与 AWS 上的弹性负载均衡一起使用。

SSL 可保护您的数据，确认您的身份，提供更好的搜索引擎排名，帮助满足支付卡行业数据安全标准 (PCI DSS) 的要求，并提高客户信任度。管理这些工作负载的开发人员和 IT 团队希望他们的 Web 应用程序和基础设施（包括 IIS 服务器和 Windows Service 器）始终符合其基准策略。

此模式包括手动从 Microsoft IIS 导出现有 SSL 凭证，将其从 Personal Information Exchange (PFX) 格式转换为 ACM 支持的 Private Enhanced Mail (PEM) 格式，然后将其导入到您的 Amazon Web Services account 中的 ACM 中。它还描述了如何为您的应用程序创建应用程序负载均衡器，以及如何将应用程序负载均衡器配置为使用您导入的凭证。然后，在应用程序负载均衡器终止 HTTPS 连接，您无需在 Web 服务器上增加额外的配置开销。有关更多信息，请参阅为您的应用程序负载均衡器创建 HTTPS 侦听器。

Windows 服务器使用 .pfx 或 .p12 文件包含公钥文件（SSL 凭证）及其唯一私钥文件。凭证颁发机构 (CA) 为您提供您的公钥文件。您可使用服务器生成创建凭证签名请求 (CSR) 的关联私钥文件。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- AWS 上的虚拟私有云（VPC），在您的目标使用的每个可用区中至少有一个私有子网和一个公有子网
- 在 Windows Server 2012 或更高版本上运行的 IIS 版本 8.0 或更高版本
- IIS 上运行的 Web 应用程序
- IIS 服务器的管理员访问权限

架构

源技术堆栈

- 使用 SSL 实施 IIS Web 服务器，确保数据在加密连接 (HTTPS) 中安全传输

源架构

目标技术堆栈

- Amazon Web Services account 中的 ACM 凭证
- 配置为使用导入凭证的应用程序负载均衡器
- 私有子网中的 Windows 服务器实例

目标架构

工具

- AWS Certificate Manager（ACM）可帮助您创建、存储和续订公有及私有 SSL/TLS X.509 证书和密钥，这些证书和密钥可保护 AWS 网站和应用程序。
- 弹性负载均衡（ELB）将传入的应用程序或网络流量分发到多个目标。例如，您可以在一个或多个可用区域中的 EC2 实例、容器和 IP 地址之间分配流量。

最佳实践

- 将 HTTP 流量强制重定向到 HTTPS。
- 为应用程序负载均衡器适当配置安全组，仅允许入站流量进入特定端口。
- 在不同的可用区启动您的 EC2 实例以确保高可用性。
- 将应用程序域配置为指向应用程序负载均衡器的 DNS 名称，而不是其 IP 地址。
- 确保应用程序负载均衡器已配置 application-layer 运行状况检查。
- 配置运行状况检查的阈值。
- 使用 Amazon CloudWatch 监控应用程序负载均衡器。

操作说明

导出 .pfx 文件

Task	描述	所需技能
从 Windows 服务器导出.pfx 文件。	要从 Windows 服务器的本地 IIS 管理器中将 SSL 凭证导出为 .pfx 文件，请执行以下操作： 1. 选择 开始、管理、Internet Information Services (IIS) Manager。 2. 选择服务器名称，然后在安全下双击服务器凭证。 3. 选择要导出的凭证，然后选择导出。 4. 在导出凭证框，为您的 .pfx 文件选择位置、路径和名称。 5. 为您的 .pfx 文件指定并确认密码。 Note 安装.pfx 文件时需要此密码。 6. 选择确定。 现在，您的 .pfx 文件应保存至指定的位置和路径。	系统管理员

将 PFX 编码的凭证转换为 PEM 格式

Task	描述	所需技能
下载和安装 OpenSSL 工具包。	1. 从 Shining Light Productions 网站下载和安装 Win32/Win64 OpenSSL 。 2. 将 OpenSSL 二进制文件的位置添加至您的系统 PATH 变量中，以便二进制文件可供命令行使用。	系统管理员
将 PFX 编码的凭证转换为 PEM 格式。	以下步骤将 PFX 编码的签名凭证文件转换为 PEM 格式的三个文件： • `cert-file.pem` 包含资源的 SSL/TLS 凭证。 • `privatekey.pem` 包含凭证的私钥，没有密码保护。 • `ca-chain.pem` 包含 CA 的根凭证。 要转换 PFX 编码的凭证，请执行以下操作： 1. 运行 Windows PowerShell。 2. 使用以下命令从 PFX 文件中提取凭证的私钥。根据系统提示输入凭证密码。 `openssl pkcs12 -in <filename>.pfx -nocerts -out withpw-privatekey.pem`	系统管理员

Task	描述	所需技能

	该命令生成名为 privatekey.pem 的 PEM 编码的私钥文件。在提示时，输入密码以保护私钥文件。
	3. 运行以下命令以移除密码。显示提示时，请提供您在步骤 2 中创建的密码。
	`openssl rsa -in withpw-privatekey.pem -out privatekey.pem`
	如果命令成功，该命令会显示“正在写入 RSA 密钥”消息。
	4. 使用以下命令将凭证从 PFX 文件传输至 PEM 文件。
	`openssl pkcs12 -in <file_name>.pfx -clcerts -nokeys -out cert-file.pem`
	这会创建名为 cert-file.pem 的 PEM 编码凭证文件。如果命令成功，该命令将显示消息“MAC 验证正常”。
	5. 通过 PFX 文件创建 CA 链文件。下面的命令创建名为 ca-chain.pem 的 CA 链文件。	

Task	描述	所需技能
	`openssl pkcs12 -in <file_name>.pfx -cacerts -nokeys -chain -out ca-chain.pem` 如果命令成功，该命令将显示消息“MAC 验证正常”。	

将凭证导入 ACM

Task	描述	所需技能
准备导入凭证。	在 ACM 控制台，选择导入凭证。	云管理员
提供凭证正文。	对于凭证正文，粘贴要导入的 PEM 编码凭证。 有关此操作说明中此任务和其他任务的命令和步骤的更多信息，请参阅 ACM 文档中的导入凭证。	云管理员
提供凭证私钥。	对于凭证私钥，粘贴与凭证的公有密钥匹配的 PEM 编码的未加密私有密钥。	云管理员
提供凭证链。	对于凭证链，粘贴 PEM 编码的凭证链，该凭证链存储在 `CertificateChain.pem` 文件中。	云管理员
导入凭证。	选择查看和导入。确认有关您的凭证的信息正确无误，然后选择导入。	云管理员

创建应用程序负载均衡器

Task	描述	所需技能
创建和配置负载均衡器与侦听器。	按照弹性负载均衡文档中的说明，配置目标组、注册目标以及创建应用程序负载均衡器和侦听器。为端口 443 添加第二个侦听器 (HTTPS)。	云管理员

故障排除

事务	解决方案
即使你将 OpenSSL 命令添加到系统路径中，Windows 也 PowerShell 无法识别该命令。	检查`$env:path` ，以确保它包含 OpenSSL 二进制文件的位置。 如果没有，请在中运行以下命令 PowerShell： `$env:path = $env:path + ";C:\OpenSSL-Win64\bin"`

相关资源

将凭证导入到 ACM

- ACM 控制台
- 凭证和密钥的导入格式
- 导入凭证
- AWS Certificate Manager 用户指南

创建应用程序负载均衡器

- 创建应用程序负载均衡器
- 应用程序负载均衡器用户指南

将消息队列从 Microsoft Azure 服务总线迁移到 Amazon SQS

由 Nisha Gambhir (AWS) 创建

摘要

此模式介绍如何使用 Microsoft Azure 服务总线队列消息传送平台将 .NET Framework 或 .NET Core Web 或控制台应用程序迁移到 Amazon Simple Queue Service（Amazon SQS）。

应用程序使用消息传递服务向其他应用程序发送数据以及从其他应用程序接收数据。这些服务有助于在云中构建解耦、高度可扩展的微服务、分布式系统和无服务器应用程序。

Azure 服务总线队列是更广泛的 Azure 消息传送基础结构的一部分，该基础结构支持排队和发布/订阅消息收发。

Amazon SQS 是一种完全托管的消息队列服务，使您能够分离和扩展微服务、分布式系统和无服务器应用程序。Amazon SQS 消除了与管理和操作面向消息的中间件相关的复杂性和开销，使开发人员能够专注于差异化工作。使用 Amazon SQS，您可以在软件组件之间以任意卷发送、存储和接收消息，而不会丢失消息或要求其他服务可用。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 使用 Azure 服务总线队列的 .NET Framework 或 .NET Core Web 或控制台应用程序（附加示例代码）

产品版本

- .NET Framework 3.5 或更高版本，或 .NET Core 1.0.1、2.0.0 或更高版本

架构

源技术堆栈

- 使用 Azure 服务总线队列发送消息的 .NET（Core 或 Framework）Web 或控制台应用程序

目标技术堆栈

- Amazon SQS

工具

工具

- Microsoft Visual Studio

代码

要为 Amazon SQS 创建 AWS Identity and Access Management (IAM) policy，请执行以下操作：

1. 登录 AWS 管理控制台并在上打开 IAM 控制台https://console.aws.amazon.com/iam/。

2. 在左侧的导航窗格中，选择 Policies (策略)，然后选择 Create policy (创建策略)。

3. 选择 JSON 选项卡，然后粘贴以下代码：

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "VisualEditor0",
         "Effect": "Allow",
         "Action": [
            "sqs:DeleteMessage",
            "sqs:GetQueueUrl",
            "sqs:ChangeMessageVisibility",
            "sqs:SendMessageBatch",
            "sqs:ReceiveMessage",
            "sqs:SendMessage",
            "sqs:GetQueueAttributes",
            "sqs:ListQueueTags",
            "sqs:ListDeadLetterSourceQueues",
            "sqs:DeleteMessageBatch",
            "sqs:PurgeQueue",
            "sqs:DeleteQueue",
            "sqs:CreateQueue",
            "sqs:ChangeMessageVisibilityBatch",
            "sqs:SetQueueAttributes"
         ],
         "Resource": "arn:aws:sqs:*:<AccountId>:*"
```

```
      },
      {
         "Sid": "VisualEditor1",
         "Effect": "Allow",
         "Action": "sqs:ListQueues",
         "Resource": "*"
      }
   ]
}
```

4. 选择查看策略，键入名称，然后选择创建策略。

5. 将新创建的策略附加到现有 IAM 角色或创建新角色。

操作说明

在 AWS 中设置 Amazon SQS

Task	描述	所需技能
为 Amazon SQS 创建 IAM policy。	创建将提供对 Amazon SQS 的访问权限的 IAM policy。有关示例策略，请参阅“代码”部分。	系统工程师
创建 AWS 配置文件。	通过运行 PowerShell 命令集的 AWS 工具来创建新的配置文件 AWSCredential。此命令将您的访问密钥和秘密密钥存储在您指定的配置文件名称下的默认凭证文件中。将您之前创建的 Amazon SQS 策略与此账户链接。保存 AWS 访问密钥 ID 和秘密访问密钥。在接下来的步骤中将需要这些内容。	系统工程师
创建 SQS 队列。	您可以创建标准队列或先进先出 (FIFO) 队列。有关说明，请参阅“参考”部分中的链接。	系统工程师

修改 .NET 应用程序代码

Task	描述	所需技能
安装 AWS Toolkit for Visual Studio。	此工具包是 Microsoft Visual Studio 的扩展，可让您更轻松地在 AWS 中构建和部署 .NET 应用程序。有关安装和使用说明，请参阅“参考”部分中的链接。	应用程序开发者
安装 AWSSDK .SQS NuGet 软件包。	你可以通过在 Visual Studio 中选择 “管理软件 NuGet 包” 或运行 “Install-Package AWSSDK .SQS” 命令来安装 AWSSDK .SQS。	应用程序开发者
在您的.NET 应用程序中创建 AWSCredentials 对象。	附件中的示例应用程序显示了如何创建继承自 AWSCredentials的 Basic AWSCredentials 对象。您可以使用之前的访问密钥 ID 和秘密访问密钥，也可以让对象在运行时从 .aws 文件夹中选取它们作为用户配置文件的一部分。	应用程序开发人员
创建 SQS 客户端对象。	为.NET 框架创建 SQS 客户端对象 (AmazonSQSClient)。这是 Amazon.SQS 命名空间的一部分。这个对象是必需的，而不是 IQueue客户端，后者是 Microsoft.Azure 的一部分。ServiceBus 命名空间。	应用程序开发者
调用 SendMessageAsync 方法向 SQS 队列发送消息。	将向队列发送消息的代码更改为使用 amazonSqsClient。SendMessageAsync 方法。有	应用程序开发者

Task	描述	所需技能
	关详细信息，请参阅随附的代码示例。	
调用 ReceiveMessageAsync 方法接收来自 SQS 队列的消息。	将接收消息的代码更改为使用 amazonSqsClient。ReceiveMessageAsync 方法。有关详细信息，请参阅随附的代码示例。	应用程序开发者
调用 DeleteMessageAsync 方法从 SQS 队列中删除消息。	要删除消息，请更改 QueueClient 中的代码。CompleteAsync 方法 amazonSqsClient. DeleteMessageAsync 方法。有关详细信息，请参阅随附的代码示例。	应用程序开发人员

相关资源

- 适用于 .NET 的 AWS SDK 开发人员指南
- 使用 Amazon SQS 进行消息收发
- 通过 适用于 .NET 的 AWS SDK 创建和使用 Amazon SQS 队列
- 发送 Amazon SQS 消息
- 接收来自 Amazon SQS 队列的消息
- 删除来自 Amazon SQS 队列的消息
- AWS Toolkit for Visual Studio

其他信息

此模式包括两个示例应用程序（请参阅附件部分）：

- AzureSbTestApp包括使用 Azure 服务总线队列的代码。
- AmazonSqsTestApp使用亚马逊 SQS。这是一个使用 .NET Core 2.2 的控制台应用程序，包括用于发送和接收消息的示例。

注意:

- QueueClient 是 Client 的对象， IQueue客户机是 Microsoft.Azure 的一部分。 ServiceBus 命名空间（包含在 Microsoft.Azure 中。 ServiceBus NuGet 包裹）。
- amazonSqsClient 是亚马逊的一个对象SQSClient，它是 Amazon.sqs 命名空间的一部分（包含在.SQS 包中） AWSSDK。 NuGet
- 根据代码的运行位置（比如代码是否在运行） EC2，角色需要拥有写入 SQS 队列的权限。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

将关系数据库迁移到 MongoDB Atlas AWS

由 Battulga Purevragchaa (AWS)、Babu Srinivasan (MongoDB) 和 Igor Alekseev (AWS) 创作

摘要

此模式描述了中从 SQL Server、MySQL 或 PostgreSQL 等关系数据库迁移到 MongoDB Atlas 的步骤。 AWS Cloud它使用 MongoDB 关系迁移器来帮助加快从关系数据库到 MongoDB Atlas 的数据迁移。

该模式与规范性指导网站上的《迁移到 MongoDB Atlas AWS》 AWS 指南一起出现。它提供了该指南中讨论的其中一个迁移场景的实施步骤。有关其他迁移方案，请参阅 AWS 规范性指导网站上的以下模式：

- 将自托管 MongoDB 环境迁移到 MongoDB Atlas AWS
- 将数据从 IBM Db2、SAP、Sybase 和其他数据库流式传输到 MongoDB Atlas AWS

该模式适用于 AWS 系统集成商 (SI) 合作伙伴和 AWS 用户。

先决条件和限制

先决条件

- 要迁移到 MongoDB Atlas 的源关系数据库（Oracle 数据库、SQL Server、PostgreSQL、MySQL、SAP/Sybase ASE 等）。
- 熟悉关系数据库、MongoDB Atlas 和。 AWS 服务这种模式从较高层次解释了一些迁移步骤。更多细节将在 future 版本中添加。

产品版本

- MongoDB 版本 5.0 或更高版本

架构

下图显示了从关系数据库管理系统 (RDBMS) 数据库迁移到 MongoDB Atlas 的过程。 AWS

有关支持不同使用场景的 MongoDB Atlas 参考架构，请参阅规范性指南网站上的迁移到 MongoDB Atlas。 AWS AWS

工具

- MongoDB Atlas 是一种完全托管的数据库即服务，用于在DBaa云中部署和管理 MongoDB 数据库。
- MongoDB 关系迁移器可将数据从传统关系数据库平稳过渡到 MongoDB。它有助于实现转换过程的自动化，并将关系数据库的结构化数据模型转换为 MongoDB 提供的灵活文档格式。关系迁移器保留了数据的完整性和关系，以简化迁移。Organizations 可以利用 MongoDB 提供的可扩展性、性能和多功能性优势，同时保持对现有数据的熟悉程度。

最佳实践

有关在 AWS上使用 MongoDB 的最佳实践，请参阅合作伙伴网络博客AWS 上的帖子。

操作说明

发现与评测

Task	描述	所需技能
确定关系数据库的参数和大小。	使用关系迁移器建议和中的总索引空间信息，估算工作集大小。`db.stats()` 假设您的数据空间中有一定比例会被频繁访问。此任务大约需要一周时间。有关此故事和本长篇故事中其他故事的更多信息和示例，请参阅相关资源部分。	应用程序所有者，数据库管理员
估计网络带宽要求。	要估计网络带宽要求，请将平均文档大小乘以每秒提供的文档数量。考虑集群中任何节点将承受的最大流量作为基础。要计算从集群到客户端应用程序的下游数据传输速率，请使用一段时间内返回的文档总数的总和。如果您的应用程序从辅助节点读取数据，请将文档总数除以可以提供读取操作的节点数。要查找数据库的平均	数据库管理员

Task	描述	所需技能
	文档大小，请使用`db.stats().avgObjSize` 命令。此任务通常需要一天的时间。	
选择 Atlas 层。	按照 MongoDB 文档中的说明选择正确的 Atlas 集群层。	数据库管理员
为转换做好计划。	制定应用程序割接计划。	数据库管理员、应用程序所有者

在上设置新的 MongoDB Atlas 环境 AWS

Task	描述	所需技能
在上创建一个新的 MongoDB Atlas 集群。 AWS	在 MongoDB Atlas 中，选择构建集群。在 “创建新集群” 对话框中，选择 AWS 作为云提供商。	数据库管理员
选择 AWS 区域 和全局群集配置。	从适用于您的 Atlas 集群 AWS 区域 的列表中进行选择。如果需要，请配置全局集群。	数据库管理员
选择集群层。	选择您的首选集群层。您的层选择决定了内存、存储和 IOPS 规格等因素。	数据库管理员
配置其他集群设置。	配置其他集群设置，例如 MongoDB 版本、备份和加密选项。有关这些选项的更多信息，请参阅相关资源部分。	数据库管理员

配置安全性和合规性

Task	描述	所需技能
配置访问列表。	要连接到 Atlas 集群，您必须在项目的访问列表中添加一个条目。Atlas 使用 TLS/SSL 来加密数据库与虚拟私有云 (VPC) 的连接。要设置项目的访问列表以及有关此长篇故事的更多信息，请参阅 “相关资源” 部分。	数据库管理员
对用户进行身份验证和授权。	您必须创建并验证将访问 MongoDB Atlas 集群的数据库用户。要访问项目中的集群，用户必须属于该项目，并且可以属于多个项目。	数据库管理员
创建自定义角色。	（可选）Atlas 支持在内置的 Atlas 数据库用户权限未涵盖您想要的权限集的情况下创建自定义角色。	数据库管理员
设置 VPC 对等连接。	（可选）Atlas 支持与其他开启的 VPC VPCs 对等。 AWS	AWS 管理员
设置 AWS PrivateLink 终端节点。	（可选）您可以使用在上 AWS 设置私有终端节点 AWS PrivateLink。有关更多信息，请参阅 Amazon VPC 文档。	AWS 管理员
启用双因素身份验证。	（可选）Atlas 支持双因素身份验证 (2FA)，以帮助用户控制对其 Atlas 账户的访问。	AWS 管理员

Task	描述	所需技能
使用 LDAP 设置用户身份验证和授权。	（可选）Atlas支持使用轻量级目录访问协议 (LDAP) 执行用户身份验证和授权。	数据库管理员
设置统一 AWS 接入。	（可选）某些 Atlas 功能（包括 Atlas Data Lake 和使用客户密钥管理进行静态加密）使用 AWS Identity and Access Management (IAM) 角色进行身份验证。	AWS 管理员
使用设置静态加密 AWS KMS。	（可选）Atlas 支持使用 AWS Key Management Service (AWS KMS) 加密存储引擎和云提供商备份。	AWS 管理员
设置客户端字段级加密。	（可选）Atlas 支持客户端字段级加密，包括字段的自动加密。	AWS 管理员

迁移数据

Task	描述	所需技能
将 MongoDB 关系迁移器添加到访问列表中。	将关系迁移器添加到源数据库的访问列表中。这有助于准备源环境以连接到目标 Atlas 集群。	数据库管理员
评估关系数据库对象。	启动 MongoDB 关系迁移器并连接到您的关系数据库。开始评估。	数据库管理员
接受迁移模式或根据您的业务需求选择对其进行更改。	要么接受 Relational Migrator 根据初始评估和性能参数推荐	数据库管理员

Task	描述	所需技能
	的数据库模式，要么选择根据您的业务需求对其进行更改。	
在 MongoDB Atlas 中启动目标副本集。	在 MongoDB Atlas 中启动目标副本集。在关系迁移器中，选择我已准备好迁移。	数据库管理员

配置操作集成

Task	描述	所需技能
连接到 MongoDB Atlas 集群。	确保 MongoDB Atlas 集群连接按预期运行。	应用程序所有者
与集群数据交互。	验证集群数据。	数据库管理员
监控您的集群。	验证您的集群设置是否正确。	数据库管理员
备份和恢复集群数据。	定期为群集数据安排备份。	数据库管理员

相关的资源

除非另有说明，否则以下所有链接均指向 MongoDB 文档中的网页。

迁移指南

- 在（规范性指南）上迁移到 MongoDB Atla AWS sAWS

发现与评测

- 内存
- 使用 Atlas 示例数据集进行大小调整示例
- 移动应用程序的大小调整示例
- 网络流量
- 集群自动扩缩

- Atlas 大小调整模板

配置安全性和合规性

- 配置 IP 访问列表条目
- 配置数据库用户
- 配置对 Atlas 用户界面的访问权限
- 配置自定义数据库角色
- 配置数据库用户
- 设置网络对等连接
- 了解 Atlas 中的私有端点
- 管理您的多重身份验证选项
- 使用 LDAP 设置用户身份验证和授权
- Atlas 数据湖
- 使用客户密钥管理进行静态加密
- 代入角色的方法（IAM 文档）
- 客户端字段级加密
- 自动加密
- MongoDB Atlas 安全控制
- MongoDB 信任中心
- 为集群配置安全功能

在上设置新的 MongoDB Atlas 环境 AWS

- 云提供商和地区
- 管理全球集群
- 选择集群层
- 配置其他设置
- 开始使用 Atlas
- 配置对 Atlas 用户界面的访问权限
- 管理集群

迁移数据

- 迁移或导入数据

监控集群

- 监控您的集群

集成操作

- 连接到集群
- 与您的数据互动
- 监控您的集群
- 备份、恢复和存档数据

博客文章

- 使用 MongoDB 文档模型实现 RDBMS 架构的现代化

将自托管 MongoDB 环境迁移到 MongoDB Atlas AWS

由 Battulga Purevragchaa (AWS)、Babu Srinivasan (MongoDB) 和 Igor Alekseev (AWS) 创作

摘要

此模式描述了从自我管理的 MongoDB 环境（包括 MongoDB 社区服务器、企业服务器、企业高级、mLab 或任何托管 MongoDB 集群）迁移到 MongoDB Atlas 的步骤。 AWS Cloud它使用 Atlas Live Migration Service 来帮助加速从 MongoDB 到 MongoDB Atlas 的数据迁移。

该模式与规范性指导网站上的《迁移到 MongoDB Atlas AWS》 AWS 指南一起出现。它提供了该指南中讨论的其中一个迁移场景的实施步骤。有关其他迁移方案，请参阅 AWS 规范性指导网站上的以下模式：

- 将关系数据库迁移到 MongoDB Atlas AWS
- 将数据从 IBM Db2、SAP、Sybase 和其他数据库流式传输到 MongoDB Atlas AWS

该模式适用于AWS 系统集成商 (SI) 合作伙伴和 AWS 用户。

先决条件和限制

先决条件

- 要迁移到 MongoDB Atlas 的源 MongoDB 企业高级版、社区服务器或其他自我管理的 MongoDB 环境。
- 熟悉 MongoDB、MongoDB Atlas 和。 AWS 服务这种模式从较高的层面解释了一些迁移步骤。更多细节将在 future 版本中添加。

产品版本

- MongoDB 版本 6.0.13 或更高版本

架构

下图显示了 Atlas 实时迁移服务，该服务用于将数据从 MongoDB Enterprise Advanced 数据库和 MongoDB 社区数据库迁移到 MongoDB Atlas 上。 AWS如果您需要将大型复杂数据库迁移到 MongoDB Atlas，同时最大限度地减少停机时间并实现持续的数据同步，请使用此服务。此模式使用 Atlas 实时迁移服务。

下图显示了 MongoDB 镜像服务 mongomirror ()，您还可以使用该服务通过安全连接将数据从 MongoDB 企业高级数据库和 MongoDB 社区数据库迁移到 MongoDB Atlas。 AWS AWS PrivateLinkmongomirror用于在本地 MongoDB 和 MongoDB Atlas 之间进行持续的数据复制。此工具非常适合灾难恢复或分阶段迁移，但不在此模式的范围之内。

有关更多支持不同使用场景的 MongoDB Atlas 参考架构，请参阅规范性指南网站上的迁移到 MongoDB Atlas。 AWS AWS

工具

- MongoDB Atlas 是一种完全托管的数据库即服务 (DBaaS)，用于在云中部署和管理 MongoDB 数据库。
- Atlas 实时迁移服务是一款免费的 MongoDB 实用工具，可帮助将数据库迁移到 Atlas。此服务使源数据库与目标数据库保持同步，直到直接割接。当您准备好割接时，您可以停止应用程序实例，将它们指向目标 Atlas 集群，然后重新启动它们。要访问此服务，请从 MongoDB Atlas 集群中选择数据库选项。
- mongomirror 是一种用于手动将数据从现有 MongoDB 副本集迁移到 MongoDB Atlas 副本集的工具。 mongomirror不要求您关闭现有的副本集或应用程序，也不需要导入用户或角色数据，也不需要复制配置数据库。你可以mongomirror从 MongoDB 文档中下载。

最佳实践

有关在 AWS上使用 MongoDB 的最佳实践，请参阅合作伙伴网络博客AWS 上的帖子。

操作说明

发现与评测

Task	描述	所需技能
确定集群大小。	使用中的信息计算总索引空间，估算工作集大小。db.stats() 假设您的数据空间中有一定比例会被频繁访问。或者，您可以根据自己	数据库管理员、应用程序所有者

Task	描述	所需技能
	的假设来估计内存需求。此任务大约需要一周时间。有关此故事和本长篇故事中其他故事的更多信息和示例，请参阅相关资源部分。	
估计网络带宽要求。	要估计网络带宽要求，请将平均文档大小乘以每秒提供的文档数量。考虑集群中任何节点将承受的最大流量作为基础。要计算从集群到客户端应用程序的下游数据传输速率，请使用一段时间内返回的文档总数的总和。如果您的应用程序从辅助节点读取数据，请将文档总数除以可以提供读取操作的节点数。要查找数据库的平均文档大小，请使用`db.stats().avgObjSize` 命令。此任务通常需要一天的时间。	数据库管理员
选择 Atlas 层。	按照 MongoDB 文档中的说明选择正确的 Atlas 集群层。	数据库管理员
为转换做好计划。	制定应用程序割接计划。	数据库管理员、应用程序所有者

在 AWS 上设置新的 MongoDB Atlas 环境

Task	描述	所需技能
在上创建新的 MongoDB Atlas 集群。 AWS	登录 Atlas 并打开项目的 “概述” 页面。选择 “创建” 按钮创建集群。有关更多信息，请参阅 MongoDB 文档。	数据库管理员

Task	描述	所需技能
选择 AWS 区域 和全局群集配置。	从适用于您的 Atlas 集群 AWS 区域 的列表中进行选择。如果需要，请配置全局集群。有关更多信息，请参阅 MongoDB 文档。	数据库管理员
选择集群层。	选择您的首选集群层。您的层选择决定了内存、存储和 IOPS 规格等因素。	数据库管理员
配置其他集群设置。	配置其他集群设置，例如 MongoDB 版本、备份和加密选项。有关这些选项的更多信息，请参阅相关资源部分。	数据库管理员

配置安全性和合规性

Task	描述	所需技能
对用户进行身份验证和授权。	您必须创建并验证将访问 MongoDB Atlas 集群的数据库用户。要访问项目中的集群，用户必须属于该项目，并且他们可以属于多个项目。 Atlas 还支持基于 AWS Identity and Access Management (IAM) 的身份验证。有关更多信息，请参阅 MongoDB 文档。	数据库管理员
创建自定义角色。	（可选）Atlas 支持在内置的 Atlas 数据库用户权限未涵盖您想要的权限集的情况下创建自定义角色。	数据库管理员

Task	描述	所需技能
设置 VPC 对等连接。	（可选）Atlas 支持虚拟私有云 (VPC) 与其他开启的 VPCs 对 AWS等。	AWS 管理员
设置 AWS PrivateLink 终端节点。	（可选）您可以使用在上 AWS 设置私有终端节点 AWS PrivateLink。有关更多信息，请参阅 Amazon VPC 文档。	AWS 管理员
启用双因素身份验证。	（可选）Atlas 支持双因素身份验证 (2FA)，以帮助用户控制对其 Atlas 账户的访问。	AWS 管理员
使用 LDAP 设置用户身份验证和授权。	（可选）Atlas支持使用轻量级目录访问协议 (LDAP) 执行用户身份验证和授权。	AWS 管理员
设置统一 AWS 接入。	（可选）某些 Atlas 功能（包括 Atlas Data Lake 和使用客户密钥管理进行静态加密）使用 IAM 角色进行身份验证。	AWS 管理员
使用设置静态加密 AWS KMS。	（可选）Atlas 支持使用 AWS Key Management Service (AWS KMS) 加密存储引擎和云提供商备份。	AWS 管理员
设置客户端字段级加密。	（可选）Atlas 支持客户端字段级加密，包括字段的自动加密。	AWS 管理员

迁移数据

Task	描述	所需技能
在 MongoDB Atlas 中选择您的目标副本集。	导航到目标 Atlas 集群并选择省略号 (...) 按钮。在群集列表中，此按钮显示在群集名称下方。在集群详细信息中，按钮显示在右侧，在 “Connect” 和 “配置” 按钮旁边。有关更多信息，请参阅 MongoDB 文档。	数据库管理员
将 Atlas 实时迁移服务添加到访问列表中。	将 Atlas 实时迁移服务添加到 AWS 源集群的访问列表中。这有助于准备源环境以连接到目标 Atlas 集群。	数据库管理员
使用 Atlas 实时迁移服务执行迁移。	选择 “开始迁移”。当 “准备直接转换” 按钮变为绿色时，执行直接转换。查看 Atlas 集群性能指标。考虑更新所有应用程序层中的数据库连接以指向新数据库。	数据库管理员

配置操作集成

Task	描述	所需技能
连接到 MongoDB Atlas 集群。	确保 MongoDB Atlas 集群连接按预期运行。	应用程序所有者
与集群数据交互。	测试集群数据。	数据库管理员
监控您的集群。	验证您的集群设置是否正确。	数据库管理员
备份和恢复集群数据。	定期为群集数据安排备份。	数据库管理员

故障排除

事务	解决方案

错误：无法访问指定来源	• 确保在源集群上的 IP 访问列表中添加了正确的子网范围。您可以在实时迁移模式窗口中找到四个必需的子网范围。
	• 确认您指定的主机名解析为公有 IP 地址。在命令提示符处，使用以下命令之一：
	`nslookup <hostname>`
	`ping <hostname>`
	• 确保您未使用与实时拉取迁移不兼容的 VPC 对等连接。如果 VPC 对等连接是您的唯一选择，请mongomirror 改用。

错误：无法解析主机名	找不到给定主机名的 IP 地址。确认给定的主机名正确且可公开访问。
任何其他错误	如果您遇到任何其他错误，请参阅 MongoDB 文档中的实时迁移疑难解答（Pull）。

相关资源

除非另有说明，否则以下所有链接均指向 MongoDB 文档中的网页。

迁移指南

- 在（规范性指南）上迁移到 MongoDB Atla AWS sAWS

旧版迁移

- 迁移旧版本的 MongoDB

发现与评测

- 内存

- 使用 Atlas 示例数据集进行大小调整示例
- 移动应用程序的大小调整示例
- 网络流量
- 集群自动扩缩
- Atlas 大小调整模板

配置安全性和合规性

- 配置 IP 访问列表条目
- 配置数据库用户
- 配置对 Atlas 用户界面的访问权限
- 配置自定义数据库角色
- 配置数据库用户
- 设置网络对等连接
- 在 Atlas 中了解私有端点
- 管理您的多重身份验证选项
- 使用 LDAP 设置用户身份验证和授权
- Atlas 数据湖
- 使用客户密钥管理进行静态加密
- 代入角色的方法（IAM 文档）
- 客户端字段级加密
- 自动加密
- MongoDB Atlas 安全控制
- MongoDB 信任中心
- 为集群配置安全功能

在上设置新的 MongoDB Atlas 环境 AWS

- 云提供商和地区
- 管理全球集群
- 选择集群层
- 配置其他设置

- 开始使用 Atlas
- 配置对 Atlas 用户界面的访问权限
- 管理集群

迁移数据

- 迁移或导入数据

监控集群

- 监控您的集群

集成操作

- 连接到集群
- 与您的数据互动
- 监控您的集群
- 备份、恢复和存档数据

训练

- 使用 MongoDB Atlas 进行实时迁移

其他信息

有关更多信息，请参阅 MongoDB 文档中的以下主题：

- 要将数据移动到无服务器实例，请使用 Compass 导出和导入数据，或者使用自管理工具迁移数据。要了解更多信息，请参阅无服务器实例限制。
- 要将数据加载到 Atlas 中的新集群，请参阅将数据加载到 Atlas 中。
- 要为测试目的制作集群副本，请参阅自管理部署的 Backup 方法。
- 如果您要迁移的应用程序需要近乎持续的正常运行时间，请联系 MongoDB Supp ort 并分享您的正常运行时间要求和集群配置。
- 有关更多信息，请参阅迁移或导入数据。

使用 Oracle 数据泵和 AWS DMS 将 Oracle JD Edwards EnterpriseOne 数据库迁移到 AWS

创建者：Thanigaivel Thirumalai (AWS)

摘要

你可以在亚马逊关系 EnterpriseOne 数据库服务 (Amazon RDS) 上迁移和运行你的 JD Edwards 数据库。当您将数据库迁移到 Amazon RDS 时，AWS 可以负责备份任务和高可用性设置，因此您可以集中精力维护 EnterpriseOne 应用程序及其功能。有关迁移过程中需要考虑的关键因素完整列表，请参阅 AWS Prescriptive Guidance 中的 Oracle 数据库迁移策略。

有多种方法可以迁移 EnterpriseOne 数据库，包括：

- 使用 Oracle Universal Batch Engine (UBE) R98403 创建架构和表，使用 AWS Database Migration Service (AWS DMS) 迁移
- 使用数据库原生工具创建架构和表，使用 AWS DMS 迁移
- 使用数据库原生工具迁移现有数据（完全加载），使用 AWS DMS 执行变更数据捕获 (CDC) 任务

此模式涵盖了第三个选项。它解释了如何使用带有 AWS DMS 的 Oracle 数据泵及其 CDC 功能，将您的本地 EnterpriseOne 数据库迁移到 Amazon R DS for Oracle。

Oracle JD Edwards EnterpriseOne 是一款企业资源规划 (ERP) 解决方案，适用于制造、构造、分销、维修或管理产品或实物资产的组织。JD Edwards EnterpriseOne 支持各种硬件、操作系统和数据库平台。

在迁移 JD Edwards 等关键 ERP 应用程序时 EnterpriseOne，最大限度地减少停机时间是关键。AWS DMS 支持从源数据库到目标数据库的满负荷和连续复制，可最大限度地减少停机时间。AWS DMS 还为迁移提供实时监控和日志记录，可帮助您识别并解决任何可能导致停机的问题。

使用 AWS DMS 复制更改时，必须指定时间或系统更改号 (SCN) 作为从数据库日志中读取更改的起点。为了确保 AWS DMS 可以访问这些更改，请务必在指定的时间内（我们推荐 15 天）保持这些日志在服务器上的可访问性。

先决条件和限制

先决条件

- 已在您的 Amazon Web Services Cloud 环境中预置为目标数据库的 Amazon RDS for Oracle 数据库。有关说明，请参阅 Amazon RDS 文档。

- 在本地运行或在 AWS 上的亚马逊弹性计算云 (Amazon EC2) 实例上运行 EnterpriseOne 的数据库。

> **Note**
>
> 这种模式专为从本地迁移到 AWS 而设计，但它是通过在 EC2 实例上使用 EnterpriseOne 数据库进行测试的。如果计划从本地环境迁移，则必须配置适当网络连接。

- 架构详细信息。确定您计划迁移到哪个 Oracle 数据库架构（例如 DV92 0）EnterpriseOne。在开始迁移进程前，请收集有关架构的以下详细信息：
  - 架构大小
  - 每种对象类型的对象数量
  - 无效对象数量

限制

- 您必须在目标 Amazon RDS for Oracle 数据库上创建任何您想要的架构，AWS DMS 不会为您创建此架构。（操作说明部分描述了如何使用 Data Pump 导出和导入架构。）必须已存在目标 Oracle 数据库的架构名称。来自源架构的表导入到用户或架构，AWS DMS 使用管理员或系统账号连接到目标实例。若要迁移多个架构，可以创建多个复制任务。您还可以将数据迁移至目标实例上的不同架构。为此，请对 AWS DMS 表映射使用架构转换规则。
- 此模式已使用演示数据集测试。建议验证数据集和自定义兼容性。
- 这种模式使用在微软 Windows 上运行 EnterpriseOne 的数据库。但是，您可在 AWS DMS 支持的其他操作系统中使用相同的进程。

架构

下图显示了一个以 Oracle 数据库作为源数据库、 EnterpriseOne 在 Amazon RDS for Oracle 数据库上作为目标数据库运行的系统。该数据通过使用 Oracle Data Pump 从源 Oracle 数据库导出，并导入至目标 Amazon RDS for Oracle 数据库，并使用 AWS DMS 复制 CDC 更改。

1. Oracle Data Pump 从源数据库提取数据，并将数据发送至 Amazon RDS for Oracle 数据库目标。
2. CDC 数据将从 AWS DMS 中的源数据库发送至源端点。
3. 数据从源端点发送至 AWS DMS 复制实例，在此执行复制任务。
4. 复制任务完成后，数据将发送至 AWS DMS 中的目标端点。

5. 数据将从目标端点发送至 Amazon RDS for Oracle 数据库实例。

工具

Amazon Web Services

- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到 Amazon Web Services Cloud，或者在云和本地设置的组合之间迁移。
- Amazon Relational Database Service (Amazon RDS) for Oracle 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展 Oracle 关系数据库。

其他服务

- Oracle Data Pump 可帮助您将数据和元数据从一个数据库高速移动至另一个数据库。

最佳实践

Migrating LOBs

如果您的源数据库包含需要迁移到目标数据库的大型二进制对象 (LOBs)，AWS DMS 会提供以下选项：

- 完整 LOB 模式 — AWS DMS 会将所有数据 LOBs 从源数据库迁移到目标数据库，无论其大小如何。尽管迁移速度比其他模式慢，但其优点是数据不会被截断。为了提高性能，您可以在新的复制实例上创建单独的任务，以迁移大 LOBs 于几兆字节的表。
- 受限 LOB 模式 — 您可以指定 LOB 列数据的最大大小，这样 AWS DMS 就可以预先分配资源并批量应用。 LOBs 如果 LOB 列的大小超过任务中指定的大小，AWS DMS 会截断数据，并向 AWS DMS 日志文件发送警告。如果您的 LOB 数据大小在有限的 LOB 大小之内，则可通过使用受限 LOB 模式提高性能。
- 内联 LOB 模式 — 您可以通过复制小型和大型来进行迁移， LOBs 而不会截断数据，也不会降低任务的性能。 LOBs首先，为 `InlineLobMaxSize` 参数指定一个值，该值仅在完全 LOB 模式时设置为 `true` 时可用。AWS DMS 任务会传输较小的 LOBs 内联，这样效率更高。然后，AWS DMS LOBs 通过从源表中执行查找来迁移大型数据。但是，内联 LOB 模式仅适用于完全加载阶段。

生成序列值

在 AWS DMS CDC 过程中，不从源数据库中复制增量序列号。为避免序列值存在差异，您必须从数据来源中为所有序列生成最新的序列值，并将其应用至目标 Amazon RDS for Oracle 数据库。

AWS Secrets Manager

若要帮助管理凭证，我们建议您按照博客文章使用 AWS Secrets Manager 管理 AWS DMS 端点凭证中的说明操作。

性能

- 复制实例 – 有关选择最佳实例大小的指导，请参阅 AWS DMS 文档中的为复制实例选择最佳大小。
- 连接选项 – 为避免延迟问题，我们建议您选择正确的连接选项。AWS Direct Connect 可提供通往 AWS 资源的最短路径，其为企业数据中心与 AWS 之间的专用连接。在传输过程中，您的网络流量仍保留在 AWS 全球网络中，且不会通过 Internet 传输。相比使用 VPN 或公共互联网，减少了遇到瓶颈或者延迟意外增加的机会。
- 网络带宽 – 如果想要优化性能，请验证您的网络吞吐量的快慢。如果在本地源数据库与 AWS DMS 之间使用 VPN 隧道，请确保带宽足以承载您的工作负载。
- 任务并行性 – 您可以通过在完全加载期间并行加载多个表来加快数据复制速度。此模式使用了 RDBMS 端点，故此选项仅适用于完全加载进程。任务并行度由 `MaxFullLoadSubTasks` 参数控制，该参数决定并行运行的满负荷子任务的数量。默认情况下，此参数设置为 8，这意味着在完整模式下将一起加载八个表（如果在表映射中选中）。您可以在任务的 JSON 脚本的完全加载任务设置部分调整此参数。
- 表并行度 – AWS DMS 还允许您使用多个并行线程加载单个大表。这对于具有数十亿条记录以及多个分区和子分区的 Oracle 源表特别有用。如果源表未分区，则可以使用列边界进行并行加载。
- 拆分负载 – 当您将负载拆分至多个任务或 AWS DMS 实例时，请在捕获更改时记住交易边界。

操作说明

使用 Oracle 数据泵导出 EnterpriseOne 架构

Task	描述	所需技能
生成 SCN。	当源数据库处于活动状态并被 EnterpriseOne 应用程序使用时，使用 Oracle Data Pump 启动数据导出。您必须首先从源数据库生成系统更改号	数据库管理员

Task	描述	所需技能

	(SCN)，以便在使用 Oracle Data Pump 导出期间保持数据一致性，并作为 AWS DMS 中 CDC 的起点。
	
	若要从源数据库生成当前 SCN，请使用以下 SQL 语句：
	
	`SQL> select current_scn from v$database;`
	
	`CURRENT_SCN`
	`-----------`
	`30009727`
	
	保存已生成 SCN。您将在导出数据与创建 AWS DMS 复制任务时使用此 SCN。	

Task	描述	所需技能

创建参数文件。	要创建用于导出架构的参数文件，您可使用以下代码。
	`directory=DMS_DATA_PUMP_DIR logfile=export_dms.log dumpfile=export_dms_data.dmp schemas=<schema name> flashback_scn=<SCN from previous command>`
	Note
	您也可以根据自己的要求使用以下命令DATA_PUMP_DIR 来定义自己的命令。
	`SQL> CREATE OR REPLACE DIRECTORY DMS_DATA_PUMP_DIR AS '<Directory for dump>'; Directory created.`
	`SQL> GRANT READ, WRITE ON DIRECTORY DMS_DATA_PUMP_DIR TO SYSTEM; Grant succeeded.`
		数据库管理员

Task	描述	所需技能
导出架构。	若要执行导出，请使用如下 expdp 实用程序： C:\Users\Administrator>expdp system/********@<DB Name> PARFILE='<Path to PAR file create above>' Export: Release 19.0.0.0.0 - Production on *** *** ** **:**:** **** Version 19.3.0.0.0 Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved. Connected to: Oracle Database 19c Standard Edition 2 Release 19.0.0.0.0 - Production Starting "SYSTEM". "SYS_EXPORT_SCHEMA_02": system/**********@<DB Name>PARFILE='E:\exp_dms_datapump.par' Processing object type SCHEMA_EXPORT/TABLE/TABLE_DATA Processing object type SCHEMA_EXPORT/TABLE/INDEX/STATISTICS/INDEX_STATISTICS Processing object type SCHEMA_EXPORT/TABL	数据库管理员

Task	描述	所需技能
	E/STATISTICS/TABLE _STATISTICS Processing object type SCHEMA_EXPORT/STAT ISTICS/MARKER Processing object type SCHEMA_EXPORT/USER Processing object type SCHEMA_EXPORT/ROLE _GRANT Processing object type SCHEMA_EXPORT/DEFA ULT_ROLE Processing object type SCHEMA_EXPORT/TABL ESPACE_QUOTA Processing object type SCHEMA_EXPORT/PRE_ SCHEMA/PROCACT_SCHEMA Processing object type SCHEMA_EXPORT/TABLE/ TABLE Processing object type SCHEMA_EXPORT/TABL E/GRANT/OWNER_GRANT/ OBJECT_GRANT Processing object type SCHEMA_EXPORT/TABLE/ INDEX/INDEX Processing object type SCHEMA_EXPORT/TABLE/ CONSTRAINT/CONSTRAINT . . exported "<Schema Name>"."<Table Name>" 228.9 MB 496397 rows Master table "SYSTEM". "SYS_EXPORT_SCHEMA _02" successfully loaded/unloaded	

Task	描述	所需技能
	``` ********** ********** ********** ********** **** Dump file set for SYSTEM.SYS_EXPORT_ SCHEMA_02 is: E:\DMSDUMP\EXPORT_ DMS_DATA.DMP Job "SYSTEM"."SYS_EXPO RT_SCHEMA_02" successfully completed at *** *** ** **:**:** **** elapsed 0 00:01:57 ```	

使用 Oracle 数据泵导入 EnterpriseOne 架构

Task	描述	所需技能
将转储文件传输至目标实例。	若要使用 DBMS_FILE_TRANSFER 实用程序传输文件，您需要创建从源数据库到 Amazon RDS for Oracle 实例的数据库链接。建立链接后，您可以使用该实用程序将 Data Pump 文件直接传输至 Amazon RDS 实例。 或者，您可以将 Data Pump 文件传输至 Amazon Simple Storage Service (Amazon S3)，然后将其导入至 Amazon RDS for Oracle 实例。有关该选项的更多信息，请参阅其他信息部分。	数据库管理员

Task	描述	所需技能

	若要创建一个用于连接到位于目标数据库实例中的 Amazon RDS 主用户的数据库链接 ORARDSDB，请在源数据库上运行以下命令：
	
	`sqlplus / as sysdba`
	
	`SQL*Plus: Release 19.0.0.0.0 on *** *** ** **:**:** **** Version 19.3.0.0.0`
	
	`Copyright (c) 1982, 2019, Oracle. All rights reserved.`
	
	`Connected to: Oracle Database 19c Standard Edition 2 Release 19.0.0.0.0 Version 19.3.0.0.0`
	
	`SQL> create database link orardsdb connect to admin identifie d by "******" using '(DESCRIPTION = (ADDRESS = (PROTOCOL = TCP)(HOST = orcl.******.us-east-1.rds.amazonaws.com)(PORT = 1521))(CONNECT_DATA = (SERVER = DEDICATED ) (SERVICE_NAME = orcl)))';`
	
	`Database link created.`
	
	`SQL>`	

Task	描述	所需技能

测试数据库链接。	测试数据库链接，以确保您可以使用 `sqlplus` 连接至 Amazon RDS for Oracle 目标数据库。
	
	`SQL> select name from v$database@orardsdb;`
	
	`NAME`
	`---------`
	`ORCL`
		数据库管理员

Task	描述	所需技能

将转储文件传输至目标数据库。	若要将转储文件复制到 Amazon RDS for Oracle 数据库，您可以使用默认 DATA_PUMP_DIR 目录，也可以使用以下代码（必须在目标 Amazon RDS 实例上运行）创建自己的目录：
	`exec rdsadmin.rdsadmin_util.create_directory(p_directory_name => 'DMS_TARGET_PUMP_DIR');`
	`PL/SQL procedure successfully completed.`
	以下脚本使用名为 orardsdb 的数据库链接，将名为 EXPORT_DMS_DATA.DMP 的转储文件从源实例复制到目标 Amazon RDS for Oracle 数据库。您必须在源数据库实例上运行脚本。
	`BEGIN DBMS_FILE_TRANSFER.PUT_FILE( source_directory_object => 'DMS_DATA_PUMP_DIR', source_file_name => 'EXPORT_DMS_DATA.DMP', destination_directory_object => 'DMS_TARGET_PUMP_DIR',`
		数据库管理员

Task	描述	所需技能

	`destination_file_name => 'EXPORT_DMS_DATA.DMP', destination_database => 'orardsdb'); END;`
	
	`PL/SQL procedure successfully completed.`	

在目标数据库中列出转储文件。	PL/SQL 过程完成后，您可使用以下代码在 Amazon RDS for Oracle 数据库中列出数据转储文件： `select * from table (rdsadmin.rds_file_util.listdir(p_directory => 'DMS_TARGET_PUMP_DIR'));`	数据库管理员

Task	描述	所需技能

在目标实例中创建 JDE-特定用户。	在目标实例中，使用以下命令创建 JD Edwards 配置文件和角色：
	
	`SQL> CREATE PROFILE "JDEPROFILE" LIMIT IDLE_TIME 15;`
	`Profile created.`
	
	`SQL> CREATE ROLE "JDE_ROLE";`
	`Role created.`
	
	`SQL> CREATE ROLE "JDEADMIN";`
	`CREATE ROLE "JDEUSER";`
	`Role created.`
	`Role created.`
	
	授予角色所需权限：
	
	`SQL> GRANT CREATE ANY SEQUENCE TO JDE_ROLE;`
	`GRANT DROP ANY SEQUENCE TO JDE_ROLE;`
	`GRANT CREATE ANY TRIGGER TO JDE_ROLE;`
	`GRANT DROP ANY TRIGGER TO JDE_ROLE;`
		数据库管理员、JDE CNC

Task	描述	所需技能

在目标实例中创建表空间。	对此迁移所涉及的架构使用以下命令，在目标实例中创建所需表空间：
	
	`SQL> CREATE TABLESPACE <Tablespace Name for Tables>;`
	`Tablespace created.`
	
	`SQL> CREATE TABLESPACE <Tablespace Name for Indexes>;`
	`Tablespace created.`
		数据库管理员、JDE CNC

Task	描述	所需技能
在目标数据库上启动导入。	在开始导入进程前，请使用数据转储文件在目标 Amazon RDS for Oracle 数据库上设置角色、架构和表空间。 若要执行导入，请使用 Amazon RDS 主用户账户访问目标数据库，并使用 `tnsnames.ora` 文件中的连接字符串名称（其中包括 Amazon RDS for Oracle 数据库 `tns-entry` ）。如有必要，可以纳入重映射选项，将数据转储文件导入不同的表空间或使用不同架构名称。 若要开始导入，请使用下面的代码： `impdp admin@orardsdb directory=DMS_TARGET_PUMP_DIR logfile=import.log dumpfile=EXPORT_DMS_DATA.DMP` 为确保成功导入，请检查导入日志文件中是否存在任何错误，并查看对象数、行数和无效对象等详细信息。如果有任何无效对象，请重新编译它们。此外，比较源数据库对象和目标数据库对象，以确认它们是否匹配。	数据库管理员

预置 AWS DMS 复制实例，包括源端点和目标端点

Task	描述	所需技能
下载 模板。	下载 AWS CloudFormation DMS_Instance.yaml 模板以配置 AWS DMS 复制实例及其源和目标终端节点。	云管理员、数据库管理员
开始创建堆栈。	1. 登录 AWS 管理控制台，然后在 https://console.aws.amazon.com/cloudformation 上打开 AWS CloudFormation 控制台。 2. 选择创建堆栈。 3. 对于指定模板，请选择上传模板文件。 4. 选择选择文件。 5. 选择 `DMS_instance.yaml` 文件。 6. 选择下一步。	云管理员、数据库管理员
指定参数。	1. 在堆栈名称中，输入堆栈名称。 2. 对于 AWS DMS 实例参数，请输入以下参数： • DMSInstance类型 — 根据您的业务需求为 AWS DMS 复制实例选择所需的实例。 • DMSStorage大小 — 根据您的迁移大小输入 AWS DMS 实例的存储大小。 3. 对于源 Oracle 数据库配置，请输入以下参数：	云管理员、数据库管理员

Task	描述	所需技能
	• SourceOracleEndpointID-源 Oracle 数据库服务器的名称 • SourceOracleDatabaseName— 源数据库服务名称或会话 ID (SID)（如果适用） • SourceOracleUserName— 源数据库的用户名（默认为`system`） • SourceOracleDBPassword-源数据库用户名的密码 • SourceOracleDBPort— 源数据库端口 4. 对于适用于 Oracle 数据库配置的目标 RDS，请输入以下参数： • 目标RDSOracle终端节点ID-目标 RDS 数据库终端节点 • 目标 RDSOracle DatabaseName-目标 RDS 数据库名称 • 目标 RSOracle UserName-目标 RDS 用户名 • 目标 RDSOracle DBPassword-目标 RDS 密码 • TargetOracleDBPort— 目标 RDS 数据库端口	

Task	描述	所需技能
	5. 对于 VPC、子网和安全组配置，请输入以下参数： • VPCID – 适用于复制实例的 VPC • VPCSecurityGroupId— 复制实例的 VPC 安全组 • DMSSubnet1— 可用区 1 的子网 • DMSSubnet2— 可用区 2 的子网 6. 选择下一步。	
创建堆栈。	1. 在配置堆栈选项页面上，对于标签，输入任何可选值。 2. 选择 Next(下一步)。 3. 在查看页面上，验证详细信息，然后选择提交。 预置应在 5-10 分钟左右完成。当 AWS CloudFormation Stacks 页面显示 CREATE_COMPLETE 时，它就完成了。	云管理员、数据库管理员
设置端点。	1. 在 https://console.aws.amazon.com/dms/v2/ 上打开 AWS DMS 控制台。 2. 对于资源管理，请选择复制实例，然后查看复制实例。 3. 对于资源管理，请选择端点，然后查看端点。	云管理员、数据库管理员

Task	描述	所需技能
测试连接。	在源端点和目标端点显示为活动状态后，测试连接。为每个端点（源端点和目标端点）选择运行测试，以确保状态显示为成功。	云管理员、数据库管理员

为实时复制创建 AWS DMS 复制任务

Task	描述	所需技能
创建复制任务。	通过使用以下步骤创建 AWS DMS 复制任务： 1. 在 https://console.aws.amazon.com/dms/v 2/ 上打开 AWS DMS 控制台。 2. 在导航窗格的迁移数据 下，选择数据库迁移任务。 3. 在任务配置框中，为任务标识符输入任务标识符。 4. 对于复制实例，请选择您创建的 DMS 复制实例。 5. 对于源数据库端点，请选择源端点。 6. 对于目标数据库端点，请选择您的目标 Amazon RDS for Oracle 数据库。 7. 对于迁移类型，请选择仅复制数据更改。如果收到需要开启补充日志记录的消息，请按照故障排除部分中的说明进行操作。	云管理员、数据库管理员

Task	描述	所需技能
	8. 在任务设置 框中，选择指定日志序列号。 9. 对于系统更改号，请输入从源 Oracle 数据库生成的 Oracle 数据库 SCN。 10选择启用验证。 11选择 “启用 CloudWatch 日志”。 通过激活此功能，您可以验证数据和 A mazon CloudWatch 日志，以查看 AWS DMS 复制实例日志。 12.在选择规则下，完成以下操作： • 对于架构，请选择输入架构。 • 在架构名称中，输入 JDE 架构名称（例如：DV920）。 • 对于表名称，请输入 %。 • 对于操作，请选择包括。 13选择创建任务。 创建任务后，AWS DMS 会将持续更改从在 CDC 启动模式下提供的 SCN 迁移至 Amazon RDS for Oracle 数据库实例。您也可以通过查看 CloudWatch 日志来验证迁移。	

Task	描述	所需技能
重复复制任务。	重复前述步骤，为迁移进程中的其他 JD Edwards 架构创建复制任务。	云管理员、数据库管理员、JDE CNC 管理员

在目标 Amazon RDS for Oracle 数据库上验证数据库架构

Task	描述	所需技能
验证数据传输。	AWS DMS 任务启动后，您可查看任务页面上的表统计数据选项卡，以查看对数据所做的更改。 您可以在控制台的数据库迁移任务页面监控正在进行的复制的状态。 有关更多信息，请参阅 AWS DMS 数据验证。	云管理员、数据库管理员

割接

Task	描述	所需技能
停止复制。	停止复制过程，并停止源应用程序服务。	云管理员、数据库管理员
启动 JD Edwards 应用程序。	在 AWS 上启动目标 JD Edwards 演示和逻辑层应用程序，并将其定向至 Amazon RDS for Oracle 数据库。 在访问应用程序时，您应该会注意到，现已通过 Amazon	数据库管理员、JDE CNC 管理员

Task	描述	所需技能
	RDS for Oracle 数据库建立了所有连接。	
关闭源数据库。	在确认源数据库不再有其他连接后，可关闭源数据库。	数据库管理员

故障排除

事务	解决方案

您将收到一条警告消息，要求在源数据库中为进行中的复制启用补充日志记录	输入以下命令，以启用补充日志记录：
	`SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (ALL) COLUMNS;`
	`SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (PRIMARY KEY) COLUMNS;`
	`SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (UNIQUE) COLUMNS;`
	`SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (FOREIGN KEY) COLUMNS;`
	`SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (PRIMARY KEY) COLUMNS;`
	`SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (UNIQUE) COLUMNS;`

AWS DMS 已禁用补充日志记录。	在 AWS DMS 中默认关闭补充日志记录。若要为源 Oracle 端点将其打开： 1. 登录 AWS 管理控制台并在 https://console.aws.amazon.com/dms/v 2/ 上打开 AWS DMS 控制台。 2. 选择端点。 3. 选择要将补充日志记录添加到的 Oracle 源端点。 4. 选择 Modify(修改)。

事务	解决方案
	5. 选择高级，然后将以下代码添加到额外的连接属性文本框中： `addSupplementalLogging=Y` 6. 选择 Modify(修改)。

在 CDB 级别未启用补充日志记录。	1. 输入以下命令：
	`SQL> alter session set container = CDB$ROOT;`
	`Session altered.`
	2. 重复启用补充日志记录的步骤。

您会收到错误消息：“测试端点失败：应用程序状态：1020912，应用程序消息：Oracle PDB 环境 LogMiner 不支持端点初始化失败。”	如果遇到此错误消息，则可以改用 Binary Reader LogMiner。 在端点设置下，将此行添加至源数据库的额外连接属性： `useLogMinerReader=N;useBfile=Y;`

相关资源

- AWS Database Migration Service 入门
- AWS Database Migration Service 最佳实践
- 将 Oracle 数据库迁移至 AWS Cloud
- 适用于 AWS 的 AWS Database Migration Service 资源类型参考 CloudFormation
- 使用 AWS Secrets Manager 管理 AWS DMS 端点凭证
- AWS Database Migration Service 中的排除迁移任务
- AWS Database Migration Service 最佳实践

其他信息

使用 Amazon S3 传输文件

要将文件传输至 Amazon S3，您可以使用 AWS CLI 或 Amazon S3 控制台。将文件传输至 Amazon S3 后，您可以使用 Amazon RDS for Oracle 实例从 Amazon S3 导入 Data Pump 文件。

如果选择使用 Amazon S3 集成作为替代方法传输转储文件，请执行以下步骤：

1. 创建 S3 存储桶。
2. 使用 Oracle Data Pump 从源数据库导出数据。
3. 将 Data Pump 文件上传至 S3 存储桶。
4. 将 Data Pump 文件从 S3 存储桶下载至目标 Amazon RDS for Oracle 数据库。
5. 使用 Data Pump 文件执行导入。

> **Note**
>
> 要在 S3 和 RDS 实例之间传输大型数据文件，我们建议您使用 Amazon S3 传输加速功能。

使用 AWS DMS 将 Oracle PeopleSoft 数据库迁移到 AWS

创建者：sampath kathirvel (AWS)

摘要

Oracle PeopleSoft 是一款适用于企业级流程的企业资源规划 (ERP) 解决方案。 PeopleSoft 具有三层架构：客户端、应用程序和数据库。 PeopleSoft 可以在亚马逊关系数据库服务 (Amazon RDS) 上运行。

如果您将 Oracle 数据库迁移到 Amazon RDS，Amazon Web Services (AWS) 可以处理备份任务和高可用性，让您可以自由地专注于维护 PeopleSoft 应用程序及其功能。有关迁移过程中需要考虑的关键因素完整列表，请参阅 AWS Prescriptive Guidance 中的 Oracle 数据库迁移策略。

此模式提供了一种解决方案，用于使用 Oracle Data Pump 与 AWS Database Migration Service (AWS DMS) 及其更改数据捕获 (CDC) 功能将本地 Oracle 数据库迁移到 Amazon RDS for Oracle。

在迁移 Oracle 等关键 ERP 应用程序时 PeopleSoft，最大限度地减少停机时间是关键。AWS DMS 支持从源数据库到目标数据库的满负荷和连续复制，可最大限度地减少停机时间。AWS DMS 还为迁移提供实时监控和日志记录，这可帮助您识别和解决任何可能导致停机的问题。

使用 AWS DMS 复制更改时，必须指定时间或系统更改号 (SCN)，作为供 AWS DMS 从数据库日志中读取更改的起点。为了确保 AWS DMS 可以访问这些更改，请务必在指定的时间内保持这些日志在服务器上的可访问性。

先决条件和限制

先决条件

- 已在您的 Amazon Web Services Cloud 环境中预调配 Amazon RDS for Oracle 数据库作为目标数据库。
- 在本地运行或在 AWS 云中的亚马逊弹性计算云 (Amazon EC2) 上运行的 Oracle PeopleSoft 数据库。

> **Note**
>
> 此模式专为从本地迁移到 AWS 而设计，但已通过在 Amazon EC2 实例上使用 Oracle 数据库进行测试。要从本地迁移，您需要配置适当的网络连接。

- 架构详细信息。将 Oracle PeopleSoft 应用程序迁移到 Amazon RDS for Oracle 时，必须确定要迁移哪个 Oracle 数据库架构（例如SYSADM）。在开始迁移进程之前，请收集有关架构的以下详细信息：
  - 大小
  - 每种对象类型的对象数量
  - 无效对象数量。

  此信息将有助于迁移进程。

限制

- 此场景仅在 PeopleSoft DEMO 数据库中进行了测试。它尚未使用大型数据集进行测试。

架构

下图显示了一个实例，该实例将 Oracle 数据库作为源数据库，将 Amazon RDS for Oracle 数据库作为目标数据库进行运行。该数据通过使用 Oracle Data Pump 从源 Oracle 数据库导出，并导入至目标 Amazon RDS for Oracle 数据库，并使用 AWS DMS 复制 CDC 更改。

1. 初始步骤涉及使用 Oracle Data Pump 从源数据库提取数据，然后将其发送到 Amazon RDS for Oracle 数据库目标。
2. 数据将从 AWS DMS 中的源数据库发送至源端点。
3. 数据从源端点发送至 AWS DMS 复制实例，在此执行复制任务。
4. 复制任务完成后，数据将发送至 AWS DMS 中的目标端点。
5. 数据将从目标端点发送至 Amazon RDS for Oracle 数据库实例。

工具

Amazon Web Services

- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到 Amazon Web Services Cloud，或者在云和本地设置的组合之间迁移。
- Amazon Relational Database Service (Amazon RDS) for Oracle 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展 Oracle 关系数据库。

其他服务

- Oracle 数据泵 可帮助您将数据和元数据从一个数据库高速移动至另一个数据库。

最佳实践

Migrating LOBs

如果您的源数据库包含需要迁移到目标数据库的大型二进制对象 (LOBs)，AWS DMS 会提供以下选项：

- 完整 LOB 模式 — AWS DMS 会将所有数据 LOBs 从源数据库迁移到目标数据库，无论其大小如何。尽管迁移速度比较慢，但其优点是数据不会被截断。为了提高性能，您可以在新的复制实例上创建单独的任务来迁移 LOBs 大于几兆字节的表。
- 受限 LOB 模式 — 您可以指定 LOB 列数据的最大大小，这样 AWS DMS 就可以预先分配资源并批量应用。 LOBs 如果 LOB 列的大小超过任务中指定的大小，AWS DMS 会截断数据，并向 AWS DMS 日志文件发送警告。如果您的 LOB 数据大小在有限的 LOB 大小之内，则可通过使用受限 LOB 模式提高性能。
- 内联 LOB 模式 — 您可以通过复制小型和大型来进行迁移， LOBs 而不会截断数据，也不会降低任务的性能。 LOBs首先，为 InlineLobMaxSize 参数指定一个值，该值仅在完整 LOB 模式设置为 true 时可用。AWS DMS 任务会传输较小的 LOBs 内联，这样效率更高。然后，AWS DMS LOBs 通过从源表中执行查找来迁移大型数据。但是，内联 LOB 模式仅适用于完全加载阶段。

生成序列值

请记住，在使用 AWS DMS 进行更改数据捕获过程中，不会从源数据库复制增量序列号。为避免序列值存在差异，您必须从数据来源中为所有序列生成最新的序列值，并将其应用至目标 Amazon RDS for Oracle 数据库。

凭证管理

为了帮助保护您的 AWS 资源，我们建议遵循 AWS Identity and Access Management (IAM) 的最佳实践。

操作说明

预置 AWS DMS 复制实例，包括源端点和目标端点

Task	描述	所需技能
下载 模板。	下载 dms_instance.yaml AWS CloudFormation 模板以配置 AWS DMS 复制实例及其源和目标终端节点。	云管理员、数据库管理员
开始创建堆栈。	1. 在 AWS 管理控制台上，选择CloudFormation。 2. 选择创建堆栈。 3. 对于指定模板，请选择上传模板文件。 4. 选择选择文件。 5. 选择 `DMS_instance.yaml` 文件。 6. 选择下一步。	云管理员、数据库管理员
指定参数。	1. 在堆栈名称中，输入堆栈名称。 2. 在 AWS DMS 实例参数下方，输入以下参数： • DMSInstance类型 — 根据您的业务需求为 AWS DMS 复制实例选择所需的实例。 • DMSStorage大小 — 根据您的迁移大小输入 AWS DMS 实例的存储大小。 3. 在源 Oracle 数据库配置下方，输入以下参数：	云管理员、数据库管理员

Task	描述	所需技能
	• SourceOracleEndpointID-源 Oracle 数据库服务器的名称 • SourceOracleDatabaseName— 源数据库服务名称或会话 ID (SID)（如果适用） • SourceOracleUserName— 源数据库用户名（默认为 system） • SourceOracleDBPassword-源数据库用户名的密码 • SourceOracleDBPort— 源数据库端口 4. 在 Oracle 数据库配置的目标 RDS 下方，输入以下参数： • 目标RDSOracle终端节点ID-目标 RDS 数据库终端节点 • 目标 RDSOracle DatabaseName-目标 RDS 数据库名称 • 目标 RSOracle UserName-目标 RDS 用户名 • 目标 RDSOracle DBPassword-目标 RDS 密码 • TargetOracleDBPort— 目标 RDS 数据库端口	

Task	描述	所需技能
	5. 在 VPC、子网和安全组配置下方，请输入以下参数： • VPCID – 适用于复制实例的 VPC • VPCSecurityGroupId— 复制实例的 VPC 安全组 • DMSSubnet1— 可用区 1 的子网 • DMSSubnet2— 可用区 2 的子网 6. 选择下一步。	
创建堆栈。	1. 在配置堆栈选项页面上，对于标签，输入任何可选值。 2. 选择 Next(下一步)。 3. 在查看页面上，验证详细信息，然后选择提交。 预置应在 5-10 分钟左右完成。当 AWS CloudFormation Stacks 页面显示 CREATE_COMPLETE 时，它就完成了。	云管理员、数据库管理员
设置端点。	1. 从 AWS 管理控制台上，选择数据库迁移服务。 2. 在资源管理下方，选择复制实例。 3. 在资源管理下方，选择端点。	云管理员、数据库管理员

Task	描述	所需技能
测试连接。	在源端点和目标端点显示为“活动”状态后，测试连接。为每个端点（源端点和目标端点）选择运行测试，以确保状态显示为成功。	云管理员、数据库管理员

使用 Oracle 数据泵将 PeopleSoft 架构从本地 Oracle 数据库导出

Task	描述	所需技能

生成 SCN。	当源数据库处于活动状态并用于应用程序，请使用 Oracle Data Pump 启动数据导出。您必须首先从源数据库生成系统更改号 (SCN)，以便在使用 Oracle Data Pump 导出期间保持数据一致性，并作为 AWS DMS 中更改数据捕获的起点。
	
	要从您的源数据库生成当前 SCN，请输入以下 SQL 语句。
	
	`SQL> select name from v $database;`
	`SQL> select name from v $database;`
	`NAME`
	`---------`
	`PSFTDMO`
	`SQL> SELECT current_s cn FROM v$database;`
	`CURRENT_SCN`
	`-----------`
	`23792008`
		数据库管理员

Task	描述	所需技能
	保存生成的 SCN，以便在导出数据时使用，并用于创建 AWS DMS 复制任务。	

Task	描述	所需技能

创建参数文件。	要创建用于导出架构的参数文件，您可使用以下代码。
	
	`$ cat exp_datapmp.par`
	`userid=system/*******`
	`directory=DATA_P`
	`UMP_DIR`
	`logfile=export_dms_`
	`sample_user.log`
	`dumpfile=export_dms_`
	`sample_data_%U.dmp`
	`schemas=SYSADM`
	`flashback_scn=237920`
	`08`
	
	Note
	您也可以根据自己的要求使用以下命令DATA_PUMP_DIR 来定义自己的命令。
	
	`SQL> CREATE OR REPLACE`
	` DIRECTORY DATA_PUMP`
	`_DIR AS '/opt/oracle/`
	`product/19c/dbhome_1/`
	`dmsdump/';`
	`Directory created.`
	`SQL> GRANT READ, WRITE`
	` ON DIRECTORY DATA_PUMP`
	`_DIR TO system;`
	`Grant succeeded.`
	`SQL>`
	`SQL> SELECT owner,`
	` directory_name,`
	` directory_path FROM`
		数据库管理员

Task	描述	所需技能
	dba_directories WHERE directory_name='DA TA_PUMP_DIR'; OWNER DIRECTORY_NAME DIRECTORY_PATH ---------- ---------- ---------- ---------- ---------- ---------- SYS DATA_PUMP_DIR /opt/ oracle/product/19c/dbh ome_1/dmsdump/	

Task	描述	所需技能

导出架构。	要执行导出，请使用 expdp 实用程序。
	
	`$ expdp parfile=exp_datapmp.par`
	`.....`
	`Transferring the dump file with DBMS_FILE_TRANSFER to Target:`
	`. . exported "SYSADM"."PS_XML_TEMPLT_LNG" 6.320 KB 0 rows`
	`. . exported "SYSADM"."PS_XML_TEMPLT_LNK" 6.328 KB 0 rows`
	`. . exported "SYSADM"."PS_XML_XLATDEF_LNG" 6.320 KB 0 rows`
	`. . exported "SYSADM"."PS_XML_XLATITM_LNG" 7.171 KB 0 rows`
	`. . exported "SYSADM"."PS_XPQRYRUNCNTL" 7.601 KB 0 rows`
	`. . exported "SYSADM"."PS_XPQRYRUNPARM" 7.210 KB 0 rows`
	`. . exported "SYSADM"."PS_YE_AMOUNTS" 9.351 KB 0 rows`
	`. . exported "SYSADM"."PS_YE_DATA" 16.58 KB 0 rows`
	`. . exported "SYSADM"."PS_YE_EE" 6.75 KB 0 rows`
	`. . exported "SYSADM"."PS_YE_W2CP_AMOUNTS" 9.414 KB 0 rows`
		数据库管理员

Task	描述	所需技能
	. . exported "SYSADM". "PS_YE_W2CP_DATA" 20.94 KB 0 rows . . exported "SYSADM". "PS_YE_W2C_AMOUNTS" 10.27 KB 0 rows . . exported "SYSADM". "PS_YE_W2C_DATA" 20.95 KB 0 rows . . exported "SYSADM". "PS_ZBD_JOBCODE_TBL" 14.60 KB 0 rows . . exported "SYSADM". "PTGRANTTBL" 5.468 KB 0 rows Master table "SYSTEM". "SYS_EXPORT_SCHEMA _01" successfully loaded/unloaded ** Dump file set for SYSTEM.SYS_EXPORT_ SCHEMA_01 is: /opt/oracle/pr oduct/19c/dbhome_1 /dmsdump/export_dm s_sample_data_01.dmp Job "SYSTEM"."SYS_EXPO RT_SCHEMA_01" successfully completed at Mon Dec 19 20:13:57 2022 elapsed 0 00:38:22	

使用 Oracle 数据泵将 PeopleSoft 架构导入 Amazon RDS for Oracle 数据库

Task	描述	所需技能
将转储文件传输至目标实例。	要使用 DBMS_FILE _TRANSFER 传输文件，	数据库管理员

Task	描述	所需技能

	您需要创建从源数据库到 Amazon RDS for Oracle 实例的数据库链接。建立链接后，您可以使用该实用程序将 Data Pump 文件直接传输至 RDS 实例。
	
	或者，您可以将 Data Pump 文件传输至 Amazon Simple Storage Service (Amazon S3)，然后将其导入至 Amazon RDS for Oracle 实例。有关该选项的更多信息，请参阅“其他信息”部分。
	
	要创建一个用于连接到位于目标数据库实例中的 Amazon RDS 主用户的数据库链接 ORARDSDB，请在源数据库上运行以下命令。
	
	`$sqlplus / as sysdba`
	`$ SQL> create database link orardsdb connect to admin identified by "*****" using '(DESCRIPTION = (ADDRESS = (PROTOCOL = TCP)(HOST = testpsft.*******.us-west-2.rds.amazonaws.com)(PORT = 1521))(CONNECT_DATA = (SERVER = DEDICATED) (SERVICE_NAME = orcl)))';`
	`Database link created.`	

Task	描述	所需技能

测试数据库链接。	测试数据库链接，以确保您可以使用 sqlplus 连接到 Amazon RDS for Oracle 目标数据库。
	`SQL>`
	`SQL> select name from v$database@orardsdb;`
	`NAME`
	`---------`
	`ORCL`
	`SQL>`
		数据库管理员

Task	描述	所需技能

将转储文件传输至目标数据库。	要将转储文件复制到 Amazon RDS for Oracle 数据库，您可以使用默认 DATA_PUMP_DIR 目录，也可以使用以下代码创建自己的目录。
	`exec rdsadmin.rdsadmin_util.create_directory(p_directory_name => 'TARGET_PUMP_DIR') ;`
	以下脚本使用名为 orardsdb 的数据库链接，将名为 export_dms_sample_data_01.dmp 的转储文件从源实例复制到目标 Amazon RDS for Oracle 数据库。
	`$ sqlplus / as sysdba SQL> BEGIN DBMS_FILE_TRANSFER .PUT_FILE(source_directory_object => 'DATA_PUMP_DIR', source_file_name => 'export_dms_sample_data_01.dmp', destination_directory_object => 'TARGET_PUMP_DIR'', destination_file_name => 'export_dms_sample_data_01.dmp', destination_database => 'orardsdb'`
		数据库管理员

Task	描述	所需技能

	``
	);
	END;
	/
	PL/SQL procedure
	 successfully completed
	.
	``	

在目标数据库中列出转储文件。	PL/SQL 过程完成后，您可使用以下代码在 Amazon RDS for Oracle 数据库中列出数据转储文件。 ``` SQL> select * from table (rdsadmin.rds_file _util.listdir(p_di rectory => 'TARGET_P UMP_DIR')); ```	数据库管理员

Task	描述	所需技能
在目标数据库上启动导入。	在开始导入进程前，请使用数据转储文件在目标 Amazon RDS for Oracle 数据库上设置角色、架构和表空间。 要执行导入，请使用 Amazon RDS 主用户账户访问目标数据库，并使用 `tnsnames.ora` 文件中的连接字符串名称，其中包括 Amazon RDS for Oracle 数据库 `tns-entry`。如有必要，可以纳入重映射选项，将数据转储文件导入不同的表空间或使用不同架构名称。 要开始导入，请使用以下代码。 `impdp admin@orardsdb directory=TARGET_PUMP_DIR logfile=import.log dumpfile=export_dms_sample_data_01.dmp` 为确保成功导入，请检查导入日志文件中是否存在任何错误，并查看对象数、行数和无效对象等详细信息。如果有任何无效对象，请重新编译它们。此外，比较源数据库对象和目标数据库对象，以确认它们是否匹配。	数据库管理员

使用 CDC 创建 AWS DMS 复制任务，以执行实时复制

Task	描述	所需技能
创建复制任务。	通过使用以下步骤创建 AWS DMS 复制任务： 1. 在 AWS DMS 控制台的转换和迁移，选择数据库迁移任务。 2. 在任务配置下方，为任务标识符输入您的任务标识符。 3. 对于复制实例，请选择您创建的 DMS 复制实例。 4. 对于源数据库端点，请选择源端点。 5. 对于目标数据库端点，请选择您的目标 Amazon RDS for Oracle 数据库。 6. 对于迁移类型，请选择仅复制数据更改。如果您收到需要开启补充日志记录的消息，请按照其他信息部分中的说明进行操作。 7. 在任务设置下方，选择指定日志序列号。 8. 对于系统更改号，请输入从源 Oracle 数据库生成的 Oracle 数据库 SCN。 9. 选择启用验证。 10选择 “启用 CloudWatch 日志”。 通过激活此功能，您可以验证数据和 A mazon	云管理员、数据库管理员

Task	描述	所需技能
	CloudWatch 日志，以查看 AWS DMS 复制实例日志。 11.在选择规则下，完成以下操作： • 对于架构，请选择输入架构。 • 对于架构名称，请输入 SYSADM。 • 对于表名称，请输入 %。 • 对于操作，请选择包含。 12.在转换规则下，完成以下操作： • 对于目标，请选择表。 • 对于架构名称，请选择输入架构。 • 对于架构名称，请输入 SYSADM。 • 对于操作，请选择重命名为。 13选择创建任务。 创建任务后，它会将 CDC 从在 CDC 启动模式下提供的 SCN 迁移到 Amazon RDS for Oracle 数据库实例。您也可以通过查看 CloudWatch 日志进行验证。	

在目标 Amazon RDS for Oracle 数据库上验证数据库架构

Task	描述	所需技能
验证数据传输。	AWS DMS 任务启动后，您可查看任务页面上的表统计数据选项卡，以查看对数据所做的更改。 您可以在控制台的数据库迁移任务页面监控正在进行的复制的状态。 有关更多信息，请参阅 AWS DMS 数据验证。	云管理员、数据库管理员

割接

Task	描述	所需技能
停止复制。	停止复制过程，并停止源应用程序服务。	云管理员、数据库管理员
启动 PeopleSoft 中间层。	在 AWS 中启动目标 PeopleSoft 中间层应用程序，并将其定向到最近迁移的 Amazon RDS for Oracle 数据库。 在访问应用程序时，您应该会注意到，现已通过 Amazon RDS for Oracle 数据库建立了所有应用程序连接。	数据库管理员、管理员 PeopleSoft
关闭源数据库。	在确认源数据库不再有其他连接后，可关闭源数据库。	数据库管理员

相关的资源

- AWS Database Migration Service 入门
- AWS Database Migration Service 最佳实践
- 将 Oracle 数据库迁移至 AWS Cloud

其他信息

使用 Amazon S3 传输文件

要将文件传输至 Amazon S3，您可以使用 AWS CLI 或 Amazon S3 控制台。将文件传输至 Amazon S3 后，您可以使用 Amazon RDS for Oracle 实例从 Amazon S3 导入 Data Pump 文件。

如果选择使用 Amazon S3 集成作为替代方法传输转储文件，请执行以下步骤：

1. 创建 S3 存储桶。
2. 使用 Oracle Data Pump 从源数据库导出数据。
3. 将 Data Pump 文件上传至 S3 存储桶。
4. 将 Data Pump 文件从 S3 存储桶下载至目标 Amazon RDS for Oracle 数据库。
5. 使用 Data Pump 文件执行导入。

> **Note**
>
> 要在 S3 和 RDS 实例之间传输大型数据文件，建议使用 Amazon S3 Transfer Acceleration 功能。

激活补充日志记录

如果您收到一条警告消息，要求在源数据库中为进行中的复制启用补充日志记录，请使用以下步骤。

```
SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (ALL) COLUMNS;
SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (PRIMARY KEY) COLUMNS;
SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (UNIQUE) COLUMNS;
SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (FOREIGN KEY) COLUMNS;
SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (PRIMARY KEY) COLUMNS
SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (UNIQUE) COLUMNS;
```

将本地 MySQL 数据库迁移至 Amazon RDS for MySQL

由 Lorenzo Mota (AWS) 编写

摘要

此模式为将本地 MySQL 数据库迁移至 Amazon Relational Database Service (Amazon RDS) for MySQL 提供了指导。该模式讨论了如何使用 AWS Database Migration Service (AWS DMS) 或原生 MySQL 工具（例如 mysqldump）进行完整数据库迁移。这种模式主要适用于 DBAs 解决方案架构师。它可以在小型或大型项目中用作测试程序（我们建议至少一个测试周期），或作为最终迁移程序。

先决条件和限制

先决条件

- 一个活跃的 AWS 账户
- 本地数据中心的 MySQL 源数据库

限制

- 数据库大小限制：64 TB

产品版本

- MySQL 版本 5.5、5.6、5.7、8.0。有关支持版本的最新列表，请参阅 AWS 文档中的 Amazon RDS 上的 MySQL。如果您正在使用 AWS DMS，另请参阅使用与 MySQL 兼容的数据库作为当前支持的 AWS DMS MySQL 版本的目标。 AWS DMS

架构

源技术堆栈

- 本地数据库。

目标技术堆栈

- 运行 MySQL 的 Amazon RDS 数据库实例

目标架构

下图介绍了迁移后 Amazon RDS for MySQL 的目标实施。

AWS 数据迁移架构

使用 AWS DMS：

下图显示了在直接转换之前使用 AWS DMS 发送完整和增量更改时的数据迁移架构。从本地到的网络连接 AWS 取决于您的要求，并且超出了这种模式的范围。

使用原生 MySQL 工具：

使用 MySQL 原生工具时的数据迁移架构如下图所示。导出转储文件被复制到亚马逊简单存储服务 (Amazon S3) Simple Service，并在转换之前导入到亚马逊 RDS for MySQL 数据库 AWS 中。从本地到的网络连接 AWS 取决于您的要求，并且超出了这种模式的范围。

备注：

- 根据停机时间要求和数据库的大小，使用 AWS DMS 或更改数据捕获 (CDC) 工具可以最大限度地减少转换时间。 AWS DMS 可以帮助将新目标的切换时间缩短到最少（通常为几分钟）。 如果数据库的大小和网络延迟允许很短的窗口，那么使用 mysqldump 的离线策略就足够了。（我们建议进行测试以获得大致时间。）
- 通常，CDC 策略比离线选项 AWS DMS 需要更多的监控和复杂性。

工具

- AWS s@@ ervices:AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到 AWS Cloud d 或在云和本地设置的组合之间迁移。有关支持的 MySQL 源数据库和目标数据库的信息 AWS DMS，请参阅将兼容 MySQL 的数据库迁移到。 AWS如果您的源数据库不受支持 AWS DMS，则必须选择其他方法来迁移数据。
- 原生 MySQL 工具：my s qldump
- 第三方工具：Per cona XtraBackup

操作说明

计划迁移

Task	描述	所需技能
验证数据库版本。	验证源数据库和目标数据库版本。	数据库管理员
确定硬件要求。	确定目标服务器的硬件要求。	数据库管理员、系统管理员
确定存储要求。	确定目标数据库的存储需求(如存储类型和容量)。	数据库管理员、系统管理员
选择实例类型。	根据容量、存储功能、网络功能选择目标实例类型。	数据库管理员、系统管理员
确定网络访问要求。	确定源和目标数据库的网络访问安全要求。	数据库管理员、系统管理员
确定不支持的对象。	确定不支持的对象（如有）并确定迁移工作。	数据库管理员
确定依赖项。	确定对远程数据库的任何依赖项。	数据库管理员
确定应用程序迁移策略。	确定客户端应用程序迁移策略。	数据库管理员、应用程序所有者、系统管理员

配置基础设施

Task	描述	所需技能
创建虚拟私有云（VPC）。	配置路由表、互联网网关、NAT 网关和子网。有关更多信息，请参阅亚马逊 RDS 文档中的VPCs 和 Amazon RDS。	系统管理员

Task	描述	所需技能
创建安全组。	IPs 根据您的要求配置端口和 CIDR 范围或特定范围。MySQL 的默认端口是 3306。有关更多信息，请参阅 Amazon RDS 文档中的使用安全组控制访问权限。	系统管理员
配置和启动 Amazon RDS for MySQL DB 实例。	有关说明，请参阅 Amazon RDS 文档中的创建 Amazon RDS 数据库实例。查看支持的版本。	系统管理员

迁移数据-选项 1（使用原生工具）

Task	描述	所需技能
使用原生 MySQL 工具或第三方工具迁移数据库对象和数据。	有关说明，请参阅 mysqldump 和 Percona XtraBackup（用于物理迁移）等 MySQL 工具的文档。 有关选项的更多信息，请参阅博客文章 MySQL 到 Amazon RDS for MySQL 或 Amazon Aurora MySQL 的迁移选项	数据库管理员

迁移数据-选项 2（使用 AWS DMS）

Task	描述	所需技能
使用迁移数据 AWS DMS。	有关说明，请参阅AWS DMS 文档。	数据库管理员

在割接之前执行初步任务

Task	描述	所需技能
修复对象计数差异。	从源数据库和新目标数据库收集对象计数。修复目标数据库差异。	数据库管理员
检查依赖项。	检查与其他数据库之间的依赖关系（链接）是否有效并按预期工作。	数据库管理员
执行测试。	如果这是测试周期，请执行查询测试、收集指标以及修复问题。	数据库管理员

割接

Task	描述	所需技能
切换至目标数据库。	将客户端应用程序切换至新基础设施。	数据库管理员、应用程序所有者、系统管理员
提供测试支持。	为功能应用测试提供支持。	数据库管理员

关闭项目

Task	描述	所需技能
关闭资源。	关闭您为迁移创建的临时 AWS 资源。	数据库管理员、系统管理员
验证项目文档。	查看和验证项目文档。	数据库管理员、应用程序所有者、系统管理员

Task	描述	所需技能
收集指标。	收集关于迁移时间、手动与工具工作的百分比、成本节省等指标。	数据库管理员、应用程序所有者、系统管理员
关闭项目。	关闭项目并提供反馈。	数据库管理员、应用程序所有者、系统管理员
停用源数据库。	当所有迁移和割接任务完成后，停用本地数据库。	数据库管理员、系统管理员

相关资源

参考

- 关系数据库的迁移策略
- AWS DMS 网站
- AWS DMS 文档
- Amazon RDS 文档
- Amazon RDS 定价
- 亚马逊 VPC 和亚马逊 RDS
- Amazon RDS 多可用区部署
- 使用 Percona、A XtraBackup mazon EFS 和 Amazon S3 将本地 MySQL 数据库迁移到 Aurora MySQL
- Amazon RDS 数据库实例存储

教程

- 入门 AWS DMS
- Amazon RDS 入门

将本地 Microsoft SQL Server 数据库迁移至 Amazon RDS for SQL Server

由 Henrique Lobao (AWS)、Jonathan Pereira Cruz (AWS) 和 Vishal Singh (AWS) 编写

摘要

此模式提供以下指导：将本地 Microsoft SQL Server 数据库迁移至 Amazon Relational Database Service (Amazon RDS) for SQL Server。它描述了两种迁移选项：使用 AWS Data Migration Service (AWS DMS) 或使用原生 Microsoft SQL Server 工具，例如复制数据库向导。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 本地数据中心的源 Microsoft SQL Server 数据库

限制

- 数据库大小限制：16 TB

产品版本

- SQL Server 2014-2019、Enterprise、Standard、Workgroup 和 Developer 版本。有关当前支持的版本和功能的最新列表，请参阅 AWS 文档中的 Microsoft SQL Server on Amazon RDS。如果您使用的是 AWS DMS，另请参阅 AWS DMS 支持的 SQL Server 版本的使用 Microsoft SQL Server 数据库作为 AWS DMS 的目标。

架构

源技术堆栈

- 本地 Microsoft SQL Server 数据库

目标技术堆栈

- Amazon RDS for SQL Server DB 实例

源架构和目标架构

使用 AWS DMS：

使用原生 SQL Server 工具：

工具

- AWS DMS 支持不同类型的源数据库和目标数据库。有关详细信息，请参阅 AWS DMS Step-by-Step 演练。如果 AWS DMS 不支持源数据库，请选择其他方法来迁移数据。
- 原生 Microsoft SQL Server 工具包括备份和恢复、复制数据库向导以及复制和附加数据库。

操作说明

计划迁移

Task	描述	所需技能
验证源数据库和目标数据库版本和引擎。		数据库管理员
确定目标服务器实例的硬件要求。		数据库管理员、系统管理员
确定存储需求（存储类型和容量）。		数据库管理员、系统管理员
根据容量、存储功能和网络功能选择正确的实例类型。		数据库管理员、系统管理员
确定源数据库和目标数据库的网络访问安全要求。		数据库管理员、系统管理员
确定应用程序迁移策略。		数据库管理员、系统管理员

配置基础设施

Task	描述	所需技能
创建虚拟私有云（VPC）。		系统管理员
创建安全组。		系统管理员
配置和启动运行 Amazon RDS 数据库实例。		数据库管理员、系统管理员

迁移数据 - 选项 1

Task	描述	所需技能
使用原生 SQL Server 工具或第三方工具迁移数据库对象和数据。		数据库管理员

迁移数据 - 选项 2

Task	描述	所需技能
使用 AWS DMS 迁移数据。		数据库管理员

迁移应用程序

Task	描述	所需技能
遵循应用程序迁移策略。		数据库管理员、应用程序所有者、系统管理员

割接

Task	描述	所需技能
将应用程序客户端切换至新基础设施。		数据库管理员、应用程序所有者、系统管理员

关闭项目

Task	描述	所需技能
关闭临时 AWS 资源。		数据库管理员、系统管理员
查看和验证项目文档。		数据库管理员、应用程序所有者、系统管理员
收集关于迁移时间、手动任务与自动任务的百分比以及成本节省等指标。		数据库管理员、应用程序所有者、系统管理员
关闭项目并提供反馈。		数据库管理员、应用程序所有者、系统管理员

相关资源

参考

- 在 Amazon Web Services 上部署 Microsoft SQL Server
- AWS DMS 网站
- Amazon RDS 定价
- AWS 上的 Microsoft 产品
- AWS 上的 Microsoft 许可
- AWS 上的 Microsoft SQL Server
- 将 Windows 身份验证与 Microsoft SQL Server 数据库实例结合使用
- Amazon RDS 多可用区部署

教程和视频

- AWS DMS 入门
- Amazon RDS 入门
- AWS DMS（视频）
- Amazon RDS（视频）

使用 Rclone 将数据从 Microsoft Azure Blob 迁移至 Amazon S3

由 Suhas Basavaraj (AWS)、Aidan Keane (AWS) 和 Corey Lane (AWS) 编写

摘要

此模式描述如何使用克隆将数据从 Microsoft Azure Blob 对象存储迁移到 Amazon Simple Storage Service (Amazon S3) 存储桶。您可使用此模式对数据执行一次性迁移或持续同步。Rclone 是用 Go 编写的命令行程序，用于跨云提供商的各种存储技术移动数据。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 存储在 Azure Blob 容器服务的数据

架构

源技术堆栈

- Azure Blob 存储容器

目标技术堆栈

- Amazon S3 存储桶
- 亚马逊弹性计算云 (亚马逊 EC2) Linux 实例

架构

工具

- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Rclone 是一款受 rsync 启发的开源命令行程序。它用于管理许多云存储平台文件。

最佳实践

将数据从 Azure 迁移至 Amazon S3 时，请注意以下注意事项，以避免不必要的成本或传输速度变慢：

- 在与 Azure 存储账户和 Blob 容器相同的地理区域中创建 AWS 基础设施，例如 AWS 区域 `us-east-1`（弗吉尼亚北部）以及 Azure 区域 `East US`。
- 如果可能，请避免使用 NAT 网关，因为它会累积入口与出口带宽的数据传输费用。
- 使用适用于 Amazon S3 的 VPC 网关端点提高性能。
- 考虑使用基于 AWS Graviton2 (ARM) 处理器的 EC2 实例，与英特尔 x86 实例相比，成本更低，性能更高。Rclone 经过大量交叉编译，并提供了预编译 ARM 二进制文件。

操作说明

准备 AWS 和 Azure 云资源

Task	描述	所需技能
准备目标 S3 存储桶。	在相应的 AWS 区域创建新 S3 存储桶，或者选择现有存储桶作为要迁移的数据的目的地。	AWS 管理员
为亚马逊创建 IAM 实例角色 EC2。	为@@ 亚马逊创建一个新的 AWS Identity and Access 管理 (IAM) 角色 EC2。此角色授予您的 EC2 实例对目标 S3 存储桶的写入权限。	AWS 管理员
将 策略附加到 IAM 实例角色。	使用 IAM 控制台或 AWS 命令行界面 (AWS CLI) Line CLI 为实例角色创建内联策略， EC2 该策略允许对目标 S3 存储桶进行写入访问权限。有关示例策略，请参阅其他信息部分。	AWS 管理员
启动实 EC2 例。	启动配置为使用新创建的 IAM 服务角色的 Amazon Linux EC2 实例。此实例还需通过互	AWS 管理员

Task	描述	所需技能
	联网访问 Azure 公共 API 端点。 **Note** 考虑使用基于 AWS Graviton 的 EC2 实例来降低成本。Rclone 提供了 ARM 编译二进制文件。	
创建 Azure AD 服务主体。	使用 Azure CLI 创建对源 Azure Blob 存储容器具有只读访问权限的 Azure Active Directory (Azure AD) 服务主体。有关说明，请参阅其他信息部分。将这些证书存储在您的 EC2 实例上，存储到相应的位置`~/azure-principal.json` 。	云管理员，Azure

安装和配置 Rclone

Task	描述	所需技能
下载并安装 Rclone。	下载和安装 Rclone 命令行程序。有关安装说明，请参阅 Rclone 安装文档。	常规 AWS，云管理员
配置 Rclone。	复制以下 `rclone.conf` 示例文件。将 `AZStorage Account` 替换为您的 Azure Storage 账户名称，将 `us-east-1` 替换为 S3 存储	常规 AWS，云管理员

Task	描述	所需技能

	桶所在的 AWS 区域。将此文件保存到您的 EC2 实例~/.config/rclone/rclone.conf 上的相应位置。
	
	`[AZStorageAccount]`
	`type = azureblob`
	`account = AZStorageAccount`
	`service_principal_file = azure-principal.json`
	
	`[s3]`
	`type = s3`
	`provider = AWS`
	`env_auth = true`
	`region = us-east-1`	

Task	描述	所需技能

验证 Rclone 配置。	若要确认 Rclone 已配置且权限是否正常运行，请验证 Rclone 是否可以解析您的配置文件，以及 Azure Blob 容器和 S3 存储桶中的对象是否可以访问。有关示例验证命令，请参阅以下内容。
	• 在配置文件中列出已配置遥控器。这会确保您的配置文件得到正确解析。查看输出，确保它与您的 `rclone.conf` 文件匹配。
	`rclone listremotes`
	`AZStorageAccount:`
	`s3:`
	• 列出已配置账户的 Azure Blob 容器。将 `AZStorageAccount` 替换 `rclone.conf` 文件中的使用的存储账户。
	`rclone lsd AZStorage`
	`Account:`
	`2020-04-29 08:29:26`
	`docs`
	• 列出 Azure Blob 容器文件。将此命令中的 docs 替换为 Azure 存储账户中的实际 Blob 容器名称。
	`rclone ls AZStorage`
	`Account:docs`
		常规 AWS，云管理员

Task	描述	所需技能
	824884 administr ator-en.a4.pdf • 列出 Amazon Web Services account 中的存储桶。 [root@ip-10-0-20-157 ~]# rclone lsd s3: 2022-03-07 01:44:40 amzn-s3-demo- bucket1 2022-03-07 01:45:16 amzn-s3-demo- bucket2 2022-03-07 02:12:07 amzn-s3-demo- bucket3 • 列出 S3 存储桶内的文件。 [root@ip-10-0-20-1 57 ~]# rclone ls s3:amzn-s3-demo-bu cket1 template0.yaml template1.yaml	

通过 Rclone 迁移数据

Task	描述	所需技能
从容器迁移数据。	运行 Rclone 复制或者同步命令。 示例：复制 此命令将数据从源 Azure Blob 容器复制至目标 S3 存储桶。	常规 AWS，云管理员

Task	描述	所需技能

	`rclone copy AZStorageAccount:blob-container s3:amzn-s3-demo-bucket1`
	
	示例：同步
	
	此命令在源 Azure Blob 容器和目标 S3 存储桶间同步数据。
	
	`rclone sync AZStorageAccount:blob-container s3:amzn-s3-demo-bucket1`
	
	⚠ Important
	使用 sync 命令时，源容器中不存在的数据将从目标 S3 存储桶中删除。	

同步容器。	初始复制完成后，运行 Rclone sync 命令以进行持续迁移，这样只会复制目标 S3 存储桶中缺少的新文件。	常规 AWS，云管理员
验证数据是否成功迁移。	若要检查数据是否已成功复制到目标 S3 存储桶，请运行Rclone lsd 和 ls 命令。	常规 AWS，云管理员

相关资源

- Amazon S3 用户指南（AWS 文档）
- 亚马逊的 IAM 角色 EC2（AWS 文档）

- 创建 Microsoft Azure Blob 容器（Microsoft Azure 文档）
- Rclone 命令（Rclone 文档）

其他信息

实例的角色策略示 EC2 例

该策略授予您的 EC2 实例对您账户中特定存储桶的读写权限。如果您的存储桶使用客户管理的密钥进行服务器端加密，则策略可能需要对 AWS Key Management Service (AWS KMS) 的额外访问权限。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:DeleteObject",
                "s3:GetObject",
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": [
              "arn:aws:s3:::amzn-s3-demo-bucket/*",
              "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "s3:ListAllMyBuckets",
            "Resource": "arn:aws:s3:::*"
        }
    ]
}
```

创建只读 Azure AD 服务主体

Azure 服务主体是客户应用程序、服务和自动化工具用来访问特定 Azure 资源的安全标识。可以将其视为具有特定角色和严格控制访问资源的权限的用户身份（登录名和密码或证书）。要创建只读服务主体、以遵循最低权限并保护 Azure 中的数据免遭意外删除，请按照以下步骤操作：

1. 登录你的 Microsoft Azure 云账户门户，在工作站上启动云命令行 PowerShell 或使用 Azure 命令行界面 (CLI)。
2. 创建服务主体，并将其配置为对 Azure Blob 存储账户的只读访问权限。将此命令 JSON 输出保存到名为 `azure-principal.json` 的本地文件中。该文件将上传到您的 EC2 实例。将大括号（{和}）中显示的占位符变量替换为您的 Azure 订阅 ID、资源组名称和存储账户名称。

```
az ad sp create-for-rbac `
--name AWS-Rclone-Reader `
--role "Storage Blob Data Reader" `
--scopes /subscriptions/{Subscription ID}/resourceGroups/{Resource Group Name}/
providers/Microsoft.Storage/storageAccounts/{Storage Account Name}
```

从 Couchbase Server 迁移至 AWS 上的 Couchbase Capella

由 Battulga Purevragchaa (AWS)、Mark Gamble 和 Saurabh Shanbhag (AWS) 编写

摘要

Couchbase Capella 是一个完全托管的 NoSQL 数据库即服务，适用于任务关键型应用程序（例如，用户配置文件或在线目录和库存管理）。DBaaCouchbase Capella 在 Couchbase 管理的亚马逊网络服务 (AW DBaa S) 账户中管理你的 S 工作负载。Capella 可以让您在单个界面中轻松运行和管理多集群、多 AWS 区域、多云和混合云复制。

Couchbase Capella 可帮助您立即扩展 Couchbase Server 应用程序，帮助您在几分钟内创建多节点集群。Couchbase Capella 支持所有 Couchbase Server 功能，包括 SQL++、全文搜索、事件服务和分析服务。它还消除了管理安装、升级、备份和一般数据库维护的需要。

本示例介绍了将自托管的 Couchbase Server 环境迁移之 Amazon Web Services Cloud 的最佳做法。该模式提供了一个可重复的过程，用于将数据和索引从本地或云中运行的 Couchbase Server 集群迁移到 Couchbase Capella。使用这些步骤可帮助您避免迁移期间出现问题并加快整个迁移过程。

此模式提供以下两个迁移项：

- 如果要迁移的索引少于 50 个，则选项 1 是合适的。
- 如果要迁移的索引超过 50 个，则选项 2 是合适的。

您还可以在自托管 Couchbase 服务器上设置示例数据，以便按照迁移指南进行操作。

如果您选择迁移选项 2，或者您使用的范围或集合不是默认值，则应参见示例配置文件其他信息部分。

先决条件和限制

先决条件

- 现有 Couchbase Capella 付费账户。您还可以在 AWS 上创建 Couchbase Capella 账户，并使用 Couchbase Capella 免费试用版，然后升级到付费账户来配置集群以进行迁移。要从试用版开始，请按照 Couchbase Capella 入门中的说明进行操作。
- 现有的自托管 Couchbase Server 环境，可以在本地部署或部署在云服务提供商上。
- 对于迁移选项 2，使用 Couchbase Shell 和配置文件。若要创建配置文件，可以使用其他信息部分中的示例文件。
- 熟悉管理 Couchbase Server 和 Couchbase Capella。

- 熟悉打开 TCP 端口及在命令行界面（CLI）中运行命令。

迁移过程还需要下表中描述的角色和专业知识。

角色	专业知识	责任
Couchbase 管理员	• 熟悉 Couchbase Server 和 Couchbase Capella • 基本的命令行知识很有帮助，但非必需的	• Couchbase Server 与 Capella 特定任务
系统管理员、IT 管理员	• 熟悉自托管的 Couchbase 服务器系统环境和管理	• 在自托管的 Couchbase Server 集群节点上打开端口并确定 IP 地址

限制

- 这种模式用于将数据、索引和 Couchbase 全文搜索索引从 Couchbase Server 迁移到 AWS 上的 Couchbase Capella。这种模式不适用于迁移 Couchbase Eventing Service 或 Couchbase Analytics。
- Couchbase Capella 在多个 AWS 区域可用。有关 Capella 支持的区域 up-to-date的信息，请参阅 Couchbase 文档中的亚马逊 Web Services。

产品版本

- Couchbase Server（Community Edition 或企业版）版本 5.x 或更高版本

架构

源技术堆栈

- Couchbase Server

目标技术堆栈

- Couchbase Capella

目标架构

1. 您可使用Capella 控制面板访问 Couchbase Capella。您可使用 Capella 控制面板来执行以下操作：
   - 控制和监控账户。
   - 管理集群和数据、索引、用户和组、访问权限、监控和事件。
2. 创建集群。
3. Capella 数据面板位于 Couchbase 管理的 Amazon Web Services account 中。创建新集群后，Couchbase Capella 会将其部署至所选 AWS 区域的多个可用区。
4. 您可在 Amazon Web Services account 的 VPC 中开发和部署 Couchbase 应用程序。通常，此 VPC 通过 VPC 对等连接访问 Capella 数据面板。

工具

- Couchbase Cross Data Center Replication (XDCR) 有助于在位于不同云提供商和不同数据中心的集群之间复制数据。它用于将数据从自托管的 Couchbase Server 集群迁移至 Couchbase Capella。

> **Note**
>
> XDCR 不能与 Couchbase Server 社区版一起使用来迁移到 Couchbase Capella。相反，您可以使用 cbexport。有关更多信息，请参阅从 Community Edition 迁移数据操作说明。

- Couchbase Shell 是 Couchbase Server 和 Couchbase Capella 访问本地和远程 Couchbase 集群的命令行外壳。在这种模式中，Couchbase Shell 可用于迁移索引。
- cbexport 是 Couchbase 实用程序，用于从 Couchbase 集群中导出数据。包含在 Couchbase Server CLI 工具中。

操作说明

准备迁移

Task	描述	所需技能
评估自托管的 Couchbase Server 集群的大小。	登录适用于 Couchbase Server 的 Couchbase Web 控制台，	Couchbase 管理员

Task	描述	所需技能
	然后评测自托管的集群的节点和存储桶。 1. 要显示集群节点列表，请选择导航栏中的服务器选项卡。 2. 记录节点数量，然后选择列表中的每个节点以显示其属性。 3. 记录每个单独节点的内存和存储。 4. 在导航栏中选择存储桶选项卡，然后在列表中选择每个存储桶以显示其属性。记录每个存储桶的 RAM 配额和冲突解决配置。 您将使用自托管的 Couchbase Server 集群配置作为在 Couchbase Capella 上调整和配置目标集群的通用指南。 如需有关更详细的 Couchbase Capella 缩放练习的帮助，请联系 Couchbase。	
在自托管的 Couchbase 服务器集群上记录 Couchbase 服务分发情况。	1. 在 Couchbase Web 控制台，选择服务器选项卡以显示集群节点列表。 2. 选择每个节点以显示其属性，然后记录每个节点（数据服务、查询服务、索引服务、搜索服务、分析服务以及事件服务）。	Couchbase 管理员

Task	描述	所需技能
记录自托管的 Couchbase Server 集群节点的 IP 地址。	（如果您使用的是Community Edition，请忽略此步骤。）记录集群中每个节点 IP 地址。稍后它们将被添加到您的 Couchbase Capella 集群上的允许列表中。	Couchbase 管理员、系统管理员

在 Couchbase Capella 上部署和配置资源

Task	描述	所需技能
Choose a template。	1. 登录 Couchbase Capella 控制面板，在主导航栏中选择控制面板选项卡或集群选项卡，然后选择创建集群。 2. 使用您在自托管的 Couchbase Server 集群评估中记录的信息，选择符合配置要求的集群模板。如果找不到合适的模板，请在集群大小编辑器中选择自定义模板。	Couchbase 管理员
选择和配置节点。	选择和配置节点以匹配您的自托管的 Couchbase Server 集群环境，包括节点数量、服务分布、计算或 RAM 以及存储。 Couchbase Capella 使用多维缩放最佳实践。只能根据部署最佳实践来选择服务与节点。这可能意味着您无法完全匹配	Couchbase 管理员

Task	描述	所需技能
	自托管 Couchbase Server 集群配置。	
部署集群。	选择支持区域与支持包，然后部署集群。有关详细步骤和说明，请参阅 Couchbase 文档中的创建集群。 **Important** 如果您使用的是Couchbase Capella免费试用版，则必须在开始迁移之前将其转换为付费帐户。要转换您的账户，请打开 Couchbase Capella 控制面板的账单部分，然后选择添加激活 ID。在您与 Couchbase 销售人员签订购买协议后，或者在您通过 Amazon Web Services Marketplace 进行购买之后，激活编号将发送到您的账单联系人电子邮件地址。	Couchbase 管理员

Task	描述	所需技能
创建数据库凭证用户。	数据库凭证用户特定于集群，由用户名、密码和一组存储桶权限组成。此用户是创建存储桶和访问存储桶数据的必要条件 在 Couchbase Capella 控制面板中，按照 Couchbase Capella 文档中配置数据库凭证中的说明为新集群创建数据库凭证。 **Note** 如果组织用户想要远程或通过 Couchbase Capella 用户界面访问特定集群上的存储桶数据，则需要为其分配组织角色证书。这与数据库凭证是分开的，后者通常由应用程序与集成使用。创建组织用户允许您在 Couchbase Capella 集群创建和管理目标存储桶。	Couchbase 管理员

Task	描述	所需技能
如果使用迁移选项 2，请安装 Couchbase Shell。	您可将 Couchbase Shell 安装在任何能够通过网络访问自托管的 Couchbase 服务器和 Couchbase Capella 集群的系统上。有关更多信息，请参阅 Couchbase Shell 文档中的安装 Couchbase Shell 版本 1.0.0-beta.5。 通过在命令行终端中测试与自托管集群的连接，确认已安装 Couchbase Shell。	Couchbase 管理员、系统管理员

Task	描述	所需技能
允许 IP 地址。	1. 在 Couchbase Capella 控制面板，选择集群，然后选择目标集群。 2. 选择集群的连接选项卡，记录在管理允许的 IP下列出的集群的连接端点。 3. 要将安装了 Couchbase Shell 的系统的 IP 地址和自托管的 Couchbase 服务器集群实例的 IP 地址添加为允许的 IP 地址，请执行以下操作： a. 在广域网，选择管理允许 IP。 b. 选择添加允许 IP，输入安装了 Couchbase Shell 的系统的 IP 地址，然后选择添加 IP。 c. 重复上一步以添加自托管的 Couchbase Server 集群实例的 IP 地址。 有关允许的 IP 地址的更多信息，请参阅 Couchbase 文档中的配置允许 IP 地址。	Couchbase 管理员、系统管理员

Task	描述	所需技能
配置证书。	1. 要下载集群的根证书，请在根证书选择下载。 2. 使用 .pem 文件扩展名将根证书保存在系统上，将运行 Couchbase Shell 的文件夹中。 3. 接下来，登录您自托管的 Couchbase Server Web 控制台，在左侧导航栏中选择安全，然后选择证书选项卡。 4. 复制您自托管的 Couchbase Server 集群的根证书，并将其作为.pem 文件保存到您保存 Couchbase Capella 集群根证书文件的同一个文件夹。有关根证书的更多信息，请参阅 Couchbase Server 文档中的根证书。	Couchbase 管理员、系统管理员

Task	描述	所需技能
创建 Couchbase Shell 的配置文件。	在 Couchbase Shell 安装的主目录中创建配置点文件（例如 `/<HOME_DIRECTORY>/.cbsh/config`）。有关更多信息，请参阅 Couchbase 文档中的 Config dotfiles。 将源集群和目标集群连接属性添加到配置文件中。您可使用其他信息部分中的示例配置文件并编辑集群的设置。 将包含更新设置的配置文件保存到 `.cbsh` 文件夹（例如 `/<HOME_DIRECTORY>/.cbsh/config`）。	Couchbase 管理员、系统管理员
创建目标存储桶。	对于每个源存储桶，请按照 Couchbase 文档中创建存储桶中的说明在您的 Couchbase Capella 集群中创建目标存储桶。 您的目标存储桶配置必须与您自托管的 Couchbase Server 集群中存储桶的存储桶名称、内存设置和冲突解决设置相匹配。	Couchbase 管理员

Task	描述	所需技能

创建范围和集合。	每个存储桶都包含一个默认作用域和带有密钥空间 `_default._default` 的集合。如果您在作用域和集合中使用任何其他密钥空间，则必须在目标 Capella 集群中创建相同的密钥空间。
	1. 在安装了 Couchbase Shell 的系统打开命令行终端。
	2. 若要启动 Couchbase Shell，请运行以下命令。
	`./cbsh`
	3. 对于要迁移的每个存储桶，通过运行以下命令在 Capella 集群中创建范围和集合。请确保将 <BUCKET_NAME> 替换为要迁移的存储桶的名称。
	`scopes --clusters "On-Prem-Cluster" --bucket <BUCKET_NAME> | select scope | where scope != "_default" | each { |it| scopes create $it.scope --clusters "Capella-Cluster" } collections --clusters "On-Prem-Cluster" --bucket <BUCKET_NAME> | select scope collection | where $it.scope != "_default" | where $it.colle`
		Couchbase 管理员

Task	描述	所需技能
	`ction != "_default" | each { |it| collectio ns create $it.colle ction --clusters "Capella-Cluster" -- bucket <BUCKET_NAME> -- scope $it.scope }`	

从 Enterprise Edition 迁移数据

Task	描述	所需技能
在自托管的 Couchbase 服务器集群节点上打开 TCP 端口。	确保在自托管的 Couchbase Server 集群的节点上打开相应的端口，以便进行 XDCR 通信。有关更多信息，请参阅 Couchbase Server 端口文档。	Couchbase 管理员、系统管理员
如果您使用的是 Couchbase Server 企业版，请设置 Couchbase XDCR。	1. 在 Couchbase Capella 控制面板主导航栏中，选择集群，然后选择要迁移的目标集群。 2. 在根证书下，选择复制。 3. 登录到自托管的 Couchbase Server Web 控制台，然后在主导航栏中选择 XDCR。然后选择添加远程。 4. 输入以下设置： • 集群名称 - Capella 集群连接的名称 • IP/主机名 — Couchbase Capella 集群连接端点	Couchbase 管理员

Task	描述	所需技能
	• 远程集群的用户名 — Couchbase Capella 集群数据库用户 • 密码 — Couchbase Capella 集群的数据库用户密码 • 启用安全连接 - 已选择 • 完整（TLS 加密密码和数据）- 已选择 5. 粘贴您之前复制的 Capella 集群根证书，然后选择保存。	
启动 Couchbase XDCR。	1. 在自托管的 Couchbase Server Web 控制台中，在主导航栏中选择 XDCR，然后选择添加复制。 2. 输入以下设置： • 从存储桶复制 — 选择要迁移的源存储桶。 • 远程存储桶 — 输入目标存储桶名称。 • 远程集群 — 选择之前创建的目标集群。 3. 选择保存复制。复制过程应在几秒钟内开始。	Couchbase 管理员

使用选项 1 迁移索引

Task	描述	所需技能
将自托管的集群索引迁移至 Couchbase Capella。	**⚠ Important** 如果您要迁移的索引少于 50 个，我们建议您执行此过程。如果您要迁移的索引超过 50 个，建议您使用迁移选项 2。 1. 在 Couchbase Web 控制台，选择索引。 2. 在索引列表中，选择要迁移的第一项索引。然后显示索引定义。 3. 使用 CREATE 语句复制索引定义，但不要复制 `WITH { "defer_build":true }` 。 例如，在以下示例索引定义中，您只需要复制 `CREATE INDEX `cityindex` ON `travel-sample`(`city`)` 。 `CREATE INDEX `cityindex` ON `travel-sample`(`city`) WITH { "defer_build":true }`	Couchbase 管理员、系统管理员

Task	描述	所需技能
	4. 在 Couchbase Capella 控制面板，选择集群，然后选择目标集群。 5. 在工具下拉列表，选择查询工作台。将之前复制的 CREATE 语句粘贴至查询编辑器，然后选择执行。这将创建和构建索引。 6. 要确认索引已创建，请从工具下拉列表中选择索引。列表显示索引已创建和构建。 7. 对必须迁移的每个索引重复此进程。	

使用选项 2 迁移索引

Task	描述	所需技能
迁移索引定义。	**Important** 如果您要迁移的索引超过 50 个，我们建议您执行此过程。如果您要迁移的索引少于 50 个，建议您使用迁移选项 1。 1. 在安装了 Couchbase Shell 的系统打开命令行终端。 2. 若要启动 Couchbase Shell，请运行以下命令。	Couchbase 管理员、系统管理员

Task	描述	所需技能
	``` ./cbsh ``` 3. 要连接至自托管 Couchbase Server 集群，请运行以下命令。 ``` cb-env cluster On-Prem-Cluster ``` 4. 要将索引定义从自托管的 Couchbase Server 集群迁移至 Couchbase Capella 集群，请对要迁移的每个存储桶运行以下命令。务必将 <BUCKET_NAME> 替换为待迁移索引对应的存储桶名称。此迁移选项要求您的目标存储桶名称与源存储桶名称相同。 ``` query indexes --definitions | where bucket =~ <BUCKET_NAME>| get definition | each { |it| query $it --clusters Capella-Cluster } ```	

Task	描述	所需技能

生成索引定义。	1. 要将上下文割接到 Couchbase Capella 集群，请运行以下命令：
	
	`cb-env cluster Capella-Cluster`
	
	2. 要构建已迁移到 Couchbase Capella 集群的索引定义，请运行以下命令，将 <BUCKET_NAME> 替换待生成索引对应的存储桶名称。
	
	`query 'SELECT RAW CONCAT("BUILD INDEX ON ", k , "(['", CONCAT2 ("','", inames), "']);") FROM system:indexes AS s LET bid = CONCAT("`",s.bucket_id, "`"), sid = CONCAT("`", s.scope_id, "`"), kid = CONCAT("`", s.keyspace_id, "`"), k = NVL2(bid, CONCAT2(".", bid, sid, kid), kid) WHERE s.namespace_id = "default" AND s.bucket_id = "" GROUP BY k LETTING inames = ARRAY_AGG(s.name) FILTER (WHERE s.state = 'deferred') HAVING ARRAY_LENGTH(iname`
		Couchbase 管理员、系统管理员

Task	描述	所需技能
	s) > 0;' | each { | it| query $it } 3. 对每个存储桶重复此操作。	

迁移全文搜索索引

Task	描述	所需技能
将自托管的集群全文搜索索引迁移至 Couchbase Capella。	1. 在 Couchbase Web 控制台，选择搜索。 2. 在全文搜索 (FTS) 索引列表中，选择要迁移的第一个 FTS 索引，选择显示索引定义 JSON，然后选择复制到剪贴板。记下索引名称及其所属的存储桶。 3. 在 Couchbase Capella 控制面板，选择集群，然后选择目标集群。 4. 在工具下拉列表，选择全文搜索。 5. 选择导入索引，然后粘贴 FTS 索引定义。 6. 输入索引名称，选择正确的存储桶（如自托管集群中所述），然后选择创建。 7. 对必须迁移的每个 FTS 索引重复此过程。	Couchbase 管理员

从 Couchbase Community Edition 迁移

Task	描述	所需技能

从自托管的 Couchbase Server Community Edition 导出数据。	加密 XDCR 在 Couchbase Community Edition 中不可用。您可以从 Couchbase Community Edition 导出数据，然后手动将数据导入 Couchbase Capella。
	
	要从源存储桶导出数据，请在命令行中使用 `cbexport`。
	
	以下命令是一个示例。
	
	`cbexport json \`
	`--cluster localhost \`
	`--bucket <SOURCE BUCKET NAME> \`
	`--format lines \`
	`--username <USERNAME> \`
	`--password <PASSWORD> \`
	`--include-key cbkey \`
	`--scope-field cbscope \`
	`--collection-field cbcoll \`
	`--output cbexported_data.json`
	
	请注意，`cbkey`、`cbscope`、`cbcoll` 和 `cbexported_data.json` 是任意标签。稍后将在此过程中引用它们，因此如果您选择以不同的方式命名它们，请记下它。
		Couchbase 管理员

Task	描述	所需技能
将数据导入 Couchbase Capella。	1. 在 Couchbase Capella 控制面板，选择集群，然后选择目标集群。 2. 在工具下拉列表，选择导入。打开一个包含以下六个步骤的向导： a. 存储桶 — 选择目标存储桶。 b. 文件 — 选择 JSON，选择行，然后选择使用 Web 浏览器。如果您有大量的数据，您可探索手动选项。选择由 `cbexport` 创建的文件。 c. 收藏夹 - 选择自定义收藏夹映射。 如果您的 Community Edition 数据库不使用范围或集合，或者仅使用 _default，则可以选择选择单个集合选项。 在集合映射表达式中，输入 `%cbscope%.%cbcoll%` 。要验证此表达式是否正常工作，您可以粘贴示例数据，如下所示。 `{ "cbscope" :"inventory", "cbcoll":"landmark`	Couchbase 管理员

Task	描述	所需技能
	<pre>", "cbkey":" landmark_3991" }</pre> d. 密钥 - 选择客户生成。(如果您不关心保留要导入的数据的密钥，可以改为选择自动生成 UUID，然后继续执行步骤 5。) 在密钥名称生成器表达式，输入 %cbkey%。要验证此表达式是否正常运行，请粘贴一些示例数据。 e. 配置 — 选择 忽略字段，然后输入 cbscope、cbcoll、cbkey。这些字段包含导入后不需要位于目标存储桶中的临时信息。保留其他设置的默认值。 f. 导入 - 查看，准备就绪后选择导入。等待上传与数据导入。 对于大文件，Couchbase Capella 支持使用 curl 执行命令行导入。您可以在 Couchbase Capella 文档中的导入数据中更详细地探索导入选项。	

测试和验证迁移

Task	描述	所需技能
验证数据迁移。	1. 在 Couchbase Capella 控制面板，选择集群，然后在集群列表中选择目标集群。 2. 为您的目标集群选择存储桶选项卡。验证目标存储桶中的项目（文档）数量是否与源存储桶中的项目数量相匹配。 3. 在目标集群的工具下拉列表中，选择文档。确认所有文档均已完成迁移。 4. （可选）迁移完所有数据后，您可通过删除复制来关闭复制。有关更多信息，请参阅 Couchbase 文档中的删除副本。	Couchbase 管理员
验证索引迁移。	在 Couchbase Capella 控制面板，在目标集群的工具下拉列表中，选择索引。验证索引是否已完成迁移和构建。	Couchbase 管理员
验证查询结果。	1. 在 Couchbase Capella 控制面板，在目标集群的工具下拉列表中，选择查询工作台。 2. 运行示例 N1QL 查询或应用程序中所使用的查询。确保收到的结果与在自托管的 Couchbase Server 集群中运行查询时的结果相同。	Couchbase 管理员

Task	描述	所需技能
验证全文搜索结果（如果您迁移了 FTS 索引，则适用）。	1. 在 Couchbase Capella 控制面板，在目标集群的工具下拉列表中，选择全文搜索。 2. 通过选择其名称，以选择 FTS 指数。 3. 选择搜索。 4. 输入搜索查询示例，然后选择搜索。 5. 验证结果是否与在自托管集群上运行搜索时的结果相同。	Couchbase 管理员

相关资源

准备迁移

- 开始使用 Couchbase Capella 免费试用版
- 云提供程序对 Couchbase Capella 的要求
- Couchbase Capella 缩放指南

迁移数据和索引

- Couchbase XDCR
- Couchbase Shell 文档

Couchbase Cap SLAs ella 和支援

- Couchbase Capella 服务等级协议 () SLAs
- Couchbase Capella 服务支持策略

其他信息

以下代码是 Couchbase Shell 的配置文件示例。

```
Version = 1

[[clusters]]
identifier = "On-Prem-Cluster"
hostnames = ["<SELF_MANAGED_COUCHBASE_CLUSTER>"]
default-bucket = "travel-sample"
username = "<SELF_MANAGED_ADMIN>"
password = "<SELF_MANAGED_ADMIN_PWD>"
tls-cert-path = "/<ABSOLUTE_PATH_TO_SELF_MANAGED_ROOT_CERT>"
data-timeout = "2500ms"
connect-timeout = "7500ms"
query-timeout = "75s"

[[clusters]]
identifier = "Capella-Cluster"
hostnames = ["<COUCHBASE_CAPELLA_ENDPOINT>"]
default-bucket = "travel-sample"
username = "<CAPELLA_DATABASE_USER>"
password = "<CAPELLA_DATABASE_USER_PWD>"
tls-cert-path = "/<ABSOLUTE_PATH_TO_COUCHBASE_CAPELLA_ROOT_CERT>"
data-timeout = "2500ms"
connect-timeout = "7500ms"
query-timeout = "75s"
```

在保存配置文件之前，请使用下表，以确保您添加了自己的源和目标集群信息。

<SELF_MANAGED_COUCHBASE_CLUSTER>	使用自托管 Couchbase 服务器集群的 IP 地址。
<SELF_MANAGED_ADMIN>	使用管理员用户来管理您的自托管的 Couchbase 服务器集群。
<ABSOLUTE_PATH_TO_SELF_MANAGED_ROOT_CERT>	使用您自托管的 Couchbase Server 集群保存的根证书文件的绝对路径。
<COUCHBASE_CAPELLA_ENDPOINT>	使用您的 Couchbase Capella 集群的连接端点。
<CAPELLA_DATABASE_USER>	使用您的 Couchbase Capella 集群数据库用户。

<CAPELLA_DATABASE_USER_PWD>	使用 Couchbase Capella 集群的数据库用户密码。
<ABSOLUTE_PATH_TO_COUCHBASE_CAPELLA_ROOT_CERT>	使用为 Couchbase Capella 集群保存的根证书文件的绝对路径。

在 Amazon 上从 IBM WebSphere 应用程序服务器迁移到 Apache Tomcat EC2

由 Neal Ardeljan (AWS) 和 Afroz Khan (AWS) 编写

摘要

此模式将引导您完成从运行 IBM WebSphere 应用程序服务器 (WAS) 的本地红帽企业 Linux (RHEL) 6.9 或更高版本系统迁移到在亚马逊弹性计算云 (Amazon) 实例上运行 Apache Tomcat 的 RHEL 8 的步骤。 EC2

该模式可以应用于以下源和目标版本：

- WebSphere 应用程序服务器 7.x 到 Apache Tomcat 8（使用 Java 7 或更高版本）
- WebSphere 应用服务器 8.x 到 Apache Tomcat 8（使用 Java 7 或更高版本）
- WebSphere 应用程序服务器 8.5.5.x 到 Apache Tomcat 9（使用 Java 8 或更高版本）
- WebSphere 应用程序服务器 8.5.5.x 到 Apache Tomcat 10（使用 Java 8 或更高版本）

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- Java 源代码，假定如下：
  - 使用 Java Development Kit (JDK)，版本的 Java 7 或更高版本
  - 使用 Spring 或 Apache Struts 框架
  - 不使用企业 Java Beans (EJB) 框架或任何其他不适合 Tomcat 的 WebSphere 服务器功能
  - 主要使用 servlet 或 Java 服务器页面 () JSPs
  - 使用 Java Database Connectivity (JDBC) 连接器连接到数据库
- 来源 IBM WebSphere 应用程序服务器 7.x 或更高版本
- 目标 Apache Tomcat 8.5 或更高版本

架构

源技术堆栈

- 使用 Apache Struts Model-View-Controller (MVC) 框架构建的 Web 应用程序

- 在 IBM WebSphere 应用服务器 7.x 或 8.x 版本上运行的 Web 应用程序
- 使用轻型目录访问协议 (LDAP) 连接器连接至 LDAP 目录 (iPlanet/eTrust) 的 Web 应用程序
- 使用 IBM Tivoli Access Manager (TAM) 连接更新 TAM 用户密码的应用程序（在本实现中，应用程序使用 PD.jar）

本地数据库

- Oracle Database 21c (21.0.0.0)
- Oracle Database 19c (19.0.0.0)
- Oracle Database 12c 第 2 版 (12.2.0.1)
- Oracle Database 12c 第 1 版 (12.1.0.2)

目标技术堆栈

- Apache Tomcat 版本 8（或更高版本）在实例上的 RHEL 上运行 EC2
- Amazon Relational Database Service (Amazon RDS) for Oracle

有关 Amazon RDS 支持的 Oracle 版本的更多信息，请参阅 Amazon RDS for Oracle 网站。

目标架构

工具

- 应用程序层：将 Java 应用程序重建至 WAR 文件。
- 数据库层：Oracle 本机备份与还原。
- 适用于 Jakarta EE 的 Apache Tomcat 迁移工具。该工具采用为在 Apache Tomcat 9 上运行的 Java EE 8 编写的 Web 应用程序，然后自动将其转换为实现 Jakarta EE 9 的 Apache Tomcat 10 上运行。

操作说明

计划迁移

Task	描述	所需技能
完成应用程序发现、当前状态占用空间与性能基准。		BA，迁移主管
验证源数据库和目标数据库版本。		数据库管理员
确定目标服务器 EC2 实例的硬件要求。		数据库管理员， SysAdmin
识别存储需求（存储类型和容量）。		数据库管理员， SysAdmin
根据容量、存储功能和网络功能选择正确的 EC2 实例类型。		数据库管理员， SysAdmin
确定源数据库和目标数据库的网络访问安全要求。		数据库管理员， SysAdmin
确定应用程序迁移策略与工具。		数据库管理员，迁移主管
完成应用程序迁移设计与迁移指南。		构建主管，迁移主管
完成应用程序迁移运行手册。		构建主管，割接主管，测试主管，迁移主管

配置基础设施

Task	描述	所需技能
创建虚拟私有云（VPC）。		SysAdmin

Task	描述	所需技能
创建安全组。		SysAdmin
配置和启动 Amazon RDS for Oracle。		数据库管理员，SysAdmin

迁移数据

Task	描述	所需技能
创建或获取对端点访问权限，以获取数据库备份文件。		数据库管理员
使用原生数据库引擎或第三方工具迁移数据库对象和数据。	有关详细信息，请参阅其他信息部分中的“迁移数据库对象和数据”。	数据库管理员

迁移应用程序

Task	描述	所需技能
提交变更申请 (CR)。以进行迁移。		割接主管
获得 CR 批准，以进行迁移。		割接主管
遵循应用程序迁移运行手册的应用程序迁移策略。	有关详细信息，请参阅其他信息部分中的设置应用程序层。	数据库管理员，迁移工程师，应用程序所有者
升级应用程序（如有必要）。		数据库管理员，迁移工程师，应用程序所有者
完成功能测试、非功能测试、数据验证、SLA 与性能测试。		测试负责人、应用程序所有者、应用程序用户

割接

Task	描述	所需技能
获得应用程序所有者或企业主的签名。		割接主管
将应用程序客户端切换至新基础设施。		数据库管理员，迁移工程师，应用程序所有者

关闭项目

Task	描述	所需技能
关闭临时 AWS 资源。		DBA，迁移工程师，SysAdmin
审核和验证项目文档。		迁移主管
收集关于迁移时间、手动任务与自动任务的百分比以及成本节省等指标。		迁移主管
关闭项目并提供反馈。		迁移主管，应用程序所有者

相关资源

参考

- Apache Tomcat 10.0 文档
- Apache Tomcat 9.0 文档
- Apache Tomcat 8.0 文档
- Apache Tomcat 8.0 安装指南
- Apache Tomcat JNDI 文档
- Amazon RDS for Oracle 网站
- Amazon RDS 定价

- Oracle 和 Amazon Web Services
- Amazon RDS 上的 Oracle
- Amazon RDS 多可用区部署

教程和视频

- Amazon RDS 入门

其他信息

迁移数据库对象与数据

例如，如果您使用本机 Oracle 备份/恢复实用程序：

1. 为数据库备份文件创建 Amazon Simple Storage Service (Amazon S3) 备份（可选）。
2. 将 Oracle Database 数据备份到网络共享文件夹。
3. 登录迁移暂存服务器，以映射网络共享文件夹。
4. 将数据从网络共享文件夹复制到 S3 存储桶。
5. 申请 Amazon RDS Multi-AZ 部署。
6. 将本地数据库备份恢复至 Amazon RDS for Oracle。

设置应用程序层

1. 从 Apache Tomcat 网站安装 Tomcat 8（或 9/10）。
2. 将应用程序和共享库打包至 WAR 文件。
3. 在 Tomcat 中部署 WAR 文件。
4. 监控启动日志，查看`Linux cat`所有缺少的共享库 WebSphere。
5. 观看`Linux cat`任何 WebSphere特定部署描述符扩展的起始记录。
6. 从 WebSphere 服务器收集所有缺失的依赖 Java 库。
7. 使用与 Tomc WebSphere at 兼容的等效项修改特定部署描述符元素。
8. 使用依赖 Java 库和更新的部署描述符重建 WAR 文件。
9. 更新 LDAP 配置、数据库配置和测试连接（请参阅 Apache Tomcat 文档中的 Realm Configuration HOW-TO 以及 JNDI Datasource HOW-TO）。
10. 针对还原的 Amazon RDS for Oracle Database 测试已安装的应用程序。

11从该 EC2 实例创建适用于 Linux 的亚马逊系统映像 (AMI)。

12.使用应用程序负载均衡器和自动扩缩组启动已完成的架构。

13更新 URLs （使用 WebSEAL 连接点）以指向 Application Load Balancer。

14配置管理数据库 (CMDB)

使用 Auto Scaling 从 IBM WebSphere 应用程序服务器迁移到亚马逊 EC2 上的 Apache Tomcat

由 Kevin Yung (AWS) 和 Afroz Khan (AWS) 创作

摘要

此模式为在启用了 Amazon Aut EC2 o Scaling 的亚马逊弹性计算云 (亚马逊 EC2) 实例上将 Java WebSphere 应用程序从 IBM 应用程序服务器迁移到 Apache Tomcat 提供了指导。

通过使用此示例，您可实现：

- 降低 IBM 许可成本
- 使用多可用区部署，实现高可用性
- 使用 Amazon A EC2 uto Scaling 提高了应用程序的弹性

先决条件和限制

先决条件

- Java 应用程序（版本 7. x 或 8. x）应该在 LAMP 堆栈中开发。
- 目标状态是在 Linux 主机托管的 Java 应用程序。此示例已在 Red Hat Enterprise Linux (RHEL) 7 环境中成功实现。其他 Linux 发行版可遵循这种模式，但应参考 Apache Tomcat 发行版的配置。
- 您应该了解 Java 应用程序的依赖项。
- 您必须有权访问 Java 应用程序源代码才能更改。

限制和更换平台的变化

- 您应该了解企业存档 (EAR) 组件，确认所有库都打包在 Web 组件 WAR 文件中。您需要配置 Apache Maven WAR Plugin 并生成 WAR 文件构件。
- 使用 Apache Tomcat 8，servlet-api.jar 和应用程序包内置 jar 文件之间存在已知冲突。若要解决此问题，请将 servlet-api.jar 从应用程序包中删除。
- 您必须配置位于 Apache Tomcat 配置的类路径中的 Web-inf/资源。默认情况下，将 JAR 库加载至以下目录。或者，您可以部署下的所有资源src/main/resources。
- 检查 Java 应用程序中是否存在任何硬编码的上下文根，并更新 Apache Tomcat 上下文根目录
- 若要设置 JVM 运行时选项，可以在 Apache Tomcat bin 文件夹中创建配置文件 setenv.sh；如 JAVA_OPTS、JAVA_HOME 等。

- 身份验证是在容器级别配置，并在 Apache Tomcat 配置中设置为一个领域。已为以下三个领域中的任何一个建立身份验证：
  - JDBC Database Realm 在 JDBC 驱动程序访问的关系数据库查找用户。
  - DataSource Database Realm 在 JNDI 访问的数据库中查找用户。
  - JNDI Directory Realm 在 JNDI 提供者可以访问的轻型目录访问协议 (LDAP) 目录中查找用户。查询需要：
    - LDAP 连接详细信息：用户搜索库、搜索筛选条件、角色库、角色筛选条件
    - 密钥 JNDI Directory Realm：连接至 LDAP、对用户进行身份验证并检索用户所属的所有群组
- 授权：如果容器具有基于角色授权，可以检查 web.xml 中的授权限制，则必须定义 Web 资源并将其与约束条件中定义的角色进行比较。如果 LDAP 没有组角色映射，则必须在 web.xml 中设置属性 < security-role-ref > 才能实现组角色映射。要查看配置文档的示例，请参见 Oracle 文档。
- 数据库连接：在 Apache Tomcat 中通过 Amazon Relational Database Service (Amazon RDS) 端点 URL 和连接详细信息创建资源 使用 JNDI 查找更新应用程序代码以引用 a DataSource 。中定义的现有数据库连接 WebSphere 不起作用，因为它使用 WebSphere的 JNDI 名称。你可以<resource-ref> 在 web.xml 中添加一个带有 JNDI 名称和 DataSource 类型定义的条目。若要查看示例配置文档，请参阅 Apache Tomcat 文档。
- 日志记录：默认情况下，Apache Tomcat 将日志记录到控制台或日志文件中。您可通过更新 logging.properties（参见Tomcat 日志记录），启用 realm 级追踪。如果您使用 Apache Log4j 将日志附加至文件中，则必须下载 tomcat-juli 并将其添加至 classpath。
- 会话管理：如果您保留 IBM WebSEAL 用于应用程序负载平衡与会话管理，则无需进行任何更改。如果您在 AWS 上使用应用程序负载均衡器或网络负载均衡器来取代 IBM WebSEAL 组件，则必须使用带有 Memcached 集群的 Amazon ElastiCache 实例来设置会话管理，并将 Apache Tomcat 设置为使用开源会话管理。
- 如果您使用的是 IBM WebSEAL 转发代理，则必须设置新的适用于 AWS 的网络负载均衡器。使用 Network Load Balancer IPs 提供的 WebSEAL 接合点配置。
- SSL 配置：我们建议您使用安全套接字层 (SSL) 进行 end-to-end通信。若要在 Apache Tomcat 中设置 SSL 服务器配置，请按照 Apache Tomcat 文档中的说明进行操作。

架构

源技术堆栈

- IBM WebSphere 应用程序服务器

目标技术堆栈

- 该架构使用 Elastic Load Balancing（版本 2）。如果您使用 IBM WebSEAL 进行身份管理和负载平衡，则可以选择适用于 AWS 的网络负载均衡器与 IBM WebSEAL 反向代理集成。
- Java 应用程序部署到 Apache Tomcat 应用程序服务器，该服务器在 A mazon A EC2 uto Scaling 组中的 EC2 实例上运行。您可以基于 Amazon CloudWatch 指标（例如 CPU 利用率）设置扩展策略。
- 如果您要停止使用 IBM WebSEAL 进行负载平衡，则可以使用 A mazon for Memcached ElastiCache 进行会话管理。
- 对于后端数据库，您可以部署 High Availability (Multi-AZ) for Amazon RDS 并选择数据库引擎类型。

目标架构

工具

- AWS CloudFormation
- AWS 命令行界面（AWS CLI）
- Apache Tomcat（版本 7.x 或 8.x）
- RHEL 7 或 Centos 7
- Amazon RDS Multi-AZ 部署
- ElastiCache 适用于 Memcached 的亚马逊（可选）

操作说明

设置 VPC

Task	描述	所需技能
创建虚拟私有云（VPC）。		
创建子网。		
必要时创建路由表。		

Task	描述	所需技能
创建网络访问控制列表 (ACLs)。		
设置 AWS Direct Connect 或企业 VPN 连接。		

更换应用程序平台

Task	描述	所需技能
重构应用程序构建 Maven 配置，以生成 WAR 构件。		
在 Apache Tomcat 中重构应用程序依赖项数据来源。		
重构应用程序源代码，以使用 Apache Tomcat 中的 JNDI 名称。		
将 WAR 神器部署至 Apache Tomcat 中。		
完成应用程序验证与测试。		

配置网络

Task	描述	所需技能
配置公司防火墙，以允许连接到依赖项服务。		
将公司防火墙配置为允许最终用户访问 Elastic Load Balancing on AWS。		

创建应用程序基础设施

Task	描述	所需技能
在 EC2 实例上创建和部署应用程序。		
创建 Amazon f ElastiCache or Memcached 集群用于会话管理。		
为后端数据库创建 Amazon RDS Multi-AZ 实例。		
创建 SSL 证书并将其导入 AWS Certificate Manager (ACM)。		
在负载均衡器上安装 SSL 证书。		
为 Apache Tomcat 服务器安装 SSL 证书。		
完成应用程序验证与测试。		

割接

Task	描述	所需技能
关闭现有基础设施。		
将数据库从生产环境恢复至 Amazon RDS。		
通过更改 DNS 割接应用程序。		

相关资源

参考

- Apache Tomcat 7.0 文档
- Apache Tomcat 7.0 安装指南
- Apache Tomcat JNDI 文档
- Amazon RDS 多可用区部署
- 亚马逊 Memcac ElastiCache hed 版

教程和视频

- Amazon RDS 入门

将 .NET 应用程序从 Microsoft Azure 应用服务迁移到 AWS Elastic Beanstalk

由 Raghavender Madamshitti (AWS) 创建

摘要

此模式介绍如何将 Microsoft Azure 应用服务上托管的 .NET Web 应用程序迁移到 AWS Elastic Beanstalk。有两种方法可以将应用程序迁移到 Elastic Beanstalk：

- 使用 AWS Toolkit for Visual Studio - 此插件适用于 Microsoft Visual Studio IDE，提供了将自定义 .NET 应用程序部署到 AWS 的最简单、最直接的方法。您可以使用此方法将 .NET 代码直接部署到 AWS，并直接从 Visual Studio 创建支持资源，例如 Amazon Relational Database Service（Amazon RDS）for SQL Server 数据库。
- 上传并部署到 Elastic Beanstalk - 每个 Azure 应用服务都包含一个名为 Kudu 的后台服务，该服务可用于捕获内存转储和部署日志、查看配置参数以及访问部署包。您可以使用 Kudu 控制台访问 Azure 应用服务内容，提取部署包，然后使用 Elastic Beanstalk 控制台中的上传和部署选项将程序包上传到 Elastic Beanstalk。

此模式介绍了第二种方法（通过 Kudu 将应用程序上传到 Elastic Beanstalk）。该模式还使用以下 AWS 服务：AWS Elastic Beanstalk、亚马逊虚拟私有云（亚马逊 VPC）、亚马逊、亚马逊弹性计算云（ CloudWatch亚马逊）Auto Scaling、亚马逊简单存储服务（亚马逊 S3 EC2）Simple Storage Scaling 和亚马逊 Route 53。

.NET Web 应用程序部署到 AWS Elastic Beanstalk 上，后者在亚马 EC2 逊 Auto Scaling 组中运行。您可以基于 Amazon CloudWatch 指标（例如 CPU 利用率）设置扩展策略。对于数据库，您可以在多可用区环境中使用 Amazon RDS 或 Amazon DynamoDB，具体取决于您的应用程序和业务需求。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 在 Azure 应用服务中运行的 .NET Web 应用程序
- 使用 Azure 应用服务 Kudu 控制台的权限

产品版本

- .NET Core (x64) 1.0.1、2.0.0 或更高版本，或 .NET Framework 4.x、3.5（请参阅 Windows Server 平台上的 .NET 历史记录）

- Internet Information Services（IIS）版本 8.0 或更高版本，在 Windows Server 2012 或更高版本上运行
- .NET 2.0 或 4.0 运行时系统。

架构

源技术堆栈

- 使用 .NET Framework 3.5 或更高版本或 .NET Core 1.0.1、2.0.0 或更高版本开发并托管在 Azure 应用服务（Web 应用或 API 应用）上的应用程序

目标技术堆栈

- AWS Elastic Beanstalk 在亚马逊 EC2 Auto Scaling 群组中运行

迁移架构

部署工作流

工具

工具

- .NET Core 或 .NET Framework
- C#
- IIS
- Kudu 控制台

Amazon Web Services 和特征

- AWS Elastic Be anstalk — Elastic Beanstalk 是一项用于部署和扩展 easy-to-use .NET Web 应用程序的服务。Elastic Beanstalk 可自动管理容量预置、负载均衡和自动扩缩。
- Amazon A EC2 uto Scaling 组 — Elastic Beanstalk 包括一个 Auto Scaling 组，负责管理环境 EC2 中的亚马逊实例。在单实例环境中，自动扩缩组可确保始终有一个正在运行的实例。在负载平衡的环

境中，您可以为该组配置一系列要运行的实例，Amazon A EC2 uto Scaling 会根据需要根据负载添加或删除实例。

- Elastic Load Balancing — 当你在 AWS Elastic Beanstalk 中启用负载平衡时，它会创建一个负载均衡器，用于在环境中的实例之间分配流量 EC2 。
- 亚马逊 CloudWatch — Elastic Beanstalk 自动 CloudWatch 使用亚马逊来提供有关您的应用程序和环境资源的信息。Amazon CloudWatch 支持标准指标、自定义指标和警报。
- Amazon Route 53 - Amazon Route 53 是一项高度可用且可扩展的云域名系统（DNS）Web 服务。您可以使用 Route 53 别名记录将自定义域名映射到 AWS Elastic Beanstalk 环境。

操作说明

设置 VPC

Task	描述	所需技能
设置虚拟私有云（VPC）。	在您的 Amazon Web Services account 中，使用所需信息创建一个 VPC。	系统管理员
创建子网。	在您的 VPC 中创建两个或更多子网。	系统管理员
创建路由表。	根据您的要求创建路由表。	系统管理员

设置 Elastic Beanstalk

Task	描述	所需技能
访问 Azure 应用服务 Kudu 控制台。	导航到应用服务控制面板，然后选择高级工具，然后转到，通过 Azure 门户访问 Kudu。或者，可以修改 Azure 应用服务 URL，如下所示：`https://<appservicename>.scm.azurewebsites.net` 。	应用程序开发人员、系统管理员

Task	描述	所需技能
从 Kudu 下载部署包。	PowerShell 通过选择相应DebugConsole选项导航到 Windows。这将打开 Kudo 控制台。转到 `wwwroot` 文件夹并下载它。这会将 Azure 应用服务部署包下载为 zip 文件。有关示例，请参阅附件。	应用程序开发人员、系统管理员
为 Elastic Beanstalk 创建程序包。	解压缩从 Azure 应用服务下载的部署包。创建名为 `aws-windows-deployment-manifest.json` 的 JSON 文件（只有 .NET Core 应用程序才需要此文件）。创建一个包含 `aws-windows-deployment-manifest.json` 和 Azure 应用服务部署包文件的 zip 文件。有关示例，请参阅附件。	应用程序开发人员、系统管理员
创建新的 Elastic Beanstalk 应用程序。	打开 Elastic Beanstalk 控制台。选择现有应用程序或创建新应用程序。	应用程序开发人员、系统管理员
创建环境。	在 Elastic Beanstalk 控制台操作菜单中，选择创建环境。选择 Web 服务器环境和 .NET/IIS 平台。对于应用程序代码，选择上传。上传您为 Elastic Beanstalk 准备的 zip 文件，然后选择创建环境。	应用程序开发人员、系统管理员

Task	描述	所需技能
配置亚马逊 CloudWatch。	默认情况下，基本 CloudWatch 监控处于启用状态。如果您想要更改配置，请在 Elastic Beanstalk 向导中选择已发布的应用程序，然后选择监控。	系统管理员
验证部署包是否位于 Amazon S3 中。	创建应用程序环境后，您可以在 S3 存储桶中找到部署包。	应用程序开发人员、系统管理员
测试应用程序。	创建环境后，使用 Elastic Beanstalk 控制台中提供的 URL 测试应用程序。	系统管理员

相关资源

- AWS Elastic Beanstack 概念（Elastic Beanstalk 文档）
- Elastic Beanstalk 上的 .NET 入门（Elastic Beanstalk 文档）
- Kudu 控制台 () GitHub
- 使用“Kudu”管理 Azure Web 应用（GS 实验室文章）
- 自定义 ASP.NET Core Elastic Beanstalk 部署（AWS Toolkit for Visual Studio 用户指南）
- Elastic Load Balancing 文档
- AWS Elastic Beanstalk 支持的平台（Elastic Beanstalk 文档）
- 将 Web 应用程序部署到 AWS（C# Corner 文章）
- 扩展 Auto Scaling 组的规模（亚马逊 EC2 文档）
- Amazon RDS 的高可用性（多可用区）（Amazon RDS 文档）

其他信息

备注

- 如果要将本地或 Azure SQL Server 数据库迁移到 Amazon RDS，还必须更新数据库连接详细信息。
- 出于测试目的，附加了一个示例演示应用程序。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

在 Amazon ECS 上从 Oracle 迁移 WebLogic 到 Apache Tomcat (ToMee)

由 Anya Epishcheva (AWS) 和 Harshad Gohil (AWS) 创作

摘要

此模式讨论了使用亚马逊弹性容器服务 (Amazon ECS) 将运行 Oracle 的本地 Oracle Solaris SPARC 系统迁移 WebLogic 到运行 Apache toMee（增加了容器支持的 Apache Tom cat）的基于 Docker 容器的安装的步骤。

有关将与要从 Oracle 迁移的应用程序关联的数据库迁移 WebLogic 到 Tomcat 的信息，请参阅此目录中的数据库迁移模式。

最佳实践

迁移 Java 和 Java Enterprise Edition (Java EE) Web 应用程序的步骤有所不同，具体取决于应用程序使用的容器特定资源的数量。基于 Spring 的应用程序通常更容易迁移，因为它们对部署容器有少量依赖项。相比之下，使用企业 JavaBeans (EJBs) 和托管容器资源（例如线程池、Java 身份验证和授权服务 (JAAS) 以及容器管理的持久性 (CMP)）的 Java EE 应用程序需要付出更多努力。

为 Oracle Application Server 开发的应用程序经常使用 Oracle Identity Management 套件。迁移至开源应用程序服务器的客户通常会选择使用基于 SAML 的联合身份验证来重新实现身份和访问管理。其他人则使用 Oracle HTTP Server Webgate 来处理无法从 Oracle 身份管理套件迁移情况。

Java 和 Java EE Web 应用程序非常适合部署至基于 Docker 的 Amazon Web Services 上，例如 AWS Fargate 和 Amazon ECS。客户经常选择预装最新版本的目标应用程序服务器（例如 TomEE）和 Java 开发工具包（JDK）的 Docker 映像。他们将应用程序安装在基础 Docker 映像之上，将其发布到 Amazon Elastic Container Registry (Amazon ECR) 注册表中，并使用它在 AWS Fargate 或 Amazon ECS 上进行应用程序的可扩展部署。

理想情况下，应用程序部署是有弹性的；也就是说，应用程序实例的数量根据流量或工作负载而缩小或缩小。这意味着应用程序实例需要上线或终止按根据需求调整容量。

将 Java 应用程序迁移至 AWS 时，请考虑将其设置为无状态。这是 AWS Well-Architected Framework 的一项关键架构原则，将使用容器化实现水平扩缩。例如，多数基于 Java 的 Web 应用程序在本地存储用户会话信息。为了避免由于亚马逊弹性计算云 (Amazon EC2) 中的自动扩展或其他原因而导致应用程序实例终止，应在全球范围内存储用户会话信息，以便 Web 应用程序用户无需重新连接或重新登录网络应用程序即可继续无缝透明地工作。这种方法有多种架构选项，包括 Amazon ElastiCache for Redis，或者将会话状态存储在全局数据库中。TomEE 等应用程序服务器具有插件，可以通过 Redis、数据库和其他全局数据存储实现会话存储和管理。

使用可轻松与 Amazon 和 AWS X-Ray 集成的通用集中式日志 CloudWatch 和调试工具。迁移提供了改进应用程序生命周期功能的机会。例如，您可能希望自动化构建过程，以便使用持续集成和持续交付 (CI/CD) 管道轻松进行更改。这可能需要对应用程序进行更改，以便可以在不停机的情况下部署。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 源代码 Java 代码和 JDK
- 使用 Oracle 构建的源应用程序 WebLogic
- 为身份和访问管理定义的解决方案（SAML 或 Oracle Webgate）
- 为应用程序会话管理定义的解决方案（移动 like-for-like或与 Amazon 一起移动 ElastiCache，或者根据需要使应用程序处于无状态状态）
- 了解团队是否需要重构特定于 J2EE 的库才能移植到 Apache ToMee（参见 Apache 网站上的 Java EE 7 实施状态）
- 根据安全要求强化 ToMee 镜像
- 带有预装目标 ToMee 容器镜像
- 同意并在需要时实施应用程序补救措施（例如，记录调试版本、身份验证）

产品版本

- Oracle WebLogic OC4 J、9i、10g
- Tomcat 7（使用 Java 1.6 或更高版本）

架构

源技术堆栈

- 使用 Oracle 构建的 Web 应用程序 WebLogic
- 使用 Oracle Webgate 或 SAML 身份验证 Web 应用程序
- 连接到 Oracle 数据库 10g 及以上版本的 Web 应用程序

目标技术堆栈

- 在Amazon ECS 上运行的 ToMee（增加了容器支持的 Apache Tomcat）（另请参阅部署 Java Web 应用程序 和 Java Microservices on Amazon ECS）
- 适用于 Oracle 的 Amazon Relational Database Service (Amazon RDS)； 有关 Amazon RDS 支持的 Oracle 版本，请参阅 Amazon RDS for Oracle

目标架构

工具

若要在 ToMee 上运行，必须将 Java 应用程序重新构建为.war 文件。在某些情况下，可能需要更改应用程序才能在 TomEE 上运行应用程序；您应该检查以确保正确定义了必要的配置选项和环境属性。

此外，还应正确定义 Java 命名和目录接口 (JNDI) 查找和 JavaServer 页面 (JSP) 命名空间。考虑检查应用程序使用的文件名，以避免与内置 T 库发生命名冲突。例如，persistence.xml 是 Apache OpenJPA 框架（在 ToMee 中与 OpenEJB 捆绑在一起）用于配置目的的文件名。PUI 中的 persistence.xml 文件包含 Spring 框架 bean 声明。

Tomee 版本 7.0.3 及更高版本（Tomcat 8.5.7 及更高版本）针对带有特殊字符的原始（未编码）返回 HTTP 400 响应（错误请求）。 URLs 服务器响应对最终用户显示为空白页。Tomee 和 Tomcat 的早期版本允许在中使用某些未编码的特殊字符 URLs；但是，正如 CVE-2016-6816 网站上所述，它被认为是不安全的。要解决网址编码问题， JavaScript 必须使用 encodeURI () 方法对直接 URLs 传递给浏览器的内容进行编码，而不是用作原始字符串。

在 Tomee 中部署.war 文件后，在 Linux cat监视任何缺失的共享库的启动日志，并监视特定于 Oracle 的扩展，以添加 Tomcat 库中缺少的组件。

一般过程

- 在 ToMee 配置应用程序。
- 识别并重新配置应用程序服务器特定的配置文件和资源，从源格式转换为目标格式。
- 识别和重新配置 JNDI 资源。
- 将 EJB 命名空间和查询调整为目标应用程序服务器所需格式（如适用）。
- 重新配置 JAAS 应用程序容器特定的安全角色和主映射（如适用）。
- 将应用程序和共享库打包至 WAR 文件。

- 使用提供的 Docker 容器在 ToMee 部署.war 文件。
- 监控启动日志，找出任何缺少的共享库和部署描述符扩展。如果找到任何任务，请返回第一项任务。
- 针对还原的 Amazon RDS 数据库测试已安装的应用程序。
- 按照部署 Docker 容器中的说明启动带有负载均衡器和 Amazon ECS 集群的完整架构。
- 更新为指 URLs 向负载均衡器。
- 配置管理数据库 (CMDB)

操作说明

计划迁移

Task	描述	所需技能
执行应用程序发现（当前状态占用空间和性能基准）。		BA，迁移主管
验证源和目标数据库的版本。		数据库管理员
验证源应用程序和目标应用程序的设计（身份和会话管理）。		数据库管理员，迁移工程师，应用程序所有者
确定目标服务器实例的硬件要求。		数据库管理员， SysAdmin
根据容量、存储功能和网络功能选择正确的实例类型。		数据库管理员， SysAdmin
确定源数据库和目标数据库的网络访问安全要求。		数据库管理员， SysAdmin
确定应用程序迁移/切换策略。		数据库管理员，迁移主管
完成应用程序迁移设计与迁移指南。		构建主管，迁移主管
完成应用程序迁移运行手册。		构建主管，割接主管，测试主管，迁移主管

配置基础设施

Task	描述	所需技能
创建虚拟私有云（VPC）。		SysAdmin
创建安全组。		SysAdmin
配置和启动 Amazon RDS 数据库实例。		数据库管理员， SysAdmin
配置 Amazon ECS 部署。		SysAdmin
将应用程序打包为 Docker 映像。		SysAdmin
将镜像推送到 Amazon ECR 注册表（或者跳过此步骤并将其推送到 Amazon ECS 集群）。		SysAdmin
配置应用程序和 Amazon ECS 服务选项的任务定义。		SysAdmin
配置集群，查看安全设置，设置 AWS Identity and Access Management (IAM) 角色。		SysAdmin
根据应用程序迁移运行手册启动设置并运行测试。		SysAdmin

迁移数据

Task	描述	所需技能
获得安全保障团队的许可，将生产数据转移至 AWS。		数据库管理员，迁移工程师，应用程序所有者
创建或获取对端点访问权限，以获取数据库备份文件。		数据库管理员

Task	描述	所需技能
使用原生 Microsoft SQL Server 工具或第三方工具迁移数据库对象和数据。		数据库管理员
从应用程序迁移运行手册中运行必要的测试，以确认数据迁移成功。		数据库管理员，迁移工程师，应用程序所有者

迁移应用程序

Task	描述	所需技能
提交变更申请 (CR) 以进行迁移。		割接主管
获得 CR 批准，以进行迁移。		割接主管
遵循应用程序迁移运行手册的应用程序迁移策略。		数据库管理员，迁移工程师，应用程序所有者
升级应用程序（如需要）。		数据库管理员，迁移工程师，应用程序所有者
完成功能测试、非功能测试、数据验证、SLA 与性能测试。		测试负责人、应用程序所有者、应用程序用户

割接

Task	描述	所需技能
获得应用程序所有者或企业主的签名。		割接主管
运行表格主题练习，演练割接运行手册的所有步骤。		数据库管理员，迁移工程师，应用程序所有者

Task	描述	所需技能
将应用程序客户端切换至新基础设施。		数据库管理员，迁移工程师，应用程序所有者

关闭项目

Task	描述	所需技能
关闭临时 AWS 资源。		DBA，迁移工程师，SysAdmin
审核和验证项目文档。		迁移主管
收集与迁移时间、工具成本节约等相关的指标。		迁移主管
关闭项目并提供反馈。		迁移主管，应用程序所有者

相关资源

参考

- Apache Tomcat 7.0 文档
- Apache Tomcat 7.0 安装指南
- Apache Tomcat JNDI 文档
- Apache ToMee 文档
- Amazon RDS for Oracle
- Amazon RDS 定价
- Oracle 和 AWS
- Oracle on Amazon RDS 文档
- Amazon RDS 多可用区部署
- Amazon ECS 入门
- Amazon RDS 入门

教程和视频

- 在 Amazon RDS 上运行 Oracle 数据库的最佳实践

使用 AWS DMS 将 Oracle 数据库从亚马逊迁移 EC2 到 Amazon RDS for Oracle

由 Chethan Gangadharaiah (AWS) 和 Brian motzer (AWS) 创作

摘要

此模式描述了使用 AWS 数据库迁移服务 (AWS DMS) 将亚马逊弹性计算云 (Amazon EC2) 上的 Oracle 数据库迁移到适用于 Oracle 的亚马逊关系数据库服务 (Amazon RDS) 的步骤。该模式还使用 Oracle SQL Developer 或 SQL *Plus 连接到您的 Oracle 数据库实例，并包括一个可自动执行某些任务的 AWS CloudFormation 模板。

迁移至 Amazon RDS for Oracle 让您专注于业务和应用程序，而 Amazon RDS 则负责数据库管理任务，例如预置数据库、备份和恢复、安全补丁、版本升级和存储管理。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 亚马逊 Oracle 数据库的亚马逊系统映像 (AMI) EC2

产品版本

- AWS DMS 支持适用于 Amazon RDS 实例数据库的 Oracle 版本 11g（版本 11.2.0.3.v1 和更高版本）、12c 和 18c，包括 Enterprise、Standard、Standard One 和 Standard Two 版本。有关支持的版本的最新信息，请参阅 AWS 文档中的使用 Oracle 数据库作为 AWS DMS 的目标。（随附的 AWS CloudFormation 模板使用 Oracle 版本 12c 作为源数据库。）
- Oracle SQL Developer 4.0.3

架构

源架构

- 亚马逊上的 Oracle 数据库 EC2

目标架构

- Amazon RDS for Oracle

迁移架构

工具

- AWS DMS – AWS Database Migration Service (AWS DMS) 可帮助您快速安全地将数据库迁移到 AWS。其支持同构与异构迁移。有关支持的 Oracle 数据库版本和版本的信息，请参阅 AWS 文档中的使用 Oracle 数据库作为 AWS DMS 的源和使用 Oracle 数据库作为 AWS DMS 的目标。
- Oracle SQL Developer 或 SQL *Plus – 此类工具允许您连接至 Amazon RDS for Oracle 数据库实例。

操作说明

设置目标数据库

Task	描述	所需技能
创建 Amazon RDS for Oracle 数据库实例。	登录 AWS 管理控制台并通过以下网址打开 Amazon RDS 控制台 https://console.aws.amazon.com/rds/。为 Oracle 数据库选择相应的引擎、模板、数据库凭证设置、实例类型、存储、多可用区设置、虚拟私有云（VPC）和配置、登录凭证以及其他设置创建 Oracle 数据库实例。有关说明，请查看 “相关资源” 部分的链接。或者使用附件中的 AWS CloudFormation 模板 (create_rds.yaml) 创建 Amazon RDS for Oracle 数据库实例。	开发人员
连接 Amazon RDS 并向 Oracle 用户授予特权。	修改安全组以打开相应端口，以便从本地计算机和 AWS DMS 复制实例连接。配置连接时，请确保选择 “可公开访问”	开发人员

Task	描述	所需技能
	选项，这样您就可以从 VPC 外部连接至数据库。使用登录凭证通过 Oracle SQL Developer 或 SQL *Plus 连接至 Amazon RDS，创建 AWS DMS 用户，并为 AWS DMS 用户提供修改数据库所需权限。	

配置源 EC2 实例的安全组

Task	描述	所需技能
检查 Oracle 数据库是否已启动并正在运行。	使用安全外壳 (SSH) 连接到 EC2 实例，然后尝试使用 SQL *Plus 连接到 Oracle 数据库。	开发人员
修改安全组。	修改 EC2 实例的安全组以打开相应的端口，这样您就可以从本地计算机和 AWS DMS 复制实例进行连接。	开发人员

设置 AWS DMS

Task	描述	所需技能
创建 AWS DMS 复制实例。	在 AWS DMS 中，在与 Amazon RDS for Oracle 数据库实例相同的 VPC 中创建复制实例。指定复制实例的名称和描述，选择实例类别和复制引擎版本（使用默认值），选择您在其中创建 Amazon RDS 数据库实例的 VPC，根据需要设置多可用区设置，分配存	数据库管理员

Task	描述	所需技能
	储，指定可用区，并配置其他设置。或者，您可以使用附件中的 AWS CloudFormation 模板 (dms.yaml) 来实现此步骤。	
连接至源数据库端点和目标数据库端点。	经指定端点标识符、引擎、服务器、端口、登录凭证和其他连接属性创建源数据库端点和目标数据库端点。对于源服务器，请使用托管 Oracle 数据库的 EC2 实例的公有 DNS。对于目标服务器，请使用 Amazon RDS for Oracle 端点。运行测试，以验证源连接和目标连接是否正常工作。或者，您可以使用附件中的 AWS CloudFormation 模板 (dms.yaml) 来实现此步骤。	数据库管理员
创建 AWS DMS 任务。	创建 AWS DMS 任务，以将数据从源端点迁移至目标端点，在源端点和目标端点之间设置复制，或同时执行两种操作。创建 AWS DMS 任务时，请指定复制实例、源端点、目标端点、迁移类型（仅数据、仅复制或二者兼而有之）、表映射和筛选条件。在 Amazon CloudWatch 中运行 AWS DMS 任务，监控任务，检查表格统计数据并查看日志。或者，您可以使用附件中的 AWS CloudFormation 模板 (dms.yaml) 来实现此步骤。	数据库管理员

相关资源

- 创建 Amazon RDS 数据库实例
- 与运行 Oracle 数据库引擎的数据库实例连接
- AWS DMS 文档
- AWS DMS 演练 Step-by-Step
- 将 Oracle 数据库迁移至 AWS Cloud

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 Logstash 将本地 Oracle 数据库迁移到亚马逊 OpenSearch 服务

由 Aditya Goteti (AWS) 编写

摘要

此模式描述了如何使用 Logstash 将数据从本地 Oracle 数据库移动到亚马逊 OpenSearch 服务。它包括架构注意事项以及一些所需技能集和建议。数据可以来自单个表，也可以来自需要执行全文搜索的多个表。

OpenSearch 服务可以在虚拟私有云 (VPC) 中配置，也可以在基于 IP 的限制下公开。此模式描述了在 VPC 中配置 OpenSearch 服务的场景。Logstash 用于从 Oracle 数据库收集数据，将其解析为 JSON 格式，然后将数据馈送到服务中。 OpenSearch

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- Java 8（Logstash 6.4.3 要求）
- 本地数据库服务器与 VPC 中的亚马逊弹性计算云 (AWS EC2) 实例之间的连接，使用 AWS 虚拟专用网络 (AWS VPN) 建立
- 用于检索要从数据库推送到 Ser OpenSearch vice 的所需数据的查询
- Oracle Java 数据库连接JDBC 驱动程序

限制

- Logstash 无法识别从数据库硬删除的记录

产品版本

- Oracle Database 12c
- OpenSearch 服务 6.3
- Logstash 6.4.3

架构

源技术堆栈

- 本地 Oracle 数据库
- 本地 AWS VPN

目标技术堆栈

- VPC
- EC2 实例
- OpenSearch 服务
- Logstash
- NAT 网关（用于 EC2 实例上的操作系统更新以及安装 Java 8、Logstash 和插件）

数据迁移架构

工具

- Logstash 6.4.3
- JDBC 输入插件（下载及更多信息）
- Logstash 输出插件 (_es) logstash-output-amazon
- Oracle JDBC 驱动程序

操作说明

计划迁移

Task	描述	所需技能
确定源数据的大小。	源数据的大小是用于确定要在索引中配置的分片数量的参数之一。	数据库管理员，数据库开发人员
分析每一列的数据类型以及对应的数据。	OpenSearch 当在文档中发现以前看不见的字段时，服务会动态映射数据类型。如果需要显式声明任何特定的数据类型	应用程序所有者、开发人员、数据库开发人员

Task	描述	所需技能
	或格式（例如日期字段），请在创建索引时标识这些字段并定义这些字段的映射。	
确定是否有任何具有主键或唯一键的列。	为了避免在更新或插入期间在 Amazon S OpenSearch ervice 中重复记录，您需要在`amazon_es` 插件的输出部分配置`document_id` 设置（例如，主键`document_id => "%{customer_id}"` 在`customer_id` 哪里）。	应用程序所有者、开发人员
分析添加新记录的数量和频率；检查记录删除频率。	需要执行此任务来了解源数据的增长率。如果数据读取密集且插入很少，则可以使用单个索引。如果频繁插入新记录并且没有删除，则分片大小很容易超过建议的最大大小 50 GB。在这种情况下，您可通过在 Logstash 中配置索引模式以及在可以使用别名访问它的代码中动态创建索引。	应用程序所有者、开发人员
确定所需副本数量		应用程序所有者、开发人员
确定在索引上配置的分片数量。		应用程序所有者、开发人员
确定专用主节点、数据节点和 EC2 实例的实例类型。	有关更多信息，请参阅相关资源部分。	应用程序所有者、开发人员
确定所需专用主节点和数据节点的数量。	有关更多信息，请参阅相关资源部分。	

迁移数据

Task	描述	所需技能
启动实 EC2 例。	在 AWS VPN 所连接的 VPC 内启动一个 EC2 实例。	Amazon VPC 结构，AWS VPN
在实例上安装 Logstash。 EC2		开发人员
安装 Logstash 插件。	安装所需 Logstash 插件 `jdbc-input` 和 `logstash-output-amazon_es` 。	开发人员
配置 Logstash。	创建 Logstash 密钥库以存储 AWS Secrets Manager 密钥和数据库凭证等敏感信息，然后将引用放置在 Logstash 配置文件中。	开发人员
配置死信队列与永久队列。	默认情况下，当 Logstash 遇到由于数据包含映射错误或其他问题而无法处理的事件时，Logstash 管道会挂起或删除不成功的事件。为了防止这种情况下的数据丢失，您可将 Logstash 配置为将不成功的事件写入死信队列而不是丢弃它们。为了防止异常终止期间的数据丢失，Logstash 具有持久队列功能，可将消息队列存储在磁盘上。持久队列提供 Logstash 中的数据持久性。	开发人员
创建亚马逊 OpenSearch 服务域名。	使用不要求使用 AWS 身份和访问管理 (IAM) 凭证签署请求的访问策略创建亚马逊 OpenSearch 服务域。亚马逊	开发人员

Task	描述	所需技能
	OpenSearch 服务域必须在同一 VPC 内创建。您还应该根据您的分析选择实例类型并设置专用节点和主节点的数量。	
配置所需的亚马逊 OpenSearch 服务日志。	有关更多信息，请参阅OpenSearch 服务文档。	
创建索引。		开发人员
开启 Logstash。	将 Logstash 以后台服务运行。Logstash 运行配置的 SQL 查询，提取数据，将其转换为 JSON 格式，然后将其提供给服务。 OpenSearch 对于初始加载，不要在 Logstash 配置文件中配置调度程序。	开发人员
检查文档。	检查索引上的文档数量以及所有文档是否都存在于源数据库中。初始加载期间，它们会被添加至索引中并用于停止 Logstash。 更改 Logstash 配置。以添加根据客户端要求固定间隔运行的调度程序，然后重新启动 Logstash。Logstash 将仅选择上次运行后更新或添加的记录，上次运行时间戳存储在 Logstash 配置文件中使用 `last_run_metadata_path => “/usr/share/logstash/.logstash_jdbc_last_run”` 属性配置的文件中。	开发人员

相关资源

- 推荐 CloudWatch 警报
- 专用 Amazon OpenSearch 服务主节点
- 调整亚马逊 OpenSearch 服务域名的大小
- Logstash 文档
- JDBC 输入插件
- Logstash 输出插件
- 亚马逊 OpenSearch 服务网站

将本地 Oracle 数据库迁移到 Amazon RDS for Oracle

创建者：Baji Shaik (AWS) 和 Pavan Pusuluri (AWS)

摘要

此模式描述将本地 Oracle 数据库迁移到 Amazon Relational Database Service（Amazon RDS）for Oracle 的步骤。作为迁移过程的一部分，您需要制定迁移计划，并根据源数据库考虑有关目标数据库基础设施的重要因素。您可以根据业务需求和用例从两个迁移选项中选择一个：

- AWS Database Migration Service（AWS DMS）– 您可以使用 AWS DMS 快速安全地将数据库迁移到 Amazon Web Services Cloud。源数据库可在迁移过程中保持全面运行，从而最大程度地为依赖该数据库的应用程序缩短停机时间。您可以使用 AWS DMS 创建一个任务，在您完成初始全负载迁移后，通过名为更改数据捕获（CDC）的过程捕获正在进行的更改，从而缩短迁移时间。
- Oracle 原生工具 — 您可以使用原生 Oracle 工具迁移数据库，例如 Oracle 和 CDC 版 Or acle GoldenGate 的数据泵导出和数据泵导入。您也可以使用原始导出实用程序和原始导入实用程序等 Oracle 本机工具来缩短全负载时间。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 本地 Oracle 数据库
- Amazon RDS Oracle 数据库（DB）实例

限制

- 数据库大小限制：64 TB

产品版本

- Oracle 版本 11g（版本 11.2.0.3.v1 及更高版本）以及最高 12.2 和 18c。有关当前支持的版本和版本列表，请参阅 AWS 文档中的 Amazon RDS for Oracle。有关 AWS DMS 支持的 Oracle 版本，请参阅 AWS DMS 文档中的使用 Oracle 数据库作为 AWS DMS 的源数据库。

架构

源技术堆栈

- 本地 Oracle 数据库

目标技术堆栈

- Amazon RDS for Oracle

源架构和目标架构

下图显示了如何使用 AWS DMS 将本地 Oracle 数据库迁移到 Amazon RDS for Oracle。

图表显示了以下工作流：

1. 创建或使用现有数据库用户，向该用户授予所需 AWS DMS 权限，打开 ARCHIVELOG 模式，然后设置补充日志。
2. 在本地和 AWS 网络之间配置互联网网关。
3. 为 AWS DMS 配置源端点和目标端点。
4. 配置 AWS DMS 复制任务，将数据从源数据库迁移到目标数据库。
5. 在目标数据库上完成迁移后活动。

下图显示了如何使用 Oracle 本机工具将本地 Oracle 数据库迁移到 Amazon RDS for Oracle。

图表显示了以下工作流：

1. 使用 Oracle 导出（`exp`）和导入（`imp`）实用程序创建或使用现有数据库用户并授予备份 Oracle 数据库所需权限。
2. 在本地和 AWS 网络之间配置互联网网关。
3. 在堡垒主机上配置 Oracle 客户端以获取备份数据库。
4. 将备份数据库上传到 Amazon Simple Storage Service（Amazon S3）存储桶。
5. 将数据库备份从 Amazon S3 恢复到 Amazon RDS for Oracle 数据库。
6. 为 CDC 配置 GoldenGate Oracle。

7. 在目标数据库上完成迁移后活动。

工具

- AWS Database Migration Service (AWS DMS) 可帮助您将数据存储迁移到 Amazon Web Services Cloud，或者在云和本地设置的组合之间迁移。
- Oracle 原生工具可帮助您执行同构迁移。您可以使用 Oracle 数据泵在源数据库和目标数据库之间迁移数据。此模式使用 Oracle 数据泵执行从源数据库到目标数据库的完全加载。
- Oracle GoldenGate 可帮助您在两个或多个数据库之间执行逻辑复制。此模式 GoldenGate 用于在初始加载后使用 Oracle 数据泵复制增量更改。

操作说明

计划迁移

Task	描述	所需技能
创建项目文档并记录数据库详细信息。	1. 记录迁移目标、迁移要求、主要项目利益相关者、项目里程碑、项目截止日期、关键指标、迁移风险和风险缓解计划。 2. 记录有关源数据库的重要信息，包括 RAM、IOPS 和。CPUs稍后，您将使用此信息来确定相应的目标数据库实例。 3. 验证源数据库和目标数据库的版本。	数据库管理员
识别存储要求。	确定并记录存储需求，包括： 1. 计算为源数据库实例分配的存储空间。 2. 收集来自源数据库实例的历史增长指标。	数据库管理员， SysAdmin

Task	描述	所需技能
	3. 预测目标数据库实例的未来增长。 **Note** 对于通用型 (gp2) 固态硬盘卷，每 1 GB 存储空间可获得三个 IOPS。 通过计算源数据库的读取和写入 IOPS 总数来分配存储。	
根据计算要求选择正确的实例类型。	1. 确定目标数据库实例的计算要求。 2. 识别性能问题。 3. 考虑以下因素来确定合适的实例类型： • 源数据库实例的 CPU 使用率 • 源数据库实例的 IOPS（读取和写入） • 源数据库实例上的内存占用	SysAdmin
识别网络访问安全要求。	1. 识别并记录源数据库和目标数据库的网络访问安全要求。 2. 配置适当的安全组，使应用程序能够与数据库通信。	数据库管理员， SysAdmin

Task	描述	所需技能
确定应用程序迁移策略。	1. 确定并记录迁移割接策略。 2. 确定并记录应用程序的恢复时间目标 (RTO) 和恢复点目标 (RPO)，然后相应地规划切换。	DBA、 SysAdmin、应用程序所有者
识别迁移风险。	评测数据库并记录特定于迁移的风险和缓解措施。例如： • 识别未记录表格，突出显示在恢复时丢失数据的风险。 • 提取源数据库用户和权限，并突出显示与 Amazon RDS 权限的冲突。 • 查看警报日志，了解任何特定于 Oracle 的错误和警告。 • 识别目标数据库实例支持和不支持的功能。 • 查看目标数据库版本引擎的已弃用功能。	数据库管理员

配置基础设施

Task	描述	所需技能
创建 VPC。	为目标数据库实例创建新的 Amazon Virtual Private Cloud（Amazon VPC）。	SysAdmin
创建安全组。	在新 VPC 中创建安全组以允许数据库实例的入站连接。	SysAdmin

Task	描述	所需技能
创建 Amazon RDS for Oracle 数据库实例。	使用新的 VPC 和安全组创建目标数据库实例，然后启动该实例。	SysAdmin

选项 1-使用原生 Oracle 工具或第三方工具迁移数据

Task	描述	所需技能
准备源数据库。	1. 创建数据泵目录或使用现有目录。 2. 创建迁移用户并授予执行数据泵数据提取的权限。 3. 以 SQL 脚本的形式从源数据库中提取角色、用户和表空间。 4. 将提取的数据泵转储传输到目标数据库实例 `data pump` 目录。	数据库管理员，SysAdmin
准备目标数据库。	1. 确认 Amazon RDS for Oracle 目标数据库实例上安装或启用了所有数据库选项（例如，文本和 Java）。 2. 创建数据泵目录或使用现有目录。 3. 创建迁移用户并授予执行数据泵数据导入的权限。 4. 在目标数据库实例上创建所需表空间、用户和角色。 5. 将传输的数据泵导出转储到目标数据库。 6. 创建在导入或创建对象期间排除的所有索引。	数据库管理员，SysAdmin

Task	描述	所需技能
	7. 创建导入期间排除的所有约束。 8. 验证或重新编译无效对象。 9. 重建无效的索引。 10验证源数据库和目标数据库之间的数据库对象计数。 11解决在对象计数之间发现的任何差异。	

选项 2-使用 AWS DMS 迁移数据

Task	描述	所需技能
准备数据。	1. 清理源数据库中的数据。 2. 创建复制实例。 3. 创建源端点和目标端点。 4. 确定要迁移的表和对象的数量。	数据库管理员
迁移数据。	1. 删除目标数据库上的外键约束和触发器。 2. 删除目标数据库上的辅助索引。 3. 配置从源数据库到目标数据库的 AWS DMS 全负载任务设置。 4. 启用外键。 5. 启用 AWS DMS CDC 以复制正在进行的更改。 6. 启用触发器。 7. 更新序列。	数据库管理员

Task	描述	所需技能
	8. 验证源数据和目标数据。	

割接至目标数据库

Task	描述	所需技能
将应用程序客户端切换至新基础设施。	1. 停止所有指向 Oracle 的应用程序服务和客户端连接。 2. 运行 AWS DMS 任务。 3. 设置回滚任务（例如，将 CDC 从 Amazon RDS 数据库反向到本地 Oracle 数据库）。 4. 验证数据。 5. 通过将 Amazon Route 53 配置到 Amazon RDS for Oracle 的新数据库实例上，在新目标数据库上启动应用程序服务。 6. 将亚马逊 CloudWatch 监控添加到您的新 Amazon RDS for Oracle 数据库实例中。	DBA、 SysAdmin、应用程序所有者
实施您的回滚计划。	1. 停止所有指向 Amazon RDS for Oracle 数据库实例的应用程序服务。 2. 使用 AWS DMS 任务将更改回滚到本地 Oracle 源数据库。 3. 停止从本地 Oracle 数据库运行到 Amazon RDS for Oracle 数据库的 AWS DMS 任务。	数据库管理员、应用程序所有者

Task	描述	所需技能
	4. 在 Oracle 源数据库上重新启动所有应用程序。 5. 确认回滚部署已完成。	

关闭迁移项目

Task	描述	所需技能
清理资源。	关闭或移除 AWS 临时资源，例如 AWS DMS 复制实例和 S3 存储桶。	数据库管理员 ， SysAdmin
查看项目文档。	查看迁移计划文档和目标，然后确认您已完成所有必需的迁移步骤。	DBA、 SysAdmin、应用程序所有者
收集指标。	记录关键迁移指标，包括完成迁移所需时间、手动任务与基于工具的任务的百分比、成本节省以及其他相关指标。	DBA、 SysAdmin、应用程序所有者
关闭项目。	结束迁移项目并收集有关迁移工作的反馈。	DBA、 SysAdmin、应用程序所有者

相关资源

参考

- 将 Oracle 数据库迁移到 AWS 云（AWS Prescriptive Guidence）
- AWS Database Migration Service（AWS DMS）（AWS DMS 文档）
- Amazon RDS 定价（Amazon RDS 文档）

教程和视频

- AWS Database Migration Service 入门（AWS DMS 文档）

- Amazon RDS 资源（Amazon RDS 文档）
- AWS Database Migration Service (DMS) (YouTube)

使用 Oracle 数据泵将本地 Oracle 数据库迁移到 Amazon RDS for Oracle

创建者：Mohan Annam（AWS）和 Brian motzer（AWS）

摘要

此模式描述了如何使用 Oracle 数据泵将 Oracle 数据库从本地数据中心迁移到 Amazon Relational Database Service（Amazon RDS）for Oracle 数据库实例。

该模式包括从源数据库创建数据转储文件，将文件存储在 Amazon Simple Storage Service（Amazon S3）存储桶中，然后将数据恢复到 Amazon RDS for Oracle 数据库实例中。当您使用 AWS Database Migration Service（AWS DMS）进行迁移遇到限制时，此模式非常有用。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 在 AWS Identity and Access Management (IAM) 中创建角色以及 Amazon S3 分段上传所需权限
- 从源数据库导出数据所需权限
- AWS 命令行界面（AWS CLI）已安装并配置

产品版本

- Oracle 数据泵仅适用于 Oracle 数据库 10g 发行版 1 (10.1) 及更高版本。

架构

源技术堆栈

- 本地 Oracle 数据库

目标技术堆栈

- Amazon RDS for Oracle
- SQL 客户端（Oracle SQL 开发人员）
- 一个 S3 存储桶

源架构和目标架构

工具

Amazon Web Services

- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。在这种模式中，IAM 用于创建将数据从 Amazon S3 迁移到 Amazon RDS for Oracle 所需角色和策略。
- Amazon Relational Database Service (Amazon RDS) for Oracle 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩缩 Oracle 关系数据库。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

其他工具

- Oracle 数据泵可帮助您将数据和元数据从一个数据库高速移动至另一个数据库。在这种模式中，Oracle 数据泵用于将数据转储 (.dmp) 文件导出到 Oracle 服务器，然后将其导入到 Amazon RDS for Oracle 中。有关更多信息，请参阅 Amazon RDS 文档中的将数据导入到 Amazon RDS 上的 Oracle。
- Oracle SQL Developer 是一个集成的开发环境，可简化传统部署和基于云的部署中 Oracle 数据库的开发和管理。它与本地 Oracle 数据库和 Amazon RDS for Oracle 交互，运行导出和导入数据所需 SQL 命令。

操作说明

创建 S3 存储桶

Task	描述	所需技能
创建存储桶。	要创建 S3 存储桶，请按照 AWS 文档中的说明进行操作。	AWS 系统管理员

创建 IAM 角色并分配策略

Task	描述	所需技能
配置 IAM 权限。	要配置权限，请按照 AWS 文档中的说明进行操作。	AWS 系统管理员

创建 Amazon RDS for Oracle 目标数据库实例并关联 Amazon S3 集成角色

Task	描述	所需技能
创建 Amazon RDS for Oracle 目标数据库实例。	要创建 Amazon RDS for Oracle 实例，请按照 AWS 文档中的说明进行操作。	AWS 系统管理员
将该角色与数据库实例关联。	要将角色与实例关联，请按照 AWS 文档中的说明进行操作。	数据库管理员

在目标数据库上创建数据库用户

Task	描述	所需技能

创建 用户。	连接来自 Oracle SQL Developer 或 SQL*Plus 的 Amazon RDS for Oracle 目标数据库，然后运行以下 SQL 命令来创建要将架构导入的用户。
	`create user SAMPLE_SCHEMA identified by <PASSWORD>;`
	`grant create session, resource to <USER NAME>;`
		数据库管理员

Task	描述	所需技能
	`alter user <USER NAME> quota 100M on users;`	

从 Oracle 源数据库创建导出文件

Task	描述	所需技能
创建数据转储文件。	要在 DATA_PUMP_DIR 目录中创建名为 sample.dmp 的转储文件（用于导出 SAMPLE_SCHEMA 用户），请使用以下脚本。 `DECLARE hdnl NUMBER; BEGIN hdnl := dbms_datapump.open(operation => 'EXPORT', job_mode => 'SCHEMA', job_name => NULL); dbms_datapump.add_file( handle => hdnl, filename => 'sample.dmp', directory => 'DATA_PUMP_DIR', filetype => dbms_datapump.ku$_file_type_dump_file);`	数据库管理员

Task	描述	所需技能

	<pre> dbms_datapump.add_
	file(handle => hdnl,
	
	         filename =>
	 'export.log',
	
	        directory =>
	 'DATA_PUMP_DIR',
	
	         filetype =>
	 dbms_datapump.ku$_
	file_type_log_file);
	
	    dbms_datapump.meta
	data_filter(hdnl,
	 'SCHEMA_EXPR', 'IN
	 (''SAMPLE_SCHEMA''
	)');
	
	    dbms_datapump.star
	t_job(hdnl);
	END;
	/</pre>
	通过查看本地 DATA_PUMP_DIR 目录中的 export.log 文件来查看导出详细信息。	

将转储文件上传到 S3 存储桶中

Task	描述	所需技能

将数据转储文件从源上传到 S3 存储桶。	使用 AWS CLI 运行以下命令。
	
	<pre>aws s3 cp sample.dmp
	 s3://<bucket_creat
	ed_epic_1>/</pre>
		数据库管理员

将导出文件从 S3 存储桶下载到 RDS 实例

Task	描述	所需技能

将数据转储文件下载到 Amazon RDS	要将转储文件 `sample.dmp` 从 S3 存储桶复制到 Amazon RDS for Oracle 数据库，请运行以下 SQL 命令。在此示例中，`sample.dmp` 文件从 S3 存储桶 `my-s3-integration1` 下载到 Oracle 目录 `DATA_PUMP_DIR` 。确保分配给 RDS 实例的磁盘空间足以容纳数据库和导出文件。
	`-- If you want to download all the files in the S3 bucket remove the p_s3_prefix line.`
	`SELECT rdsadmin.rdsadmin_s3_tasks.download_from_s3( p_bucket_name => 'my-s3-integration', p_s3_prefix => 'sample.dmp', p_directory_name => 'DATA_PUMP_DIR') AS TASK_ID FROM DUAL;`
	前述命令输出一个任务 ID。要通过查看任务 ID 中的数据来查看下载状态，请运行以下命令。
	`SELECT text FROM table(rdsadmin.rds_file_util.read_te`
		AWS 系统管理员

Task	描述	所需技能
	xt_file('BDUMP','d btask-<task_id>.lo g'));	
	要查看 DATA_PUMP_DIR 目录中的文件，请运行以下命令。 SELECT filename, type,filesize/1024 /1024 size_megs ,to_char(mtime,'DD -MON-YY HH24:MI:SS') timestamp FROM TABLE(rdsadmin.rds _file_util.listdir (p_directory => upper('DATA_PUMP_D IR'))) order by 4;	

将转储文件导入到目标数据库中

Task	描述	所需技能
将架构和数据恢复到 Amazon RDS。	要将转储文件导入 sample_schema 数据库架构，请从 SQL Developer 或 SQL*Plus 中运行以下 SQL 命令。 DECLARE hdnl NUMBER; BEGIN hdnl := DBMS_DATA PUMP.OPEN(operation => 'IMPORT', job_mode	数据库管理员

Task	描述	所需技能
	=> 'SCHEMA', job_name=>null); DBMS_DATAPUMP.ADD_FILE(handle => hdnl, filename => 'sample.dmp', directory => 'DATA_PUMP_DIR', filetype => dbms_datapump.ku$_file_type_dump_file); DBMS_DATAPUMP.ADD_FILE(handle => hdnl, filename => 'import.log', directory => 'DATA_PUMP_DIR', filetype => dbms_datapump.ku$_file_type_log_file); DBMS_DATAPUMP.METADATA_FILTER(hdnl,'SCHEMA_EXPR','IN (''SAMPLE_SCHEMA'')'); DBMS_DATAPUMP.START_JOB(hdnl); END; / 要查看导入后的日志文件，请运行以下命令。 SELECT text FROM table(rdsadmin.rds_file_util.read_te	

Task	描述	所需技能
	xt_file('DATA_PUMP _DIR','import.log'));	

从 DATA_PUMP_DIR 目录中移除转储文件

Task	描述	所需技能

列出并清理导出文件。	列出并删除 DATA_PUMP_DIR 目录中的导出文件，运行以下命令。
	`-- List the files`
	`SELECT filename,`
	`type,filesize/1024`
	`/1024 size_megs`
	`,to_char(mtime,'DD`
	`-MON-YY HH24:MI:S`
	`S') timestamp FROM`
	` TABLE(rdsadmin.rds`
	`_file_util.listdir`
	`(p_directory =>`
	` upper('DATA_PUMP_D`
	`IR'))) order by 4;`
	`-- Remove the files`
	`EXEC UTL_FILE.`
	`FREMOVE('DATA_PUMP`
	`_DIR','sample.dmp');`
	`EXEC UTL_FILE.FREMOVE('`
	`DATA_PUMP_DIR','im`
	`port.log');`
		AWS 系统管理员

相关资源

- Amazon S3 集成
- 创建数据库实例

- 将数据导入到 Amazon RDS 上的 Oracle
- Amazon S3 文档
- IAM 文档
- Amazon RDS 文档
- Oracle 数据泵文档
- Oracle SQL Developer

使用 pglogical 从 EC2 亚马逊上的 PostgreSQL 迁移到适用于 PostgreSQL 的亚马逊 RDS

由 Rajesh Madiwale (AWS) 编写

摘要

此模式概述了使用 PostgreSQL pglogical 扩展将 PostgreSQL 数据库（版本 9.5 及更高版本）从亚马逊弹性计算云（ EC2亚马逊）迁移到适用于 PostgreSQL 的亚马逊关系数据库服务（亚马逊 RDS）的步骤。Amazon RDS 现在支持 PostgreSQL 版本 10 的 pglogical 扩展。

先决条件和限制

先决条件

- 选择正确的 Amazon RDS 实例类型。有关更多信息，请参阅 Amazon RDS 实例类型。
- 确保 PostgreSQL 的源版本与目标版本相同。
- 在亚马逊上安装 p glogical 扩展并将其与 P ostgreSQL 集成。 EC2

产品版本

- Amazon RDS 上的 PostgreSQL 版本 10 及以上版本，Amazon RDS 支持这些功能（参见 AWS 文档中的PostgreSQL on Amazon RDS）。这种模式是通过在 Amazon RDS 上将 PostgreSQL 9.5 迁移至 PostgreSQL 版本 10 进行测试，但它也适用于最新版 PostgreSQL on Amazon RDS。

架构

数据迁移架构

工具

- pglogical 扩展
- PostgreSQL 原生实用程序：pg_dump 和 pg_restore

操作说明

使用 pglogical 扩展迁移数据

Task	描述	所需技能
创建 Amazon RDS PostgreSQL 数据库实例。	在 Amazon RDS 中设置 PostgreSQL 数据库实例。有关说明，请参阅 Amazon RDS for PostgreSQL 文档。	数据库管理员
从源 PostgreSQL 数据库获取架构转储并将其恢复至目标 PostgreSQL 数据库中。	1. 使用带有 `-s`选项的 pg_dump 实用程序，从源数据库生成架构文件。 2. 使用带有 `-f`选项的 psql 实用程序，将架构加载至目标数据库中。	数据库管理员
启用逻辑解码。	在 Amazon RDS 数据库参数组，将 `rds.logical_replication` 静态参数设置为 1。有关说明，请参阅 Amazon RDS 文档。	数据库管理员
在源数据库和目标数据库创建 pglogical 扩展。	1. 在源 PostgreSQL 数据库上创建 `pglogical` 扩展： `psql -h <amazon-ec2-endpoint> -d target-dbname -U target-dbuser -c "create extension pglogical ;"` 2. 在目标 PostgreSQL 数据库上创建 `pglogical` 扩展：	数据库管理员

Task	描述	所需技能
	`psql -h <amazon-rds-endpoint> -d source-dbname -U source-dbuser -c "create extension pglogical ;"`	
在源 PostgreSQL 数据库上创建发布者。	若要创建发布者，请运行： `psql -d dbname -p 5432 <<EOF SELECT pglogical .create_node( node_name := 'provider1', dsn := 'host=<ec2-endpoint> port=5432 dbname=source-dbname user=source-dbuser' ); EOF`	数据库管理员
创建复制集，添加表格和序列。	要在源 PostgreSQL 数据库上创建复制集并将表和序列添加到复制集，请运行： `psql -d dbname -p 5432 <<EOF SELECT pglogical .replication_set_add_all_tables('default', '{public}'::text[],synchronize_data := true); EOF`	数据库管理员

Task	描述	所需技能
创建订阅用户。	若要对 PostgreSQL 数据库创建订阅用户，请运行： `psql -h <rds-endpoint> -d target-dbname -U target-dbuser <<EOF SELECT pglogical.create_node( node_name := 'subscriber1', dsn := 'host=<rds-endpoint> port=5432 dbname=target-dbname password=postgres user=target-dbuser' ); EOF`	数据库管理员
创建订阅。	若要对 PostgreSQL 数据库创建订阅用户，请运行： `psql -h <rds-endpoint> -d target -U postgres <<EOF SELECT pglogical.create_subscription( subscription_name := 'subscription1', replication_sets := array['default'], provider_dsn := 'host=<ec2-endpoint> port=5432 dbname=<source-dbname> password=<password> user=source-dbuser' ); EOF`	数据库管理员

验证数据

Task	描述	所需技能
检查源数据库和目标数据库。	检查源数据库和目标数据库，以确认数据迁移正在成功复制。您可以使用 `select count(1)` 对源表和目标表执行基本验证。	数据库管理员

相关资源

- Amazon RDS
- Amazon RDS 上 PostgreSQL 的逻辑复制（Amazon RDS 文档）
- pglogical（存储库）GitHub
- pglogical（GitHub 存储库自述文件）的局限性
- 使用逻辑@@复制将 PostgreSQL 从本地或亚马逊迁移 EC2 到 Amazon RDS（AWS 数据库博客）

将本地 PostgreSQL 数据库迁移到 Aurora PostgreSQL

创建者：Baji Shaik (AWS) 和 Jitender Kumar (AWS)

摘要

Amazon Aurora PostgreSQL-Compatible Edition 将高端商业数据库的性能和可用性与开源数据库的简单性和成本效益结合在一起。Aurora 通过在同一 AWS 区域的三个可用区扩展存储来提供这些优势，并支持多达 15 个只读副本实例，用于横向扩展读取工作负载并在单个区域内提供高可用性。通过使用 Aurora 全局数据库，您最多可以在五个区域中复制 PostgreSQL 数据库，以便在区域出现故障时进行远程读取访问和灾难恢复。此模式描述了将本地 PostgreSQL 源数据库迁移到 Aurora PostgreSQL-Compatible 数据库的步骤。该模式包括两个迁移选项：使用 AWS 数据迁移服务（AWS DMS）或使用原生 PostgreSQL 工具（例如 pg_dump、pg_restore 和 psql）或第三方工具。

此模式中描述的步骤也适用于亚马逊关系数据库服务 (Amazon RDS) 和亚马逊弹性计算云 (Amazon) 实例上的目标 PostgreSQL 数据库。 EC2

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 本地数据中心中的 PostgreSQL 源数据库
- Aurora PostgreSQL-Compatible 数据库实例或 Amazon RDS for PostgreSQL 数据库实例

限制

- Amazon RDS for PostgreSQL 的数据库大小限制为 64 TB，Aurora PostgreSQL-Compatible 的数据库大小限制为 128 TB。
- 如果您使用的是 AWS DMS 迁移选项，请查看使用 PostgreSQL 数据库作为源的 AWS DMS 限制。

产品版本

- 有关 Amazon RDS 对 PostgreSQL 主要版本和次要版本的支持，请参阅 Amazon RDS 文档中的 Amazon RDS for PostgreSQL 更新。
- 有关 Aurora 中 PostgreSQL 的支持，请参阅 Aurora 文档中的 Amazon Aurora PostgreSQL 更新。
- 如果您使用的是 AWS DMS 迁移选项，请参阅 AWS DMS 文档中支持的 PostgreSQL 版本。

架构

源技术堆栈

- 本地 PostgreSQL 数据库

目标技术堆栈

- Aurora PostgreSQL-Compatible 数据库实例

源架构

目标架构

数据迁移架构

使用 AWS DMS

使用原生 PostgreSQL 工具

工具

- AWS Database Migration Service（AWS DMS）可帮助您将数据存储迁移到 Amazon Web Services Cloud，或者在云和本地配置的组合之间迁移。该服务支持不同的源数据库和目标数据库。有关如何验证 AWS DMS 支持的 PostgreSQL 源数据库和目标数据库版本和版本的信息，请参阅使用 PostgreSQL 数据库作为 AWS DMS 源。建议使用最新版本的 AWS DMS，以获得最全面的版本和功能支持。
- PostgreSQL 原生工具包括 pg_dump、pg_restore 和psql。

操作说明

分析迁移

Task	描述	所需技能
验证源数据库和目标数据库的版本。	如果您使用的是 AWS DMS。请确保您使用的是 PostgreSQL 的受支持版本。	数据库管理员
确定存储类型和容量。	1. 计算为源数据库实例分配的存储空间。 2. 收集源数据库实例的历史增长指标。 3. 预测目标数据库实例的未来增长预测。 4. 通过计算源数据库的读取和写入 IOPS 总数来分配存储空间。通用固态硬盘（gp2）卷为每 1 GB 的存储空间提供 3 个 IOPS。	数据库管理员、系统管理员
选择正确的实例类型、容量、存储功能和网络功能。	确定目标数据库实例的计算要求。查看可能需要额外注意的已知性能问题。要确定合适的实例类型，请考虑以下因素： • 源数据库实例的 CPU 使用率 • 源数据库实例的 IOPS（读写操作） • 源数据库实例上的内存占用 有关更多信息，请参阅 Amazon Aurora 用户指南中的 Aurora 数据库实例类。	数据库管理员、系统管理员

Task	描述	所需技能
确定源数据库和目标数据库的网络访问安全要求。	确定使应用程序能够与数据库通信的相应安全组。	数据库管理员、系统管理员
确定应用程序迁移策略。	• 根据应用程序的复杂性确定迁移割接策略。 • 确定应用程序的恢复时间目标（RTO）和恢复点目标（RPO），并相应地计划割接。	数据库管理员、应用程序所有者、系统管理员

配置基础设施

Task	描述	所需技能
创建 VPC。	为目标数据库实例创建新的虚拟私有云（VPC）。	系统管理员
创建安全组。	在 VPC 内创建安全组（如前一个操作说明中所述），以允许数据库实例的入站连接。	系统管理员
配置并启动 Aurora 数据库集群。	使用新的 VPC 和安全组创建目标数据库实例并启动该实例。	系统管理员

迁移数据 – 选项 1（使用 AWS DMS）

Task	描述	所需技能
完成迁移前步骤。	1. 清理源数据库中的数据。 2. 创建复制实例。 3. 创建源和目标端点。 4. 确定要迁移的可用表和对象的数量。	数据库管理员

Task	描述	所需技能
完成迁移步骤。	1. 删除目标数据库上的外键约束和触发器。 2. 删除目标数据库上的辅助索引。 3. 使用满载任务将数据从源数据库迁移到目标数据库。 4. 启用外键。 5. 如果您使用的是闪存快捷迁移，并且应用程序需要最少的停机时间，请启用更改数据捕获（CDC）以复制正在进行的更改 6. 启用触发器。 7. 更新序列。 8. 验证源数据和目标数据。	数据库管理员
验证数据。	要确保您的数据从源准确迁移到目标，请按照 AWS DMS 文档中的数据验证步骤进行操作。	数据库管理员

迁移数据-选项 2（使用 pg_dump 和 pg_restore）

Task	描述	所需技能
准备源数据库。	1. 创建一个用于存储 pg_dump 备份的目录（如果尚不存在）。 2. 创建一个有权对数据库对象运行 pg_dump 的迁移用户。	数据库管理员

Task	描述	所需技能
	3. 连接到 EC2 实例并运行 pg_dump 备份。 有关更多信息，请参阅 pg_dump 文档和 AWS DMS 文档中的演练。	
准备目标数据库。	1. 创建一个有权对数据库对象使用 pg_restore 的迁移用户。 2. 使用 pg_restore 导入数据库转储。 有关更多信息，请参阅 pg_restore 文档和 AWS DMS 文档中的演练。	数据库管理员
验证数据。	1. 比较源数据库和目标数据库之间的数据库对象计数。 2. 解决在对象计数之间发现的任何差异。	数据库管理员

迁移应用程序

Task	描述	所需技能
遵循应用程序迁移策略。	实施您在第一个操作说明中创建的应用程序迁移策略。	数据库管理员、应用程序所有者、系统管理员

割接至目标数据库

Task	描述	所需技能
将应用程序客户端切换至新基础设施。	1. 停止所有指向本地 PostgreSQL 数据库的应用程序服务和客户端连接。 2. 运行 AWS DMS 任务。 3. 如果需要，可以设置回滚任务（从 Aurora PostgreSQL-Compatible 到本地 PostgreSQL 数据库的反向 CDC）。 4. 验证数据。 5. 通过将 Amazon Route 53 配置为新 Aurora PostgreSQL-Compatible 数据库实例，在新目标上启动应用程序服务。 6. 在兼容 Aurora PostgreSQL 的新数据库实例上添加亚马逊 CloudWatch 和 Performance Ins ights 监控功能。	数据库管理员、应用程序所有者、系统管理员
如果您需要回滚迁移。	1. 停止所有指向 Aurora PostgreSQL-Compatible 数据库的应用程序服务。 2. 使用您在上一个情节中创建的 AWS DMS 任务，将变更回滚到本地 PostgreSQL 源数据库。 3. 停止从本地 PostgreSQL 数据库运行到 Aurora PostgreSQL-Compatible 数据库的 AWS DMS 任务。	数据库管理员、应用程序所有者

Task	描述	所需技能
	4. 配置应用程序，使其指回到本地 PostgreSQL 源数据库。 5. 确认所有回滚部署均已完成。	

关闭项目

Task	描述	所需技能
关闭资源。	关闭临时 AWS 资源。	数据库管理员、系统管理员
验证文档。	查看和验证项目文档。	数据库管理员、应用程序所有者、系统管理员
收集指标。	收集与迁移时间、手动与工具占比、成本节约等相关的指标。	数据库管理员、应用程序所有者、系统管理员
关闭项目。	关闭项目并提供任何反馈。	数据库管理员、应用程序所有者、系统管理员

相关资源

参考

- AWS 数据库迁移服务
- VPCs 还有亚马逊 Aurora
- Amazon Aurora 定价
- 使用 PostgreSQL 数据库作为 AWS DMS 源
- 如何创建 AWS DMS 复制实例
- 如何使用 AWS DMS 创建源端点和目标端点

其他资源

- AWS DMS 入门
- 数据迁移 step-by-step演练
- Amazon Aurora 资源

将本地微软 SQL Server 数据库迁移到 EC2 运行 Linux 的亚马逊上的微软 SQL Server

由 Tirumala Dasari (AWS) 创建

摘要

此模式描述了如何使用备份和还原实用程序从在微软 Windows 上运行的本地 Microsoft SQL Server 数据库迁移到亚马逊弹性计算云 (亚马逊 EC2) Linux 实例上的微软 SQL Server。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 带微软 SQL Server 的亚马逊 EC2 Linux AMI（亚马逊机器映像）
- 在本地 Windows 和 Linux EC2 实例上的微软 SQL Server 之间的 AWS Direct Connect

架构

源技术堆栈

- 本地 Microsoft SQL Server 数据库

目标技术堆栈

- 带有微软 SQL Server 数据库的 Linux EC2 实例

数据库迁移架构

工具

- WinSCP - 此工具使 Windows 用户能够轻松地与 Linux 用户共享文件。
- Sqlcmd - 此命令行实用程序允许您将 T-SQL 语句或批处理提交到 SQL Server 的本地和远程实例。该实用程序对于重复的数据库任务（例如批处理或单元测试）非常有用。

操作说明

使用 SQL Server 准备 EC2 Linux 实例

Task	描述	所需技能
选择提供 Linux 操作系统并包括 Microsoft SQL Server 的 AMI。		系统管理员
配置 AMI 以创建 EC2 实例。		系统管理员
创建安全组的入站和出站规则。		系统管理员
为微软 SQL Server 数据库配置 Linux EC2 实例。		数据库管理员
创建用户并提供与源数据库中一样的权限。		应用程序所有者、数据库管理员
在 Linux 实例上安装 SQL Server 工具和 sqlcmd 实用程序。 EC2		数据库管理员

备份数据库并将备份文件移至 Linux EC2 实例

Task	描述	所需技能
备份本地 SQL Server 数据库。		数据库管理员
在 Microsoft SQL Server 上安装 WinSCP。		数据库管理员
将备份文件移至运行微软 SQL Server 的 Linux EC2 实例。		数据库管理员

在运行 SQL Server 的 Linux EC2 实例上恢复数据库

Task	描述	所需技能
使用 sqlcmd 实用程序从数据库备份文件恢复数据库。		数据库管理员
验证数据库对象和数据。		开发人员、测试工程师

在 Linux EC2 实例上从 Windows SQL 服务器切换到 Windows SQL Server

Task	描述	所需技能
验证数据库对象和数据。		开发人员、测试工程师
从本地微软 SQL Server 数据库切换到运行微软 SQL Server 的 Linux EC2 实例。		数据库管理员

相关的资源

- 如何在亚马逊 Linux 和 Ubuntu 上配置 SQL Server 2017 AMIs
- 在 Linux 实例上安装 SQL 工具
- 在 Linux EC2 实例上从本地微软 SQL Server 数据库备份和恢复到微软 SQL Server

使用链接服务器将本地 Microsoft SQL Server 数据库迁移至 Amazon RDS for SQL Server

由 Kevin Yung（AWS）、Vishal Singh（AWS）和 Viqash Adwani（AWS）创作

摘要

链接服务器使 Microsoft SQL Server 能够在数据库服务器的其他实例上运行 SQL 语句。此模式描述了如何将本地 Microsoft SQL Server 数据库迁移到 Amazon Relational Database Service (Amazon RDS) for Microsoft SQL Server，以实现更低的成本和更高的可用性。目前，Amazon RDS for Microsoft SQL Server 不支持 Amazon Virtual Private Cloud (Amazon VPC) 网络外部的连接。

您可使用此模式来实现以下目标：

- 在不中断链接服务器功能的情况下将 Microsoft SQL Server 迁移至 Amazon RDS for Microsoft SQL Server。
- 在不同的批次中确定优先级并迁移链接的 Microsoft SQL Server。

先决条件和限制

先决条件

- 检查Amazon RDS 上的 Microsoft SQL Server 是否支持需要的功能。
- 确保您可使用具有默认排序规则的 Amazon RDS for Microsoft SQL Server，或者在数据库级别上设置排序规则。

架构

源技术堆栈

- 本地数据库 (Microsoft SQL Server)

目标技术堆栈

- Amazon RDS for SQL Server

源状态架构

目标状态架构

在目标状态下，您可使用链接服务器将 Microsoft SQL Server 迁移至 Amazon RDS for Microsoft SQL Server。此架构使用网络负载均衡器将流量从 Amazon RDS for Microsoft SQL Server 代理到运行 Microsoft SQL Server 的本地服务器。下图显示了网络负载均衡器反向代理功能。

工具

- AWS CloudFormation
- 网络负载均衡器
- 多个可用区（多AZs）中的 Amazon RDS for SQL Server
- AWS Database Migration Service (AWS DMS)

操作说明

创建登录区 VPC

Task	描述	所需技能
创建 CIDR 分配。		AWS SysAdmin
创建虚拟私有云（VPC）。		AWS SysAdmin
创建 VPC 子网。		AWS SysAdmin
创建子网访问控制列表 (ACLs)。		AWS SysAdmin
创建子网路由表。		AWS SysAdmin

Task	描述	所需技能
通过 AWS Direct Connect 或 AWS 虚拟专用网络（VPN）创建连接。		AWS SysAdmin

将数据库迁移到 Amazon RDS

Task	描述	所需技能
创建 Amazon RDS for Microsoft SQL Server DB 实例。		AWS SysAdmin
创建 AWS DMS 复制实例。		AWS SysAdmin
在 AWS DMS 中创建源和目标数据库端点。		AWS SysAdmin
创建迁移任务，并在满负荷后将连续复制设置为“打开”。		AWS SysAdmin
请求更改防火墙，以允许 Amazon RDS for Microsoft SQL Server 访问本地 SQL Server 数据库。		AWS SysAdmin
创建网络负载均衡器。		AWS SysAdmin
创建针对数据中心中的数据库服务器的目标组	我们建议您在目标设置中使用主机名来合并数据中心 (DC) 故障转移事件。	AWS SysAdmin
运行 SQL 语句执行链接服务器设置。	使用 Microsoft SQL 管理工具针对 Amazon RDS for Microsoft SQL Server 数据库实例运行用于添加链接服务器的 SQL 语句。在 SQL 语	AWS SysAdmin

Task	描述	所需技能
	句中，设置 @datasrc 以使用网络负载均衡器主机名。使用 Microsoft SQL 管理工具针对 Amazon RDS for Microsoft SQL Server 数据库实例添加链接服务器登录凭证。	
测试和验证 SQL Server 函数。		AWS SysAdmin
创建割接。		AWS SysAdmin

相关资源

- Amazon RDS 上 Microsoft SQL Server 的常见管理任务
- Microsoft SQL Server 的排序规则和字符集
- 网络负载均衡器文档
- 使用 Amazon RDS for Microsoft SQL Server 实施链接服务器（博客文章）

使用本机备份和还原将本地 Microsoft SQL Server 数据库迁移到 Amazon RDS for SQL Server。

由 Tirumala Dasari (AWS)、David Queiroz(AWS) 和 Vishal Singh(AWS) 编写

摘要

此模式描述如何将本地 Microsoft SQL Server 数据库迁移到 SQL Server 数据库实例的 Amazon Relational Database Service (Amazon RDS)（同质迁移）。迁移进程基于本机 SQL Server 备份和还原方法。它使用 SQL Server Management Studio (SSMS) 创建数据库备份文件，并使用 Amazon Simple Storage Service (Amazon S3) 存储桶来存储备份文件，然后再将其恢复到 Amazon RDS for SQL Server 中。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- AWS Identity and Access Management(IAM) 角色策略，用于访问 S3 存储桶和 Amazon RDS for SQL Server 数据库实例。

限制

- 此模式中描述的进程仅迁移数据库。SQL 登录名或数据库用户（包括任何 SQL Server 代理作业）不会迁移，因为它们需要额外的步骤。

产品版本

- SQL Server 2012-2017。有关当前支持的版本和功能的最新列表，请参阅 AWS 文档中的 Microsoft SQL Server on Amazon RDS。

架构

源技术堆栈

- 本地 Microsoft SQL Server 数据库

目标技术堆栈

- Amazon RDS for SQL Server DB 实例

数据迁移架构

工具

- Microsoft SQL Server Management Studio (SSMS) 是一个用于管理 SQL Server 基础设施的集成环境。它提供了用户界面和一组工具，其中包含与 SQL Server 交互的丰富脚本编辑器。

操作说明

创建 Amazon RDS for SQL Server 数据库实例

Task	描述	所需技能
在 Amazon RDS for SQL Server 中选择 SQL Server 作为数据库引擎。		数据库管理员
选择 SQL Server Express Edition。		数据库管理员
指定数据库详细信息。	有关创建数据库实例的信息，请参阅 Amazon RDS 文档。	数据库管理员、应用程序所有者

从本地 SQL Server 数据库创建备份文件

Task	描述	所需技能
通过 SSMS 连接至本地 SQL Server 数据库。		数据库管理员
创建数据库的备份。	有关说明，请参阅 SSMS 文档。	数据库管理员、应用程序所有者

将备份文件上载到 Amazon S3

Task	描述	所需技能
在 Amazon S3 中创建一个桶。	有关更多信息，请参阅 Amazon S3 文档。	数据库管理员
将备份文件上传到 S3 存储桶。	有关更多信息，请参阅 Amazon S3 文档。	SysOps 管理员

还原 Amazon RDS for SQL Server 数据库

Task	描述	所需技能
向 Amazon RDS 添加选项组。	1. 在 https://console.aws.amazon.com/rds/ 处打开 Amazon RDS 控制台。 2. 在导航窗格中，依次选择选项组和创建组。 3. 填写选项组信息，然后选择创建。 4. 将选项组添加 `SQLSERVER_BACKUP_RESTORE` 选项，然后选择添加选项。 有关更多信息，请参阅 Amazon RDS 文档。	SysOps 管理员
还原数据库。	1. 使用 SSMS 连接到 Amazon RDS for SQL Server 2. 调用 `msdb.dbo.rds_restore_database` 存储过程还原数据库。	数据库管理员

校验目标数据库

Task	描述	所需技能
验证对象和数据。	验证源数据库与 Amazon RDS for SQL Server 之间的对象和数据。 **Note** 此任务仅迁移数据库。不迁移登录名和作业。	应用程序所有者，数据库管理员

割接

Task	描述	所需技能
重定向应用程序流量。	验证后，将应用程序流量重定向至 Amazon RDS for SQL Server 数据库实例。	应用程序所有者，数据库管理员

相关资源

- Amazon S3 文档
- Amazon RDS for SQL Server 文档
- 适用于 Microsoft SQL Server 数据库引擎的选项

使用 AWS DMS 和 AWS SCT 将 Microsoft SQL Server 数据库迁移到 Aurora MySQL

由 Mark Szalkiewicz (AWS) 和 Pavan Pusuluri (AWS) 创作

摘要

此模式描述了如何将本地或亚马逊弹性计算云 (亚马逊 EC2) 实例上的微软 SQL Server 数据库迁移到亚马逊 Aurora MySQL。该模式使用 AWS Database Migration Service (AWS DMS) 和 AWS Schema Conversion Tool (AWS SCT) 进行数据迁移和架构转换。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 本地数据中心或 EC2 实例上的 Microsoft SQL Server 源数据库
- AWS SCT 连接器的 Java 数据库连接 (JDBC) 驱动程序，安装在本地计算机或安装了 AWS SCT 的 EC2 实例上

限制

- 数据库大小限制：64 TB

产品版本

- Microsoft SQL Server 2008、2008 R2、2012、2014、2016 和 2017（Enterprise、Standard、Workgroup 和 Developer 版）。AWS DMS 不支持 Web 和 Express 版本。有关受支持版本的最新列表，请参阅将 Microsoft SQL Server 数据库作为 AWS DMS 的源。建议使用最新版本的 AWS DMS，以获得最全面的版本和功能支持。有关 AWS SCT 支持的 Microsoft SQL Server 版本的信息，请参阅 AWS SCT 文档。
- MySQL 版本 5.5、5.6 和 5.7。有关受支持版本的最新列表，请参阅将 MySQL 兼容数据库作为 AWS DMS 的目标。

架构

源技术堆栈

下列情况之一：

- 本地 Microsoft SQL Server 数据库
- EC2 实例上的微软 SQL Server 数据库

目标技术堆栈

- Aurora MySQL

数据迁移架构

- 从在 Amazon Web Services Cloud 中运行的 Microsoft SQL Server 数据库

- 从本地数据中心运行的 Microsoft SQL Server 数据库

工具

- AWS DMS - AWS Data Migration Service (AWS DMS) 可帮助您将数据迁移到广泛使用的商业和开源数据库，包括 Oracle、SQL Server、MySQL 和 PostgreSQL。您可以使用 AWS DMS 将数据迁移到 Amazon Web Services Cloud、本地实例之间（通过 Amazon Web Services Cloud 设置）或云和本地设置的组合之间。
- AWS SCT - AWS Schema Conversion Tool (AWS SCT) 通过自动将源数据库架构和大部分自定义代码转换为与目标数据库兼容的格式，使异构数据库迁移变得轻松。

操作说明

准备迁移

Task	描述	所需技能
验证源数据库和目标数据库版本和引擎。		数据库管理员
为源数据库和目标数据库创建出站安全组。		SysAdmin

Task	描述	所需技能
如果需要，为 AWS SCT 创建和配置 EC2 实例。		数据库管理员
下载最新版本的 AWS SCT 和相关驱动程序。		数据库管理员
在源数据库中添加和验证必备用户和授权。		数据库管理员
为工作负载创建 AWS SCT 项目并连接到源数据库。		数据库管理员
生成评测报告并评测可行性。		数据库管理员

准备目标数据库

Task	描述	所需技能
使用 Amazon Aurora 作为数据库引擎创建目标 Amazon RDS 数据库实例。		数据库管理员
从源中提取用户、角色和授权的列表。		数据库管理员
将现有数据库用户映射到新的数据库用户。		应用程序所有者
在目标数据库中创建用户。		数据库管理员
将上一步的角色应用至目标数据库。		数据库管理员
查看源数据库中的数据库选项、参数、网络文件和数据库		数据库管理员

Task	描述	所需技能
链接，然后评估它们对目标数据库的适用性。		
将任何相关设置应用于目标。		数据库管理员

传输对象

Task	描述	所需技能
为目标数据库配置 AWS SCT 连接。		数据库管理员
使用 AWS SCT 转换架构。	AWS SCT 会自动将源数据库架构和大多数自定义代码转换为与目标数据库兼容的格式。该工具无法自动转换的任何代码都会被清楚地标记出来，以便您可以自己转换。	数据库管理员
查看生成的 SQL 报告并保存所有错误和警告。		数据库管理员
将自动架构更改应用于目标或将其另存为 .sql 文件。		数据库管理员
验证 AWS SCT 是否在目标创建了对象。		数据库管理员
手动重写、拒绝或重新设计任何无法自动转换的项目。		数据库管理员
应用生成的角色和用户授权并查看任何例外情况。		数据库管理员

迁移数据

Task	描述	所需技能
确定迁移方法。		数据库管理员
从 AWS DMS 控制台创建复制实例。	有关使用 AWS DMS 的详细信息，请参阅“相关资源”部分中的链接。	数据库管理员
创建源端点和目标端点。		数据库管理员
创建复制任务。		数据库管理员
启动复制任务并监控日志。		数据库管理员

迁移应用程序

Task	描述	所需技能
使用 AWS SCT 分析和转换应用程序代码中的 SQL 项。	当您将数据库架构从一个引擎转换到另一个引擎时，还需要更新应用程序中的 SQL 代码，以便与新数据库引擎 (而非旧引擎) 进行交互。您可以查看、分析、编辑和保存转换后的 SQL 代码。有关使用 AWS SCT 的详细信息，请参阅“相关资源”部分中的链接。	应用程序所有者
在 AWS 上创建新应用程序服务器。		应用程序所有者
将应用程序代码迁移至新服务器。		应用程序所有者
为目标数据库和驱动程序配置应用程序服务器。		应用程序所有者

Task	描述	所需技能
修复应用程序中特定于源数据库引擎的任何代码。		应用程序所有者
优化目标引擎的应用程序代码。		应用程序所有者

割接

Task	描述	所需技能
将任何新用户、授权和代码更改应用于目标。		数据库管理员
锁定应用程序以进行任何更改。		应用程序所有者
验证所有更改是否都已传播到目标数据库。		数据库管理员
将新的应用程序服务器指向目标数据库。		应用程序所有者
重新检查所有内容。		应用程序所有者
上线。		应用程序所有者

关闭项目

Task	描述	所需技能
关闭临时的 AWS 资源（AWS DMS 复制 EC2 实例和用于 AWS SCT 的实例）。		数据库管理员、应用程序所有者

Task	描述	所需技能
更新内部团队对于 AWS DMS 流程的反馈。		数据库管理员、应用程序所有者
如有必要，请修改 AWS DMS 流程并改进模板。		数据库管理员、应用程序所有者
审查和验证项目文档。		数据库管理员、应用程序所有者
收集有关迁移时间、手动成本与工具成本节省百分比等指标。		数据库管理员、应用程序所有者
关闭项目并提供任何反馈。		数据库管理员、应用程序所有者

相关资源

参考

- AWS DMS 用户指南
- AWS SCT 用户指南
- Amazon Aurora 定价

教程和视频

- AWS Database Migration Service 入门
- AWS Schema Conversion Tool 入门
- Amazon RDS 资源
- AWS DMS 演练 Step-by-Step

使用原生工具将本地 MariaDB 数据库迁移至 Amazon RDS for MariaDB

由 Shyam Sunder Rakhecha (AWS) 创作

摘要

此模式为使用原生工具将本地 MariaDB 数据库迁移至 Amazon Relational Database Service (Amazon RDS) for MariaDB 提供了指导。如果安装了 MySQL 工具，则可使用 mysql 和 mysqldump。如果安装了 Mariadb 工具，则可使用 mariadb 和 mariadb-dump。MySQL 和 MariaDB 工具的源相同，但二者在 MariaDB 版本 10.6 及更高版本中存在细微差别。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 本地数据中心的 MariaDB 源数据库

限制

- 数据库大小限制：64 TB

产品版本

- MariaDB 版本 10.0-10.6（有关支持版本的最新列表，请参阅 AWS 文档中的 Amazon RDS 上的 MariaDB）

架构

源技术堆栈

- 本地数据中心的 MariaDB 数据库

目标技术堆栈

- Amazon RDS for MariaDB 数据库实例

目标架构

数据迁移架构

工具

- 原生 MySQL 工具：mysql 和 mysqldump
- 原生 MariaDB 工具：mariadb 和 mariadb-dump

操作说明

计划迁移

Task	描述	所需技能
验证源数据库和目标数据库版本和引擎。		数据库管理员
确定目标服务器实例的硬件要求。		数据库管理员、系统管理员
识别存储需求（存储类型和容量）。		数据库管理员、系统管理员
根据容量、存储功能和网络功能选择正确的实例类型。		数据库管理员、系统管理员
确定源数据库和目标数据库的网络访问安全要求。		数据库管理员、系统管理员
确定应用程序迁移策略。		数据库管理员、应用程序所有者、系统管理员

配置基础设施

Task	描述	所需技能
创建虚拟私有云（VPC）。		系统管理员
创建安全组。		系统管理员
配置并启动运行 MariaDB 的 Amazon RDS 数据库实例。		系统管理员

迁移数据

Task	描述	所需技能
使用原生工具迁移数据库对象和数据。	在源数据库中，使用 mysqldump 或 mariadb-dump 创建包含数据库对象和数据的输出文件。在目标数据库中，使用 mysql 或 mariadb 恢复数据。	数据库管理员
验证数据。	检查源数据库和目标数据库，以确认数据迁移已成功。	数据库管理员

迁移应用程序

Task	描述	所需技能
遵循应用程序迁移策略。		数据库管理员、应用程序所有者、系统管理员

割接

Task	描述	所需技能
将应用程序客户端切换至新基础设施。		数据库管理员、应用程序所有者、系统管理员

关闭项目

Task	描述	所需技能
关闭临时 AWS 资源。		系统管理员
查看和验证项目文档。		数据库管理员、应用程序所有者、系统管理员
收集与迁移时间、工具成本节约等相关的指标。		数据库管理员、应用程序所有者、系统管理员
关闭项目并提供反馈。		数据库管理员、应用程序所有者、系统管理员

相关资源

Amazon RDS 参考

- Amazon RDS for MariaDB
- 亚马逊 Virtual Private Cloud VPCs 和亚马逊 RDS
- Amazon RDS 多可用区部署
- Amazon RDS 定价

MySQL 和 MariaDB 参考

- mariadb-dump/mysqldump
- mysql 命令行客户端

教程和视频

- Amazon RDS 入门

将本地 MySQL 数据库迁移至 Aurora MySQL

由伊戈尔·奥布拉多维奇 (AWS) 创作

摘要

此模式说明了如何将本地 MySQL 源数据库迁移到兼容 Amazon Aurora MySQL 的版本。它描述了两个迁移选项：使用 AWS Database Migration Service (AWS DMS) 或使用原生 MySQL 工具，例如 mysql dbcopy 和 mysqld ump。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 本地数据中心的源 MySQL 数据库

限制

- 数据库大小限制：128 TB

产品版本

- MySQL 版本 8.0（Aurora MySQL 版本 3）在标准支持下可用。
- MySQL 版本 5.7（Aurora MySQL 版本 2）在扩展支持下提供，但需要额外付费。

有关支持版本的最新列表，请参阅 AWS 文档中的 Amazon Aurora 版本。如果您正在使用 AWS DMS，另请参阅使用与 M ySQL 兼容的数据库作为目标适用于支持的 AWS DMS MySQL 版本。 AWS DMS

架构

源技术堆栈

- 本地数据库。

目标技术堆栈

- Amazon Aurora MySQL 兼容版

目标架构

Aurora 数据存储在集群卷中，集群卷是一个使用固态硬盘 (SSDs) 的单个虚拟卷。集群卷由单个 AWS 区域中三个可用区域的数据副本组成。由于数据会自动跨可用区复制，因此非常耐用，数据丢失的可能性更小。

Aurora 会自动将您的数据库卷划分为分布在多个磁盘上的 10 GB 分段。数据库卷的每个 10 GB 区块都通过六种方式在三个可用区中进行复制。下图说明了 Aurora 数据库集群中的集群卷、写入器数据库实例和读取器数据库实例之间的关系，以及计算容量和存储的分离。有关此架构的更多信息，请参阅 Aurora 文档和常见问题解答。

数据迁移架构

使用 AWS DMS：

下图说明了使用本地 MySQL 数据库迁移到与 Aurora MySQL 兼容的 AWS Cloud集群的情况。 AWS DMS

使用原生 MySQL 工具：

下图说明了使用 mysqldbcopy 和 mysqldump 等原生 MySQL 工具将本地 MySQL 数据库迁移到中兼容 Aurora MySQL 的集群的情况。 AWS Cloud

工具

- AWS Database Migration Service (AWS DMS) 支持多个源数据库和目标数据库引擎。有关支持的 MySQL 源数据库和目标数据库的信息 AWS DMS，请参阅将兼容 MySQL 的数据库迁移到。 AWS 我们建议您使用最新版本的， AWS DMS 以获得最全面的版本和功能支持。
- mysqldbcopy 是一个 MySQL 实用程序，它可以在单台服务器上或在服务器之间复制 MySQL 数据库。
- my@@ sqldump 是一个 MySQL 实用工具，它从 MySQL 数据库创建转储文件用于备份或迁移。

操作说明

计划迁移

Task	描述	所需技能
验证版本和引擎。	验证源数据库和目标数据库的数据库版本和引擎。	数据库管理员
确定硬件要求。	确定目标服务器实例的硬件要求。	数据库管理员、系统管理员
确定存储要求。	识别存储需求（存储类型和容量）。	数据库管理员、系统管理员
选择实例类型。	根据您的计算、存储和网络要求选择正确的实例类型。	数据库管理员、系统管理员
确定网络访问安全要求。	确定源数据库和目标数据库的网络访问安全要求。	数据库管理员、系统管理员
确定策略。	确定应用程序迁移策略。	数据库管理员、应用程序所有者、系统管理员

配置基础设施

Task	描述	所需技能
创建虚拟私有云（VPC）。	有关说明，请参阅亚马逊虚拟私有云（亚马逊 VPC）文档中的创建 VPC。	系统管理员
创建安全组。	有关说明，请参阅 Amazon VPC 文档中的为您的 VPC 创建安全组。	系统管理员

Task	描述	所需技能
在中配置并启动与 Aurora MySQL 兼容的数据库集群。 AWS 账户	有关说明，请参阅 Aurora 文档中的创建 Amazon Aurora 数据库集群。	系统管理员

迁移数据 - 选项 1

Task	描述	所需技能
使用原生 MySQL 工具或第三方工具迁移数据库对象和数据。	有关说明，请参阅 mysql dbcopy 和 mysqldump 等 Mysqld ump 等工具的文档。	数据库管理员

迁移数据 - 选项 2

Task	描述	所需技能
使用迁移数据 AWS DMS。	有关说明，请参阅文档中的使用与 MySQL 兼容的数据库作为源和使用 MySQL 兼容的数据库作为目标。 AWS DMS	数据库管理员

迁移应用程序

Task	描述	所需技能
遵循策略。	遵循应用程序迁移策略。	数据库管理员、应用程序所有者、系统管理员

割接

Task	描述	所需技能
切换应用程序客户端。	切换应用程序客户端以连接到新的 Aurora 集群终端节点。	数据库管理员、应用程序所有者、系统管理员

关闭项目

Task	描述	所需技能
关闭资源。	关闭临时 AWS 资源。	数据库管理员、系统管理员
查看文档。	查看和验证项目文档。	数据库管理员、应用程序所有者、系统管理员
收集指标。	收集有关迁移时间、手动步骤与工具使用量的百分比、成本节省等方面的指标。	数据库管理员、应用程序所有者、系统管理员
完成迁移项目。	关闭项目并提供反馈。	应用程序所有者，数据库管理员，系统管理员

相关资源

参考

- 将数据迁移到 Amazon Aurora MySQL 数据库集群
- AWS DMS 网站
- AWS DMS 文档
- Amazon Aurora 定价
- 创建并连接到 Aurora MySQL 数据库集群
- 亚马逊 VPC 和亚马逊 RDS
- Amazon Aurora 文档

教程和视频

- 入门 AWS DMS
- Amazon Aurora 入门

使用 Percona、A XtraBackup mazon EFS 和 Amazon S3 将本地 MySQL 数据库迁移到 Aurora MySQL

创建者：Rohan Jamadagni (AWS)、sajith menon (AWS) 和 Udayasimha Theepireddy (AWS)

摘要

此模式描述了如何使用 Percon XtraBackup a 将大型本地 MySQL 数据库高效地迁移到 Amazon Aurora MySQL。Percona XtraBackup 是一款适用于基于 MySQL 的服务器的开源、非阻塞备份实用程序。该模式显示了如何使用 Amazon Elastic File System（Amazon EFS）来缩短将备份上传到 Amazon Simple Storage Service（Amazon S3）的时间，以及将备份恢复到 Amazon Aurora MySQL 的时间。该模式还详细介绍了如何进行 Percona 增量备份，以最大限度地减少要应用于目标 Aurora MySQL 数据库的二进制日志数量。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 创建 AWS Identity and Access Management（IAM）角色和策略的权限
- 本地 MySQL 数据库与 AWS 上的虚拟私有云（VPC）之间的网络连接

限制

- 源服务器必须是基于 Linux 的系统，可以安装网络文件系统 (NFS) 客户端 (nfs-utils/nfs-common)。
- 用于上传备份文件的 S3 存储桶仅支持服务器端加密(SSE-S3/SSE-KMS)。
- Amazon S3 将备份文件大小限制为 5 TB。如果备份文件超过 5 TB，则可以将其分为多个较小的文件。
- 上传到 S3 存储桶的源文件个数不能超过 100 万个。
- 该模式仅支持 Percona XtraBackup 完整备份和增量备份。它不支持使用 `--tables`、`--tables-exclude`、`--tables-file`、`--databases`、`--databases-exclude` 或 `--databases-file` 的部分备份。
- Aurora 不会从 MySQL 源数据库中恢复用户、函数、存储过程或时区信息。

产品版本

- 源数据库必须是 MySQL 版本 5.5、5.6 或 5.7。

- 对于 MySQL 5.7，你必须使用 Percona XtraBackup 2.4。
- 对于 MySQL 5.6 和 5.6，你必须使用 Percona XtraBackup 2.3 或 2.4。

架构

源技术堆栈

- 基于 Linux 的操作系统
- MySQL 服务器
- Percona XtraBackup

目标技术堆栈

- Amazon Aurora
- Amazon S3
- Amazon EFS

目标架构

工具

Amazon Web Services

- Amazon Aurora 是一款完全托管型关系数据库引擎，可以让您通过简单且经济高效的方式设置、操作和扩展 MySQL 部署。Aurora MySQL 是 MySQL 的插拔式替换。
- Amazon Elastic File System（Amazon EFS）可帮助您在 Amazon Web Services Cloud 中创建和配置共享文件系统。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

其他工具

- Percona XtraBackup 是一个开源实用程序，可在不中断或阻塞数据库的情况下执行 MySQL 数据库的流式备份、压缩备份和增量备份。

操作说明

创建 Amazon EFS 文件系统

Task	描述	所需技能
创建一个与 Amazon EFS 挂载目标关联的安全组。	在 VPC 中创建一个安全组，该组配有通过 AWS Transit Gateway 连接到本地数据库的 VPN。有关本文和其他文章中描述的命令和步骤的更多信息，请参阅此模式末尾的“相关资源”部分中的链接。	AWS DevOps /数据库管理员
编辑安全组规则。	添加入站规则，使用类型 NFS、端口 2049 和本地数据库服务器的 IP 范围作为源。默认情况下，出站规则允许所有流量离开。如果不是这种情况，请添加出站规则以打开 NFS 端口的连接。再添加两个入站规则：端口 2049（来源：同一安全组的安全组 ID）和端口 22（来源：您将从中连接到 EC2 实例的 IP 范围）。	AWS DevOps /数据库管理员
创建文件系统。	在挂载目标中，使用您在上一个情节中创建的 VPC 和安全组。根据本地数据库的 I/O 要求选择吞吐量模式和性能。或者，启用静态加密。	AWS DevOps /数据库管理员

挂载文件系统

Task	描述	所需技能
创建要与实例关联的 IAM EC2 实例配置文件角色。	创建有权上传和访问 Amazon S3 中的对象的 IAM 角色。选择将备份存储为策略资源的 S3 存储桶。	AWS DevOps
创建实 EC2 例。	启动基于 Linux 的 EC2 实例，并附加您在上一步中创建的 IAM 实例配置文件角色和之前创建的安全组。	AWS DevOps
安装 NFS 客户端。	在本地数据库服务器和 EC2 实例上安装 NFS 客户端。有关安装说明，请参阅“其他信息”部分。	DevOps
挂载 Amazon EFS 文件系统。	在本地和 EC2 实例上安装 Amazon EFS 文件系统。在每台服务器上，创建一个用于存储备份的目录，然后使用挂载目标端点挂载文件系统。有关示例，请参阅“其他信息”部分。	DevOps

生成 MySQL 源数据库的备份

Task	描述	所需技能
安装 Percona XtraBackup。	在本地数据库服务器上安装 Percona XtraBackup 2.3 或 2.4（取决于您的 MySQL 数据库的版本）。有关安装链接，请查看“相关资源”部分。	数据库管理员

Task	描述	所需技能
计数源数据库中的架构和表。	收集并记下 MySQL 源数据库中架构和对象的数量。迁移后，您将使用这些计数来验证 Aurora MySQL 数据库。	数据库管理员
（可选）记下源数据库中最新的二进制日志序列。	如果要在源数据库和 Aurora MySQL 之间建立二进制日志复制以最大限度地减少停机时间，请执行此步骤。必须启用日志箱，且 server_id 必须是唯一的。在启动备份之前，请记下源数据库中当前的二进制日志序列。如果您计划仅使用完整备份，请在完整备份之前执行此步骤。如果您计划在完整备份后进行增量备份，请先执行此步骤，然后再在 Aurora MySQL 数据库实例上恢复最终增量备份。	数据库管理员
创建 MySQL 源数据库的完整备份。	使用 Percon XtraBackup a 对 MySQL 源数据库进行完整备份。有关完整备份和增量备份的命令示例，请参阅“其他信息”部分。	数据库管理员

Task	描述	所需技能
（可选）使用 Percon XtraBackup a 进行增量备份。	增量备份可用于减少将源数据库与 Aurora MySQL 同步所需二进制日志量。大型和事务密集型数据库可能会在备份期间生成大量二进制日志。通过进行增量备份并将其存储在共享的 Amazon EFS 文件系统上，您可以显著缩短备份和上传数据库的时间。有关详细信息，请参阅“其他信息”部分。继续进行增量备份，直到准备好开始向 Aurora 的迁移过程。	数据库管理员
准备备份。	在此步骤中，将事务日志应用于备份期间传输中的事务的备份。继续对每个增量备份应用事务日志 (--apply-log-only) 以合并除上次备份之外的备份。有关示例，请参阅“其他信息”部分。完成此步骤后，完整的合并备份将位于 ~/<efs_mount_name>/fullbackup。	数据库管理员
压缩并拆分最终合并的备份。	准备好最终的合并备份后，使用 tar、zip 和 split 命令创建较小的备份压缩文件。有关示例，请参阅“其他信息”部分。	数据库管理员

将备份恢复到 Aurora MySQL 数据库集群

Task	描述	所需技能
将备份文件上传到 Amazon S3。	存储备份文件的 Amazon EFS 文件系统同时安装在本地数据库和 EC2 实例上，因此备	AWS DevOps

Task	描述	所需技能
	份文件随时可供 EC2 实例使用。<bucket_name>使用安全外壳 (SSH) 连接到 EC2 实例，然后将压缩后的备份文件上传到新的或现有的 S3 存储桶；例如：aws s3 sync ~/<efs_mount_name>/fullbackup s3: ///fullbackup s3: //fullbackup。有关其他详细信息，请查看“相关资源”部分的链接。	
为 Aurora 创建服务角色以访问 Amazon S3。	创建具有信任“rds.amazonaws.com”的 IAM 角色和一个允许 Aurora 访问存储备份文件的 S3 存储桶的策略。所需的权限是 ListBucket GetObject、和 GetObjectVersion。	AWS DevOps
为 Aurora 创建联网配置。	创建一个集群数据库子网组，该子网组具有至少两个可用区和一个允许对源数据库进行出站连接的子网路由表配置。创建一个安全组，允许对本地数据库进行出站连接，并允许管理员连接到 Aurora 数据库集群。有关更多信息，请参阅“相关资源”部分中的链接。	AWS DevOps /数据库管理员

Task	描述	所需技能
将备份恢复到 Aurora MySQL 数据库集群。	通过上传到 Amazon S3 的备份恢复数据。指定源数据库的 MySQL 版本，提供上传备份文件的 S3 存储桶名称和文件夹路径前缀（例如，“其他信息”部分中的示例为“fullbackup”），并提供您为授权 Aurora 访问 Amazon S3 而创建的 IAM 角色。	AWS DevOps /数据库管理员
验证 Aurora MySQL 数据库。	根据您从源数据库获得的计数来验证已恢复的 Aurora 数据库集群中的架构和对象计数。	数据库管理员
设置二进制日志复制。	在创建恢复到 Aurora 数据库集群的最后一次备份之前，请使用您之前记下的二进制日志序列。在源数据库上创建复制用户，然后按照“其他信息”部分中的说明提供相应的权限，在 Aurora 上启用复制，并确认复制已同步。	AWS DevOps /数据库管理员

相关资源

创建 Amazon EFS 文件系统

- 创建安全组（Amazon VPC 文档）
- 传输网关 VPN 连接（Amazon VPC 文档）
- 使用 AWS Transit Gateway 扩展 VPN 吞吐量（联网和内容交付博客）
- 创建 Amazon EFS 文件系统（Amazon EFS 文档）
- 创建挂载目标（Amazon EFS 文档）
- 加密静态数据（Amazon EFS 文档）

挂载 EFS 文件系统

- 亚马逊@@ 的 IAM 角色 EC2（亚马逊 EC2 文档）
- 启动亚马逊 EC2 Linux 实例（亚马逊 EC2 文档）
- 安装 NFS 客户端（Amazon EFS 文档）
- 在本地客户端上安装 Amazon EFS 文件系统（Amazon EFS 文档）
- 安装 EFS 文件系统（Amazon EFS 文档）

创建 MySQL 源数据库的备份

- 安装 Percona XtraBackup 2.3（Per XtraBackup cona 文档）
- 安装 Percona XtraBackup 2.4（Per XtraBackup cona 文档）
- 设置复制主配置（MySQL 文档）
- 将数据从外部 MySQL 数据库迁移到 Aurora MySQL 数据库集群（Aurora 文档）
- 增量备份（Percona 文档 XtraBackup ）

将备份恢复到 Amazon Aurora MySQL

- 创建存储桶（Amazon S3 文档）
- 使用 SSH 连接到 Linux 实例（Amazon EC2 文档）
- 配置 AWS CLI（AWS CLI 文档）
- 同步命令（AWS CLI 命令参考）
- 创建 IAM policy 以访问 Amazon S3 资源（Aurora 文档）
- 数据库集群先决条件（Aurora 文档）
- 使用数据库子网组（Aurora 文档）
- 为私有数据库实例创建 VPC 安全组（Aurora 文档）
- 从 S3 存储桶恢复 Aurora MySQL 数据库集群（Aurora 文档）
- 使用 MySQL 或其他 Aurora 数据库集群设置复制（Aurora 文档）
- mysql.rds_set_external_master procedure（Amazon RDS 上的 MySQL 参考）
- mysql.rds_start_replication procedure（Amazon RDS 上的 MySQL SQL 参

其他参考资料

- 将数据从外部 MySQL 数据库迁移到 Aurora MySQL 数据库集群（Aurora 文档）
- MySQL 服务器下载（Oracle 网站）

教程和视频

- 使用 Amazon S3 将 MySQL 数据迁移到 Aurora MySQL 数据库集群（AWS Knowledge Center）
- Amazon EFS 设置和挂载（视频）

其他信息

安装 NFS 客户端

- 如果您使用的是 Red Hat 或类似的 Linux 操作系统，请使用以下命令：

```
$ sudo yum -y install nfs-utils
```

- 如果您使用的是 Ubuntu 或类似的 Linux 操作系统，请使用以下命令：

```
$ sudo apt-get -y install nfs-common
```

有关更多信息，请参阅 Amazon EFS 文档中的演练。

挂载 Amazon EFS 文件系统

使用命令：

```
mkdir ~/<efs_mount_name>
$ sudo mount -t nfs -o
 nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport mount-
target-IP:/ ~/<efs_mount_name>
```

有关更多信息，请参阅 Amazon EFS 文档中的演练和挂载 EFS 文件系统。

生成 MySQL 源数据库的备份

完整备份

使用如下命令，该命令获取备份，将其压缩，然后将其拆分为每个大小 1 GB 的小块：

```
xtrabackup --backup --user=dbuser --password=<password> --binlog-info=AUTO --stream=tar
 --target-dir=~/<efs_mount_name>/fullbackup | gzip - | split -d --bytes=1024MB - ~/
<efs_mount_name>/fullbackup/backup.tar.gz &
```

如果您计划在完整备份后进行后续增量备份，请不要压缩和拆分备份。使用类似于以下内容的命令：

```
xtrabackup --backup --user=dbuser --password=<password> --target-dir=~/
<efs_mount_name>/fullbackup/
```

增量备份

使用完整备份路径作为 `--incremental-basedir` 参数；例如：

```
xtrabackup --backup --user=dbuser --password=<password> --target-dir=~/
<efs_mount_name>/incremental/backupdate --incremental-basedir=~/<efs_mount_name>/
fullbackup
```

其中 basedir 是完整备份和 xtrabackup_checkpoints 文件的路径。

有关更多信息，请参阅 Aurora 文档中的将数据从外部 MySQL 数据库迁移到 Amazon Aurora MySQL 数据库集群。

准备备份

要准备完整备份，请执行以下操作：

```
xtrabackup --prepare --apply-log-only --target-dir=~/<efs_mount_name>/fullbackup
```

要准备增量备份，请执行以下操作：

```
xtrabackup --prepare --apply-log-only --target-dir=~/<efs_mount_name>/fullbackup --
incremental-dir=~/<efs_mount_name>/incremental/06062020
```

要准备最终备份，请执行以下操作：

```
xtrabackup --prepare --target-dir=~/<efs_mount_name>/fullbackup --incremental-dir=~/
<efs_mount_name>/incremental/06072020
```

有关更多信息，请参阅 Percona XtraBackup 文档中的增量备份。

压缩和拆分合并的备份

要将合并的备份压缩到 ~/<efs_mount_name>/fullbackup，请执行以下操作：

```
tar -zcvf <backupfilename.tar.gz> ~/<efs_mount_name>/fullbackup
```

要拆分备份，请执行以下操作：

```
split -d -b1024M --verbose  <backupfilename.tar.gz> <backupfilename.tar.gz>
```

设置二进制日志复制

要在源数据库上创建复制用户并提供相应的权限，请执行以下操作：

```
CREATE USER 'repl_user'@'' IDENTIFIED BY ''; GRANT REPLICATION CLIENT, REPLICATION
 SLAVE ON *.* TO 'repl_user'@'';
```

要通过连接到 Aurora 数据库集群在 Aurora 上启用复制，请在数据库集群参数组中启用二进制日志。设置 `binlog_format = mixed`（首选混合模式）。此更改要求您重新启动实例才能应用更新。

```
CALL mysql.rds_set_external_master ('sourcedbinstanceIP', sourcedbport, 'repl_user',
 '', 'binlog_file_name', binlog_file_position, 0); CALL mysql.rds_start_replication;
```

要确认复制是否处于同步状态，请执行以下操作：

```
SHOW Slave Status \G;
```

落后秒数主字段显示 Aurora 与本地数据库相比落后了多远。

使用 AWS App2Container 将本地 Java 应用程序迁移到 AWS

创建者：Dhananjay Karanjkar (AWS)

摘要

注意： CodeCommit AWS 不再向新客户开放。AWS 的现有客户 CodeCommit 可以继续照常使用该服务。了解更多

AWS App2Container（A2C）是一款命令行工具，无需更改代码即可帮助将虚拟机上运行的现有应用程序转换至容器。A2C 发现服务器上运行的应用程序、识别依赖项并生成相关构件，以便无缝部署到 Amazon Elastic Container Service (Amazon ECS) 和 Amazon Elastic Kubernetes Service (Amazon EKS)。

此模式提供了通过 Worker 计算机，使用 App2Container 将部署在应用程序服务器上的本地 Java 应用程序远程迁移到 AWS Fargate 或 Amazon EKS 的步骤。

Worker 计算机可用于以下用例：

- 运行 Java 应用程序的应用程序服务器上不允许安装 Docker，或者安装 Docker 不可用。
- 您必须管理部署在不同物理或虚拟服务器上的多项应用程序迁移。

此模式使用 AWS CodeCommit AWS CodePipeline、和 AWS CodeBuild。

先决条件和限制

先决条件

- Java 应用程序在 Linux 服务器上运行的应用程序服务器
- 装有 Linux 操作系统的 Worker 计算机
- 至少有 20 GB 可用磁盘空间的 Worker 计算机

限制

- 并非所有应用程序都受支持。有关更多信息，请参阅支持的应用程序（适用于 Linux）。

架构

源技术堆栈

- 在 Linux 服务器运行的 Java 应用程序

目标技术堆栈

- AWS CodeBuild
- AWS CodeCommit
- AWS CodeDeploy
- AWS CodePipeline
- Amazon Elastic Container Registry
- AWS Fargate

目标架构

工具

工具

- AWS App2Container – AWS App2Container（A2C）是一款命令行工具，可帮助您直接迁移在本地部署的数据中心或在虚拟机上运行的应用程序，以便它们在由 Amazon ECS 或 Amazon EKS 托管的容器中运行。
- AWS CodeBuild — AWS CodeBuild 是一项完全托管的云端构建服务。 CodeBuild 编译您的源代码，运行单元测试，并生成随时可以部署的工件。
- AWS CodeCommit — AWS CodeCommit 是一项由 Amazon Web Services 托管的版本控制服务，您可以使用它来私下存储和管理云中的资产（例如文档、源代码和二进制文件）。
- AWS CodePipeline — AWS CodePipeline 是一项持续交付服务，可用于对发布软件所需的步骤进行建模、可视化和自动化。
- Amazon ECS – Amazon Elastic Container Service (Amazon ECS) 是一项可扩展性高的快速容器管理服务，可用于运行、停止和管理集群上的容器。
- Amazon ECR – Amazon Elastic Container Registry (Amazon ECR) 是一项 AWS 托管容器映像注册表服务，它安全、可扩展且可靠。
- Amazon EKS – Amazon Elastic Kubernetes Service (Amazon EKS) 是一项托管服务，可让您在 AWS 上轻松运行 Kubernetes，而无需安装、操作和维护您自己的 Kubernetes 控制面板或节点。

- AWS Fargate — AWS Fargate 是一项可以与 Amazon ECS 一起使用的技术，无需管理服务器或亚马逊弹性计算云 (Amazon) 实例集群即可运行容器。 EC2使用 Fargate，您不必再预配置、配置或扩展虚拟机集群即可运行容器。

操作说明

设置凭证

Task	描述	所需技能
创建访问应用程序服务器的密钥。	要从 Worker 计算机远程访问应用程序服务器，请在 AWS Secrets Manager 中创建密钥。对于您的机密，您可以使用 SSH 私有密钥或证书和 SSH 私有密钥。有关更多信息，请参阅管理 AWS App2Container 的密钥。	DevOps，开发者

设置 Worker 计算机

Task	描述	所需技能
安装 tar 文件。	运行 `sudo yum install -y tar`。	DevOps，开发者
安装 Amazon CLI。	要安装 Amazon 命令行界面（AWS CLI），请运行 `curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"` 。 解压缩 `awscliv2.zip` 。	DevOps，开发者

Task	描述	所需技能
	运行 `sudo ./aws/install` 。	

安装 App2Container。	运行以下 命令：
	
	`curl -o AWSApp2Container-installer-linux.tar.gz https://app2container-release-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/AWSApp2Container-installer-linux.tar.gz`
	
	`sudo tar xvf AWSApp2Container-installer-linux.tar.gz`
	
	`sudo ./install.sh`
		DevOps，开发者

配置这些配置文件。	要配置 AWS 默认配置文件，请运行 `sudo aws configure` 。 要配置已命名的 AWS 默认配置文件，请运行 `sudo aws configure --profile <profile name>`。	DevOps，开发者

Task	描述	所需技能

安装 Docker。	运行以下命令。
	
	`sudo yum install -y docker`
	
	`sudo systemctl enable docker & sudo systemctl restart docker`	

Task	描述	所需技能
初始化 App2Container。	要初始化 App2Container，您需要以下信息： • `workspace` ：存储应用程序容器化构件。我们建议提供至少具有 20 GB 可用磁盘空间的目录路径。 • `awsProfile` ：在服务器上配置的 AWS 配置文件。这是将构件上传到 Amazon S3、运行 `containerize` 命令以及生成 AWS 构件以供在 Amazon ECS 或 Amazon EKS 上进行部署的必要条件。 • `s3Bucket`：提取和存储 AWS 构件。 • `metricsReportPermission` ：收集和存储报告的指标。 • `dockerContentTrust` ：对 Docker 映像进行签名。 运行 `sudo app2container init`。	DevOps，开发者

配置 Worker 计算机

Task	描述	所需技能
配置 Worker 计算机，以远程连接并在应用程序服务器上运行 App2Container 命令。	要配置 Worker 计算机，需要以下信息： • `Server FQDN`：应用程序服务器的完全限定域名。 • `Server IP address`：应用程序服务器 IP 地址。FQDN 或 IP 地址已足够。 • `SecretARN` ：密钥的 Amazon 资源名称（ARN），用于连接到应用程序服务器并存储在 Secrets Manager 中。 • `AuthMethod` ：`key` 或 `cert` 身份验证方法。 运行 `sudo app2container remote configure` 。	DevOps，开发者

发现、分析和提取 Worker 计算机上的应用程序

Task	描述	所需技能
探索本地 Java 应用程序。	要远程发现应用程序服务器上运行的所有应用程序，请运行以下命令。 `sudo app2container remote inventory --target <FQDN/IP of App server>`	开发者， DevOps

Task	描述	所需技能
	此命令生成 `inventory.json` 中已部署应用程序的列表。	
分析所发现的应用程序。	要通过使用在清单阶段获得的 application-id 来远程分析每个应用程序，请运行以下命令。 `sudo app2container remote analyze --application-id <java-app-id> --target <FQDN/IP of App Server>` 这将在工作区位置生成 `analysis.json` 文件。生成此文件后，您可根据需要更改容器化参数。	开发者，DevOps
提取分析的应用程序。	要为分析的应用程序生成应用程序存档，请远程运行以下命令，这将在工作区位置生成 tar 包。 `sudo app2container remote extract --application-id <application id> --target <FQDN/IP of App Server>` 提取的构件可以在本地 Worker 计算机上生成。	开发者，DevOps

在 Worker 计算机上对提取的构件进行容器化

Task	描述	所需技能
对提取的构件进行容器化。	运行以下命令，将上一步中提取的构件容器化。 `sudo app2container containerize --input-archive <tar bundle location on worker machine>`	开发者，DevOps
最终确定目标。	要最终确定目标，请打开 `containerize` 命令运行时创建的 `deployment.json` 。要将 AWS Fargate 指定为目标，请将 `createEcsArtifacts` 设置为 `true`。要将 Amazon EKS 设置为目标，请将 `createEksArtifacts` 设置为 true。	开发者，DevOps

生成和预调配 AWS 构件

Task	描述	所需技能
在 Worker 计算机上生成 AWS 部署项目。	要生成部署构件，请运行以下命令。 `sudo app2container generate app-deployment --application-id <application id>`	DevOps

Task	描述	所需技能
	这将在工作空间中生成 `ecs-master.yml` AWS CloudFormation 模板。	
预调配构件。	要进一步配置生成的项目，请运行以下命令部署 AWS CloudFormation 模板。 `aws cloudformation deploy --template-file <path to ecs-master.yml> --capabilities CAPABILITY_NAMED_IAM --stack-name <application id>-ECS`	DevOps
生成管线。	根据您的需求，修改在上一个情节中创建的 `pipeline.json` 。然后运行 `generate pipeline` 命令以生成管线部署构件。	DevOps

相关资源

- 什么是 App2Container？
- AWS App2Container 博客文章
- AWS CLI 配置基础知识
- Amazon ECS 的 Docker 基本信息
- Docker 命令

在 AWS 大规模迁移中迁移共享文件系统

由 Amit Rudraraju (AWS)、Sam Apa (AWS)、Bheemeswararao Balla (AWS)、Wally Lu (AWS) 和 Sanjeev Prakasam (AWS) 编写

摘要

迁移 300 台或更多服务器时，就被视为大规模迁移。大规模迁移的目的是将工作负载从其现有的本地数据中心迁移至 AWS Cloud，而这些项目通常侧重于应用程序和数据库工作负载。但是，共享文件系统需要集中精力，并制定单独的迁移计划。此模式描述了共享文件系统的迁移进程，并提供了在大规模迁移项目中成功迁移共享文件系统的最佳实践。

共享文件系统 (SFS)，也称为网络或集群文件系统，是装载到多个服务器上的文件共享。共享文件系统可通过 Network File System (NFS)、Common Internet File System (CIFS) 或 Server Message Block (SMB)等协议进行访问。

这些系统不会使用 AWS 应用程序迁移服务等标准迁移工具进行迁移，因为它们既不是专用迁移主机，也未表示为块设备。尽管大多数主机依赖项都为透明迁移，但依赖文件系统的协调和管理必须分开处理。

您可通过下几个阶段迁移共享文件系统：发现、规划、准备、割接和验证。使用此模式和随附的工作簿，您可以将共享文件系统迁移到 AWS 存储服务，例如亚马逊弹性文件系统 (Amazon EFS)、亚马逊 FSx 版 NetApp ONTAP 或亚马 FSx 逊 Windows 文件服务器。要传输文件系统，您可以使用 AWS DataSync 或第三方工具，例如 NetApp SnapMirror。

> **Note**
>
> 这种模式是 AWS Prescriptive Guidence 系列的一部分，该系列讲述了向 AWS 云的大规模迁移。此模式包括最佳实践和说明，用于将服务器 SFSs 纳入波浪计划。如果您要在大型迁移项目之外迁移一个或多个共享文件系统，请参阅 Amazon EFS、Amazon for Windows 文件服务器和 A maz on FSx for NetApp ONTAP 的 AWS 文档中的数据传输说明。 FSx

先决条件和限制

先决条件

先决条件可能会不同，具体取决于您的源和目标共享文件系统以及您的用例。最常见的条件如下：

- 一个有效的 Amazon Web Services account。

- 您已完成了大规模迁移项目的应用程序组合发现，并开始制定波次计划。有关更多信息，请参见 AWS 大规模迁移产品组合手册。
- 虚拟私有云 (VPCs) 和安全组，允许本地数据中心和您的 AWS 环境之间的入口和出口流量。有关更多信息，请参阅 Network-to-Amazon VPC 连接选项和 AWS DataSync 网络要求。
- 创建 AWS CloudFormation 堆栈的权限或创建 Amazon EFS 或亚马逊 FSx 资源的权限。有关更多信息，请参阅CloudFormation 文档、Amazon EFS 文档或亚马逊 FSx 文档。
- 如果您使用 AWS DataSync 执行迁移，则需要以下权限：
  - AWS DataSync 向 AWS 日志组发送 CloudWatch 日志的权限。有关更多信息，请参阅允许将日志上传 DataSync 到 CloudWatch 日志组。
  - 访问 CloudWatch 日志组的权限。有关更多信息，请参阅管理 CloudWatch 日志资源访问权限概述。
  - 在中创建代理和任务的权限 DataSync。有关更多信息，请参阅使用 AWS 所需的 IAM 权限 DataSync。

限制

- 这种模式旨在 SFSs 作为大型迁移项目的一部分进行迁移。它包括最佳实践和说明，用于 SFSs 将其纳入迁移应用程序的浪潮计划。如果您要在大型迁移项目之外迁移一个或多个共享文件系统，请参阅 Amazon EFS、Amazon for Windows 文件服务器和 A maz on FSx for NetApp ONTAP 的 AWS 文档中的数据传输说明。 FSx
- 这种模式基于常用的架构、服务与迁移模式。但是，大规模迁移项目和策略可能会因组织而异。您可能需要根据自己的要求自定义此解决方案或工作簿。

架构

源技术堆栈

下列一个或多个：

- Linux (NFS) 文件服务器
- Windows (SMB) 文件服务器
- NetApp 存储阵列
- Dell EMC Isilon 存储阵列

目标技术堆栈

下列一个或多个：

- Amazon Elastic File System
- FSx 适用于 NetApp ONTAP 的亚马逊
- FSx 适用于 Windows 文件服务器的亚马逊

目标架构

此图显示以下流程：

1. 您可以使用 AWS Direct Connect 或 AWS Site-to-Site VPN 等 AWS 服务，在本地数据中心和 AWS 云之间建立连接。
2. 您在本地数据中心安装 DataSync 代理。
3. 根据您的 Wave 计划，您可以使用将数据从源共享文件系统复制 DataSync 到目标 AWS 文件共享。

迁移阶段

下图介绍了在大规模迁移项目中迁移 SFS 的阶段和高级步骤。

此模式的操作说明部分包含有关如何完成迁移以及使用所附工作簿的详细说明。以下是此分阶段方法步骤的高度概述。

阶段	步骤
发现	1. 使用发现工具，您可收集有关共享文件系统的数据，包含服务器、挂载点和 IP 地址。 2. 使用配置管理数据库 (CMDB) 或迁移工具，您可收集有关服务器的详细信息，包含有关迁移波次、环境、应用程序所有者、IT 服务管理 (ITSM) 服务名称、组织单位和应用程序 ID 的信息。

规划	3. 使用收集到的有关 SFSs 和服务器的信息，创建 SFS 波浪计划。 4. 使用构建工作表中的信息，为每个 SFS 选择目标 Amazon Web Services 以及迁移工具。
准备	5. 在 Amazon EFS、 FSx 适用于 NetApp ONTAP 的亚马逊或适用于 Windows 文件服务器 FSx 的亚马逊中设置目标基础架构。 6. 设置数据传输服务，例如 DataSync，然后开始初始数据同步。初始同步完成后，您可将重复同步设置为按计划运行。 7. 使用有关目标文件共享的信息 (例如 IP 地址或路径) 更新 SFS 波次计划。
割接	8. 停止主动访问源 SFS 的应用程序。 9. 在数据传输服务中，执行最终数据同步。 10. 同步完成后，通过查看日志中的 CloudWatch 日志数据来验证同步是否完全成功。
验证	11. 在服务器上，将挂载点更改为新 SFS 路径。 12. 重新启动和验证应用程序。

工具

Amazon Web Services

- Amazon CloudWatch Logs 可帮助您集中管理来自所有系统、应用程序和 AWS 服务的日志，以便您可以监控它们并安全地将其存档。
- AWS DataSync 是一项在线数据传输和发现服务，可帮助您在 AWS 存储服务之间移动文件或对象数据。
- Amazon Elastic File System (Amazon EFS) 可帮助您在 Amazon Web Services Cloud 中创建和配置共享文件系统。

- Amazon FSx 提供的文件系统支持行业标准的连接协议，并可在 AWS 区域之间实现高可用性和复制。

其他工具

- SnapMirror是一种 NetApp 数据复制工具，可将数据从指定的源卷或 qtre e 分别复制到目标卷或 qtree。您可以使用此工具将 NetApp 源文件系统迁移到 Amazon FSx for ONTAP。
- Robocopy 是 Roboost File Copy 的缩写，是 Windows 的命令行目录和命令。您可以使用此工具将 Windows 源文件系统迁移到亚马逊 Window FSx s 文件服务器版。

最佳实践

波次规划方法

在为大规模迁移项目规划波次时，请考虑延迟和应用程序性能。当 SFS 和依赖应用程序在不同的位置运行时，例如一个在云端，一个在本地数据中心，可能会增加延迟并影响应用程序性能。创建波次计划时可用的选项如下：

1. 在同一波中迁移 SFS 和所有依赖项服务器 — 这种方法可以防止性能问题并最大限度地减少返工，例如多次重新配置挂载点。当应用程序和 SFS 间需要非常低的延迟时，建议使用它。但是，波次规划很复杂，目标通常是从依赖项分组中移除变量，而不是将其添加到依赖项分组中。此外，如果许多服务器访问相同 SFS，则不建议使用这种方法，因为这会使波次过大。
2. 最后一台依赖服务器迁移后迁移 SFS — 例如，如果多台服务器访问一个 SFS，并且这些服务器计划在第 4、6 和第 7 波中迁移，请安排 SFS 在第 7 波中迁移。

   对于大规模迁移，这种方法通常最合乎逻辑，推荐用于对延迟敏感的应用程序。它降低了数据传输的相关成本。它还可以最大限度地缩短 SFS 和更高级别的应用程序（例如生产）之间的延迟时间，因为更高级别的应用程序通常计划在开发和 QA 应用程序之后最后迁移。

   但是，这种方法仍然需要发现、规划以及敏捷性。您可能需要在较早的波次中迁移 SFS。确认应用程序能够承受从第一个依赖波到包含 SFS 的波次之间的时间段内的额外延迟。与应用程序所有者进行发现会话，并在同一波次中迁移对延迟最敏感的应用程序。如果在迁移依赖应用程序后发现性能问题，请快速调整以尽快迁移 SFS。
3. 在大规模迁移项目结束时迁移 SFS — 如果延迟并非重要因素，例如不经常访问 SFS 中的数据或对应用程序性能不重要时，建议使用这种方法。这种方法简化了迁移并简化了割接任务。

您可根据应用程序的延迟敏感度来混合这些方法。例如，您可以使用方法 1 或 2 迁移延 SFSs 迟敏感型，然后使用方法 3 迁移其余部分。 SFSs

选择 AWS 文件系统服务

AWS 提供多种文件存储云服务。每种方法在性能、规模、可访问性、集成、合规性和成本优化方面有不同的优势和限制。包含合乎逻辑的默认选项。例如，如果你当前的本地文件系统运行的是 Windows Server，那么亚马逊 FSx 版 Windows 文件服务器是默认选择。或者，如果本地文件系统运行的是 NetApp ONTAP，则默认选择 Amazon FSx fo NetApp r ONTAP。但是，您可根据应用程序的要求或实现其他云运营优势来选择目标服务。有关更多信息，请参阅为您的部署选择合适的 AWS 文件存储服务（AWS 峰会演示文稿）。

选择迁移工具

Amazon EFS 和亚马逊 FSx 支持使用 AWS 将共享文件系统迁移 DataSync 到 AWS 云。有关支持的存储系统和服务、优势和用例的更多信息，请参阅什么是 AWS DataSync。有关使用传输文件的过程概述 DataSync ，请参阅 AWS DataSync 传输的工作原理。

还有几种第三方工具可用，包含：

- 如果您选择 Amazon FSx for NetApp ONTAP，则可以使用将文件从本地数据中心迁移 NetApp SnapMirror 到云端。 SnapMirror 使用块级复制，它可以比数据传输过程更快 DataSync 并缩短持续时间。有关更多信息，请参阅使用 NetApp SnapMirror迁移到 FSx ONTAP 版。
- 如果你选择 Amazon f FSx or Windows 文件服务器，则可以使用 Robocopy 将文件迁移到云端。有关更多信息，请参阅使用 Robocopy 将 FSx 现有文件迁移到 Windows 文件服务器。

操作说明

发现

Task	描述	所需技能
准备 SFS 发现工作簿。	1. 在此模式的附件部分下载工作簿。它包含两个文件，即 SFS-Discovery-Workbook.xlsx 和 SFS-Wave-Plan-Workbook.xlsx。	迁移工程师，迁移主管

Task	描述	所需技能
	2. 在 Microsoft Excel 中打开 SFS-Discovery-Workbook 文件。 3. 在控制面板 上，执行以下操作： • 在 A 列，更新环境名称。 • 在 B 列，更新环境的顺序，使其按从最低 (1) 优先级到最高优先级的顺序排列。 • 在 D—E 列中，更新波次时间表。 • 在 C 和 K 列，更新 Amazon Web Services account 名称。 • 在 L 列中，更新 VPC IDs。 • 在 M—O 列中，更新子网。 IDs 4. 查看工作簿模板的其余部分，并更新组织或用例所需任何其他值。 5. 保存工作簿。	

Task	描述	所需技能
收集有关源 SFS 的信息。	1. 使用您的首选发现工具，识别所有适用的存储设备、Linux 服务器以及 Windows 服务器上的所有 SFS 挂载。通常，您需要收集以下信息： • 客户端设备 • 客户端 IP 地址 • SFS 详细信息 • 挂载点 **Note** 您可以将装载点详细信息添加到迁移运行手册中，以便在迁移后重新装载 SFS。 2. 打开 SFS-Discovery-Workbook 文件。 3. 在 Wave-Sheet 工作表，执行以下操作： • 在服务器位置 (D) 列的公式中，确认本地源 CIDR 范围格式是否适用于您的范围。例如，如果您的 CIDR 范围是 `10.0.0.0/8` ，请输入 `10.*.*.*`。 • 在 SFS 位置 (E) 列公式中，确认目标 VPC 的 CIDR 范围格式是否适用于您的范围。例如，	迁移工程师，迁移主管

Task	描述	所需技能
	如果您的 CIDR 范围是 `176.16.0.0/16` ，请输入 `176.16.*.*` 。 4. 在 SFS-Data 工作表，执行以下操作： • 在服务器名称 (A) 列中，输入装载 SFS 服务器的名称。 • 在 SFS 路径 (B) 列，输入 SFS 的名称。 • 在 IP 地址 (C) 列，输入服务器的 IP 地址。 • 添加您在发现期间收集的任何其他相关信息，例如挂载点以及 SFS 大小。您可稍后使用这些数据来修改波次计划计算。 5. 保存工作簿。	

Task	描述	所需技能
收集有关服务器的信息。	1. 使用您的 CMDB 或迁移工具中的记录数据，识别有关装有 SFS 的服务器的所有以下信息： • 服务器名称 • IP 地址 • 波次 • 组织部门（OU） • 服务器环境，例如 DEV、QA 或 PROD • 应用程序名称 • 应用程序所有者和联系信息 2. 打开 SFS-Discovery-Workbook 文件。 3. 在 Server-Data 工作表的 A—H 列，输入您收集的有关源服务器的信息。请注意以下几点： • 在 Wave # (C) 列中，输入波浪名称（例如`Wave1`）、 out-of-scope (OOS) 或`Retire`。 • 如果应用程序所有者联系人 (H) 列，请验证电子邮件地址是否正确。此电子邮件地址是根据您在应用程序所有者 (G) 列提供的姓名自动生成的。如有必要，请手动更新该值，以反映正确的电子邮件地址。	迁移工程师，迁移主管

Task	描述	所需技能
	• 不要修改包含公式的 I—J 列。 4. 保存工作簿。	

规划

Task	描述	所需技能
制定 SFS 波次计划。	1. 打开 SFS-Discovery-Workbook 文件。 2. 验证在发现阶段收集的所有信息是否准确和最新。 3. 在 Wave-Sheet 工作表，根据1值筛选 SFS 波次 (K) 列。这是第一波 SFSs 中所有内容的清单。 **Note** 此列0中的值为表示 SFS 已超出迁移范围。这可能是因为 SFS 已托管在 AWS，或者因为访问共享的服务器超出了迁移范围。 4. 确认您要在此浪潮 SFSs 中迁移它们。有关如何分配 SFSs 给波浪的更多信息，请参阅 “最佳实践” 部分中的波浪规划方法。	构建主管，割接主管，迁移工程师，迁移主管

Task	描述	所需技能
	5. 选择和复制包含筛选值的单元格。切勿复制包含列标题的标题行。 6. 打开您之前下载的 SFS-Wave-Plan-Workbook 文件。 7. 在 Export-from-Discovery 工作表，选择单元格 A2。 8. 粘贴复制的数据。 9. 保存 SFS-Discovery-Workbook 和 SFS-Wave-Plan-Workbook 文件。	

Task	描述	所需技能
选择目标 Amazon Web Services 与迁移工具。	1. 在SFS-Wave-Plan-Workbook 文件Exported-from-Discovery 的工作表中，选择并复制旧路径 (C) 列中的值。 2. 在 Build-Wave 工作表上，选择单元格 A2。 3. 粘贴复制的数据。此工作表中的 B—M 列会自动更新，以反映与此路径关联的其他数据。 4. 移除 A 列中的所有重复值。有关说明，请参阅移除重复值(Microsoft Support 网站)。 5. 在目标模式或服务 (F) 列，查看推荐的目标 Amazon Web Services 并根据需要进行更新。有关更多信息，请参阅此模式的最佳实践部分中的选择 AWS 文件系统服务。 6. 在迁移方法 (G) 列，查看推荐的迁移工具并根据需要进行更新。有关更多信息，请参阅此模式的最佳实践部分中的选择迁移工具。 7. 保存 SFS-Discovery-Workbook 文件。您已经完成了为此波次创建的波次计划。 8. 重复这些说明，为每个波次准备波次计划。由于迁移期间波次计划可能会发生变化	迁移工程师，迁移主管

Task	描述	所需技能
	，因此我们建议您提前计划不超过 5 个波次。	

准备

Task	描述	所需技能
设置目标文件系统。	根据波次计划中记录的详细信息，在目标 Amazon Web Services account 、VPC 和子网中设置目标文件系统。有关说明，请参阅以下 AWS 文档。 • Amazon EFS • FSx 适用于 NetApp ONTAP 的亚马逊 • FSx 适用于 Windows 文件服务器的亚马逊	迁移工程师，迁移主管，AWS 管理员
设置迁移工具和传输数据。	1. 如果您使用的是 AWS DataSync，请为 DataSync 任务配置日志。有关说明，请参阅记录您的 AWS DataSync 任务活动。 2. 设置迁移工具，并根据所选工具说明执行初始数据传输： • 对于 Amazon EFS，请参见以下内容： • 使用 AWS 将文件传输到亚马逊 EFS DataSync	AWS 管理员，云管理员，迁移工程师，迁移主管

Task	描述	所需技能

	• 有关适用 FSx 于 ONTAP 的 Amazon，请参阅以下内容：
	  • FSx 使用 ONTAP 迁移到 NetApp SnapMirror
	  • 使用 AWS 迁移到 FSx ONTAP DataSync
	• FSx 对于亚马逊 Windows 文件服务器，请参阅以下内容：
	  • 使用 AWS 将 FSx 现有文件迁移到 Windows 文件服务器 DataSync
	  • 使用 Robocopy 将 FSx 现有文件迁移到 Windows 文件服务器
	3. 在初始传输期间或之后，可能会更改源 SFS。设置源文件系统和目标文件系统间的重复数据传输，以保持数据同步：
	• 如果您正在使用 DataSync，请参阅安排 AWS DataSync 任务。DataSync 仅传输源 SFS 中修改过的文件或新文件。
	• 如果您使用第三方工具，请参阅所选工具的文档。	

Task	描述	所需技能
更新波次计划。	1. 打开当前波次的 SFS-Wave-Plan-Workbook 文件。 2. 在 Build—Wave 工作表，在新路径 IP 地址 (N) 列，输入目标文件系统的 IP 地址。执行以下任一操作，查找 IP 地址： • 对 FSx 于 Windows 文件服务器，在亚马逊 FSx 控制台上，选择文件系统，选择您的文件系统，然后查看 “网络和安全” 部分。 • FSx 有关 ONTAP 的信息，请参阅装入卷。 • 有关 Amazon EFS，请参阅使用 IP 地址挂载。 3. 在新路径 (O) 列，输入新的挂载路径。挂载路径为文件系统的 DNS 名称。执行以下任一操作，以找到挂载路径： • 对 FSx 于 Windows 文件服务器，在亚马逊 FSx 控制台上选择文件系统，选择您的文件系统，然后选择附加。 • FSx 有关 ONTAP 的信息，请参阅文件系统详细信息页面。有关说明，请参阅挂载卷。 • 有关 Amazon EFS，请参阅获取信息。	迁移工程师，迁移主管

Task	描述	所需技能
	4. 在 Remount-Summary 工作表，确认新路径 (C) 和新路径 IP 地址 (D) 列是否反映了更新的值。 5. 确认您的组织已准备好在割接后重新挂载 Linux 和 Windows 文件系统运行手册。有关一般说明，请参阅以下： • 挂载 EFS 文件系统 • 访问 Window FSx s 文件服务器文件共享 • FSx 为 ONTAP 卷装载 6. 如果此波次中未包含任何依赖服务器，请将其记录在 App-Team-Communication 工作表中。通知相应的应用程序或服务器拥有者，因为他们可能不包含在标准波次通信中。 7. 如果 SFSs 在完成波浪计划后从波浪中移除，请在 Descop ed 工作表中对其进行跟踪。	

割接

Task	描述	所需技能
停止应用程序。	如果应用程序或客户端正在源 SFS 主动执行读写操作，请在执行最终数据同步之前将其停止。有关说明，请参	应用程序所有者、应用程序开发人员

Task	描述	所需技能
	阅应用程序文档或停止读写活动的内部流程。例如，请参阅启动或停止 Web 服务器 (IIS 8)（Microsoft 文档）或使用 systemctl 管理系统服务（Red Hat 文档）。	
执行最后数据传输。	1. 在迁移工具中，手动运行最终的数据传输任务或者作业，将目标文件系统与源 SFS 同步。有关说明，请参阅启动 DataSync 任务或查看所选第三方迁移工具的文档。 2. 请等待数据传输任务完成。有关更多信息，请参阅 AWS 通过 A mazon 监控 AWS DataSync 活动 CloudWatch和通过命令行监控您的 DataSync 任务。	迁移工程师，迁移主管

Task	描述	所需技能
验证数据传输。	如果您使用的是 AWS DataSync，请执行以下操作以验证最终数据传输成功完成： 1. 在 AWS DataSync 控制台中，记下任务和执行 ID，例如`task-0000-exec-1111` 。 2. 导航到任务的 “任务记录” 部分。 DataSync 3. 选择CloudWatch 日志组链接。 4. 在日志中，搜索任务与执行 ID。 5. 记录任何传输错误。有关更多信息，请参阅 DataSync 文档中的常见错误。 6. 请验证以下内容： • 比较源端和目标端的文件列表 SFSs ，确认所有数据均已传输 • 比较源和目标之间的文件访问权限 SFSs。 如果您使用第三方工具，请参阅所选迁移工具文档中的数据传输验证说明。	迁移工程师，迁移主管

验证

Task	描述	所需技能
重新安装文件系统，并验证应用程序的功能和性能。	1. 如果在此波次中迁移了依赖服务器，则在 SFS-Wave-Plan-Workbook 文件的 Remount-Summary 工作表中，在新服务器 IP 地址 (F) 列中输入新服务器 IP 地址。 2. 在所有服务器，将文件系统的装载点从旧路径更新到新路径。使用组织的运行手册进行重新挂载，之前在准备阶段介绍过。 3. 通过检查挂载和验证文件是否存在，确认文件系统已正确装载并且可以访问。基础设施团队通常会执行此活动。 4. 根据应用程序需要，重新启动应用程序并与应用程序所有者或 QA 团队接触，以完成应用程序的功能和性能测试。	AWS 系统管理员、应用程序所有者

故障排除

事务	解决方案
Microsoft Excel 中的单元格值不会更新。	通过拖动填充手柄，复制示例行中的公式。有关更多信息，请参阅 Windows 或 Mac 的说明（Microsoft 支持网站）

相关资源

AWS 文档

- AWS DataSync 文档
- Amazon EFS 文档
- 亚马逊 FSx 文档
- 向 Amazon Web Services Cloud 大规模迁移
  - AWS 大规模迁移指南
  - AWS 大规模迁移产品组合手册

故障排除

- 对 AWS DataSync 问题进行故障排除
- 故障排除 Amazon EFS
- 对适用 FSx 于 Windows 的亚马逊文件服务器进行故障排除
- FSx 对亚马逊版 NetApp ONTAP 进行故障排除

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 Oracle GoldenGate 平面文件适配器将 Oracle 数据库迁移到 Amazon RDS for Oracle

由 Dhairya Jindani (AWS) 和 Baji Shaik (AWS) 创建

摘要

Oracle GoldenGate 是一项适用于异构数据库和 IT 环境的实时数据捕获和复制服务。但是，该服务目前不支持适用于 Oracle 的 Amazon Relational Database Service (Amazon RDS)。有关支持的数据库的列表，请参阅 Oracle GoldenGate 异构数据库（Oracle 文档）。此模式描述了如何使用 Oracle GoldenGate 和 Oracle GoldenGate 平面文件适配器从源 Oracle 数据库生成平面文件，这些文件可以在本地或亚马逊弹性计算云 (Amazon EC2) 实例上。然后，您可以将这些平面文件导入至 Amazon RDS for Oracle 数据库实例。

在这种模式中，您可以使用 Oracle GoldenGate 从源 Oracle 数据库中提取跟踪文件。数据泵将跟踪文件复制到集成服务器（即 Amazon EC2 实例）。在集成服务器上，Oracle GoldenGate 使用平面文件适配器根据跟踪文件的事务数据捕获生成一系列连续的平面文件。Oracle 将数据 GoldenGate 格式化为分隔符分隔的值或长度分隔的值。然后，您可以使用 Oracle SQL*Loader 将平面文件导入至目标 Amazon RDS for Oracle 数据库实例。

目标受众

这种模式适用于那些对 Oracle GoldenGate 基本构件有经验和了解的人。有关更多信息，请参阅 Oracle GoldenGate 架构概述（Oracle 文档）。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- 甲骨文 GoldenGate 许可证。
- Oracle GoldenGate 适配器的单独许可证。
- 一种源 Oracle 数据库，可以在本地运行或在 Amazon EC2 实例上运行。
- 用作集成服务器的 Amazon EC2 Linux 实例。有关更多信息，请参阅亚马逊 EC2 Linux 实例入门（亚马逊 EC2 文档）。
- 目标 Amazon RDS for Oracle 数据库实例。有关更多信息，请参阅创建 Oracle 数据库实例（Amazon RDS 文档）。

产品版本

- Oracle Database Enterprise Edition 版本 10g、11g、12c 或更高版本
- 甲骨文 GoldenGate 版本 12.2.0.1.1 或更高版本

架构

源技术堆栈

Oracle 数据库（本地数据库或在亚马逊 EC2 实例上）

目标技术堆栈

Amazon RDS for Oracle

源架构和目标架构

1. Oracle 从源数据库日志中 GoldenGate 提取跟踪。
2. Data Pump 提取跟踪，并将其迁移至集成服务器。
3. Oracle GoldenGate 平面文件适配器读取轨迹、源定义和数据提取参数。
4. 您可以退出可生成控制文件和平面数据文件的提取。
5. 您可以在中将平面数据文件迁移到 Amazon RDS for Oracle 数据库实例 AWS Cloud。

工具

AWS 服务

- 亚马逊弹性计算云 (Amazon EC2) 在中提供可扩展的计算容量 AWS Cloud。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- 适用于 Oracle 的 Amazon Relational Database Service（Amazon RDS）可帮助您在中设置、操作和扩展 Oracle 关系数据库 AWS Cloud。

其他服务

- Oracle GoldenGate 是一项服务，可帮助您将数据从一个数据库复制、筛选和转换到另一个异构数据库或另一个目标拓扑（例如平面文件）。
- Oracle GoldenGate 应用程序适配器使 Oracle GoldenGate 能够根据源数据库的跟踪文件中捕获的事务数据生成一系列顺序平面文件和控制文件。这些适配器广泛用于数据仓库应用程序以及专有或

旧版应用程序中的提取、转换、加载（ETL）操作。Oracle GoldenGate 执行此捕获并近乎实时地将其应用于异构数据库、平台和操作系统。此类适配器支持不同的输出文件格式，如 CSV 或 Apache Parquet。您可加载此类已生成文件，以便将数据加载至不同的异构数据库。

操作说明

在源数据库服务器 GoldenGate 上设置 Oracle

Task	描述	所需技能
下载甲骨文 GoldenGate。	在源数据库服务器上，下载 Oracle GoldenGate 版本 12.2.0.1.1 或更高版本。有关说明，请参阅下载 Oracle GoldenGate（Oracle 文档）。	数据库管理员
安装甲骨文 GoldenGate。	有关说明，请参阅安装 Oracle GoldenGate（Oracle 文档）。	数据库管理员
设置 Oracle GoldenGate。	有关说明，请参阅为 Oracle 准备数据库 GoldenGate（Oracle 文档）。	数据库管理员

在集成服务器 GoldenGate 上设置 Oracle

Task	描述	所需技能
下载甲骨文 GoldenGate。	在集成服务器上，下载 Oracle GoldenGate 版本 12.2.0.1.1 或更高版本。有关说明，请参阅下载 Oracle GoldenGate（Oracle 文档）。	数据库管理员
安装甲骨文 GoldenGate。	创建目录，设置管理器进程，并为异构环境创建 defgen文件。有关说明，请参阅安装	数据库管理员

Task	描述	所需技能
	Oracle GoldenGate（Oracle 文档）。	

更改 Oracle GoldenGate 数据捕获配置

Task	描述	所需技能
准备 Oracle GoldenGate 适配器。	在集成服务器上，设置 Oracle GoldenGate 适配器软件。执行以下操作： 1. 从 Oracle Software Delivery Cloud 下载 ggs_Adapters_Linux_x64.zip。 2. 解压缩 ggs_Adapters_Linux_x64.zip。 3. 运行以下命令安装适配器： `tar -xvf ggs_Adapters_Linux_x64.tar`	数据库管理员
配置数据泵。	在源服务器上，配置 Data Pump，以将跟踪文件从源服务器传输至集成服务器。创建数据泵参数文件和跟踪文件目录。有关说明，请参阅配置平面文件适配器（Oracle 文档）。	数据库管理员

生成并迁移平面文件

Task	描述	所需技能
生成平面文件。	创建提取文件和控制文件，然后在集成服务器上启动提取过程。这将提取数据库更改，并将源数据库写入平面文件。有关说明，请参阅使用平面文件适配器（Oracle 文档）。	数据库管理员
将平面文件加载至目标数据库。	将平面文件加载至目标 Amazon RDS for Oracle 数据库实例。有关更多信息，请参阅 使用 Oracle SQL*Loader 导入（Amazon RDS 文档）。	数据库管理员

故障排除

事务	解决方案
Oracle GoldenGate 平面文件适配器生成错误。	有关适配器错误的描述，请参阅 定位错误消息（Oracle 文档）。有关故障排除说明，请参阅平面文件适配器故障排除（Oracle 文档）。

相关资源

- 安装 Oracle GoldenGate（甲骨文文档）
- 配置 Oracle GoldenGate（甲骨文文档）
- 了解 Oracle GoldenGate 适配器（Oracle 文档）
- 配置平面文件适配器（Oracle 文档）

更改 Python 和 Perl 应用程序以支持数据库从Microsoft SQL Server 迁移至兼容 Amazon Aurora PostgreSQL 的版本

由 Dwarika Patra (AWS) 和 Deepesh Jayaprakash (AWS) 编写

摘要

此模式描述了将数据库从 Microsoft SQL Server 迁移到 Amazon Aurora PostgreSQL 兼容版时可能需要对应用程序存储库进行的更改。该模式假设这些应用程序基于 Python 或 Perl，并为这些脚本语言提供单独的指令。

将 SQL Server 数据库迁移至兼容 Aurora PostgreSQL 的数据库涉及架构转换、数据库对象转换、数据迁移和数据加载。由于 PostgreSQL 和 SQL Server 之间存在差异（与数据类型、连接对象、语法和逻辑有关），因此最困难的迁移任务是对代码库进行必要的更改，使其能够在 PostgreSQL 中正常运行。

对于基于 Python 的应用程序，连接对象和类分散在整个系统中。此外，Python 代码库可能使用多个库来连接到数据库。如果数据库连接接口发生变化，运行应用程序内联查询的对象也需要更改。

对于基于 Perl 的应用程序，更改涉及连接对象、数据库连接驱动程序、静态和动态内联 SQL 语句以及应用程序如何处理复杂的动态 DML 查询和结果集。

迁移应用程序时，您还可以考虑 AWS 上可能的增强功能，例如使用 Amazon Simple Storage Service (Amazon S3) 访问替换 FTP 服务器。

应用程序迁移过程涉及以下挑战：

- 连接对象。如果连接对象分散在具有多个库和函数调用的代码中，您可能必须找到一种通用方法来更改它们以支持 PostgreSQL。
- 记录检索或更新期间的错误或者异常处理。如果对返回变量、结果集或数据帧的数据库进行条件创建、读取、更新和删除 (CRUD) 操作，则任何错误或异常都可能导致应用程序错误并产生级联效应。应通过适当的验证和保存点来仔细处理这些问题。此类保存点之一是调用 `BEGIN...EXCEPTION...END` 块内的大型内联 SQL 查询或数据库对象。
- 控制事务及其验证。其中包括手动和自动提交与回滚。Perl 的 PostgreSQL 驱动程序要求您始终明确设置自动提交属性。
- 处理动态 SQL 查询。这需要对查询逻辑和迭代测试有深入的了解，以确保查询按预期工作。
- 性能。您应该确保代码更改不会导致应用程序性能下降。

此模式详细解释了转换进程。

先决条件和限制

先决条件

- Python 和 Perl 语法工作知识。
- SQL Server 和 PostgreSQL 基本技能。
- 了解现有的应用程序架构。
- 访问您的应用程序代码、SQL Server 数据库以及 PostgreSQL 数据库。
- 使用开发、测试和验证应用程序更改的凭证访问 Windows 或 Linux (或其他 Unix)开发环境。
- 对于基于 Python 的应用程序，您的应用程序可能需要的标准 Python 库，例如用于处理数据帧的 Pandas，以及用于数据库连接的 psycopg2。SQLAlchemy
- 对于基于 Perl 应用程序，需要带有依赖库或模块的 Perl 包。全面的 Perl 存档网络 (CPAN) 模块可支持大多数应用程序要求。
- 所有必需依赖自定义库或模块。
- 用于对 SQL Server 进行读取访问以及对 Aurora 进行读/写访问的数据库凭证。
- PostgreSQL 通过服务和用户验证和调试应用程序更改。
- 在应用程序迁移期间访问开发工具，例如Visual Studio Code、 Sublime Text 或 pgAdmin。

限制

- 某些 Python 或 Perl 版本、模块、库以及包与云环境不兼容。
- 某些用于 SQL Server 的第三方库和框架无法替换支持 PostgreSQL 迁移。
- 性能变化可能需要更改应用程序、内联 Transact-SQL (T-SQL) 查询、数据库函数以及存储过程。
- PostgreSQL 支持表名、列名和其他数据库对象小写名称。
- 某些数据类型（例如 UUID 列）仅以小写形式存储。Python 和 Perl 应用程序必须要处理此类大小写差异。
- 必须使用 PostgreSQL 数据库相应文本列的正确数据类型来处理字符编码差异。

产品版本

- Python 3.6 或更高版本（使用支持您的操作系统的版本）
- Perl 5.8.3 或更高版本（使用支持您的操作系统的版本）
- 兼容 Aurora PostgreSQL 的版本 4.2 或更高版本（查看详细信息）

架构

源技术堆栈

- 脚本(应用程序编程)语言：Python 2.7 或更高版本，或 Perl 5.8
- 数据库：Microsoft SQL Server 版本 13
- 操作系统：Red Hat Enterprise Linux (RHEL) 7

目标技术堆栈

- 脚本(应用程序编程)语言：Python 3.6 或更高版本，或 Perl 5.8 或更高版本
- 数据库：Aurora PostgreSQL-Compatible 4.2
- 操作系统：RHEL 7

迁移架构

工具

AWS 工具和服务

- Aurora PostgreSQL–Compatible Edition 是一个完全托管式、兼容 PostgreSQL 和 ACID 的关系数据库引擎，结合了高端商用数据库的速度和可靠性，同时还具有开源数据库的成本效益。Aurora PostgreSQL 是 PostgreSQL 的直接替代品，可以让您通过简单且经济高效的方式设置、运行和扩展新的和现有的 PostgreSQL 部署。
- AWS 命令行界面（AWS CLI）是一种开源工具，它使您能够使用命令行 Shell 中的命令与 Amazon Web Services 交互。

其他工具

- Python 和 P ostgresSQL 数据库连接库，比如 psycopg2 和 SQLAlchemy
- Perl 及其 DBI 模块
- PostgreSQL 交互式终端 (psql)

操作说明

将您的应用程序存储库迁移至 PostgreSQL — 高级步骤

Task	描述	所需技能
按照以下代码转换步骤将您的应用程序迁移至 PostgreSQL。	1. 为 PostgreSQL 设置特定于数据库的 ODBC 驱动程序与库。例如，你可以将其中一个 CPAN 模块用于 Perl 和 p yodbc、psycop g 2 或 Python。SQLAlchemy 2. 使用这些库转换数据库对象以连接到 Aurora PostgreSQL 兼容。 3. 在现有应用程序模块中应用代码更改，以获得兼容的 T-SQL 语句。 4. 在应用程序代码中重写数据库特定的函数调用与存储过程。 5. 处理对应用程序变量及其用于内联 SQL 查询的数据类型的更改。 6. 处理不兼容的数据库专用函数。 7. 完成对已转换的用于数据库迁移的应用程序代码的 end-to-end测试。 8. 将来自 Microsoft SQL Server 的结果与您迁移至 PostgreSQL 的应用程序进行比较。	应用程序开发人员

Task	描述	所需技能
	9. 在 Microsoft SQL Server 和 PostgreSQL 之间执行应用程序性能基准测试。 10.修改应用程序调用的存储过程或者内联 T-SQL 语句以提高性能。 以下操作说明详细说明了 Python 和 Perl 应用程序的一些转换任务。	
为迁移的每个步骤使用清单。	将以下内容添加到应用程序迁移的每个步骤（包括最后一步）的清单中： • 请查看 PostgreSQL 文档，确保所有更改均与 PostgreSQL 标准兼容。 • 检查列的整数值与浮点值。 • 确定插入、更新和提取的行数，以及列名和日期/时间戳。您可以使用 diff 实用程序或编写脚本自动执行这些检查。 • 完成大型内联SQL语句的性能检查，检查应用程序的整体性能。 • 使用多个 try/catch 块检查数据库操作的错误处理是否正确以及程序正常退出。 • 检查以确保适当的日志记录流程到位。	应用程序开发人员

分析和更新您的应用程序 — Python 代码库

Task	描述	所需技能
分析现有的 Python 代码库。	您的分析应包含以下内容，以简化应用程序迁移过程： • 识别代码的所有连接对象。 • 识别所有不兼容的内联 SQL 查询 (例如 T-SQL 语句和存储过程) 并分析所需更改。 • 查看您的代码文档并追踪控制流以了解代码功能。稍后当您测试应用程序性能或负载比较时，这将很有帮助。 • 了解应用程序用途，以便在数据库转换后对其进行有效测试。大多数可通过数据库迁移进行转换的 Python 应用程序要么是将数据从其他来源加载到数据库表的订阅源，要么是从表中检索数据并将其转换为适合创建报告或进行 API 调用以执行验证的不同输出格式（例如 CSV、JSON 或平面文件）的提取器。	应用程序开发人员

将您的数据库连接转换至支持 PostgreSQL。	大多数 Python 应用程序使用 pyodbc 库连接 SQL Server 数据库，如下所示。
	`import pyodbc`
	`....`
	`try:`
	`conn_string = "Driver=ODBC Driver 17 for SQL`
		应用程序开发人员

Task	描述	所需技能

	Server;UID={};PWD={};Server={};Database={}".format (conn_user, conn_password, conn_server, conn_database)
	conn = pyodbc.connect(conn_string)
	cur = conn.cursor()
	result = cur.execute(query_string)
	for row in result:
	print (row)
	except Exception as e:
	print(str(e))
	
	将数据库连接转换至支持 PostgreSQL，如下所示。
	
	import pyodbc
	import psycopg2
	....
	try:
	conn_string = 'postgresql+psycopg2://'+
	conn_user+':'+conn_password+'@'+conn_server+'/'+conn_database
	conn = pyodbc.connect(conn_string, connect_args={'options':'-csearch_path=dbo'})
	cur = conn.cursor()
	result = cur.execute(query_string)
	for row in result:
	print (row)	

Task	描述	所需技能
	except Exception as e: print(str(e))	

Task	描述	所需技能

将内联 SQL 查询更改为 PostgreSQL。	将您的内联 SQL 查询转换为与 PostgreSQL 兼容的格式。例如，以下 SQL Server 查询从表中检索字符串。
	
	`dtype = “type1”`
	`stm = ‘“SELECT TOP 1 searchcode FROM TypesTable (NOLOCK) WHERE code=”’ + “’” + str(dtype) + “’”`
	`# For Microsoft SQL Server Database Connection`
	`engine = create_engine(‘mssql+pyodbc:///?odbc_connect=%s’ % urllib.parse.quote_plus(conn_string), connect_args={‘connect_timeout’:login_timeout})`
	`conn = engine_connect()`
	`rs = conn.execute(stm)`
	`for row in rs:`
	`print(row)`
	
	转换后，与 PostgreSQL 兼容的内联 SQL 查询如下所示。
	
	`dtype = “type1”`
	`stm = ‘“SELECT searchcode FROM TypesTable WHERE code=”’ + “’” + str(dtype) + “’ LIMIT 1”`
	`# For PostgreSQL Database Connection`
		应用程序开发人员

Task	描述	所需技能

	`engine = create_engine(‘postgres+psycopg2://%s’ %conn_string, connect_args={‘connect_timeout’:login_timeout})`
	`conn = engine.connect()`
	`rs = conn.execute(stm)`
	`for row in rs:`
	`  print(row)`	

Task	描述	所需技能

处理动态 SQL 查询。	动态 SQL 可以出现在一个脚本或多个 Python 脚本中。前面的示例展示了如何使用 Python 的字符串替换函数插入变量以构建动态 SQL 查询。另一种方法是在适用的情况下在查询字符串中附加变量。
	
	在以下示例中，查询字符串是根据函数返回的值动态构造的。
	
	`query = '"SELECT id from equity e join issues i on e.permId=i.permId where e.id'"`
	`query += get_id_filter(ids) + " e.id is NOT NULL`
	
	这些类型的动态查询在应用程序迁移过程中非常常见。请按照以下步骤处理动态查询：
	
	• 检查整体语法(例如，带有子句的 SELECT 语 JOIN 句的语法)。
	• 验证查询中使用的所有变量或列名，例如 i 和 id。
	• 检查查询中使用的函数、参数和返回值（例如 get_id_filter 及其参数 ids）。
		应用程序开发人员

Task	描述	所需技能

处理结果集、变量与数据框。	对于 Microsoft SQL Server，您可以使用 Python 方法（例如 `fetchone()` 或 `fetchall()`）从数据库中检索结果集。您也可以使用 `fetchmany(size)` 并指定要从结果集中返回的记录数。为此，您可使用 pyodbc 连接对象，如以下示例中所示。
	
	pyodbc (Microsoft SQL Server)
	
	`import pyodbc`
	`server = 'tcp:myse`
	`rver.database.wind`
	`ows.net'`
	`database = 'exampledb'`
	`username = 'exampleu`
	`sername'`
	`password = 'examplep`
	`assword'`
	`conn = pyodbc.co`
	`nnect('DRIVER={ODB`
	`C Driver 17 for SQL`
	` Server};SERVER='+s`
	`erver+';DATABASE='`
	`+database+';UID='+`
	`username+';PWD='+`
	` password)`
	`cursor = conn.cursor()`
	`cursor.execute("SELECT`
	` * FROM ITEMS")`
	`row = cursor.fe`
	`tchone()`
	`while row:`
	`    print(row[0])`
	`    row = cursor.fe`
	`tchone()`
		应用程序开发人员

Task	描述	所需技能
	在 Aurora 中，要执行类似的任务，例如连接到 PostgreSQL 和获取结果集，你可以使用 psycopg2 或。SQLAlchemy这些 Python 库提供了连接模块和游标对象来遍历 PostgreSQL 数据库记录，如以下示例所示。 psycopg2（兼容Aurora PostgreSQL） `import psycopg2 query = "SELECT * FROM ITEMS;" //Initialize variables host=dbname=user= password=port=sslmode=connect_timeout="" connstring = "host='{host}' dbname='{dbname}' user='{user}' \ password='{password}'port='{port}' ".format(host=host,dbname=dbname,\ user=user,password=password,port=port) conn = psycopg2.connect(connstring) cursor = conn.cursor() cursor.execute(query) column_names = [column[0] for column in cursor.description]`	

Task	描述	所需技能
	print("Column Names: ", column_names) print("Column values: " for row in cursor: print("itemid :", row[0]) print("itemdescrpt ion :", row[1]) print("it emprice :", row[3]))	
	SQLAlchemy（兼容 Aurora PostgreSQL） from sqlalchemy import create_engine from pandas import DataFrame conn_string = 'postgres ql://core:database @localhost:5432/ex ampledatabase' engine = create_en gine(conn_string) conn = engine.co nnect() dataid = 1001 result = conn.exec ute("SELECT * FROM ITEMS") df = DataFrame (result.fetchall()) df.columns = result.ke ys() df = pd.DataFrame() engine.connect() df = pd.read_sql_query(sql_query, engine, coerce_float=False) print(“df=”, df)	

Task	描述	所需技能
在迁移期间和迁移之后测试应用程序。	测试迁移的 Python 应用程序是持续的过程。由于迁移包括连接对象更改（psycopg2 或 SQLAlchemy）、错误处理、新功能（数据框）、内联 SQL 更改、批量复制功能（bcp而不是COPY）和类似更改，因此在应用程序迁移期间和迁移之后都必须对其进行仔细测试。检查： • 错误条件和处理 • 迁移后出现任何记录不匹配的情况 • 记录更新或删除内容 • 运行应用程序所需时间	应用程序开发人员

分析和更新您的应用程序 — Perl 代码库

Task	描述	所需技能
分析现有 Perl 代码库。	您的分析应包含以下内容，以简化应用程序迁移过程。您应确定： • 任何 INI 或基于配置的代码 • 数据库特定的标准开放式数据库连接 (ODBC)Perl 驱动程序或任何自定义驱动程序 • 内联和 T-SQL 查询需要更改代码 • 各种 Perl 模块之间的交互（例如，由多个功能组件调	应用程序开发人员

Task	描述	所需技能
	用或使用的单个 Perl ODBC 连接对象） • 数据集与结果集处理 • 外部依赖 Perl 库 • 应用程序中 APIs 使用的任何内容 • Perl 版本兼容性以及与兼容 Aurora PostgreSQL 驱动程序兼容性	

Task	描述	所需技能

转换 Perl 应用程序和 DBI 模块的连接以支持 PostgreSQL。	基于Perl的应用程序通常使用 Perl DBI 模块，它是 Perl 编程语言的标准数据库访问模块。您可以为 SQL Server 和 PostgreSQL 使用相同 DBI 模块和不同的驱动程序。
	
	有关所需 Perl 模块、安装和其他说明的更多信息，请参阅 DBD::Pg documentation。以下示例连接到与 Aurora PostgreSQL 兼容的网址 `exampletest-aurorapg-database.cluster-sampleclusture.us-east-.rds.amazonaws.com` 。
	
	`#!/usr/bin/perl`
	`use DBI;`
	`use strict;`
	`my $driver = "Pg";`
	`my $hostname = “examplet`
	`est-aurorapg-datab`
	`ase-sampleclusture`
	`.us-east.rds.amazo`
	`naws.com”`
	`my $dsn = "DBI:$driver:`
	` dbname = $hostname`
	`;host = 127.0.0.1;port`
	` = 5432";`
	`my $username = "postgres`
	`";`
	`my $password = "pass123"`
	`;`
	`$dbh = DBI->conn`
	`ect("dbi:Pg:dbname`
	`=$hostname;host=$h`
		应用程序开发人员

Task	描述	所需技能
	ost;port=$port;opt ions=$options", $username, $password, {AutoCommit => 0, RaiseError => 1, PrintError => 0});	

Task	描述	所需技能

将内联 SQL 查询更改为 PostgreSQL。	您的应用程序可能包含带有 SELECT、DELETE、UPDATE 的内联 SQL 查询，以及包含 PostgreSQL 不支持的查询子句的类似语句。例如，PostgreSQL 中不支持 TOP 和 NOLOCK 等查询关键字。以下示例说明如何处理 TOP、NOLOCK 和 Boolean 变量。
	
	SQL Server 中：
	
	`$sqlStr = $sqlStr . "WHERE a.student_id in (SELECT TOP $numofRecords c_student_id \ FROM active_student_record b WITH (NOLOCK) \ INNER JOIN student_contributor c WITH (NOLOCK) on c.contributor_id = b.c_st)`
	
	对于 PostgreSQL，请转换为：
	
	`$sqlStr = $sqlStr . "WHERE a.student_id in (SELECT TOP $numofRecords c_student_id \ FROM active_student_record b INNER JOIN student_contributor c \`
		应用程序开发人员

Task	描述	所需技能
	on c.contributor_id = b.c_student_contr_id WHERE b_current_1 is true \ LIMIT $numofRecords)"	

Task	描述	所需技能
处理动态 SQL 查询与 Perl 变量。	动态 SQL 查询是在应用程序运行时生成 SQL 语句。这些查询是在应用程序运行时根据某些条件动态构建的，因此直到运行时才知道查询的全文。一个例子是一个金融分析应用程序，它每天分析排名前 10 的股票，并且这些股票每天都在变化。SQL 表是根据最佳执行者创建的，并且直到运行时才知道这些值。 假设此示例的内联 SQL 查询被传递给包装函数以获取变量中的结果集，然后变量使用条件来确定表是否存在： • 如果该表存在，则不创建它；做一些处理。 • 如果该表不存在，则创建该表并进行处理。 下面是变量处理的示例，后面是该用例的 SQL Server 和 PostgreSQL 查询。 `my $tableexists = db_read( arg 1, $sql_qry, undef, 'writer'); my $table_already_exists = $tableexists->[0]{table_exists}; if ($table_already_exists){ # do some thing`	应用程序开发人员

Task	描述	所需技能

	}
	else {
	# do something else
	}	

	SQL Server： my $sql_qry = “SELECT OBJECT_ID('$backen dTable', 'U') table_exi sts", undef, 'writer') ";	
	PostgreSQL： my $sql_qry = “SELECT TO_REGCLASS('$back endTable', 'U') table_exists", undef, 'writer')";	
	以下示例在内联 SQL 中使用一个 Perl 变量，该变量使用带有 JOIN 的 SELECT 语句来获取表的主键和键列的位置。 SQL Server： my $sql_qry = "SELECT column_name', character_maxi mum_length \ FROM INFORMATION_SCHEMA .COLUMNS \ WHERE TABLE_SCH EMA='$example_sche maInfo' \ AND TABLE_NAME='$examp le_table' \	

Task	描述	所需技能
	AND DATA_TYPE IN ('varchar','nvarchar');"; PostgreSQL： my $sql_qry = "SELECT c1.column_name, c1.ordinal_position \ FROM information_schema.key_column_usage AS c LEFT \ JOIN information_schema.table_constraints AS t1 \ ON t1.constraint_name = c1.constraint_name \ WHERE t1.table_name = $example_schemaInfo'.'$example_table' \ AND t1.constraint_type = 'PRIMARY KEY' ;";	

对基于 Perl 或 Python 的应用程序进行其他更改以支持 PostgreSQL

Task	描述	所需技能
将其他 SQL Server 结构转换至 PostgreSQL。	以下更改适用于所有应用程序，无论编程语言如何。 • 使用新的、适当的模式名称来限定应用程序使用的数据库对象。 • 使用PostgreSQL 中的排序规则功能处理LIKE运算符进行区分大小写的匹配。	应用程序开发人员

Task	描述	所需技能
	• 处理不支持的数据库特定函数 DATEDIFF，例如 DATEADD、GETDATE、CONVE 和 CAST 运算符。有关与 PostgreSQL 兼容的等效函数，请参阅其他信息部分中的原生或内置 SQL 函数。 • 处理比较语句中的布尔值。 • 处理函数返回值。这可能是记录集、数据框、变量和布尔值。根据应用程序的要求处理这些问题和支持 PostgreSQL。 • 使用新的用户定义的 PostgreSQL 函数处理匿名块（例如 BEGIN TRAN）。 • 转换行批量插入。与从应用程序内部调用的 SQL Server 批量复制 (bcp) 实用程序的 PostgreSQL 等效项是 COPY。 • 转换列连接运算符。SQL Server 使用字符串连接 +，但 PostgreSQL 使用字符串连接 ||。	

提高性能。

Task	描述	所需技能
利用 Amazon Web Services 提高性能。	迁移至 Amazon Web Services Cloud 时，您可以完善应用程序和数据库设计以利用	应用程序开发人员、云架构师

Task	描述	所需技能
	Amazon Web Services。例如，如果来自连接到 Aurora PostgreSQL 兼容数据库服务器的 Python 应用程序的查询比原始 Microsoft SQL Server 查询花费更多时间，您可以考虑将历史数据直接创建到 Amazon Simple Storage 来自 Amazon Simple Storage Service (Amazon S3) 存储桶，并使用基于 Amazon Athena 的 SQL 查询为用户控制面板生成报告和分析数据查询。	

相关资源

- Perl
- Perl DBI 模块
- Python
- psycopg2
- SQLAlchemy
- 批量复制 - PostgreSQL
- 批量复制 — Microsoft SQL Server
- PostgreSQL
- 使用 Amazon Aurora PostgreSQL

其他信息

Microsoft SQL Server 和 Aurora PostgreSQL 兼容均符合 ANSI SQL。但是，在将 Python 或 Perl 应用程序从 SQL Server 迁移至 PostgreSQL 时，您仍应注意语法、列数据类型、本地数据库专用函数、批量插入和区分大小写等方面的任何不兼容之处。

以下部分提供有关每个不一致地方的更多信息。

数据类型比较

从 SQL Server 到 PostgreSQL 的数据类型更改可能会导致应用程序操作的结果数据出现显着差异。有关数据类型的比较，请参阅 Sqlines 网站的表格。

原生或内置 SQL 函数

SQL Server 和 PostgreSQL 数据库之间的某些函数的行为有所不同。下表提供了对比。

Microsoft SQL Server	描述	PostgreSQL
`CAST`	将值从一个数据类型转换为另一个数据类型。	PostgreSQL `type ::` `operator`
`GETDATE()`	以某种 `YYYY-MM-DD hh:mm:ss.mmm` 格式返回当前数据库系统的日期和时间。	`CLOCK_TIMESTAMP`
`DATEADD`	为日期添加时间/日期间隔。	`INTERVAL` 表达式
`CONVERT`	将值转换为特定数据格式。	`TO_CHAR`
`DATEDIFF`	返回两个日期字段相差的天数。	`DATE_PART`
`TOP`	限制 `SELECT` 结果集中的行数。	`LIMIT/FETCH`

匿名区块

结构化 SQL 查询分为声明、可执行文件以及异常处理等部分。下表比较了 Microsoft SQL Server 和 PostgreSQL 版本的简单匿名块。对于复杂匿名块，我们建议您在应用程序中调用自定义数据库函数。

Microsoft SQL Server

```
my $sql_qry1=
my $sql_qry2 =
```

PostgreSQL

```
my $sql_qry1=
my $sql_qry2 =
```

```
my $sqlqry = "BEGIN TRAN
$sql_qry1 $sql_qry2
if @\@error !=0 ROLLBACK
TRAN
else COMIT TRAN";
```

```
my $sql_qry = " DO \$\$
BEGIN
$header_sql $content_sql
END
\$\$";
```

其他区别

- 批量插入行： Microsoft SQL Server bcp utility 实用程序的 PostgreSQL 等效工具是 COPY。
- 区分大小写：在 PostgreSQL 中，列名区分大小写，因此您必须将 SQL Server 列名转换为小写或大写。当您提取或比较数据，或者将列名放置在结果集或变量中时，这将成为一个因素。以下示例标识了可能以大写或小写形式存储值列。

```
my $sql_qry = "SELECT $record_id FROM $exampleTable WHERE LOWER($record_name) =
 \'failed transaction\'";
```

- 串联：SQL Server 使用 + 作为字符串连接的运算符，而 PostgreSQL 则使用 | | 。
- 验证：在 PostgreSQL 的应用程序代码中使用内联 SQL 查询和函数前，应对其进行测试和验证。
- ORM 库的包含：你也可以寻找包含现有数据库连接库，或者用 PynoModb 等 Python ORM 库替换现有数据库连接库。SQLAlchemy这将有助于使用面向对象的范例轻松地查询和操作数据库中的数据。

将数据从 IBM Db2、SAP、Sybase 和其他数据库流式传输到 MongoDB Atlas AWS

由 Battulga Purevragchaa (AWS)、Babu Srinivasan (MongoDB) 和 Igor Alekseev (AWS) 创作

摘要

此模式描述了将数据从 IBM Db2 和其他数据库（例如大型机数据库和 Sybase）迁移到 MongoDB Atlas 的步骤。 AWS Cloud它用于AWS Glue帮助加快向 MongoDB Atlas 的数据迁移。

该模式与规范性指导网站上的《迁移到 MongoDB Atlas AWS》 AWS 指南一起出现。它提供了该指南中讨论的其中一个迁移场景的实施步骤。有关其他迁移方案，请参阅 AWS 规范性指导网站上的以下模式：

- 将自托管 MongoDB 环境迁移到 MongoDB Atlas AWS
- 将关系数据库迁移到 MongoDB Atlas AWS

该模式适用于 Manage AWS d Services 合作伙伴和 AWS 用户。

先决条件和限制

先决条件

- 要迁移到 MongoDB Atlas 的源数据库，例如 SAP、Sybase、IBM Db2 等。
- 熟悉 SAP、Sybase、IBM Db2、MongoDB Atlas 等数据库。 AWS 服务

产品版本

- MongoDB 版本 5.0 或更高版本。

架构

下图说明了使用 AWS Glue Studio Amazon Kinesis Data Streams 和 MongoDB Atlas 进行批量数据加载和数据流式传输。

此参考架构用于创建 AWS Glue Studio 提取、转换和加载 (ETL) 管道，以将数据迁移到 MongoDB Atlas。 AWS Glue 爬网程序 与 MongoDB Atlas 集成，以促进数据治理。数据可以批量移植，也可以使用 Amazon Kinesis Data Streams 流式传输到 MongoDB Atlas。

批量数据加载

有关批量数据迁移的更多信息，请参阅 AWS 博客文章使用 M ongoDB Atlas 编写 ETL 作业。 AWS Glue

数据流

有关支持不同使用场景的 MongoDB Atlas 参考架构，请参阅规范性指南网站上的迁移到 MongoDB Atlas。 AWS AWS

工具

- AWS Glue是一项完全托管的 ETL 服务。它可以帮助您在数据存储和数据流之间对数据进行可靠地分类、清理、扩充和移动。
- Amazon Kinesis D ata Streams 可帮助您实时收集和处理大量数据记录流。
- MongoDB Atlas 是一种完全托管的数据库即服务 (DBaaS)，用于在云中部署和管理 MongoDB 数据库。

最佳实践

有关指南，请参阅 MongoDB 存储库中的 MongoDB 最佳实践指南。 GitHub

操作说明

发现与评测

Task	描述	所需技能
确定集群大小。	使用中的信息计算总索引空间，估算工作集大小。`db.stats()` 假设您的数据空间中有一定比例会被频繁访问。或者，您可以根据自己的假设来估算您的内存需求。此任务大约需要一周时间。有关此故事和本长篇故事中其他	MongoDB 数据库管理员，应用程序架构师

Task	描述	所需技能
	故事的更多信息和示例，请参阅 “相关资源” 部分中的链接。	
估计网络带宽要求。	要估计网络带宽要求，请将平均文档大小乘以每秒提供的文档数量。考虑集群中任何节点将承受的最大流量作为基础。要计算从集群到客户端应用程序的下游数据传输速率，请使用一段时间内返回的文档总数的总和。如果您的应用程序从辅助节点读取数据，请将文档总数除以可以提供读取操作的节点数。要查找数据库的平均文档大小，请使用`db.stats().avgObjSize` 命令。此任务通常需要一天的时间。	MongoDB 数据库管理员
选择 Atlas 层。	按照 MongoDB 文档中的说明选择正确的 Atlas 集群层。	MongoDB 数据库管理员
为转换做好计划。	制定应用程序割接计划。	MongoDB 数据库管理员，应用程序架构师

在 AWS 上设置新的 MongoDB Atlas 环境

Task	描述	所需技能
在上创建新的 MongoDB Atlas 集群。 AWS	在 MongoDB Atlas 中，选择构建集群，然后 AWS 选择作为云提供商。	MongoDB 数据库管理员
选择 AWS 区域 和全局群集配置。	从适用于您的 Atlas 集群 AWS 区域 的列表中进行选择。如果需要，请配置全局集群。	MongoDB 数据库管理员

Task	描述	所需技能
选择集群层。	选择您的首选集群层。您的层选择决定了内存、存储和 IOPS 规格等因素。	MongoDB 数据库管理员
配置其他集群设置。	配置其他集群设置，例如 MongoDB 版本、备份和加密选项。有关这些选项的更多信息，请参阅相关资源部分。	MongoDB 数据库管理员

配置安全性和合规性

Task	描述	所需技能
配置访问列表。	要连接到 Atlas 集群，您必须在项目的访问列表中添加一个条目。Atlas 使用传输层安全性协议（TLS）/安全套接字层 (SSL) 来加密数据库与虚拟私有云（VPC）的连接。要设置项目的访问列表以及有关此长篇故事的更多信息，请参阅 “相关资源” 部分中的链接。	MongoDB 数据库管理员
对用户进行身份验证和授权。	您必须创建并验证将访问 MongoDB Atlas 集群的数据库用户。要访问项目中的集群，用户必须属于该项目，并且他们可以属于多个项目。 您也可以使用 AWS Identity and Access Management (IAM) 启用授权。有关更多信息，请参阅 MongoDB 文档中的使用 IAM 设置身份验证。	MongoDB 数据库管理员

Task	描述	所需技能
创建自定义角色。	（可选）如果内置的 Atlas 数据库用户权限未涵盖您想要的权限集，Atlas 支持创建自定义角色。	MongoDB 数据库管理员
设置 VPC 对等连接。	（可选）Atlas 支持与其他 AWS 的 VPCsVPC 对等。	MongoDB 数据库管理员
设置 AWS PrivateLink 终端节点。	（可选）您可以使用在上 AWS 设置私有终端节点AWS PrivateLink。	MongoDB 数据库管理员
启用双因素身份验证。	（可选）Atlas 支持双因素身份验证 (2FA)，以帮助用户控制对其 Atlas 账户的访问。	MongoDB 数据库管理员
使用 LDAP 设置用户身份验证和授权。	（可选）Atlas支持使用轻量级目录访问协议 (LDAP) 执行用户身份验证和授权。	MongoDB 数据库管理员
设置统一 AWS 接入。	（可选）某些 Atlas 功能（包括 Atlas Data Lake 和使用客户密钥管理进行静态加密）使用 IAM 角色进行身份验证。	MongoDB 数据库管理员
使用设置静态加密 AWS KMS。	（可选）Atlas 支持使用 AWS Key Management Service (AWS KMS) 加密存储引擎和云提供商备份。	MongoDB 数据库管理员
设置 CSFLE。	（可选）Atlas 支持客户端字段级加密 (CSFLE)，包括字段的自动加密。	MongoDB 数据库管理员

迁移数据

Task	描述	所需技能
在 MongoDB Atlas 中启动目标副本集。	在 MongoDB Atlas 中启动目标副本集。在 Atlas Live 迁移服务中，选择我已准备好迁移。	MongoDB 数据库管理员
AWS Glue 与 MongoDB Atlas 建立连接。	使用连接 AWS Glue MongoDB Atlas（目标数据库）。 AWS Glue 爬网程序 此步骤有助于为迁移做好目标环境的准备。有关更多信息，请参阅 AWS Glue 文档。	MongoDB 数据库管理员
AWS Glue 与源数据库或源流建立连接。	这有助于为迁移做好目标环境的准备。	MongoDB 数据库管理员
设置数据转换。	配置转换逻辑，将数据从传统结构化架构迁移到 MongoDB 的灵活架构。	MongoDB 数据库管理员
迁移数据。	在中安排迁移 AWS Glue Studio。	MongoDB 数据库管理员

配置操作集成

Task	描述	所需技能
Connect 连接到集群。	连接到 MongoDB Atlas 集群。	应用程序开发人员
与数据交互。	与集群数据交互。	应用程序开发人员
监控集群。	监控您的 MongoDB Atlas 集群。	MongoDB 数据库管理员
备份和恢复数据。	备份和恢复集群数据。	MongoDB 数据库管理员

故障排除

事务	解决方案
如果你遇到问题	参见 Mongo CloudFormation DB Atlas 资源库中的疑难解答。

相关资源

除非另有说明，否则以下所有链接均指向 MongoDB 文档中的网页。

迁移指南

- 在（规范性指南）上迁移到 MongoDB Atla AWS sAWS

发现与评测

- 内存
- 使用 Atlas 示例数据集进行大小调整示例
- 移动应用程序的大小调整示例
- 网络流量
- 集群自动扩缩
- Atlas 大小调整模板

配置安全性和合规性

- 配置 IP 访问列表条目
- 配置数据库用户
- 配置对 Atlas 用户界面的访问权限
- 配置自定义数据库角色
- 配置数据库用户
- 设置网络对等连接
- 在 Atlas 中了解私有端点
- 管理您的多重身份验证选项

- 使用 LDAP 设置用户身份验证和授权
- Atlas 数据湖
- 使用客户密钥管理进行静态加密
- 代入角色的方法（IAM 文档）
- 客户端字段级加密
- 自动加密
- MongoDB Atlas 安全控制
- MongoDB 信任中心
- 为集群配置安全功能

在上设置新的 MongoDB Atlas 环境 AWS

- 云提供商和地区
- 管理全球集群
- 选择集群层
- 配置其他设置
- 开始使用 Atlas
- 配置对 Atlas 用户界面的访问权限
- 管理集群

迁移数据

- 迁移或导入数据

监控集群

- 监控您的集群

集成操作

- 连接到集群
- 与您的数据互动
- 监控您的集群

- 备份、恢复和存档数据

GitHub 存储库

- 使用以下命令将数据流式传输到 MongoDB Atlas AWS Glue

按照工作负载分类的迁移模式

主题

- IBM
- Microsoft
- 不适用
- 开源
- Oracle
- SAP

IBM

- 使用 AWS DMS 将 Db2 数据库从亚马逊迁移 EC2 到兼容 Aurora MySQL
- 使用日志传送将适用于 LUW 的 Db2 迁移 EC2 到 Amazon 以减少中断时间
- 将适用于 LUW 的 Db2 迁移到 EC2 具有高可用性灾难恢复功能的 Amazon
- 使用 AWS DMS 和 AWS SCT 从亚马逊上的 IBM Db2 迁移 EC2 到兼容 Aurora PostgreSQL
- 在 Amazon 上从 IBM WebSphere 应用程序服务器迁移到 Apache Tomcat EC2
- 将数据从 IBM Db2、SAP、Sybase 和其他数据库流式传输到 MongoDB Atlas AWS

Microsoft

- 加快 Microsoft 工作负载的发现和迁移到 AWS
- 更改 Python 和 Perl 应用程序以支持数据库从Microsoft SQL Server 迁移至兼容 Amazon Aurora PostgreSQL 的版本
- 使用微软 Excel 和 Python 为 AWS DMS 任务创建 AWS CloudFormation 模板
- 使用 AWS DMS 将 Microsoft SQL Server 数据库导出至 Amazon S3
- 从 SQL Server 迁移到 PostgreSQL 时，对 PII 数据实施 SHA1 哈希处理
- 将 EC2 Windows 实例提取并迁移到 AWS Managed Services 账户
- 将消息队列从 Microsoft Azure 服务总线迁移到 Amazon SQS
- 使用 AWS DMS 将微软 SQL Server 数据库从亚马逊迁移 EC2 到亚马逊 DocumentDB
- 使用 AWS DMS 和 AWS SCT 将 Microsoft SQL Server 数据库迁移到 Aurora MySQL
- 将 .NET 应用程序从 Microsoft Azure 应用服务迁移到 AWS Elastic Beanstalk
- 将本地微软 SQL Server 数据库迁移到亚马逊 EC2
- 将本地 Microsoft SQL Server 数据库迁移至 Amazon RDS for SQL Server
- 使用链接服务器将本地 Microsoft SQL Server 数据库迁移至 Amazon RDS for SQL Server
- 使用本机备份和还原将本地 Microsoft SQL Server 数据库迁移到 Amazon RDS for SQL Server。
- 使用 AWS DMS 将本地 Microsoft SQL Server 数据库迁移至 Amazon Redshift
- 使用 AWS SCT 数据提取代理将本地 Microsoft SQL Server 数据库迁移至 Amazon Redshift
- 将本地微软 SQL Server 数据库迁移到 EC2 运行 Linux 的亚马逊上的微软 SQL Server
- 使用 Rclone 将数据从 Microsoft Azure Blob 迁移至 Amazon S3
- 将关系数据库迁移到 MongoDB Atlas AWS
- 使用 ACM 将 Windows SSL 凭证迁移到应用程序负载均衡器
- 在 Amazon Web Services Cloud 中重新托管本地工作负载：迁移核对清单
- 解决将 Microsoft SQL Server 迁移至 Amazon Web Services Cloud 后出现的连接错误
- 使用 Amazon 为 SQL Server Always On FCI 设置多可用区基础架构 FSx

不适用

- 在重新托管迁移到的过程中，为防火墙请求创建批准流程 AWS

开源

- 在 Aurora PostgreSQL 兼容中创建应用程序用户和角色
- 使用并使用和，将 Amazon RDS for Oracle 迁移到适用于 PostgreSQL AWS SCT 的 Amazon RDS AWS DMSAWS CLIAWS CloudFormation
- 使用原生工具将本地 MariaDB 数据库迁移至 Amazon RDS for MariaDB
- 将本地 MySQL 数据库迁移到亚马逊 EC2
- 将本地 MySQL 数据库迁移至 Amazon RDS for MySQL
- 将本地 MySQL 数据库迁移至 Aurora MySQL
- 将本地 PostgreSQL 数据库迁移到 Aurora PostgreSQL
- 将 Couchbase 服务器数据库迁移到亚马逊 EC2
- 使用 Auto Scaling 从 IBM WebSphere 应用程序服务器迁移到亚马逊 EC2 上的 Apache Tomcat
- 从 Oracle 迁移 GlassFish 到 AWS Elastic Beanstalk
- 使用 pglogical 从 EC2 亚马逊上的 PostgreSQL 迁移到适用于 PostgreSQL 的亚马逊 RDS
- SageMaker 使用 AWS 开发人员工具将 ML 构建、训练和部署工作负载迁移到 Amazon
- 使用 AWS App2Container 将本地 Java 应用程序迁移到 AWS
- 使用 Percona、A XtraBackup mazon EFS 和 Amazon S3 将本地 MySQL 数据库迁移到 Aurora MySQL
- 将 Oracle 外部表迁移到 Amazon Aurora PostgreSQL-Compatible
- 将 Redis 工作负载迁移至 Redis Enterprise Cloud on AWS
- 重启 RHEL 源服务器 SELinux 后，无需禁用 AWS 复制代理即可自动重启 AWS 复制代理
- 使用 pg_transport 在两个 Amazon RDS 数据库实例之间传输 PostgreSQL 数据库

Oracle

- 将 Oracle 的 VARCHAR2 (1) 种数据类型转换为亚马逊 Aurora PostgreSQL 的布尔数据类型
- 通过使用兼容 PostgreSQL 的 Aurora 全局数据库来模拟 Oracle 灾难恢复
- 使用 Oracle SQL Developer 和 AWS SCT 以增量方式从 Amazon RDS for Oracle 迁移至 Amazon RDS for PostgreSQL
- 使用兼容 Aurora PostgreSQL 的文件编码将 BLOB 文件加载至文本中
- 使用 AWS DMS 在 SSL 模式下将 Amazon RDS for Oracle 迁移到 Amazon RDS for PostgreSQL
- 将 Amazon RDS for Oracle 数据库迁移到另一个 AWS 区域 数据库 AWS 账户 并 AWS DMS 用于持续复制
- 将 Amazon RDS for Oracle 数据库实例迁移至另一个 VPC
- 使用 Oracle 数据泵将本地 Oracle 数据库迁移到亚马逊 EC2
- 使用 Logstash 将本地 Oracle 数据库迁移到亚马逊 OpenSearch 服务
- 使用 AWS DMS 和 AWS SCT 将本地 Oracle 数据库迁移至 Amazon RDS for MySQL
- 将本地 Oracle 数据库迁移到 Amazon RDS for Oracle
- 使用通过数据库链接直接导入 Oracle Data Pump 将本地 Oracle 数据库迁移到 Amazon RDS for Oracle
- 使用 Oracle 数据泵将本地 Oracle 数据库迁移到 Amazon RDS for Oracle
- 使用 Oracle Bystander 和 AWS DMS 将本地 Oracle 数据库迁移到 Amazon RDS for PostgreSQL
- 将本地 Oracle 数据库迁移到亚马逊上的 Oracle EC2
- 使用 AWS DMS 和 AWS SCT 将 Oracle 数据库从亚马逊迁移 EC2 到适用于 MariaDB 的亚马逊 RDS
- 使用 AWS DMS 将 Oracle 数据库从亚马逊迁移 EC2 到 Amazon RDS for Oracle
- 使用 AWS DMS 将 Oracle 数据库迁移至 Amazon DynamoDB
- 使用 Oracle GoldenGate 平面文件适配器将 Oracle 数据库迁移到 Amazon RDS for Oracle
- 使用 AWS DMS 和 AWS SCT 将 Oracle 数据库迁移至 Amazon Redshift
- 使用 AWS DMS 和 AWS SCT 将 Oracle 数据库迁移至 Aurora PostgreSQL
- 使用 Oracle 数据泵和 AWS DMS 将 Oracle JD Edwards EnterpriseOne 数据库迁移到 AWS
- 使用 AWS DMS 将 Oracle 分区表迁移到 PostgreSQL
- 使用 AWS DMS 将 Oracle PeopleSoft 数据库迁移到 AWS
- 将数据从本地 Oracle 数据库迁移到 Aurora PostgreSQL
- 从 Amazon RDS for Oracle 迁移到 Amazon RDS for MySQL

- 使用实体化视图和 AWS DMS 从 Oracle 8i 或 9i 迁移至 Amazon RDS for PostgreSQL
- 使用和 AWS DMS 从 Oracle 8i 或 9i 迁移到适用于 SharePlex PostgreSQL 的亚马逊 RDS
- 使用 Oracle 从 Oracle 数据库迁移到 Amazon RDS for PostgreSQL GoldenGate
- 使用 AWS DMS 和 AWS SCT 从亚马逊上的 Oracle 迁移 EC2 到适用于 MySQL 的亚马逊 RDS
- 使用 AWS DMS 从 Oracle 迁移至 Amazon DocumentDB
- 在 Amazon ECS 上从 Oracle 迁移 WebLogic 到 Apache Tomcat (ToMee)
- 将基于函数的索引从 Oracle 迁移到 PostgreSQL
- 将遗留应用程序从 Oracle Pro*C 迁移到 ECPG
- 将 Oracle CLOB 值迁移到 AWS 上 PostgreSQL 中的单独的行
- 将 Oracle 数据库错误代码迁移到 Amazon Aurora PostgreSQL-Compatible 数据库
- 将 Oracle 电子商务套件迁移到 Amazon RDS Custom
- 使用扩展将 Oracle 原生函数迁移到 PostgreSQL
- 将 Oracle 迁移 PeopleSoft 到亚马逊 RDS 定制版
- 将 Oracle ROWID 功能迁移到 AWS 上的 PostgreSQL
- 将 Oracle SERIALLY_REUSABLE pragma 包迁移至 PostgreSQL
- 将虚拟生成的列从 Oracle 迁移至 PostgreSQL
- 在 Aurora PostgreSQL-Compatible 上设置 Oracle UTL_FILE 功能
- 从 Oracle 迁移至 Amazon Aurora PostgreSQL 后验证数据库对象

SAP

- 将本地 SAP ASE 数据库迁移到亚马逊 EC2
- 使用 AWS DMS 从 SAP ASE 迁移至 Amazon RDS for SQL Server
- 使用 AWS SCT 和 AWS DMS EC2 将亚马逊上的 SAP ASE 迁移到兼容 Aurora PostgreSQL 的亚马逊 Aurora
- 使用 Application Migration Service 缩短同构 SAP 迁移割接时间

更多模式

- 通过安装， AWS 服务 从 IBM z/OS 进行访问 AWS CLI
- 使用 CAST Highlight 评测迁移至 Amazon Web Services Cloud 的应用程序就绪情况
- 评测将 SQL Server 数据库迁移至 MongoDB Atlas on AWS 的查询性能
- 使用 DR Orchestrator 框架自动执行跨区域故障转移和故障恢复
- 使用 AWS Lambda 任务计划程序在亚马逊 EC2 上运行的 SQL Server Express 版中自动执行数据库任务
- 在 Amazon Web Services Cloud 中构建高级大型机文件查看器
- 使用混合链接模式将数据中心扩展配置为 VMware Cloud on AWS
- 通过私有网络连接到 Application Migration Service 数据和控制面板
- 对经过 Blu Age 现代化改造的大型机工作负载进行容器化
- 将 JSON Oracle 查询转换至 PostgreSQL 数据库 SQL
- 将 Teradata 标准化时态功能转换为 Amazon Redshift SQL
- 将 Teradata RESET WHEN 功能转换为 Amazon Redshift SQL
- 使用跨账户复制 Amazon DynamoDB 表 AWS Backup
- 使用私有静态在 Amazon EC2 上部署 Cassandra 集群 IPs 以避免重新平衡
- 使用 AWS CDK 部署多堆栈应用程序 TypeScript
- 使用 Aurora PostgreSQL 中的自定义端点模拟 Oracle RAC 工作负载
- 使用 AWR 报告估计 Oracle 数据库的 Amazon RDS 引擎大小
- 使用 AWS 大型机现代化和 Amazon Q 生成数据见解 QuickSight
- 在 Aurora PostgreSQL 中处理动态 SQL 语句中的匿名块
- 在 Aurora PostgreSQL 兼容中处理重载的 Oracle 函数
- 迁移到 Amazon ECR 存储库时自动识别重复的容器映像
- 在 AWS 上将 VMware vRealize 网络洞察与 VMware 云集成
- 将 Amazon RDS for Oracle 数据库实例迁移到使用 AMS 的其他账户
- 使用将本地 Apache Kafka 集群迁移到亚马逊 MSK MirrorMaker
- 使用 AWS Glue 将 Apache Cassandra 工作负载迁移到亚马逊密钥空间
- 将你的容器工作负载从 Azure 红帽 OpenShift (ARO) 迁移到 AWS 云端 Red Hat OpenShift 服务 (ROSA)
- 使用和 AWS DMS 从 Oracle 8i 或 9i 迁移到适用于 Oracle 的 Amazon RD SharePlex S

- 使用迁移器将 Hadoop 数据迁移到 Amazon S3 WANdisco LiveData
- 将含有 100 多个参数的 Oracle 函数和过程迁移到 PostgreSQL
- 将 Oracle OUT 绑定变量迁移到 PostgreSQL 数据库
- 使用具有相同主机名的 SAP HSR 将 SAP HANA 迁移至 AWS
- 使用分布式可用性组将 SQL Server 迁移至 AWS
- 使用 HCX 操作系统辅助迁移在 AWS 上迁移 VMs 到 VMware 云端
- 使用 Micro Focus 企业服务器和 LRS VPSX/MFI 在 AWS 上实现大型机在线打印工作负载的现代化
- 使用 Rocket Enterprise Server 和 L PageCenter RS X 实现大型机输出管理的现代化 AWS
- 在 AWS 上从 F5 迁移到应用程序负载均衡器时修改 HTTP 标头
- 使用 VMware Aria 日志操作 AWS 将日志从 VMware Cloud 发送到 Splunk
- 使用 Terraform 为数据库迁移设置 CI/CD 管道
- 使用 AWS 弹性灾难恢复为 Oracle JD Edwar EnterpriseOne ds 设置灾难恢复
- 使用 AWS Private CA 和 AWS RAM 简化私有证书管理
- 以 CSV 文件形式将大规模 Db2 z/OS 数据传输到 Amazon S3

现代化

主题

- 在 CAST Imaging 中分析和可视化软件架构
- 使用 CAST Highlight 评测迁移至 Amazon Web Services Cloud 的应用程序就绪情况
- 使用 DynamoDB TTL 自动将项目归档到 Amazon S3
- 在 Amazon 服务中构建多租户无服务器架构 OpenSearch
- 使用 AWS CDK 部署多堆栈应用程序 TypeScript
- 使用 AWS SAM 自动部署嵌套应用程序
- 使用 AWS Lambda 令牌售卖机为 Amazon S3 实施 SaaS 租户隔离
- 使用 AWS Step Functions 实施无服务器 saga 模式
- 通过使用 AWS CDK 设置 Amazon ECS Anywhere 来管理本地容器应用程序
- 在 AWS 上实现 ASP.NET Web 表单应用程序的现代化
- 使用 AWS Fargate 大规模运行事件驱动型和计划性工作负载
- 使用 C# 和 AWS CDK 在 SaaS 架构中为孤岛模型进行租户登录
- 使用 CQRS 和事件溯源将整体分解为微服务
- 更多模式

在 CAST Imaging 中分析和可视化软件架构

由 Arpita Sinha (Cast Software) 和 James Hurrell (Cast Software) 创建

摘要

此模式展示了如何使用 CAST Imaging 直观地导航复杂的软件系统，并对软件结构进行精确分析。通过以这种方式使用 CAST Imaging，您可以对应用程序的架构做出更明智的决策，特别是出于现代化目的。

要在 CAST Imaging 中查看应用程序的架构，您必须首先通过 CAST 控制台加载应用程序的源代码。然后，控制台将应用程序的数据发布到 CAST Imaging，您可以在其中逐层可视化和导航应用程序架构。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 用于 CAST Imaging 的亚马逊机器映像（AMI）
- 包含以下内容的亚马逊弹性计算云 (Amazon EC2) 实例（建议使用内存优化的 r5.xlarg e EC2 亚马逊实例）：
  - 4 个 vCPU
  - 32 GB RAM
  - 最低 500 GB 通用型固态硬盘 (SSD) (gp3) 卷
- CAST 控制台和 CAST Imaging 许可证密钥（要获取所需许可证密钥，请通过 aws.contact-me@castsoftware.com 联系 CAST）
- 要以压缩 (.zip) 格式分析的应用程序的完整源代码
- Microsoft Edge、Mozilla Firefox 或 Google Chrome

架构

下图显示了通过 CAST 控制台载入应用程序源代码，然后在 CAST Imaging 中查看应用程序源代码的示例工作流：

图表显示了以下工作流：

1. CAST 通过对前端、中间件和后端代码进行逆向工程来生成应用程序源代码元数据。
2. CAST 生成的应用数据会自动导入到 CAST Imaging 中，并可在其中进行可视化和分析。

以下是此过程工作原理的快照：

工具

- CAST Imaging 是一款基于浏览器的应用程序，可帮助您直观地查看和导航软件系统，以便您可以就其架构做出明智的决策。
- CAST 控制台是一个基于浏览器的应用程序，可帮助您配置、运行和管理 CAST AIP 分析。

> **Note**
>
> CAST Imaging 和 CAST 控制台包含在 CAST 映像的 AMI 中。

操作说明

设置 CAST Imaging 环境

Task	描述	所需技能
运行初始 CAST 控制台配置。	1. 打开 Web 浏览器，然后输入以下 URL 连接到 CAST 控制台：http://localhost:8081 2. 出现提示时，输入您的 CAST 控制台许可证密钥。然后选择下一步。 3. 审核配置设置。如果不需要进行任何更改，请选择保存并完成。	软件架构师、开发人员、技术主管
运行初始 CAST Imaging 配置。	1. 打开 Web 浏览器，然后输入以下 URL 连接到 CAST	软件架构师、开发人员、技术主管

Task	描述	所需技能
	Imaging：http://localhost:8083 2. 出现提示时，通过输入 admin 作为用户名和密码登录。 3. 出现提示时，输入您的 CAST Imaging 许可证密钥。然后，选择更新以保存密钥。	

Task	描述	所需技能
配置 CAST Extend 本地服务器。	（可选）默认情况下，CAST Extend 本地服务器配置为在脱机模式下运行。如果这是可以接受的，则无需进行其他配置。但是，如果您希望在联机/代理模式下配置 CAST Extend 本地服务器，并直接连接到 CAST Extend，请按照下列步骤操作。 **Note** 有关 CAST 扩展凭证，请参阅 CAST 扩展注册页面。 1. 使用桌面上的 CAST Extend 管理中心快捷方式加载 Web 浏览器并连接到 CAST Extend 本地服务器。 2. 选择在线选项。 3. 输入您的 CAST Extend 凭证（电子邮件和密码），然后选择保存以完成该过程。	软件架构师、开发人员、技术主管

将您的应用程序载入 CAST Imaging

Task	描述	所需技能
为您的应用程序准备源代码。	将应用程序的源代码保存在单个压缩的 .zip 文件中。	软件架构师、开发人员、技术主管

Task	描述	所需技能
将您的应用程序添加到 CAST 控制台。	1. 打开 Web 浏览器，然后输入以下 URL 连接到 CAST 控制台：http://localhost:8081 2. 出现提示时，通过输入 admin 作为用户名和密码登录。 3. 选择添加应用程序。然后，输入应用程序名称并选择添加。	软件架构师、开发人员、技术主管
打开源代码交付向导。	在 CAST 控制台中找到您创建的应用程序。然后，选择添加版本。	软件架构师、开发人员、技术主管
上传您的应用程序源代码。	请执行以下操作之一： • 将包含应用程序源代码的 .zip 文件拖放到源代码交付向导中。– 或– • 选择上传云图标。然后，打开包含应用程序源代码的 .zip 文件。	软件架构师、开发人员、技术主管

Task	描述	所需技能
启动分析过程。	1. 在交付向导中，提供版本详细信息并指定配置选项。有关详细信息，请参阅 CAST Imaging 文档中的 CAST Imaging 的标准载入。 2. 确保选中发布到 CAST Imaging 选项。然后，选择继续。 **Note** 选择 “继续” 将启动源代码的分析过程。CAST 控制台中的进度窗口显示分析过程的每个步骤，并在分析完成时显示通知。	软件架构师、开发人员、技术主管

验证发布到 CAST Imaging 的分析结果和数据

Task	描述	所需技能
检查状态和日志。	当所有分析操作完成后，验证进度窗口中是否有成功消息。 **Note** 每个分析操作完成后，您可以立即查看每个分析操作的各个日志。要查看特定操作的日志，	软件架构师、开发人员、技术主管

Task	描述	所需技能
	请在进度窗口中选择查看日志。	
检查应用程序详细信息。	在应用程序详细信息面板中，查看有关分析结果的详细信息。请务必查看已发现的技术和源代码组织。	软件架构师、开发人员、技术主管
验证并访问 CAST Imaging。	1. 在 CAST 控制台的应用程序管理窗格中，验证应用程序的版本状态是否为映像已处理。此时将显示 CAST Imaging 图标。 2. 选择 CAST Imaging 图标以直接导航到 CAST Imaging 中的应用程序数据。 **Note** 图像处理状态表示源代码已经过分析并上传到您的 CAST Imaging 实例。	软件架构师、开发人员、技术主管

开始使用 CAST Imaging 分析您的应用

Task	描述	所需技能
登录 CAST Imaging。	打开 Cast Imaging 并输入默认管理员凭证（admin/admin）。此时将显示应用程序的数据。	软件架构师、开发人员、技术主管

Task	描述	所需技能
在 CAST Imaging 中探索您的应用程序数据。	使用 CAST Imaging 功能开始查看您的软件架构。 有关如何使用 CAST Imaging 功能的快速教程，请选择帮助图标以显示 CAST Imaging Helper。 有关更多信息，请参阅《CAST Imaging 用户指南》。	软件架构师、开发人员、技术主管

相关资源

CAST 控制台文档

- 登录
- 通过 CAST 控制台配置选项

CAST Imaging 文档

- CAST Imaging 应用程序入门 - 先决条件
- 为 CAST Imaging 添加新应用程序
- CAST Imaging 的标准入门指南 – 检查结果
- 登录
- 配置选项 - 管理中心 GUI

有关 AWS 上的 CAST Imaging 的更多资源

- CAST 加速了 AWS 的应用程序现代化 — 技术（AWS PartnerCast 网络研讨会，需要免费账户）
- 使用 CAST 和 AWS Migration Hub Refactor Spaces 实现传统应用程序现代化（AWS Blog 文章）
- 使用 CAST Imaging 实现应用程序向 AWS 架构的现代化（AWS 研讨会）
- Amazon Web Services Marketplace：CAST Imaging

- AWS 资源上的所有 CAST

使用 CAST Highlight 评测迁移至 Amazon Web Services Cloud 的应用程序就绪情况

由 Greg Rivera (Cast Software) 创建

摘要

CAST Highlight 是一款软件即服务（SaaS）解决方案，用于执行应用程序组合的快速分析。此示例介绍了如何配置和使用 CAST Highlight，以评测组织的 IT 组合中定制软件应用程序的云就绪性，以及如何规划 Amazon Web Services (AWS) Cloud 的现代化和迁移技术。

CAST Highlight 可以深入了解应用程序的云就绪情况，识别迁移前需要删除的代码拦截器，估算移除这些拦截器的技术，并推荐各个应用程序在迁移后可以使用的 Amazon Web Services。

此示例介绍了设置和使用 CAST Highlight 的进程，其中包括五个步骤：新用户设置、应用程序管理、活动管理、源代码分析和结果分析。您必须完成此模式的操作部分所述所有步骤，以确保成功进行应用程序扫描和分析。

先决条件和限制

先决条件

- 具有 Portfolio Manager 许可的活动 CAST Highlight 账号。
- 本地计算机上至少有 300 MB 的可用磁盘空间和 4 GB 的内存，以用于安装 CAST Highlight 本地代理。
- Microsoft Windows 8 或更高版本。
- 您的应用程序源代码必须存储至可从安装本地代理的计算机上访问的文本文件中。没有源代码离开本地，所有代码都是在本地扫描的。

架构

下图说明了 CAST Highligh 的使用工作流。

工作流程由以下步骤组成：

1. 登录 CAST Highlight 门户，下载本地代理，然后将其安装至本地计算机上。Amazon Simple Storage Service (Amazon S3) 存储本地代理安装软件包。

2. 扫描源代码文件并生成结果文件。
3.

> **Important**
>
> 将结果文件上传至 CAST Highlight门户。：结果文件中不包含源代码。

4. 回答您扫描的每个应用程序中的调查问题。
5. 查看 CAST Highlight内容门户中提供的控制面板与报告。Amazon Relational Database Service (Amazon RDS) 存储代码扫描、分析结果和 CAST Highligh 软件数据。

技术堆栈

CAST Highlight 支持以下技术分析应用程序云就绪情况：

- Java
- COBOL
- C#
- C++
- Clojure
- PHP
- JavaScript
- TypeScript
- Python
- Microsoft Transact-SQL
- VB.net
- Kotlin
- Scala
- Swift

自动化和扩缩

- CLI 分析器可用于自动执行 CAST Highlight分析过程。

工具

如满足所有先决条件，则无需使用任何工具。但是，您可选择使用可选工具来管理源代码文件，例如源代码管理 (SCM) 实用程序、代码提取器或其他工具。

操作说明

新用户设置

Task	描述	所需技能
激活您的 CAST Highlight 账户并选择你的密码。	所有首次使用 CAST Highlight 的用户都将会收到一封账户激活电子邮件。点击激活链接，激活您的 CAST Highlight 账户，然后输入密码以完成激活过程。	不适用
登录 CAST Highlight 门户。	输入新密码后，将显示 CAST Highlight 主页。通过您的用户凭证登录 CAST Highlight 门户。	不适用

应用程序管理

Task	描述	所需技能
创建应用程序记录。	在 CAST Highlight 门户中，导航到管理产品组合部分的管理应用程序选项卡。在屏幕顶部的应用程序图块中，选择添加。	不适用
选择应用程序名称。	输入您的应用程序名称，然后选择保存。此名称用于在 CAST Highlight 中的申请记录。	不适用

Task	描述	所需技能
对所有应用程序重复此步骤。	针对您要扫描的每个应用程序重复这些步骤。	不适用

活动管理

Task	描述	所需技能
创建市场活动。	CAST Highlight 使用“活动”描述一组将在特定时间进行分析的应用程序。在 CAST Highlight 门户中，导航到管理组合部分的管理活动选项卡。选择创建活动以启动活动创建屏幕。	不适用
输入名称，并选择活动的截止日期。	输入活动的名称，然后选择活动截止日期。 **Important** 投稿人不能在活动结束日期之后提交申请分析结果。	不适用
决定包含源代码扫描、调查答案以及域和应用程序范围。	选择一个或多个标准调查，以使用定性信息增强源代码分析数据。调查类别为 “业务影响”、“软件维护工作” CloudReady、“应用程序属性” 和 “绿色影响”。选择在活动期间分析的域名与应用程序。	不适用

Task	描述	所需技能
	Important 在开始活动之前，请务必在 “管理应用程序” 部分中添加所有要扫描的应用程序。	
自定义启动消息。	自定义启动消息，该消息将通过电子邮件发送至与活动中的应用程序关联的所有参与者。	不适用
启动活动。	选择完成以启动活动。	不适用

源代码分析

Task	描述	所需技能
下载 CAST Highlight 本地代理。	在 CAST Highlight 门户，选择应用程序扫描并将 Local Agent 下载到您的本地计算机。	不适用
安装 Local Agent。	启动 CASTHighlight Setup.exe 安装程序，然后按照显示的安装说明进行操作。安装 Local Agent 后，就可以分析应用程序了。	不适用
定义本地代理代码扫描范围。	代码分析是在文件级别执行的，不考虑文件间的逻辑链接或依赖关系。所有文件都被视为平等，并且是应用程序的一部分。	不适用

Task	描述	所需技能
	若要提供准确一致的结果，请使用 Local Agent 中提供的文件或文件夹排除功能来准备代码扫描范围。	
包含开源或 COTS 软件包。	（可选）如果要包含开源或商业 off-the-shelf (COTS) 软件包，请确保它们包含在计划扫描的文件夹中。通常，外部库被分组到一个名为“第三方”或类似的子文件夹中，主代码通常位于“src/main”文件夹中。	不适用
排除测试类。	通常将测试类排除在源代码分析以外，因为它们通常不属于已编译的应用程序。不过，如有必要，您可以选择将它们纳入在扫描中。	不适用
排除 SCM、生成以及部署文件夹。	为了获得更一致的结果，应避免在扫描中包含 SCM、生成或部署文件夹（例如 .git 或 .svn 文件）。	不适用
包括依赖项文件。	如果您想深入了解其物理文件是否属于您正在扫描的文件夹的框架和依赖项，请确保包含依赖项文件（例如 pom.xml、build.gradle、package.json 或.vcsproj 文件）。	不适用
调用 Local Agent.。	在本地 Windows 计算机上运行 Local Agent.。	不适用

Task	描述	所需技能
选择包含源代码的文件夹。	选择包含源代码的文件夹。您可添加多个要由本地代理发现的文件夹。尽管 Local Agent. 确实支持通过网络路径发现源代码，但您应确保源文件夹位于本地计算机上。 **Important** 如果您的源文件夹中的文件超过 10,000 个，我们建议您进行多次扫描。	不适用
开始文件发现。	在 Local Agent 控制面板，选择发现文件。Local Agent 会发现您的文件夹和子文件夹中的文件，并检测其技术。您可以随时选择取消按钮以取消发现。 文件发现完成后，本地代理会列出找到的文件夹与文件。技术列显示相关的技术和文件数量。路径列显示文件夹和文件的位置。	不适用

Task	描述	所需技能
完善源代码的扫描配置。	（可选）要优化本地代理扫描，可以对特定文件夹或文件停用一种或多种技术。如果所有技术都已停用，则您的文件夹或文件将排除在扫描范围外。 若要停用技术，请选择要停用的技术的黄色标签。将鼠标悬停至文件或文件夹上时，也可以选择筛选器图标，将技术与特定文件或文件夹相关联。这些设置已保存，以加快文件夹或文件的发现过程。	不适用
开始源代码扫描。	配置扫描后，选择“扫描文件”以开始扫描过程。	不适用
检查绿色或者灰色标签。	源代码扫描完成后，将在文件夹和文件级显示状态标签。 绿色标签表示使用相关技术扫描了正确文件。 灰色标签表示文件未被扫描，并被排除在外。当您将鼠标悬停至每个文件的标签上时，会显示排除它们的原因。排除文件的可能原因包括：二进制文件、无法读取的文件、丢失的文件、外部库、编码的文件、生成的文件、语法错误、不符合预期语言的内容、不符合足够分析标准的代码、超过大小限制 (10 MB) 的文件、超时问题或分析器不可用。	不适用

Task	描述	所需技能
修改扫描配置，并重新扫描代码。	（可选）您可以修改扫描配置设置，然后选择扫描文件 以再次扫描文件。	不适用
确认扫描结果。	如果扫描结果符合您的要求，请选择确认结果。	不适用
查看 Local Agent 找到的框架和软件库。	查看您的应用程序使用或引用的框架和软件库，以及本地代理在代码扫描期间发现的框架和软件库。您可以通过选择单独的割接按钮保留或忽略这些列表中的元素。 选择确认依赖项继续。 **Important** 如果某个框架已关闭，则该框架不会在 CAST Highlight 门户中列出，也不会附加到您的应用程序中。	不适用

Task	描述	所需技能
保存代码扫描结果。	Local Agent 显示按技术分组的代码扫描结果摘要。选择保存，然后指定要将结果保存到的文件夹。Local Agent 每次扫描都会生成一个.zip 文件，其中包含所有分析结果。 根据不同技术和根源文件夹的数量，本地代理会自动生成一个或多个命名结构为.technology.date.csv 的 FolderName.csv 文件。	不适用
将代码扫描结果上传至 CAST Highlight 门户。	在 CAST Highlight 门户中，选择您在应用程序扫描部分中分析的应用程序。选择上传结果，然后选择 .csv 文件。您也可以单独上传 .csv 文件。每个文件上传后，屏幕上将显示上传记录。	不适用
如有需要，请删除分析结果文件。	（可选）在上传过程中，通过选择垃圾桶图标，可随时删除分析结果文件。 ⚠ Important 只有拥有投资组合管理员权限的用户或上传结果的贡献者才能删除结果。	不适用

Task	描述	所需技能
回答申请调查。	调查按钮在需要调查的应用程序上显示。选择调查，回答调查每个部分的问题，完成后选择提交。 您的调查进度将会显示在屏幕顶部。提交所有必填信息后，您就可提交结果。但是，您可通过回答所有问题来，丰富组织的 CAST Highlight 实例数据。	不适用
提交代码扫描结果。	上传应用程序的所有 .csv 结果文件并完成调查问题后，在应用程序扫描部分中选择提交。必须执行此步骤才能完成该过程，并确保结果在 CAST Highlight 门户中可用。	不适用

结果分析

Task	描述	所需技能
查看 CAST Highlight 门户主页。	CAST Highlight 门户主页包含包含有关您的应用程序组合的高级信息的图块 CloudReady，例如软件运行状况以及整个产品组合的开源安全评分。主页还包含已载入应用程序的数量。有关 CAST Highlight 指标定义和衡量方法的更多信息，请参阅 CAST 亮点——指标和方法（Microsoft PowerPoint 演示文稿）。	不适用

Task	描述	所需技能
查看 CloudReady 控制面板。	选择图CloudReady 块以打开 CloudReady 仪表板。这是主要的产品组合级别控制面板，用于评测应用程序的云准备情况。它可帮助您规划和制定云迁移的产品组合路线图	不适用

Task	描述	所需技能
查看 Portfolio Advisor for Cloud 控制面板。	Portfolio Advisor for Cloud 控制面板会自动将应用程序划分为推荐的迁移类别。按每个应用程序的技术特征进行细分。因素包括源代码分析（云就绪性、软件弹性等）和来自调查的业务影响。在右上角，选择计算以生成初始分段建议。 控制面板顶部图表中的气泡代表产品组合中的每个应用程序，按推荐的细分进行组织。图表下方的数据表中还列出了每项应用程序，包括每个应用程序的相关指标。 建议的可能细分包括： • 更换主机 — 建议更改应用程序的基础架构配置，以便使用基础设施即服务 (IaaS) 解决方案将其直接迁移到云端。 • 重构 — 建议在不更改架构或功能的情况下对应用程序代码进行适度修改，以便可以使用容器即服务 (CaaS) 或平台即服务（PaaS）解决方案对其进行迁移。 • 重新架构 — 建议大幅修改应用程序代码以改善应用程序的运行状况，并使用 PaaS 解决方案为迁移做好准备，或者使用函数即服务	不适用

Task	描述	所需技能
	（FaaS）解决方案将其部署为无服务器应用程序。 • 重建 — 建议丢弃应用程序代码，使用 PaaS 解决方案在云中重新开发，或者使用 FaaS 解决方案将其重新开发为无服务器应用程序。 • 停用 — 建议完全放弃该应用程序，或者可能将其替换为商业软件即服务（SaaS）替代方案。	
修改细分建议。	在某些情况下，您可能会选择更改 CAST Highligh 推荐部分。为此，您可浏览数据表中的应用程序，然后从应用程序名称旁边的下拉列表中选择不同的区段。然后选择右上角的保存来保存您的更改。 您也可以随时通过选择右上角的导出以导出这些数据。	不适用
选择待分析应用程序。	在 Portfolio Advisor for Cloud 控制面板，选择一个应用程序气泡，以分析该应用程序。在气泡图之后的表格中选择应用程序名称，以开始更深入的分析。 可以使用不同的控制面板来分析各个应用程序，例如 Code Insights（软件运行状况模式）、趋势和软件组合（开源风险）。	不适用

Task	描述	所需技能
分析单个应用程序的 CloudReady 结果。	选择显示应用程序总 CloudReady 分的CloudReady选项卡。该分数是基于 CloudReady 调查答案和 CloudReady 代码扫描组合的加权平均值。调查问题的答案显示在图块下的表格中。 选择 “CloudReady 代码扫描” 以查看代码扫描结果。有一个扫描应用程序代码的 CloudReady 模式列表。此列表包括以下列： • 云需求是特定代码模式。 • 技术是模式编程语言。“影响” 是模式对应用程序的影响（C = 代码，F = 框架，A = 架构）。 • 重要性是在迁移之前解决这种模式的重要程度。 • 贡献是这种模式对总 CloudReady 分的贡献方式。如果图案为绿色，则为助推器并增加 CloudReady 分数。如果图案为红色，则表示它会阻挡并降低 CloudReady 分数。如果图案没有颜色，则它是一个未被检测到的阻挡物，因此会增加 CloudReady 分数。 • Roadblocks 模式出现的个别次数。选择障碍编号，以显	不适用

Task	描述	所需技能
	示检测到该模式的源代码文件列表。 • 东部时间 工作量是对修复每行障碍所需天数的估计值。	
将数据导出至 Microsoft Excel。	（可选）选择导出至 Excel，以导出数据以供进一步分析。应用程序分析结果数据可用于进一步分析应用程序云准备情况，并确定在迁移之前必须更新哪些代码。	不适用
查看建议。	选择 “CloudReady 代码扫描” 旁边的 “推荐”，查看 “云服务推荐” 屏幕。这可根据应用程序的特性，确定应用程序可以采用的 Amazon Web Services。 重复此步骤，以查看针对您分析的所有应用程序的建议。	不适用

相关资源

活动管理

- CAST Highlight 基础认证培训第 3 节：产品组合配置（视频）

源代码分析

- CAST Highlight 基础认证培训第 4 节：应用分析（视频）

其他资源

- Amazon Web Services Marketplace 中的 CAST Highlight
- AWS 和 CAST：加快应用程序现代化

- CAST Highlight — 文档、产品教程和第三方工具
- CAST Highlight — 云就绪产品演示（视频）
- 借助 CAST Highlight实现应用程序组合现代化(AWS 研讨会)

使用 DynamoDB TTL 自动将项目归档到 Amazon S3

创建者：Tabby Ward（AWS）

摘要

此模式提供了从 Amazon DynamoDB 表中删除较旧的数据并将其存档到 Amazon Web Services（AWS）上的 Amazon Simple Storage Service（Amazon S3）存储桶的步骤，而无需管理服务器实例集。

此模式使用 Amazon DynamoDB 生存时间（TTL）自动删除旧项目，使用 Amazon DynamoDB Streams 来捕获 TTL 过期项目。然后，它将 DynamoDB Streams 连接到 AWS Lambda，后者无需预调配或管理任何服务器即可运行代码。

向 DynamoDB 流中添加新项目时，Lambda 函数会启动，并将数据写入亚马逊数据 Firehose 传输流。Firehose 提供了一种简单、完全托管的解决方案，可将数据作为存档加载到 Amazon S3 中。

DynamoDB 通常用于存储时间序列数据，例如来自传感器和联网设备的网页点击流数据或物联网（IoT）数据。许多客户不想删除访问频率较低的项目，而是将其存档以供审计。TTL 根据时间戳属性自动删除项目，从而简化了存档。

DynamoDB Streams 中可以识别由 TTL 删除的项目，DynamoDB Streams 可捕获按时间排序的项目级修改序列，并将该序列存储在日志中长达 24 个小时。这些数据可由 Lambda 函数使用并存档在 Amazon S3 存储桶中，以降低存储成本。为了进一步降低成本，可以创建 Amazon S3 生命周期规则，自动将（创建的）数据立即转换为成本最低的存储类别，例如 S3 Glacier 即时检索或 S3 Glacier 灵活检索，或者用于长期存储的 Amazon S3 Glacier Deep Archive。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- AWS 命令行界面（AWS CLI）1.7 或更高版本，已在 macOS、Linux 或 Windows 上安装并配置。
- Python 3.7 或更高版本。
- Boto3，已安装并配置。如果Boto3 尚未安装，请运行 `python -m pip install boto3` 命令进行安装。

架构

技术堆栈

- Amazon DynamoDB
- Amazon DynamoDB Streams
- Amazon Data Firehose
- AWS Lambda
- Amazon S3

1. TTL 会删除项目。
2. DynamoDB 流触发器调用 Lambda 流处理器函数。
3. Lambda 函数以批处理格式将记录放入 Firehose 交付流中。
4. 数据记录存档在 S3 存储桶中。

工具

- AWS CLI – AWS 命令行界面（AWS CLI）是用于管理 Amazon Web Services 的统一工具。
- Amazon DynamoDB – Amazon DynamoDB 是一个键值和文档数据库，在任何规模上都能提供个位数的毫秒性能。
- Amazon DynamoDB 生存时间（TTL） – Amazon DynamoDB TTL 可以定义每个项目的时间戳，以确定何时不再需要某个项目。
- Amazon DynamoDB Streams – Amazon DynamoDB Streams 可在任何 DynamoDB 表中捕获按时间排序的项目级修改序列，并将这类信息存储在日志中长达 24 个小时。
- Amazon Data Firehos e — Amazon Data Firehose 是将流数据可靠地加载到数据湖、数据存储和分析服务的最简单方法。
- AWS Lambda – AWS Lambda 无需预调配或管理服务器即可运行代码。您只需按使用的计算时间付费。
- Amazon S3 – Amazon Simple Storage Service（Amazon S3）是一种对象存储服务，提供行业领先的可扩展性、数据可用性、安全性和性能。

代码

此模式的代码可在使用 DynamoDB TTL 存储库将项目 GitHub 存档到 S3 中找到。

操作说明

设置 DynamoDB 表、TTL 和 DynamoDB 流

Task	描述	所需技能

创建 DynamoDB 表。	使用 AWS CLI 在 DynamoDB 中创建一个名为 `Reservation` 的表。选择随机读取容量单位（RCU）和写入容量单位（WCU），并给表两个属性：`ReservationID` 和 `ReservationDate`。
	
	`aws dynamodb create-table \`
	`--table-name Reservation \`
	`--attribute-definitions AttributeName=ReservationID,AttributeType=S AttributeName=ReservationDate,AttributeType=N \`
	`--key-schema AttributeName=ReservationID,KeyType=HASH AttributeName=ReservationDate,KeyType=RANGE \`
	`--provisioned-throughput ReadCapacityUnits=100,WriteCapacityUnits=100`
	
	`ReservationDate` 是一个纪元时间戳，将用于开启 TTL。
		云架构师、应用程序开发人员

Task	描述	所需技能

开启 DynamoDB TTL。	使用 AWS CLI 为 `ReservationDate` 属性开启 DynamoDB TTL。
	
	`aws dynamodb update-time-to-live \`
	`--table-name Reservation\`
	`  --time-to-live-specification Enabled=true,AttributeName=ReservationDate`
		云架构师、应用程序开发人员

Task	描述	所需技能
开启 DynamoDB 流。	使用 AWS CLI 通过 NEW_AND_OLD_IMAGES 流类型为 Reservation 表打开 DynamoDB 流。 `aws dynamodb update-table \ --table-name Reservation \ --stream-specification StreamEnabled=true,StreamViewType=NEW_AND_OLD_IMAGES` 此流将包含新项目、更新项目、已删除项目和由 TTL 删除的项目的记录。由 TTL 删除的项目的记录包含一个额外的元数据属性，用于将其与手动删除的项目区分开来。TTL 删除的 userIdentity 字段表示 DynamoDB 服务执行了删除操作。 在这种模式中，只有通过 TTL 删除的项目才会被存档，但只能存档 eventName 为 REMOVE 以及 userIdentity 包含等于 dynamodb.amazonaws.com 的 principalId 的记录。	云架构师、应用程序开发人员

创建和配置 S3 存储桶

Task	描述	所需技能
创建 S3 存储桶。	使用 AWS CLI 在您的 AWS 区域中创建目标 S3 存储桶，`us-east-1` 替换为您的区域，将 amzn-s3-demo-destination-bucket 替换为存储桶的名称。 `aws s3api create-bucket \ --bucket amzn-s3-demo-destination-bucket \ --region us-east-1` 请确保您的 S3 存储桶名称是全球唯一的，因为命名空间由所有 AWS 账户共享。	云架构师、应用程序开发人员
为 S3 存储桶创建 30 天的生命周期策略。	1. 登录 AWS 管理控制台，并打开 Amazon S3 控制台。 2. 选择包含来自 Firehose 的数据的 S3 存储桶。 3. 在 S3 存储桶中，选择管理选项卡，然后选择添加生命周期规则。 4. 在生命周期规则对话框内输入规则名称，并为存储桶配置 30 天生命周期规则。	云架构师、应用程序开发人员

创建 Firehose 传送流

Task	描述	所需技能
创建和配置 Firehose 传送流。	从 GitHub 存储库下载并编辑`CreateFireHoseToS3.py` 代码示例。 此代码是用 Python 编写的，向您展示了如何创建 Firehose 传输流和 AWS 身份和访问管理 (IAM) 角色。IAM 角色将有一个策略，Firehose 可以使用该策略写入目标 S3 存储桶。 要运行该脚本，使用以下命令和命令行参数。 参数 1= <Your_S3_bucket_ARN> ，这是您之前创建的存储桶的 Amazon 资源名称（ARN） 参数 2= 你的 Firehose 名字（这个飞行员正在`firehose_to_s3_stream` 使用。） 参数 3= IAM 角色名称（此试点正在使用 `firehose_to_s3` 。） `python CreateFireHoseToS3.py <Your_S3_Bucket_ARN> firehose_to_s3_stream firehose_to_s3` 如果指定的 IAM 角色不存在，则该脚本将创建一个具有可信	云架构师、应用程序开发人员

Task	描述	所需技能
	关系策略和授予充足 Amazon S3 权限的策略的分派角色。有关这些政策的示例，请参阅其他信息部分。	
验证 Firehose 的传送流。	使用 AWS CLI 来验证传输流是否已成功创建，从而描述 Firehose 传输流。 `aws firehose describe-delivery-stream --delivery-stream-name firehose_to_s3_stream`	云架构师、应用程序开发人员

创建一个 Lambda 函数来处理 Firehose 的交付流

Task	描述	所需技能
创建 Lambda 函数的信任策略。	请使用以下信息创建信任策略文件。 `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "Service": "lambda.amazonaws.com" }, "Action": "sts:AssumeRole" }`	云架构师、应用程序开发人员

Task	描述	所需技能

	``
	    ]
	  }
	``
	
	这向您的函数授予访问 AWS 资源的权限。	

创建 Lambda 函数的执行角色。	要创建执行角色，请运行以下代码。 ``` aws iam create-role --role-name lambda- ex --assume-role-poli cy-document file://Tr ustPolicy.json ```	云架构师、应用程序开发人员

Task	描述	所需技能

向角色添加权限。	要向角色添加权限，请使用 `attach-policy-to-role` 命令。
	
	`aws iam attach-role-policy --role-name lambda-ex --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole`
	`aws iam attach-role-policy --role-name lambda-ex --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaDynamoDBExecutionRole`
	`aws iam attach-role-policy --role-name lambda-ex --policy-arn arn:aws:iam::aws:policy/AmazonKinesisFirehoseFullAccess`
	`aws iam attach-role-policy --role-name lambda-ex --policy-arn arn:aws:iam::aws:policy/IAMFullAccess`
		云架构师、应用程序开发人员

Task	描述	所需技能

创建一个 Lambda 函数。	通过运行以下命令从代码存储库中压缩 `LambdaStreamProcessor.py` 文件。
	`zip function.zip LambdaStreamProcessor.py`
	当创建 Lambda 函数时，您将需要 Lambda 执行角色 ARN。要获取 ARN，请运行以下代码。
	`aws iam get-role \ --role-name lambda-ex`
	要创建 Lambda 函数，请运行以下代码。
	`# Review the environment variables and replace them with your values.`
	`aws lambda create-function --function-name LambdaStreamProcessor \ --zip-file fileb://function.zip --handler LambdaStreamProcessor.handler --runtime python3.8 \ --role {Your Lamda Execution Role ARN}\ --environment Variables="{firehose_name=firehose_t`
		云架构师、应用程序开发人员

Task	描述	所需技能
	`o_s3_stream,bucket_arn = <Your_S3_bucket_ARN>,iam_role_name = firehose_to_s3, batch_size=400}"`	
配置 Lambda 函数触发器。	使用 AWS CLI 配置触发器（DynamoDB Streams），该触发器将调用 Lambda 函数。批量大小为 400 是为了避免遇到 Lambda 并发问题。 `aws lambda create-event-source-mapping --function-name LambdaStreamProcessor \ --batch-size 400 --starting-position LATEST \ --event-source-arn <Your Latest Stream ARN From DynamoDB Console>`	云架构师、应用程序开发人员

测试功能

Task	描述	所需技能
将时间戳已过期的项目添加到预留表。	要测试该功能，请将带有过期纪元时间戳的项目添加到 `Reservation` 表中。TTL 将根据时间戳自动删除项目。 Lambda 函数是在 DynamoDB Stream 活动时启动的，它会筛选事件以识别 `REMOVE` 活动或已删除的项目。然后，它以批	云架构师

Task	描述	所需技能
	处理格式将记录放入 Firehose 传送流。 Firehose 交付流将项目传输到带有前缀的目标 S3 存储桶。`firehosetos3example/year=current year/month=current month/day=current day/hour=current hour/` **⚠ Important** 要优化数据检索，请使用`Prefix`和配置 Amazon `S3ErrorOutputPrefix` ，详见其他信息部分。	

清理资源

Task	描述	所需技能
删除所有资源。	删除所有资源，确保不会为未使用的任何服务付费。	云架构师、应用程序开发人员

相关资源

- 管理存储生命周期
- Amazon S3 存储类
- 适用于 Python 的 Amazon SDK（Boto3）文档

其他信息

创建和配置 Firehose 传送流-策略示例

Firehose 信任关系策略示例文档

```
firehose_assume_role = {
        'Version': '2012-10-17',
        'Statement': [
            {
                'Sid': '',
                'Effect': 'Allow',
                'Principal': {
                    'Service': 'firehose.amazonaws.com'
                },
                'Action': 'sts:AssumeRole'
            }
        ]
    }
```

S3 权限策略示例

```
s3_access = {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Sid": "",
                "Effect": "Allow",
                "Action": [
                    "s3:AbortMultipartUpload",
                    "s3:GetBucketLocation",
                    "s3:GetObject",
                    "s3:ListBucket",
                    "s3:ListBucketMultipartUploads",
                    "s3:PutObject"
                ],
                "Resource": [
                    "{your s3_bucket ARN}/*",
                    "{Your s3 bucket ARN}"
                ]
            }
        ]
    }
```

测试功能 – Amazon S3 配置

选择具有以下 Prefix 和 ErrorOutputPrefix 前缀的 Amazon S3 配置来优化数据检索。

prefix

```
firehosetos3example/year=! {timestamp: yyyy}/month=! {timestamp:MM}/day=!
 {timestamp:dd}/hour=!{timestamp:HH}/
```

Firehose 首先在 S3 存储桶下创建一个firehosetos3example名为的基本文件夹。然后，它使用 Java DateTimeFormatter格式!{timestamp:HH}将表达式!{timestamp:yyyy}!{timestamp:MM}!{timestamp:dd}、、和计算为年、月、日和小时。

例如，在 Unix 纪元时间中，1604683577 的近似到达时间戳的评估结果为 year=2020、month=11、day=06 和 hour=05。因此，Amazon S3 中数据记录的传输位置评估为 firehosetos3example/year=2020/month=11/day=06/hour=05/。

ErrorOutputPrefix

```
firehosetos3erroroutputbase/!{firehose:random-string}/!{firehose:error-output-type}/!
{timestamp:yyyy/MM/dd}/
```

ErrorOutputPrefix 结果会直接在 S3 存储桶下生成名为 firehosetos3erroroutputbase 的基本文件夹。表达式 !{firehose:random-string} 的评估结果为 11 个字符的随机字符串，例如 ztWxkdg3Thg。传输失败记录的 Amazon S3 对象的位置可以评估为 firehosetos3erroroutputbase/ztWxkdg3Thg/processing-failed/2020/11/06/。

在 Amazon 服务中构建多租户无服务器架构 OpenSearch

由 Tabby Ward (AWS) 和 Nisha Gambhir (AWS) 编写

摘要

Amazon S OpenSearch ervice 是一项托管服务，可以轻松部署、操作和扩展 Elasticsearch，这是一款流行的开源搜索和分析引擎。 OpenSearch 该服务为日志和指标等流式数据提供自由文本搜索以及近乎实时的摄取和仪表板管理。

软件即服务 (SaaS) 提供商经常使用 OpenSearch 服务来解决各种用例，例如以可扩展和安全的方式获取客户见解，同时降低复杂性和停机时间。

在多租户环境中使用 OpenSearch 服务会引入一系列注意事项，这些注意事项会影响 SaaS 解决方案的分区、隔离、部署和管理。SaaS 提供程序必须考虑如何随着不断变化的工作负载有效扩展其 Elasticsearch 集群。他们还需要考虑分层和嘈杂的邻居条件如何影响他们的分区模型。

此模式介绍了用于通过 Elasticsearch 构造表示和隔离租户数据的模型。此外，该模式以简单的无服务器参考架构为例，演示在多租户环境中使用 Serv OpenSearch ice 进行索引和搜索。它实现了池数据分区模型，在所有租户之间共享相同的索引，同时保持租户的数据隔离。此模式使用以下 AWS 服务：亚马逊 API Gateway AWS Lambda、亚马逊简单存储服务 (Amazon S3) Service 和服务。 OpenSearch

有关池模型和其他数据分区模型的更多信息，请参阅其他信息部分。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- AWS Command Line Interface (AWS CLI) 版本 2.x，已在 macOS、Linux 或 Windows 上安装和配置
- Python 版本 3.9
- pip3 — Python 源代码以.zip 文件形式提供，要部署至 Lambda 函数中。若要在本地使用代码或对其进行自定义，请按照以下步骤开发和重新编译源代码：
  1. 通过在与 Python 脚本相同的目录中运行以下命令以生成 `requirements.txt` 文件：`pip3 freeze > requirements.txt`
  2. 安装依赖项：`pip3 install -r requirements.txt`

限制

- 此代码在 Python 中运行，当前不支持其他编程语言。
- 示例应用程序不包括 AWS 跨区域或灾难恢复 (DR) 支持。
- 此模式仅用于演示目的。不应在生产环境中使用。

架构

下图展示了此模式的高级架构。该架构包括以下内容：

- 使用 Lambda 对内容进行索引和查询
- OpenSearch 执行搜索的服务
- API Gateway 用于提供与用户的 API 交互
- Amazon S3，用于存储原始（未编入索引）的数据
- 亚马逊 CloudWatch 将监控日志
- AWS Identity and Access Management (IAM) 用于创建租户角色和策略

自动化和扩缩

为简单起见，该模式 AWS CLI 用于配置基础架构和部署示例代码。您可以创建 AWS CloudFormation 模板或 AWS Cloud Development Kit (AWS CDK) 脚本来自动执行模式。

工具

AWS 服务

- AWS CLI是一个统一的工具，用于通过在命令行外壳中使用命令来管理和 AWS 服务 资源。
- Lambda 是一项计算服务，允许您在不预配置或管理服务器的情况下运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩展，从每天几个请求扩展到每秒数千个请求。
- API Gateway AWS 服务 用于创建、发布、维护、监控和保护任何规模的 RES WebSocket APIs T、HTTP。
- Amazon S3 是一项对象存储服务，可让您随时从网络上的任何位置存储和检索任意数量的信息。
- OpenSearch 服务是一项完全托管的服务，可让您轻松地大规模部署、保护和运行 Elasticsearch，经济实惠。

代码

附件提供了此模式的示例文件。这些指令包括：

- `index_lambda_package.zip`— Lambda 函数，用于使用池模型对 OpenSearch 服务中的数据进行索引。
- `search_lambda_package.zip`— 用于在服务中搜索数据的 Lambda 函数。 OpenSearch
- `Tenant-1-data` — Tenant-1 的原始(非索引)数据示例。
- `Tenant-2-data` — Tenant-2 的原始(非索引)数据示例。

> **Important**
>
> 这种模式中的故事包括针对 Unix、Linux 和 macOS 进行格式化的 AWS CLI 命令示例。对于 Windows，请将每行末尾的反斜杠 (\) Unix 行继续符替换为脱字号 (^)。

> **Note**
>
> 在 AWS CLI 命令中，将尖括号 (<>) 内的所有值替换为正确的值。

操作说明

创建和配置 S3 存储桶

Task	描述	所需技能
创建 S3 存储桶。	在您的中创建 S3 存储桶 AWS 区域。此存储桶将保存示例应用程序未编入索引的租户数据。确保 S3 存储桶的名称是全球唯一的，因为命名空间由所有人共享 AWS 账户。 要创建 S3 存储桶，您可以按如下方式使用 AWS CLI create-bu cket 命令：	云架构师、云管理员

Task	描述	所需技能
	`aws s3api create-bucket \ --bucket <tenantrawdata> \ --region <your-AWS-Region>` 其中 `tenantrawdata` 是 S3 存储桶名称。(您可使用任何符合存储桶命名准则的唯一名称。)	

创建和配置 Elasticsearch 集群

Task	描述	所需技能
创建 OpenSearch 服务域。	运行 AWS CLI create-elasticsearch-domain命令创建 OpenSearch 服务域： `aws es create-elasticsearch-domain \ --domain-name vpc-cli-example \ --elasticsearch-version 7.10 \ --elasticsearch-cluster-config InstanceType=t3.medium.elasticsearch,InstanceCount=1 \ --ebs-options EBSEnabled=true,VolumeType=gp2,VolumeSize=10 \ --domain-endpoint-options "{\"EnforceHTTPS\": true}" \`	云架构师、云管理员

Task	描述	所需技能
	--encryption-at-rest-options "{\"Enabled\": true}" \ --node-to-node-encryption-options "{\"Enabled\": true}" \ --advanced-security-options "{\"Enabled\": true, \"InternalUserDatabaseEnabled\": true, \ \"MasterUserOptions\": {\"MasterUserName\": \"KibanaUser\", \ \"MasterUserPassword\": \"NewKibanaPassword@123\"}}" \ --vpc-options "{\"SubnetIds\": [\"<subnet-id>\"], \"SecurityGroupIds\": [\"<sg-id>\"]}" \ --access-policies "{\"Version\": \"2012-10-17\", \"Statement\": [{ \"Effect\": \"Allow\", \ \"Principal\": {\"AWS\": \"*\" }, \"Action\":\"es:*\", \ \"Resource\": \"arn:aws:es:<region>:<account-id>:domain\/vpc-cli-example\/*\" }] }" 实例计数设置为 1，因为该域用于测试。您需要使用	

Task	描述	所需技能
	`advanced-security-options` 参数启用精细的访问控制，因为创建域后无法更改详细信息。 此命令会创建主用户名 (`KibanaUser` ）和密码，您可使用这些用户名和密码登录 Kibana 控制台。 由于该域是虚拟私有云（VPC）的组成部分，因此您必须通过指定要使用的访问策略来确保可以访问 Elasticsearch 实例。 有关更多信息，请参阅 AWS 文档中的在 VPC 内启动您的 Amazon OpenSearch 服务域。	

Task	描述	所需技能
设置堡垒主机。	将亚马逊弹性计算云 (亚马逊 EC2) Windows 实例设置为访问 Kibana 控制台的堡垒主机。Elasticsearch 安全组必须允许来自亚马逊 EC2 安全组的流量。有关说明，请参阅博客文章使用堡垒服务器控制对 EC2 实例的网络访问。 当堡垒主机设置完毕并且您拥有与可用实例关联的安全组后，请使用 AWS CLI authorize-security-group-ingress命令向 Elasticsearch 安全组添加权限，允许从 Amazon EC2 （堡垒主机）安全组中访问 443 端口。 `aws ec2 authorize-security-group-ingress \ --group-id <SecurityGroupIdfElasticSearch> \ --protocol tcp \ --port 443 \ --source-group <SecurityGroupIdfBashionHostEC2>`	云架构师、云管理员

创建并配置 Lambda 索引函数

Task	描述	所需技能

创建 Lambda 执行角色。	运行 AWS CLI create-role 命令以授予 Lambda 索引函数访问和资源的权限： AWS 服务
	`aws iam create-role \ --role-name index-lambda-role \ --assume-role-policy-document file://lambda_assume_role.json`
	其中`lambda_assume_role.json` 是一个授予 Lambda 函数`AssumeRole` 权限的 JSON 文档，如下所示：
	`{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "Service": "lambda.amazonaws.com" }, "Action": "sts:AssumeRole" } ] }`
		云架构师、云管理员

Task	描述	所需技能

将托管式策略附加到 Lambda 角色。	运行 AWS CLI attach-role-policy命令将托管策略附加到上一步中创建的角色。这两个策略授予角色创建弹性网络 interface 和向日志写入 CloudWatch 日志的权限。
	
	`aws iam attach-role-policy \`
	`  --role-name index-lambda-role \`
	`  --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole`
	
	`aws iam attach-role-policy \`
	`  --role-name index-lambda-role \`
	`  --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaVPCAccessExecutionRole`
		云架构师、云管理员

Task	描述	所需技能

创建策略，以授予 Lambda 索引函数读取 S3 对象的权限。	运行 AWS CLI create-policy 命令以授予 Lambda 索引函数读取 S3 存储桶中对象的`s3:GetObject` 权限：
	`aws iam create-policy \ --policy-name s3-permission-policy \ --policy-document file://s3-policy.json`
	该文件`s3-policy.json` 是一个 JSON 文档，如下所示，它授予允许`s3:GetObject` 对 S3 对象进行读取访问的权限。如果您在创建 S3 存储桶时使用了其他名称，请在以下`Resource` 部分中提供正确的存储桶名称：
	`{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": "s3:GetObject", "Resource": "arn:aws:s3:::<tenantrawdata>/*" } ] }`
		云架构师、云管理员

Task	描述	所需技能
将 Amazon S3 权限策略附加至 Lambda 执行角色。	运行 AWS CLI attach-role-policy命令将您在上一步中创建的 Amazon S3 权限策略附加到 Lambda 执行角色： `aws iam attach-role-policy \ --role-name index-lambda-role \ --policy-arn <PolicyARN>` 其中 `PolicyARN` 是 Amazon S3 权限策略的 Amazon 资源名称（ARN）。您可从上一条命令的输出中获得此值。	云架构师、云管理员

Task	描述	所需技能
创建 Lambda 索引函数。	运行 AWS CLI create-function 命令创建 Lambda 索引函数，该函数将访问服务：OpenSearch `aws lambda create-function \ --function-name index-lambda-function \ --zip-file fileb://index_lambda_package.zip \ --handler lambda_index.lambda_handler \ --runtime python3.9 \ --role "arn:aws:iam::account-id:role/index-lambda-role" \ --timeout 30 \ --vpc-config "{\"SubnetIds\": [\"<subnet-id1\>", \"<subnet-id2>\"], \ \"SecurityGroupIds\": [\"<sg-1>\"]}"`	云架构师、云管理员

Task	描述	所需技能

允许 Amazon S3 调用 Lambda 索引函数。	运行 add-p AWS CLI ermission 命令授予 Amazon S3 调用 Lambda 索引函数的权限：
	
	`aws lambda add-permission \`
	`  --function-name index-lambda-function \`
	`  --statement-id s3-permissions \`
	`  --action lambda:InvokeFunction \`
	`  --principal s3.amazonaws.com \`
	`  --source-arn "arn:aws:s3:::<tentantrawdata>" \`
	`  --source-account "<account-id>"`
		云架构师、云管理员

Task	描述	所需技能
为 Amazon S3 事件添加 Lambda 触发器。	运行 AWS CLI put-bucket-notification-configuration命令以在检测到 Amazon S3 `ObjectCreated` 事件时向 Lambda 索引函数发送通知。每当将对象上传至 S3 存储桶时，索引函数就会运行。 `aws s3api put-bucket-notification-configuration \ --bucket <tenantrawdata> \ --notification-configuration file://s3-trigger.json` 文件 `s3-trigger.json` 是当前文件夹中的一个 JSON 文档，用于在 Amazon S3 `ObjectCreated` 事件发生时向 Lambda 函数添加资源策略。	云架构师、云管理员

创建并配置 Lambda 搜索函数

Task	描述	所需技能
创建 Lambda 执行角色。	运行 AWS CLI create-role 命令以授予 Lambda 搜索函数访问和资源的权限： AWS 服务 `aws iam create-role \ --role-name search-lambda-role \`	云架构师、云管理员

Task	描述	所需技能

	--assume-role-policy-document file://lambda_assume_role.json
	
	其中 lambda_assume_role.json 是当前文件夹中用于授予 Lambda 函数 AssumeRole 权限的 JSON 文档，如下所示：
	
	{
	"Version": "2012-10-17",
	"Statement": [
	{
	"Effect": "Allow",
	"Principal": {
	"Service": "lambda.amazonaws.com"
	},
	"Action": "sts:AssumeRole"
	}
	]
	}	

Task	描述	所需技能

将托管式策略附加到 Lambda 角色。	运行 AWS CLI attach-role-policy命令将托管策略附加到上一步中创建的角色。这两个策略授予角色创建弹性网络 interface 和向日志写入 CloudWatch 日志的权限。
	
	`aws iam attach-role-policy \`
	`  --role-name search-lambda-role \`
	`  --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole`
	
	`aws iam attach-role-policy \`
	`  --role-name search-lambda-role \`
	`  --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaVPCAccessExecutionRole`
		云架构师、云管理员

Task	描述	所需技能

创建 Lambda 搜索函数。	运行 AWS CLI create-function 命令创建 Lambda 搜索函数，该函数将访问服务：OpenSearch
	
	`aws lambda create-function \`
	`  --function-name search-lambda-function \`
	`  --zip-file fileb://search_lambda_package.zip \`
	`  --handler lambda_search.lambda_handler \`
	`  --runtime python3.9 \`
	`  --role "arn:aws:iam::account-id:role/search-lambda-role" \`
	`  --timeout 30 \`
	`  --vpc-config "{\"SubnetIds\": [\"<subnet-id1\>", \"<subnet-id2>\"], \`
	`  \"SecurityGroupIds\": [\"<sg-1>\"]}"`
		云架构师、云管理员

创建和配置租户角色

Task	描述	所需技能
创建租户 IAM 角色。	运行 AWS CLI create- role 命令创建两个用于测试搜索功能的租户角色： `aws iam create-role \`	云架构师、云管理员

Task	描述	所需技能

	`--role-name Tenant-1-role \ --assume-role-policy-document file://assume-role-policy.json`
	
	`aws iam create-role \ --role-name Tenant-2-role \ --assume-role-policy-document file://assume-role-policy.json`
	
	文件 `assume-role-policy.json` 是当前文件夹中的一个 JSON 文档，用于向 Lambda 执行角色授予 `AssumeRole` 权限：
	
	`{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "AWS": "<Lambda execution role for index function>", "AWS": "<Lambda execution role for search function>" }, "Action": "sts:AssumeRole" } ]`	

Task	描述	所需技能
	} ```	

Task	描述	所需技能

创建租户 IAM policy。	运行 AWS CLI create-policy 命令来创建授予对 Elasticsearch 操作的访问权限的租户策略：
	
	`aws iam create-policy \ --policy-name tenant-policy \ --policy-document file://policy.json`
	
	文件 `policy.json` 是当前文件夹中授予对 Elasticsearch 的权限的 JSON 文档：
	
	`{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "es:ESHttpDelete", "es:ESHttpGet", "es:ESHttpHead", "es:ESHttpPost", "es:ESHttpPut", "es:ESHttpPatch" ], "Resource": [`
		云架构师、云管理员

Task	描述	所需技能

	<pre>"<ARN of Elasticsearch domain created earlier>"
	]
	}
	]
	}</pre>	

将租户 IAM policy 附加至租户角色。	运行 AWS CLI attach-role-policy命令将租户 IAM 策略附加到您在前面的步骤中创建的两个租户角色：
	<pre>aws iam attach-role-policy \
	--policy-arn arn:aws:iam::account-id:policy/tenant-policy \
	--role-name Tenant-1-role
	
	aws iam attach-role-policy \
	--policy-arn arn:aws:iam::account-id:policy/tenant-policy \
	--role-name Tenant-2-role</pre>
	策略 ARN 来自上一步中的输出。
		云架构师、云管理员

Task	描述	所需技能

创建 IAM policy，以向 Lambda 授予代入角色的权限。	运行 AWS CLI create-policy 命令为 Lambda 创建担任租户角色的策略：
	
	`aws iam create-policy \ --policy-name assume-tenant-role-policy \ --policy-document file://lambda_policy.json`
	
	文件 `lambda_policy.json` 是当前文件夹中的一个 JSON 文档，用于授予对 `AssumeRole` 的权限。
	
	`{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": "sts:AssumeRole", "Resource": "<ARN of tenant role created earlier>" } ] }`
	
	对于 `Resource`，您可使用通配符来避免为每个租户创建新策略。
		云架构师、云管理员

Task	描述	所需技能

创建 IAM policy，向 Lambda 索引角色授予 Amazon S3 访问权限。	运行 AWS CLI create-pol icy 命令以授予 Lambda 索引角色访问 S3 存储桶中对象的权限：
	`aws iam create-policy \ --policy-name s3-permission-policy \ --policy-document file://s3_lambda_policy.json`
	文件 `s3_lambda_policy.json` 是当前文件夹中以下 JSON 策略文档：
	`{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": "s3:GetObject", "Resource": "arn:aws:s3:::tenantrawdata/*" } ] }`
		云架构师、云管理员

Task	描述	所需技能

将该策略附加到 Lambda 执行角色。	运行 AWS CLI attach-role-policy命令将上一步中创建的策略附加到您之前创建的 Lambda 索引和搜索执行角色：
	
	`aws iam attach-role-policy \`
	`  --policy-arn arn:aws:iam::account-id:policy/assume-tenant-role-policy \`
	`  --role-name index-lambda-role`
	
	`aws iam attach-role-policy \`
	`  --policy-arn arn:aws:iam::account-id:policy/assume-tenant-role-policy \`
	`  --role-name search-lambda-role`
	
	`aws iam attach-role-policy \`
	`  --policy-arn arn:aws:iam::account-id:policy/s3-permission-policy \`
	`  --role-name index-lambda-role`
	
	策略 ARN 来自上一步中的输出。
		云架构师、云管理员

创建和配置搜索 API

Task	描述	所需技能
在 API Gateway 中创建 REST API。	运行 AWS CLI create-rest-api命令创建 REST API 资源： `aws apigateway create-rest-api \ --name Test-Api \ --endpoint-configuration "{ \"types\": [\"REGIONAL\"] }"` 对于端点配置类型，您可以指定EDGE而不是使用边缘位置而不是特定的边缘位置 AWS 区域。REGIONAL 记录命令输出中 id 字段的值。这是您将在后续命令使用的 API ID。	云架构师、云管理员
创建资源，以用于搜索 API。	搜索 API 资源使用名为 search 的资源启动 Lambda 搜索函数。(您不必为 Lambda 索引函数创建 API，因为当对象上传到 S3 存储桶时，它会自动运行。) 1. 运行 AWS CLI get-resources 命令以获取根路径的父 ID： `aws apigateway get-resources \ --rest-api-id <API-ID>`	云架构师、云管理员

Task	描述	所需技能

	记下 ID 字段的值。您将在下一命令中使用此父 ID。
	
	`{ "items": [ { "id": "zpsri964ck", "path": "/" } ] }`
	
	2. 运行 AWS CLI create-resource 命令为搜索 API 创建资源。对于 `parent-id` ，请指定前一命令的 ID。
	
	`aws apigateway create-resource \ --rest-api-id <API-ID> \ --parent-id <Parent-ID> \ --path-part search`	

Task	描述	所需技能

为搜索 API 创建 GET 方法。	运行 AWS CLI put-met hod 命令为搜索 API 创建GET 方法：
	
	`aws apigateway put-method \`
	`  --rest-api-id <API-ID> \`
	`  --resource-id <ID from the previous command output> \`
	`  --http-method GET \`
	`  --authorization-type "NONE" \`
	`  --no-api-key-required`
	
	对于 `resource-id` ，请从 `create-resource` 命令输出中指定 ID。
		云架构师、云管理员

Task	描述	所需技能
为搜索 API 创建响应方法。	运行 AWS CLI put-method-response命令为搜索 API 添加方法响应： `aws apigateway put-method-response \ --rest-api-id <API-ID> \ --resource-id <ID from the create-resource command output> \ --http-method GET \ --status-code 200 \ --response-models "{\"application/json\": \"Empty\"}"` 对于`resource-id` ，请从先前`create-resource` 命令的输出中指定 ID。	云架构师、云管理员

Task	描述	所需技能

为搜索 API 设置代理 Lambda 集成。	运行 AWS CLI put- integration 命令设置与 Lambda 搜索函数的集成：
	
	`aws apigateway put-integration \`
	`  --rest-api-id <API-ID> \`
	`  --resource-id  <ID from the create-resource command output> \`
	`  --http-method GET \`
	`  --type AWS_PROXY \`
	`  --integration-http-method GET \`
	`  --uri arn:aws:apigateway:region:lambda:path/2015-03-31/functions/arn:aws:lambda:<region>:<account-id>:function:<function-name>/invocations`
	
	对于 `resource-id` ，请指定先前 `create-resource` 命令中的 ID。
		云架构师、云管理员

Task	描述	所需技能
授予 API Gateway 调用 Lambda 搜索函数的权限。	运行 add-p AWS CLI ermission 命令授予 API Gateway 使用搜索功能的权限： `aws lambda add-permission \ --function-name <function-name> \ --statement-id apigateway-get \ --action lambda:InvokeFunction \ --principal apigateway.amazonaws.com \ --source-arn "arn:aws:execute-api:<region>:<account-id>:api-id/*/GET/search` 如果您使用其他名为 `search` 的 API 资源，则更改 `source-arn` 路径。	云架构师、云管理员
部署搜索 API。	运行 AWS CLI create-deployment 命令创建名为：`dev` `aws apigateway create-deployment \ --rest-api-id <API-ID> \ --stage-name dev` 如果您更新 API，则可以使用相同的 AWS CLI 命令将其重新部署到同一阶段。	云架构师、云管理员

创建和配置 Kibana 角色

Task	描述	所需技能
登录到 Kibana 控制台。	1. 在 OpenSearch 服务控制台的域名控制面板上找到 Kibana 的链接。URL 的格式为：<domain-endpoint>/_plugin/kibana/ 。 2. 使用您在第一部操作说明中配置的堡垒主机访问 Kibana 控制台。 3. 使用创建 OpenSearch 服务域时之前步骤中的主用户名和密码登录 Kibana 控制台。 4. 当系统提示选择租户时，选择私人。	云架构师、云管理员
创建和配置 Kibana 角色。	为了提供数据隔离并确保一个租户无法检索另一租户的数据，您需要使用文档安全性，它允许租户仅访问包含其租户 ID 的文档。 1. 在 Kibana 控制台的导航窗格中，选择安全性、角色。 2. 创建新租户角色。 3. 将集群权限设置为indices_all ，这将授予对 OpenSearch 服务索引的创建、读取、更新和删除 (CRUD) 权限。 4. 限制对索引的 tenant-data 索引权限。(索引名称应	云架构师、云管理员

Task	描述	所需技能
	与 Lambda 搜索和索引函数中的名称相匹配。） 5. 将索引权限设置为 `indices_all` ，使用户能够执行所有与索引相关的操作。（根据您的要求，您可限制操作以获得更精细的访问权限。） 6. 为确保文档级安全，请使用以下策略按租户 ID 筛选文档，为共享索引中的租户提供数据隔离： `{ "bool": { "must": { "match": { "TenantId": "Tenant-1" } } } }` 索引名称、属性和值区分大小写。	

Task	描述	所需技能
将用户映射至角色。	1. 为角色选择映射用户选项卡，然后选择映射用户。 2. 在后端角色部分，指定您之前创建的 IAM 租户角色的 ARN，然后选择映射。这会将 IAM 租户角色映射至 Kibana 角色，因此租户特定的搜索仅返回该租户的数据。例如，如果 Tenant-1 的 IAM 角色名称为 `Tenant-1-Role` ，则在 Tenant-1 Kibana 角色的后端角色框中指定`Tenant-1-Role` （来自创建和配置租户角色操作说明）的 ARN。 3. 对租户 2 重复步骤 1 与 2。 我们建议您在租户入职时自动创建租户与 Kibana 角色。	云架构师、云管理员

Task	描述	所需技能
创建 tenant-data 索引。	在导航窗格的管理下，选择开发工具，然后运行以下命令。此命令创建 `tenant-data` 索引，以定义 `TenantId` 属性的映射。 `PUT /tenant-data { "mappings": { "properties": { "TenantId": { "type": "keyword"} } } }`	云架构师、云管理员

为 Amazon S3 创建 VPC 终端节点和 AWS STS

Task	描述	所需技能
为 Amazon S3 创建和配置 VPC 端点	运行 AWS CLI create-vpc-endpoint命令为 Amazon S3 创建 VPC 终端节点。终端节点允许 VPC 中的 Lambda 索引函数访问 Amazon S3。 `aws ec2 create-vpc-endpoint \ --vpc-id <VPC-ID> \ --service-name com.amazonaws.us-east-1.s3 \ --route-table-ids <route-table-ID>`	云架构师、云管理员

Task	描述	所需技能
	对于 `vpc-id`，请指定您用于 Lambda 索引函数的 VPC。对于 `service-name` ，请使用 Amazon S3 端点的正确网址。对于 `route-table-ids` ，请指定与 VPC 端点关联的路由表。	

Task	描述	所需技能
为创建 VPC 终端节点 AWS STS。	运行 AWS CLI create-vpc-endpoint命令为 AWS Security Token Service (AWS STS) 创建 VPC 终端节点。终端节点允许 VPC 中的 Lambda 索引和搜索功能进行访问。 AWS STS这些函数在担任 IAM 角色 AWS STS 时使用。 `aws ec2 create-vpc-endpoint \ --vpc-id <VPC-ID> \ --vpc-endpoint-type Interface \ --service-name com.amazonaws.us-east-1.sts \ --subnet-id <subnet-ID> \ --security-group-id <security-group-ID>` 对于 `vpc-id`，请指定您用于 Lambda 索引和搜索功能的 VPC。对于 `subnet-id` ，请提供应在其中创建此端点的子网。对于 `security-group-id` ，请指定要与该端点关联的安全组。（它可能与 Lambda 使用的安全组相同。）	云架构师、云管理员

测试多租户与数据隔离

Task	描述	所需技能
更新索引与搜索函数的 Python 文件。	1. 在文件中，编辑`index_lambda_package.zip lamba_index.py` 文件以更新 AWS 账户 ID AWS 区域、和 Elasticsearch 端点信息。 2. 在文件中，编辑`search_lambda_package.zip lambda_search.py` 文件以更新 AWS 账户 ID AWS 区域、和 Elasticsearch 端点信息。 您可以从 OpenSearch 服务控制台的概述选项卡获取 Elasticsearch 终端节点。其格式为 `<AWS-Region>.es.amazonaws.com` 。	云架构师、应用程序开发人员
更新 Lambda 代码。	使用 AWS CLI update-function-code命令使用您对 Python 文件所做的更改来更新 Lambda 代码： `aws lambda update-function-code \ --function-name index-lambda-function \ --zip-file fileb://index_lambda_package.zip`	云架构师、应用程序开发人员

Task	描述	所需技能
	`aws lambda update-function-code \ --function-name search-lambda-function \ --zip-file fileb://search_lambda_package.zip`	

将原始数据上传到 S3 存储桶。	使用 AWS CLI cp 命令将 Tenant-1 和 Tenant-2 对象的数据上传到`tenantrawdata` 存储桶（指定您为此目的创建的 S3 存储桶的名称）：
	`aws s3 cp tenant-1-data s3://tenantrawdata`
	`aws s3 cp tenant-2-data s3://tenantrawdata`
	S3 存储桶设置为每当上传数据时运行 Lambda 索引函数，便在 Elasticsearch 中为文档编制索引。
		云架构师、云管理员

在 Kibana 控制台搜索数据。	在 Kibana 控制台上，运行以下查询： `GET tenant-data/_search` 此查询显示了 Elasticsearch 中编制索引的所有文档。在这种情况下，您应该会看到两个单独的 Tenant-1 和 Tenant-2 文档。	云架构师、云管理员

Task	描述	所需技能

从 API Gateway 中测试搜索 API。	1. 在 API Gateway 控制台，打开搜索 API，在搜索资源中选择 GET 方法，然后选择测试。
	2. 在测试窗口中，为租户 ID 提供以下查询字符串(区分大小写)，然后选择测试。
	`TenantId=Tenant-1`
	Lambda 函数向 OpenSearch 服务发送查询，该查询根据文档级别的安全性筛选租户文档。该方法返回 Tenant-1 的文档。
	3. 将查询字符串更改为：
	`TenantId=Tenant-2`
	此查询返回 Tenant-2 的文档。
	有关屏幕插图，请参阅其他信息部分。
		云架构师、应用程序开发人员

清理资源。	清理您创建的所有资源，以防止向您的账户收取额外费用。	AWS DevOps、云架构师、云管理员

相关资源

- AWS SDK for Python (Boto)
- AWS Lambda 文档
- API Gateway 文档

- Amazon S3 文档
- 亚马逊 OpenSearch 服务文档
  - Amazon 服务中的精细访问控制 OpenSearch
  - 使用 Amazon OpenSearch 服务创建搜索应用程序
  - 在 VPC 内启动您的亚马逊 OpenSearch 服务域

其他信息

数据分区模型

多租户系统中常用的数据分区模型包含三种：孤岛、池和混合。您选择的模型取决于环境合规性、噪音邻居、操作和隔离需求。

孤岛模型

在孤岛模型中，每个租户数据都存储在不同的存储区域中，租户数据不会混合。您可以使用两种方法通过 Serv OpenSearch ice 实现孤岛模型：每个租户的域和每个租户的索引。

- 每个租户的域名 — 您可以为每个租户使用单独的 OpenSearch 服务域（与 Elasticsearch 集群同义）。将每个租户置于自己的域内，可以获得将数据置于独立构造中的所有好处。但是这种方法带来了管理和敏捷性方面的挑战。它的分布性质使得汇总和评测租户的运营状况和活动变得更困难。这是一个成本高昂的选项，要求每个 OpenSearch 服务域至少有三个主节点和两个数据节点来处理生产工作负载。

- 每个租户的索引-您可以将租户数据放在 OpenSearch 服务集群内的单独索引中。使用这种方法，您可以在创建索引和命名索引时使用租户标识符，方法是在索引名称前面加上租户标识符。按租户编制索引的方法可帮助实现孤岛目标，而无需为每个租户引入完全独立的集群。但是，如果索引数量增加，您可能会遇到内存压力，因为这种方法需要更多分片，而主节点必须处理更多的分配和重新平衡。

孤岛模型中的隔离 — 在孤岛模型中，您可使用 IAM policy 来隔离存放每个租户数据的域或索引。这些策略阻止一个租户访问另外一个租户的数据。要实现您的孤岛隔离模型，您可创建基于资源的策略来控制对租户资源的访问权限。这通常是一个域访问策略，用于指定委托人可以对域的子资源（包括

Elasticsearch 索引和）执行哪些操作。 APIs借助基于 IAM 身份的策略，您可以指定对域、索引或服务内部允许或 APIs 拒绝的操作。 OpenSearch IAM policy 的 `Action` 元素描述该策略允许或拒绝的特定操作，并且 `Principal` 元素指定受影响的账户、用户或角色。

以下示例策略仅授予 Tenant-1 对 `tenant-1` 域中子资源的完全访问权限（如所指定 `es:*`）。`Resource` 元素中的尾随 `/*` 指示此策略适用于域的子资源，而不是域本身。此策略生效后，不允许租户在现有域创建新域或修改设置。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::<aws-account-id>:user/Tenant-1"
         },
         "Action": "es:*",
         "Resource": "arn:aws:es:<Region>:<account-id>:domain/tenant-1/*"
      }
   ]
}
```

要实现每个索引的租户孤岛模型，您需要修改此示例策略，通过指定索引名称，进一步将 Tenant-1 限制在指定的一个或多个索引范围内。以下示例策略将 Tenant-1 限制为 `tenant-index-1` 索引。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::123456789012:user/Tenant-1"
         },
         "Action": "es:*",
         "Resource": "arn:aws:es:<Region>:<account-id>:domain/test-domain/tenant-
index-1/*"
      }
   ]
}
```

池模型

在池模型中，所有租户数据都存储至同一域内的索引中。租户标识符包含在数据（文档）中并用作分区键，因此您可确定哪些数据属于哪个租户。此模式减少了管理开销。操作和管理池索引比管理多个索引更加容易、更高效。但是，由于租户数据混合在同一个索引，因此您将失去孤岛模型提供的自然租户隔离。这种方法还可能会因为邻居噪音效应而降低性能。

池模型中的租户隔离 — 通常，租户隔离难以在池模型中实现。与孤岛模型一起使用的 IAM 机制不允许您根据存储在文档内的租户 ID 来描述隔离。

另一种方法是使用开放发行版为 Elasticsearch 提供的精细访问控制 (FGAC) 支持。FGAC 允许您控制索引、文档或字段级别权限。对于每个请求，FGAC 都会评估用户凭证，然后对用户执行身份验证或拒绝访问。如果 FGAC 对用户进行身份验证，它将获取映射到该用户的所有角色，并使用完整的权限集来确定如何处理请求。

要在池化模型中实现所需隔离，您可使用文档级安全性，这样可以将角色限制为索引中文档的子集。以下示例角色将查询限制为 Tenant-1。通过将此角色应用于 Tenant-1，您可实现必要的隔离。

```
{
   "bool": {
     "must": {
       "match": {
         "tenantId": "Tenant-1"
       }
     }
   }
 }
```

混合模型

混合模型在同一环境中使用孤岛和池模型的组合，为每个租户级别（例如免费、标准和高级等级）提供独特的体验。每层均遵循池模型中使用的相同安全配置文件。

混合模型中的租户隔离 — 在混合模型中，您应遵循与池模型相同的安全配置文件，其中在文档级别使用 FGAC 安全模型可提供租户隔离。尽管此策略简化了集群管理并提供了敏捷性，但它使架构的其他方面变得复杂。例如，您的代码需要额外的复杂性来确定哪个模型与每个租户关联。您还必须确保单租户查询不会饱和整个域并降低其他租户的体验。

在 API Gateway 中测试

测试 Tenant-1 查询窗口

测试 Tenant-2 查询窗口

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS CDK 部署多堆栈应用程序 TypeScript

由 Rahul Sharad Gaikwad 博士 (AWS) 编写

摘要

此模式提供了 step-by-step一种使用 AWS Cloud Development Kit (AWS CDK) 在亚马逊网络服务 (AWS) 上部署应用程序的方法。 TypeScript例如，该模式部署无服务器实时分析应用程序。

此模式可构建和部署嵌套堆栈应用程序。父 AWS CloudFormation 堆栈调用子堆栈或嵌套堆栈。 每个子堆栈都构建和部署堆 CloudFormation 栈中定义的 AWS 资源。AWS CDK Toolkit，即命令行界面 (CLI) 命令`cdk`，是 CloudFormation 堆栈的主要接口。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 现有虚拟私有云（VPC）和子网
- AWS CDK Toolkit 已安装并配置
- 具有管理员权限且配备一组访问密钥的用户。
- Node.js
- AWS 命令行界面（AWS CLI）

限制

- 由于 AWS CDK 使用 AWS CloudFormation，因此 AWS CDK 应用程序受 CloudFormation 服务配额的限制。有关更多信息，请参阅 AWS CloudFormation 配额。

产品版本

此模式已使用以下工具和版本构建和测试。

- AWS CDK Toolkit 1.83.0
- Node.js 14.13.0
- npm 7.0.14

此模式应该适用于任何版本的 AWS CDK 或 npm。请注意，13.0.0 至 13.6.0 版本的 Node.js 与 AWS CDK 不兼容。

架构

目标技术堆栈

- AWS Amplify Console
- Amazon API Gateway
- AWS CDK
- Amazon CloudFront
- Amazon Cognito
- Amazon DynamoDB
- Amazon Data Firehose
- Amazon Kinesis Data Streams
- AWS Lambda
- Amazon Simple Storage Service(Amazon S3)

目标架构

下图显示了使用带有 AWS CDK 的多堆栈应用程序部署。 TypeScript

下图显示了示例无服务器实时应用程序架构。

工具

工具

- AWS Amplify Console 是在 AWS 中部署全栈网络和移动应用程序的控制中心。Amplify Console hosting 提供了基于 Git 的工作流程，用于托管持续部署的全栈无服务器 Web 应用程序。管理用户界面是一个为前端 Web 和移动开发人员提供的可视化界面，使其可以在 Amazon Web Services Console 外部创建和管理应用程序后端。

- A@@ mazon API Gateway 是一项 AWS 服务，用于创建、发布、维护、监控和保护任何规模的 RES WebSocket APIs T、HTTP。
- AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，可帮助您在代码中定义和预调配 Amazon Web Services Cloud 基础设施。
- AWS CDK Toolkit 是一个命令行云开发套件，可帮助您与 AWS CDK 应用程序进行交互。cdkCLI 命令是与 AWS CDK 应用程序交互的主要工具。它运行您的应用程序，查询您定义的应用程序模型，并生成和部署由 AWS CDK 生成的 AWS CloudFormation 模板。
- 亚马逊 CloudFront是一项网络服务，可加快静态和动态网页内容（例如.html、.css、.js 和图像文件）的分发。 CloudFront 通过名为边缘位置的全球数据中心网络提供内容，以降低延迟并提高性能。
- Amazon Cognito 为您的 Web 和移动应用程序提供身份验证、授权和用户管理。您的用户可以直接登录，也可通过第三方登录。
- Amazon DynamoDB 是一种全托管 NoSQL 数据库服务，提供快速而可预测的性能，能够实现无缝扩展。
- Amazon Data Firehose 是一项完全托管的服务，用于向亚马逊 S3、亚马逊 Redshift、 OpenSearch 亚马逊服务、Splunk 等目的地以及受支持的第三方服务提供商拥有的任何自定义 HTTP 终端节点或 HTTP 终端节点提供实时流式传输数据。
- Amazon Kinesis Data Streams 是一项实时收集和处理大型数据记录流的服务。
- AWS Lambda 是一项计算服务，支持无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。您只需为消耗的计算时间付费 - 代码未运行时不产生费用。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

代码

此模式代码已随附。

操作说明

安装 AWS CDK Toolkit

Task	描述	所需技能
安装 AWS CDK Toolkit。	若要在全局安装 AWS CDK Toolkit，请运行以下命令。	DevOps

Task	描述	所需技能
	`npm install -g aws-cdk`	
验证 版本。	若要验证 AWS CDK Toolkit 的版本，请运行以下命令。 `cdk --version`	DevOps

设置 AWS 凭证

Task	描述	所需技能

设置凭证。	若要设置凭证，请运行 `aws configure` 命令并按照提示进行操作。
	
	`$aws configure`
	`AWS Access Key ID`
	`[None]:`
	`AWS Secret Access Key`
	`[None]: your_secr`
	`et_access_key`
	`Default region name`
	`[None]:`
	`Default output format`
	`[None]:`
		DevOps

下载项目代码

Task	描述	所需技能
下载随附的项目代码。	有关目录和文件结构的更多信息，请参阅其他信息部分。	DevOps

引导 AWS CDK 环境

Task	描述	所需技能
引导环境。	要将 AWS CloudFormation 模板部署到您要使用的账户和 AWS 区域，请运行以下命令。 `cdk bootstrap <account>/<Region>` 有关更多信息，请参阅 AWS 文档。	DevOps

构建并部署项目

Task	描述	所需技能
构建 项目。	若要构建项目代码，请运行 `npm run build` 命令。	DevOps
部署项目。	若要部署项目代码，请运行 `cdk deploy` 命令。	

验证输出

Task	描述	所需技能
验证堆栈创建。	在 AWS 管理控制台上，选择 CloudFormation。在项目的堆栈中，验证是否已创建了一个父堆栈和两个子堆栈。	DevOps

测试应用程序

Task	描述	所需技能
将数据发送至 Kinesis 数据流。	配置您的 Amazon Web Services account，以使用 Amazon Kinesis Data Generator (KDG) 将数据发送至 Kinesis Data Streams。有关更多信息，请参阅 Amazon Kinesis Data Generator。	DevOps
创建 Amazon Cognito 用户。	要创建 Amazon Cognito 用户，请从 Kinesis 数据生成器帮助页面的 “创建亚马逊 Cognito 用户” 部分下载 cognito-setup.j CloudFormation son 模板。初始化模板，然后输入 Amazon Cognito 用户名和密码。 输出选项卡列出了 Kinesis Data Generator URL。	DevOps
登录至 Kinesis Data Generator	若要登录 KDG，请使用您提供的 Amazon Cognito 凭证和 Kinesis Data Generator URL。	DevOps
测试应用程序。	在 KDG 的记录模板、模板 1 中，粘贴其他信息部分中的测试代码，然后选择发送数据。	DevOps
测试 API 网关。	提取数据后，使用 GET 方法检索数据，以测试 API 网关。	DevOps

相关资源

参考

- AWS Cloud Development Kit
- AWS CDK 开启 GitHub
- 使用嵌套堆栈
- AWS 示例 - 无服务器实时分析

其他信息

目录与文件详细信息

此模式设置了以下三个堆栈。

- `parent-cdk-stack.ts` – 此堆栈充当父堆栈，并将两个子应用程序调用为嵌套堆栈。
- `real-time-analytics-poc-stack.ts` – 此嵌套堆栈包含基础设施和应用程序代码。
- `real-time-analytics-web-stack.ts` – 此嵌套堆栈仅包含静态 Web 应用程序代码。

重要文件及其功能

- `bin/real-time-analytics-poc.ts` – AWS CDK 应用程序的接入点。其可加载 `lib/` 定义的所有堆栈。
- `lib/real-time-analytics-poc-stack.ts` – AWS CDK 应用程序堆栈的定义 (`real-time-analytics-poc`)。
- `lib/real-time-analytics-web-stack.ts` – AWS CDK 应用程序堆栈的定义 (`real-time-analytics-web-stack`)。
- `lib/parent-cdk-stack.ts` – AWS CDK 应用程序堆栈的定义 (`parent-cdk`)。
- `package.json`— npm 模块清单，其中包含应用程序名称、版本和依赖项。
- `package-lock.json` – 由 npm 维护。
- `cdk.json` – 用于运行应用程序的工具包。
- `tsconfig.json`— 项目的 TypeScript 配置。
- `.gitignore` – Git 应从源代码中排除的文件列表。
- `node_modules` – 由 npm 维护；包括项目的依赖项。

父堆栈中的以下代码部分将子应用程序调用为嵌套 AWS CDK 堆栈。

```
import * as cdk from '@aws-cdk/core';
```

```
import { Construct, Stack, StackProps } from '@aws-cdk/core';
import { RealTimeAnalyticsPocStack } from './real-time-analytics-poc-stack';
import { RealTimeAnalyticsWebStack } from './real-time-analytics-web-stack';


export class CdkParentStack extends Stack {
  constructor(scope: Construct, id: string, props?: StackProps) {
    super(scope, id, props);


    new RealTimeAnalyticsPocStack(this, 'RealTimeAnalyticsPocStack');
    new RealTimeAnalyticsWebStack(this, 'RealTimeAnalyticsWebStack');
  }
}
```

测试代码

```
session={{date.now('YYYYMMDD')}}|sequence={{date.now('x')}}|
reception={{date.now('x')}}|instrument={{random.number(9)}}|
l={{random.number(20)}}|price_0={{random.number({"min":10000,
 "max":30000})}}|price_1={{random.number({"min":10000, "max":30000})}}|
price_2={{random.number({"min":10000, "max":30000})}}|
price_3={{random.number({"min":10000, "max":30000})}}|
price_4={{random.number({"min":10000, "max":30000})}}|
price_5={{random.number({"min":10000, "max":30000})}}|
price_6={{random.number({"min":10000, "max":30000})}}|
price_7={{random.number({"min":10000, "max":30000})}}|
price_8={{random.number({"min":10000, "max":30000})}}|
```

正在测试 API 网关

在 API Gateway 控制台上，使用 GET 方法测试 API Gateway。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS SAM 自动部署嵌套应用程序

由 Rahul Sharad Gaikwad 博士 (AWS)、Dmitry Gulin (AWS)、Ishwar Chauthaiwale (AWS) 和 Tabby Ward (AWS) 编写

摘要

在亚马逊网络服务 (AWS) 上，AWS 无服务器应用程序模型 (AWS SAM) Model 是一个开源框架，它提供用于表达函数 APIs、数据库和事件源映射的速记语法。每个资源只需几行，您就可以定义所需应用程序并使用 YAML 对其进行建模。在部署过程中，SAM 将 SAM 语法转换并扩展为 AWS CloudFormation 语法，您可以使用该语法更快地构建无服务器应用程序。

AWS SAM 简化了 AWS 平台上的无服务器应用程序的开发、部署和管理。它提供标准化框架、更快的部署、本地测试功能、资源管理、与开发工具的无缝集成以及支持社区。这些功能使其成为了高效构建无服务器应用程序的宝贵工具。

该模式使用 AWS SAM 模板自动部署嵌套应用程序。嵌套应用程序是另一应用程序中的应用程序。父应用程序调用其子应用程序。这些是无服务器架构的松耦合组件。

使用嵌套应用程序，您可重复使用独立编写和维护但使用 AWS SAM 和 Serverless Application Repository 组成的服务或组件，从而快速构建高度复杂的无服务器架构。嵌套应用程序可帮助您构建更强大的应用程序，避免重复工作，并确保整个团队和组织的一致性和最佳实践。为了演示嵌套应用程序，该模式部署了示例 AWS 无服务器购物车应用程序。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 现有的虚拟私有云（VPC）和子网
- 集成开发环境，例如 Visual Studio Code（有关更多信息，请参阅在 AWS 上构建的工具）
- 使用 pip install wheel 安装 Python wheel 库（如果尚未安装）

限制

- 无服务器应用程序中可以嵌套的最大应用程序数量为 200。
- 嵌套应用程序的最大参数数量可以是 60。

产品版本

- 此解决方案基于 AWS SAM 命令行界面 (AWS SAM CLI) 版本 1.21.1 构建，但此架构应适用于更高版本的 AWS SAM CLI。

架构

目标技术堆栈

- Amazon API Gateway
- AWS SAM
- Amazon Cognito
- Amazon DynamoDB
- AWS Lambda
- Amazon Simple Queue Service (Amazon SQS) 队列

目标架构

下图显示了用户如何通过呼叫向购物服务发出请求 APIs。用户的请求（包括所有必要信息）将发送给 Amazon API Gateway 和 Amazon Cognito 授权机构，后者负责执行身份验证和授权机制。 APIs

当在 DynamoDB 中添加、删除或更新项目时，事件会被放入 DynamoDB Streams 中，DynamoDB Streams 又会启动 Lambda 函数。为了避免在同步工作流中立即删除旧项目，将消息放到 SQS 队列中，该队列会启动工作函数来删除消息。

在此解决方案设置中，AWS SAM CLI 充当 AWS CloudFormation 堆栈的接口。AWS SAM 模板自动部署嵌套应用程序。父 SAM 模板调用子模板，父 CloudFormation 堆栈部署子堆栈。每个子堆栈都构建 AWS SAM CloudFormation 模板中定义的 AWS 资源。

1. 构建并部署堆栈。
2. 身份验证 CloudFormation 堆栈包含 Amazon Cognito。
3. 产品 CloudFormation 堆栈包含 Lambda 函数和 Amazon API Gateway
4. 购物 CloudFormation 堆栈包含 Lambda 函数、亚马逊 API Gateway、SQS 队列和亚马逊 DynamoDB 数据库。

工具

工具

- Amazon API Gateway 可帮助您创建、发布、维护、监控和保护任何规模的 RES WebSocket APIs T、HTTP。
- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。
- Amazon Cognito 为您的 Web 和移动应用程序提供身份验证、授权和用户管理。
- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Serverless Application Model (AWS SAM) 是一个帮助用于构建 AWS Cloud中无服务器应用程序的开源框架。
- Amazon Simple Queue Service (Amazon SQS) 提供了一个安全、持久且可用的托管队列，它可帮助您集成和分离分布式软件系统与组件。

代码

此模式的代码可在 GitHub AWS SAM 嵌套堆栈示例存储库中找到。

操作说明

安装 AWS SAM CLI

Task	描述	所需技能
安装 AWS SAM CLI。	要安装 AWS SAM CLI，请参阅 AWS SAM 文档中的说明。	DevOps 工程师
设置 AWS 凭证	要设置 AWS 凭证以便 AWS SAM CLI 可以代表您调用 Amazon Web Services，请运行 `aws configure` 命令并按照提示进行操作。	DevOps 工程师

Task	描述	所需技能

	`$aws configure`
	`AWS Access Key ID`
	` [None]: <your_acc`
	`ess_key_id>`
	`AWS Secret Access Key`
	` [None]: your_secr`
	`et_access_key`
	`Default region name`
	` [None]:`
	`Default output format`
	` [None]:`
	
	有关设置凭证的更多信息，请参阅身份验证和访问凭证。	

初始化 AWS SAM 项目

Task	描述	所需技能

克隆 AWS SAM 代码存储库。	1. 输入以下命令，克隆本模式的 aws sam 嵌套堆栈示例存储库
	
	`git clone https://g`
	`ithub.com/aws-samp`
	`les/aws-sam-nested-`
	`stack-sample.git`
	
	2. 通过输入以下命令导航到克隆的目录。
	
	`cd aws-sam-nested-sta`
	`ck-sample`
		DevOps 工程师

部署模板，以初始化项目。	要初始化项目，请运行 SAM `init` 命令。当系统提示您选	DevOps 工程师

Task	描述	所需技能
	择模板来源时，选择 Custom Template Location。	

编译和构建 SAM 模板代码

Task	描述	所需技能
查看 AWS SAM 应用程序模板。	查看嵌套应用程序模板。此示例使用以下嵌套应用程序模板： • auth.yaml — 此模板设置了与身份验证相关的资源，例如 Amazon Cognito 和 AWS Systems Manager Parameter Store。 • product-mock.yaml — 此模板部署与产品相关的资源，例如 Lambda 函数和 Amazon API Gateway。 • shoppingcart-service.yaml — 此模板用于设置与购物车相关的资源，例如 AWS 身份和访问管理 (IAM)、DynamoDB 表和 Lambda 函数。	DevOps 工程师
查看父级模板。	查看将调用嵌套应用程序模板的模板。在此示例中，父模板是 template.yml 。所有单独的应用程序都嵌套在单父模板 template.yml 中。	DevOps 工程师
编译和构建 AWS SAM 模板代码。	使用 AWS SAM CLI，运行以下命令。	DevOps 工程师

Task	描述	所需技能
	`sam build`	

部署 AWS SAM 模板

Task	描述	所需技能
部署应用程序。	要启动用于创建嵌套应用程序 CloudFormation 堆栈并在 AWS 环境中部署代码的 SAM 模板代码，请运行以下命令。 `sam deploy --guided --stack-name shopping-cart-nested-stack --capabilities CAPABILITY_IAM CAPABILITY_AUTO_EXPAND` 此命令将提示几个问题。用 y 回答所有问题。	DevOps 工程师

验证部署

Task	描述	所需技能
验证堆栈。	要查看在 AWS SAM 模板中定义的 AWS CloudFormation 堆栈和 AWS 资源，请执行以下操作： 1. 登录 AWS 管理控制台，然后导航到CloudFormation控制台。	DevOps 工程师

Task	描述	所需技能
	2. 确认已列出父堆栈和子堆栈。 在此示例中，`sam-shopping-cart` 是调用嵌套身份验证、产品和购物堆栈的父堆栈。 产品堆栈输出产品 API 网关网址链接。	

相关资源

参考

- AWS Serverless Application Model (AWS SAM)
- AWS SAM 开启 GitHub
- 无服务器购物车微服务（AWS 示例应用程序）

教程和视频

- 构建无服务器应用程序
- AWS 在线技术讲座：使用 AWS SAM 构建与部署无服务器应用程序

其他信息

所有代码都准备就绪后，该示例包含以下目录结构：

- sam_stacks — 此文件夹包含 `shared.py` 层。层是包含库、自定义运行时系统或其他依赖项的文件存档。利用层，您可在函数中使用库，而不必将库包含在部署包中。
- product-mock-service— 此文件夹包含所有与产品相关的 Lambda 函数和文件。
- shopping-cart-service— 此文件夹包含所有与购物相关的 Lambda 函数和文件。

使用 AWS Lambda 令牌售卖机为 Amazon S3 实施 SaaS 租户隔离

由 Tabby Ward (AWS)、Sravan Periyathambi (AWS) 和 Thomas Davis (AWS) 编写

摘要

多租户 SaaS 应用程序必须实施系统，以确保保持租户隔离。当您将租户数据存储在同一个 Amazon Web Services (AWS) 资源上时，例如多个租户将数据存储在同一个 Amazon Simple Storage Service (Amazon S3) 存储桶中，则必须确保不会发生跨租户访问。代币自动售货机 (TVMs) 是提供租户数据隔离的一种方式。这些机器提供了一种获取令牌的机制，同时减少了这些令牌生成方式的复杂性。开发人员可以在不详细了解 TVM 如何生成令牌的情况下使用 TVM。

此模式通过 AWS Lambda 实现 TVM。TVM 生成令牌，该令牌由临时安全令牌服务 (STS) 凭证组成，这些凭证限制对 S3 存储桶中单个 SaaS 租户数据的访问。

TVMs，以及此模式中提供的代码，通常与源自 JSON Web Tokens (JWTs) 的声明一起使用，以将对 AWS 资源的请求与租户范围的 AWS Identity and Access Management (IAM) 策略关联起来。您可以使用此模式中的代码为基础实现 SaaS 应用程序，该应用程序根据 JWT 令牌中提供的语句生成范围内的临时 STS 凭证。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- AWS 命令行界面（AWS CLI）版本 1.19.0 或更高版本，已在 macOS、Linux 或 Windows 上安装并配置。或者，您可以使用 AWS CLI 2.1 版或更高版本。

限制

- 此代码在 Java 中运行，当前不支持其他编程语言。
- 示例应用程序不包含 AWS 跨区域或灾难恢复 (DR) 支持。
- 此模式介绍了适用于 SaaS 应用程序的 Lambda TVM 如何提供限定范围的租户访问权限。它不用于生产环境。

架构

目标技术堆栈

- AWS Lambda
- Amazon S3
- IAM
- AWS Security Token Service (AWS STS)

目标架构

工具

Amazon Web Services

- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Security Token Service (AWS STS) 可帮助您为用户申请临时、权限有限的凭证。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

代码

此模式的源代码以附件形式提供，包含以下文件：

- `s3UploadSample.jar` 提供将 JSON 文档上传至 S3 存储桶的 Lambda 函数的源代码。
- `tvm-layer.zip` 提供了一个可重复使用的 Java 库，该库为 Lambda 函数提供用于访问 S3 存储桶和上传 JSON 文档的令牌（STS 临时凭证）。
- `token-vending-machine-sample-app.zip` 提供了用于创建这些构件的源代码和编译说明。

要使用这些文件，请按照下一节中的说明操作。

操作说明

确定变量值

Task	描述	所需技能
确定变量值。	此模式的实现包含几个必须一致使用的变量名。确定每个变量应使用的值，并在后续步骤请求时提供该值。 <AWS Account ID> – 与您实施此模式的 Amazon Web Services account 关联的 12 位账户 ID。有关如何查找您的 AWS 账号 ID 的信息，请参阅 IAM 文档中的您 Amazon Web Services account 及其别名。 <AWS Region> – 您正在实施此模式的 AWS 区域。有关 AWS 区域 的更多信息，请参阅 AWS 网站中的区域和可用区。 < sample-tenant-name > – 要在应用程序中使用的租户的名称。为简单起见，我们建议您在此值仅使用字母数字字符，但您可以为 S3 对象密钥使用任何有效名称。 < sample-tvm-role-name > – 附加到运行 TVM 和示例应用程序的 Lambda 函数的 IAM 角色的名称。角色名称是由大小写字母数字字符组成的字符串（不包含空格）。您还可以包	云管理员

Task	描述	所需技能
	含以下任何字符：下划线(_)、加号(+)、等号(=)、逗号(,)、句号(.)、@(@)、连字符(-)。角色名称在账户中必须是唯一的。 < sample-app-role-name > – Lambda 函数在生成限定范围的临时 STS 证书时所扮演的 IAM 角色的名称。角色名称是由大小写字母数字字符组成的字符串（不包含空格）。您还可以包含以下任何字符：下划线(_)、加号(+)、等号(=)、逗号(,)、句号(.)、@(@)、连字符(-)。角色名称在账户中必须是唯一的。 < sample-app-function-name > – Lambda 函数的名称。这是一个长度最多 64 个字符的字符串。 < sample-app-bucket-name > – 必须使用限定于特定租户的权限访问的 S3 存储桶的名称。S3 存储桶名称： • 长度必须介于 3-63 个字符之间。 • 只能由小写字母、数字、句点 (.) 和连字符 (-) 组成。 • 必须以字母或数字开头和结尾。 • 不得采用 IP 地址格式（例如，192.168.5.4）。	

Task	描述	所需技能
	• 在分区中必须是唯一的。分区是一组区域。AWS 目前有三个分区：aws （标准区域）、aws-cn（中国区域）aws-us-gov 和（ GovCloud AWS [美国] 区域）。	

创建 S3 存储桶

Task	描述	所需技能
为示例应用程序创建一个 S3 存储桶。	使用以下AWS CLI 命令创建 S3 存储桶。在代码片段中提供 < sample-app-bucket-name > 值： `aws s3api create-bucket --bucket <sample-app-bucket-name>` Lambda 示例应用程序将 JSON 文件上传至此存储桶。	云管理员

创建 IAM TVM 角色和策略

Task	描述	所需技能
创建 TVM 角色。	通过下列 AWS CLI 命令之一创建 IAM 角色。在命令中提供 < sample-tvm-role-name > 值。 对于 macOS 或 Linux shell： `aws iam create-role \`	云管理员

Task	描述	所需技能

	--role-name <sample-tvm-role-name> \
	--assume-role-policy-document '{
	"Version": "2012-10-17",
	"Statement": [
	{
	"Effect": "Allow",
	"Principal": {
	"Service": "lambda.amazonaws.com"
	},
	"Action": "sts:AssumeRole"
	}
	]}'	

	对于 Windows 命令行： aws iam create-role ^ --role-name <sample-tvm-role-name> ^ --assume-role-policy-document "{\"Version\": \"2012-10-17\", \"Statement\": [{\"Effect\": \"Allow\", \"Principal\": {\"Service\": \"lambda.amazonaws.com\"}, \"Action\": \"sts:AssumeRole\"}]}"	
	调用应用程序时，Lambda 示例应用程序将代入此角色。通	

Task	描述	所需技能
	过范围策略代入应用程序角色的能力，为代码提供了更广泛的权限来访问 S3 存储桶。	

Task	描述	所需技能

创建内联 TVM 角色策略。	通过下列 AWS CLI 命令之一创建 IAM policy。在< AWS Account ID>命令中提供 < sample-tvm-role-name sample-app-role-name > 、和 < > 值。
	
	对于 macOS 或 Linux shell：
	
	`aws iam put-role-policy \ --role-name <sample-tvm-role-name> \ --policy-name assume-app-role \ --policy-document '{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": "sts:AssumeRole", "Resource": "arn:aws:iam::<AWS Account ID>:role/<sample-app-role-name>" } ]}'`
	
	对于 Windows 命令行：
	
	`aws iam put-role-policy ^ --role-name <sample-tvm-role-name> ^`
		云管理员

Task	描述	所需技能

	`--policy-name assume-app-role ^`
	`--policy-document "{\"Version\": \"2012-10-17\", \"Statement\": [{\"Effect\": \"Allow\", \"Action\": \"sts:AssumeRole\", \"Resource\": \"arn:aws:iam::<AWS Account ID>:role/<sample-app-role-name>\"}]}"`
	
	此策略附加在 TVM 角色上。它使代码能够代入应用程序角色，该角色具有更广泛的 S3 存储桶访问权限。	

Task	描述	所需技能

附加托管 Lambda 策略。	使用以下 AWS CLI 命令附加 `AWSLambdaBasicExecutionRole` IAM policy。在命令中提供 < sample-tvm-role-name > 值：
	`aws iam attach-role-policy \ --role-name <sample-tvm-role-name> \ --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole`
	对于 Windows 命令行：
	`aws iam attach-role-policy ^ --role-name <sample-tvm-role-name> ^ --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole`
	此托管策略附加到 TVM 角色，允许 Lambda 向亚马逊发送日志。 CloudWatch
		云管理员

创建 IAM 应用程序角色与策略

Task	描述	所需技能
创建应用程序角色。	通过下列 AWS CLI 命令之一创建 IAM 角色。在<AWS	云管理员

Task	描述	所需技能

	Account ID>命令中提供 < sample-app-role-name sample-tvm-role-name > 、和 < > 值。
	
	对于 macOS 或 Linux shell：
	
	`aws iam create-role \ --role-name <sample-app-role-name> \ --assume-role-policy-document '{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "AWS": "arn:aws:iam::<AWS Account ID>:role/<sample-tvm-role-name>" }, "Action": "sts:AssumeRole" } ]}'`
	
	对于 Windows 命令行：
	
	`aws iam create-role ^ --role-name <sample-app-role-name> ^ --assume-role-policy-document "{\"Version\": \"2012-10-17\", \"Statement\":`	

Task	描述	所需技能
	[{\"Effect\": \"Allow\",\"Principal\": {\"AWS\": \"arn:aws:iam::<AWS Account ID>:role/<sample-tvm-role-name>\"},\"Action\": \"sts:AssumeRole\"}]}" Lambda 示例应用程序通过限定范围的策略代入此角色，以获得基于租户的 S3 存储桶访问权限。	

Task	描述	所需技能
创建内联应用程序角色策略。	通过下列 AWS CLI 命令之一创建 IAM policy。在命令中提供 < sample-app-role-name sample-app-bucket-name > 和 < > 值。 对于 macOS 或 Linux shell： `aws iam put-role-policy \ --role-name <sample-app-role-name> \ --policy-name s3-bucket-access \ --policy-document '{ "Version": "2012-10-17", "Statement": [{ "Effect": "Allow", "Action": ["s3:PutObject", "s3:GetObject", "s3:DeleteObject"], "Resource": "arn:aws:s3:::<sample-app-bucket-name>/*" }, { "Effect": "Allow", "Action": ["s3:ListBucket"],`	云管理员

Task	描述	所需技能

	<pre>"Resource
	": "arn:aws:s3:::<sam
	ple-app-bucket-name>"
	}
	]}'</pre>
	对于 Windows 命令行：
	<pre>aws iam put-role-policy
	^
	--role-name <sample-a
	pp-role-name> ^
	--policy-name s3-bucket
	-access ^
	--policy-documen
	t "{\"Version\":
	\"2012-10-17\",
	\"Statement\":
	[{\"Effect\": \"Allow
	\", \"Action\":
	[\"s3:PutObject\",
	\"s3:GetObject\",
	\"s3:DeleteObject\
	"], \"Resource\":
	\"arn:aws:s3:::<sa
	mple-app-bucket-na
	me>/*\"}, {\"Effect\":
	\"Allow\", \"Action\
	": [\"s3:ListBucket
	\"], \"Resource\":
	\"arn:aws:s3:::<sa
	mple-app-bucket-name>
	\"}]}"</pre>
	此策略附加在应用程序角色上。提供了对 S3 存储桶中对象的广泛访问权限。当示例应用程序代入该角色时，这些权	

Task	描述	所需技能
	限将限定为使用 TVM 动态生成的策略的特定租户。	

使用 TVM 创建 Lambda 示例应用程序

Task	描述	所需技能
下载编译后源文件。	下载 `s3UploadSample.jar` 和 `tvm-layer.zip` 文件，它们包含在附件内。`token-vending-machine-sample-app.zip` 中提供了用于创建这些构件的源代码和编译说明。	云管理员

创建 Lambda 层。	使用以下 AWS CLI 命令创建 Lambda 层，这样 Lambda 就可访问 TVM。
	Note
	如果您不是从下载位置运行此命令 `tvm-layer.zip` ，请在`--zip-file` 参数`tvm-layer.zip` 中提供正确的路径。
	`aws lambda publish-layer-version \`
	`--layer-name sample-token-vending-machine \`
	`--compatible-runtimes java11 \`
		云管理员、应用程序开发人员

Task	描述	所需技能

	`--zip-file fileb://t vm-layer.zip`
	
	对于 Windows 命令行：
	
	`aws lambda publish-l ayer-version ^ --layer-name sample-to ken-vending-machine ^ --compatible-runtimes java11 ^ --zip-file fileb://t vm-layer.zip`
	
	此命令创建 Lambda 层，其中包含可重复使用的 TVM 库。	

Task	描述	所需技能

创建 Lambda 函数。	使用以下 AWS CLI 命令创建 Lambda 函数。在<AWS Account ID><AWS Region>命令中提供 < sample-app-function-name sample-tvm-role-name > 、、、< sample-app-bucket-name >、< sample-app-role-name > 和 < > 值。
	
	Note
	如果您不是从下载位置运行此命令`s3UploadSample.jar`，请在`--zip-file`参数`s3UploadSample.jar`中提供正确的路径。
	
	`aws lambda create-function \`
	`--function-name <sample-app-function-name> \`
	`--timeout 30 \`
	`--memory-size 256 \`
	`--runtime java11 \`
	`--role arn:aws:iam::<AWS Account ID>:role/<sample-tvm-role-name> \`
	`--handler com.amazon.aws.s3UploadSample.App \`
		云管理员、应用程序开发人员

Task	描述	所需技能
	--zip-file fileb://s3UploadSample.jar \ --layers arn:aws:lambda:<AWS Region>:<AWS Account ID>:layer:sample-token-vending-machine:1 \ --environment "Variables={S3_BUCKET=<sample-app-bucket-name>, ROLE=arn:aws:iam::<AWS Account ID>:role/<sample-app-role-name>}" 对于 Windows 命令行： aws lambda create-function ^ --function-name <sample-app-function-name> ^ --timeout 30 ^ --memory-size 256 ^ --runtime java11 ^ --role arn:aws:iam::<AWS Account ID>:role/<sample-tvm-role-name> ^ --handler com.amazon.aws.s3UploadSample.App ^ --zip-file fileb://s3UploadSample.jar ^ --layers arn:aws:lambda:<AWS Region>:<AWS Account ID>:layer:sample-token-vending-machine:1 ^ --environment "Variables={S3_BUCKET=<samp	

Task	描述	所需技能
	le-app-bucket-name >,ROLE=arn:aws:iam ::<AWS Account ID>:role/<sample-app-role-name>}" 此命令创建 Lambda 函数，其中包含示例应用程序代码和附加的 TVM 层。它还设置了两个环境变量：S3_BUCKET 和 ROLE。示例应用程序使用这些变量来确定要代入的角色以及要将 JSON 文档上传到的 S3 存储桶。	

测试示例应用程序和 TVM。

Task	描述	所需技能
调用 Lambda 示例应用程序。	使用下列 AWS CLI 命令之一启动有预期有效负载的 Lambda 示例应用程序。在命令中提供 < sample-app-function-name sample-tenant-name > 和 < > 值。 对于 macOS 和 Linux shell： aws lambda invoke \ --function <sample-app-function-name> \ --invocation-type RequestResponse \ --payload '{"tenant": "<sample-tenant-name>"}' \	云管理员、应用程序开发人员

Task	描述	所需技能
	--cli-binary-format raw-in-base64-out response.json 对于 Windows 命令行： aws lambda invoke ^ --function <sample-app-function-name> ^ --invocation-type RequestResponse ^ --payload "{\"tenant\": \"<sample-tenant-name>\"}" ^ --cli-binary-format raw-in-base64-out response.json 此命令调用 Lambda 函数并在 response.json 文档中返回结果。在许多基于 Unix 的系统上，您可以将 response.json 更改为 /dev/stdout ，将结果直接输出到 Shell，而无需创建其他文件。 **Note** 在后续调用此 Lambda 函数时更改 < sample-tenant-name > 值会改变 JSON 文档的位置和令牌提供的权限。	

Task	描述	所需技能
查看 S3 存储桶，以查看创建的对象。	浏览到您之前创建的 S3 存储桶 (< sample-app-bucket-name >)。此存储桶包含一个 S3 对象前缀，其值为 < sample-tenant-name >。在此前缀下，您将找到以 UUID 命名的 JSON 文档。多次调用示例应用程序将添加更多 JSON 文档。	云管理员

Task	描述	所需技能
查看示例应用程序 Cloudwatch 日志。	查看与名为 sample-app-function-name < > 的 Lambda 函数关联的 Cloudwatch 日志。有关说明，请参阅 AWS Lambda 文档中的访问 AWS Lambda 的亚马逊 CloudWatch 日志。您可在这些日志中查看 TVM 生成的租户范围策略。此租户范围策略向 Amazon S3、、、和授予使用示例应用程序的权限 PutObjectGetObjectDeleteObjectListBucket APIs，但仅限于与 < > 关联的对象前缀。sample-tenant-name在后续调用示例应用程序时，如果您更改 < sample-tenant-name >，TVM 会更新作用域策略，使其与调用负载中提供的租户相对应。此动态生成的策略显示了如何在 SaaS 应用程序中使用 TVM 维护租户范围访问权限。 TVM 功能在 Lambda 层中提供，因此无需复制代码即可将其附加至应用程序使用的其他 Lambda 函数。 有关动态生成的策略的说明，请参阅其他信息部分。	云管理员

相关资源

- 使用动态生成的 IAM policy 隔离租户（博客文章）

- 在 SaaS 环境中应用动态生成的隔离策略（博客文章）
- AWS SaaS Boost（一种开源参考环境，可帮助您将自己的 SaaS 产品迁移到 AWS 上）

其他信息

以下 Amazon CloudWatch 日志显示了在这种模式下由 TVM 代码生成的动态生成的策略。在此屏幕截图中，< sample-app-bucket-name > 是DOC-EXAMPLE-BUCKET，< sample-tenant-name > 是`test-tenant-1`。此范围策略返回的 STS 凭证无法对 S3 存储桶中的对象执行任何操作，但与对象密钥前缀 `test-tenant-1` 关联的对象除外。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS Step Functions 实施无服务器 saga 模式

由 Tabby Ward (AWS)、Rohan Mehta (AWS) 和 Rimpy Tewani (AWS) 编写

摘要

在微服务架构中，主要目标是构建解耦且独立的组件，以提高应用程序的敏捷性、灵活性和更快的上市时间。解耦后，每个微服务组件都配有自己的数据持久层。在分布式架构中，业务事务可跨越多个微服务。由于这些微服务不能使用单个原子性、一致性、隔离性、持久性 (ACID) 事务，最终可能会出现不完全的事务。在这种情况下，需要一些控制逻辑撤销已经处理的事务。分布式 saga 模式通常用于该目的。

Saga 模式是一种故障管理模式，有助于在分布式应用程序中建立一致性，并协调多个微服务之间的事务以保持数据一致性。当您使用 saga 模式时，每个执行事务的服务都会发布事件，该事件会触发后续服务执行链中的下一个事务。这种情况一直持续至链中的最后一笔事务完成。如果业务事务处理失败，saga 会编排一系列补偿性事务，以撤销先前事务所做的更改。

此模式演示了如何使用 AWS Step Functions、AWS Lambda 和 Amazon DynamoDB 等无服务器技术自动设置和部署示例应用程序（用于处理差旅预订）。示例应用程序还使用 Amazon API Gateway 和 Amazon Simple Notification Service (Amazon SNS) 实现 saga 执行协调器。该模式可以通过 AWS Cloud Development Kit (AWS CDK)、AWS Serverless Application Model (AWS SAM) 或 Terraform 等基础设施即代码（IaC）框架进行部署。

有关 saga 模式和其他数据持久性模式的更多信息，请参阅 AWS Prescriptive Guidance 网站上的在微服务中启用数据持久性指南。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 创建 AWS CloudFormation 堆栈的权限。有关更多信息，请参阅 CloudFormation 文档中的控制访问权限。
- 使用您的 Amazon Web Services account 配置您选择的 IaC 框架（AWS CDK、AWS SAM 或 Terraform），以便您可使用框架 CLI 部署应用程序。
- NodeJS，用于在本地构建和运行应用程序。
- 您选择的代码编辑器（例如 Visual Studio Code、Sublime 或 Atom）。

产品版本

- NodeJS 版本 14
- AWS CDK 版本 2.37.1
- AWS SAM 版本 1.71.0
- Terraform 版本 1.3.7

限制

事件源是在微服务架构中实现 saga 编排模式的一种自然方式，在这种架构中，所有组件都是松耦合的，彼此之间没有直接的了解。如果您的事务涉及少量步骤（三到五步），那么 saga 模式可能非常合适。但是，复杂性会随微服务数量和步骤数量的增加而增加。

使用此类设计时，测试和调试可能会变得困难，因为必须运行所有服务才能模拟事务模式。

架构

目标架构

拟议架构使用 AWS Step Functions 构建 saga 模式，用于预订航班、预订租车和处理度假付款。

以下工作流图介绍了旅行预订系统的典型流程。工作流程包括预订航空旅行 (“ReserveFlight”)、预订汽车 (“ReserveCarRental”)、处理付款 (“ProcessPayment”)、确认航班预订 (“ConfirmFlight”) 和确认租车 (“ConfirmCarRental”)，然后在这些步骤完成后发出成功通知。但是，如果系统在运行其中任何一个事务时遇到任何错误，它会开始向后失败。例如，付款处理错误 (“ProcessPayment”) 会触发退款 (“RefundPayment”)，然后退款会触发取消租车和航班（“CancelRentalReservation” 和 “CancelFlightReservation”），从而以失败消息结束整个交易。

这种模式为图表中突出显示的每项任务部署了单独的 Lambda 函数，还为航班、汽车租赁和付款部署了三项 DynamoDB 表。每个 Lambda 函数都创建、更新或删除相应的 DynamoDB 表中的行，具体取决于事务是确认还是回滚。该模式使用 Amazon SNS 向订阅用户发送短信 (SMS) 消息，通知其事务失败或成功。

自动化和扩缩

您可以使用其中一个 IaC 框架为此架构创建配置。通过以下链接之一获取首选 IaC。

- 使用 AWS CDK 部署
- 使用 AWS SAM 部署
- 使用 Terraform 部署

工具

Amazon Web Services

- AWS Step Functions 是一项无服务器编排服务，可让您搭配使用 AWS Lambda 函数和其他 Amazon Web Services 来构建业务关键型应用程序。通过 Step Functions 图形控制台，您可以将应用程序的工作流视为一系列事件驱动的步骤。
- Amazon DynamoDB 是一种全托管 NoSQL 数据库服务，提供快速而可预测的性能，能够实现无缝扩展。您可以使用 DynamoDB 创建一个数据库表来存储和检索任意量级的数据，并支持任何级别的请求流量。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。
- A@@ mazon API Gateway 是一项 AWS 服务，用于创建、发布、维护、监控和保护任何规模的 RES WebSocket APIs T、HTTP。
- Amazon Simple Notification Service (Amazon SNS) 是一项托管服务，提供从发布者至订阅用户的消息传输。
- AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，用于使用熟悉的编程语言（例如，Python TypeScript JavaScript、Java 和 C#/Net）来定义您的云应用程序资源。
- AWS Serverless Application Model (AWS SAM) 是一个用于构建无服务器应用程序的开源框架。它提供了用于表达函数 APIs、数据库和事件源映射的速记语法。

代码

可以在以下链接中找到演示 saga 模式的示例应用程序的代码，包括 IaC 模板 (AWS CDK、AWS SAM 或 Terraform)、Lambda 函数和 DynamoDB 表。按照首个操作说明中的说明安装这些工具。

- 使用 AWS CDK 部署
- 使用 AWS SAM 部署
- 使用 Terraform 部署

操作说明

安装软件包、编译与构建

Task	描述	所需技能
安装 NPM 软件包。	创建一个新目录，在终端中导航到该目录，然后在此模式前面的 “代码” 部分中克隆您选择的 GitHub 存储库。 在包含 `package.json` 文件的根文件夹中，运行以下命令下载并安装所有 Node Package Manager (NPM) 软件包： `npm install`	开发人员、云架构师
编译脚本。	在根文件夹中，运行以下命令以指示 TypeScript 转译器创建所有必需 JavaScript 的文件： `npm run build`	开发人员、云架构师
注意更改和重新编译。	在根文件夹，在单独的终端窗口中运行以下命令，以监视代码更改，并在检测到更改时编译代码： `npm run watch`	开发人员、云架构师
运行单元测试（仅限 AWS CDK）。	如果您使用的是 AWS CDK，请在根文件夹中运行以下命令，以执行 Jest 单元测试： `npm run test`	开发人员、云架构师

将资源部署至目标 Amazon Web Services account

Task	描述	所需技能

将演示堆栈部署至 AWS。	**Important**
	该应用程序与 AWS 区域无关。如果您使用配置文件，则必须在 AWS 命令行界面（AWS CLI）配置文件中或通过 AWS CLI 环境变量明确声明区域。
	
	在根文件夹中，运行以下命令，以创建部署程序集并将其部署至默认 Amazon Web Services account 和区域。
	
	AWS CDK：
	`cdk bootstrap`
	`cdk deploy`
	
	AWS SAM：
	`sam build`
	`sam deploy --guided`
	
	Terraform：
	`terraform init`
	`terraform apply`
		开发人员、云架构师

Task	描述	所需技能
	此步骤可能需要几分钟时间才能完成。此命令使用为 AWS CLI 配置的默认凭证。 记下部署完成后控制台上所示 API Gateway 网址。您将需要这些信息测试 saga 的执行流程。	

将已部署的堆栈与当前状态比较。	在根文件夹中，运行以下命令，将已部署的堆栈与更改源代码后的当前状态比较：
	
	AWS CDK：
	
	`cdk diff`
	
	AWS SAM：
	
	`sam deploy`
	
	Terraform：
	
	`terraform plan`
		开发人员、云架构师

测试执行流程

Task	描述	所需技能
测试 saga 执行流程。	导航至您在部署堆栈时在前面的步骤中记下的 API Gateway 网址。此 URL 将触发状态机启动。有关如何通过传递不同的 URL 参数操纵状态机流程的	开发人员、云架构师

Task	描述	所需技能
	更多信息，请参阅其他信息部分。 要查看结果，请登录 AWS 管理控制台 并导航到 Step Functions 控制台。在这里，您可以看到 saga 状态机的每一个步骤。您还可以查看 DynamoDB 表以查看已插入、更新或删除记录。如果您经常刷新屏幕，则可以看到事务状态从 `pending` 变为 `confirmed` 。 您可以通过使用您的手机号码更新 `stateMachine.ts` 文件中的代码，以订阅 SNS 主题，以便在预订成功或失败时接收 SMS 消息。有关更多信息，请参阅其他信息部分中的 Amazon SNS。	

清理

Task	描述	所需技能
清理资源。	若要清理为此应用程序部署的资源，可以使用以下命令之一。 AWS CDK： `cdk destroy` AWS SAM：	应用程序开发人员、云架构师

Task	描述	所需技能

	`sam delete`
	
	Terraform：
	
	`terraform destroy`	

相关资源

技术论文

- 在 AWS 上实施微服务
- 无服务器应用程序剖析
- 在微服务中启用数据持久性

Amazon Web Services 文档

- AWS CDK 入门
- AWS SAM 入门
- AWS Step Functions
- Amazon DynamoDB
- AWS Lambda
- Amazon API Gateway
- Amazon SNS

教程

- 无服务器计算实践研讨会

其他信息

代码

出于测试目的，此模式部署了 API Gateway 和用于触发 Step Functions 状态机的测试 Lambda 函数。使用 Step Functions，您可以通过传递`run_type`参数来模仿 “、”、“” ReserveFlight、“” 和 “ReserveCarRental” 中的故障ProcessPayment，从而控制旅行预订系统的功能 ConfirmCarRental。ConfirmFlight

`saga` Lambda 函数 (`sagaLambda.ts`) 从 API Gateway 网址中的查询参数中获取输入，创建以下 JSON 对象，然后将其传递至 Step Functions 执行：

```
let input = {
"trip_id": tripID, //  value taken from query parameter, default is AWS request ID
"depart_city": "Detroit",
"depart_time": "2021-07-07T06:00:00.000Z",
"arrive_city": "Frankfurt",
"arrive_time": "2021-07-09T08:00:00.000Z",
"rental": "BMW",
"rental_from": "2021-07-09T00:00:00.000Z",
"rental_to": "2021-07-17T00:00:00.000Z",
"run_type": runType // value taken from query parameter, default is "success"
};
```

您可通过传递以下 URL 参数试验 Step Functions 状态机的不同流程：

- 成功执行 – https://{api gateway url}
- 预订航班失败 – https://{api gateway url}？ runType= failFlightsReservation
- 确认飞行失败 – https://{api gateway url}？ runType= failFlightsConfirmation
- 预约租车失败 – https://{api gateway url}？ runType= 预留 failCarRental
- 确认租车失败 – https://{api gateway url}？ runType= 确认 failCarRental
- 处理付款失败 – https://{api gateway url}?runType=failPayment
- 传递行程 ID – https://{api gateway url}?tripID={默认情况下，行程 ID 是 AWS 请求 ID}

IaC 模板

链接的存储库包含 IaC 模板，您可使用这些模板来创建整个示例旅行预订应用程序。

- 使用 AWS CDK 部署
- 使用 AWS SAM 部署
- 使用 Terraform 部署

DynamoDB 表

以下是关于航班、租车和付款表的数据模型。

```
Flight Data Model:
 var params = {
      TableName: process.env.TABLE_NAME,
      Item: {
        'pk' : {S: event.trip_id},
        'sk' : {S: flightReservationID},
        'trip_id' : {S: event.trip_id},
        'id': {S: flightReservationID},
        'depart_city' : {S: event.depart_city},
        'depart_time': {S: event.depart_time},
        'arrive_city': {S: event.arrive_city},
        'arrive_time': {S: event.arrive_time},
        'transaction_status': {S: 'pending'}
      }
    };

Car Rental Data Model:
var params = {
      TableName: process.env.TABLE_NAME,
      Item: {
        'pk' : {S: event.trip_id},
        'sk' : {S: carRentalReservationID},
        'trip_id' : {S: event.trip_id},
        'id': {S: carRentalReservationID},
        'rental': {S: event.rental},
        'rental_from': {S: event.rental_from},
        'rental_to': {S: event.rental_to},
        'transaction_status': {S: 'pending'}
      }
    };

Payment Data Model:
var params = {
      TableName: process.env.TABLE_NAME,
      Item: {
        'pk' : {S: event.trip_id},
        'sk' : {S: paymentID},
        'trip_id' : {S: event.trip_id},
        'id': {S: paymentID},
```

```
        'amount': {S: "750.00"}, // hard coded for simplicity as implementing any
 monetary transaction functionality is beyond the scope of this pattern
        'currency': {S: "USD"},
        'transaction_status': {S: "confirmed"}
      }
    };
```

Lambda 函数

将创建以下函数，以支持 Step Functions 中的状态机流程和执行：

- 预订航班：在 DynamoDB 航班表中插入一条带有 `pending` 的 `transaction_status` 的记录以预订航班。
- 确认航班：更新 DynamoDB 航班表中的记录，将 `transaction_status` 设置为 `confirmed`，以确认航班。
- 取消航班预订：从 DynamoDB 航班表内删除记录，以取消待处理的航班。
- 预订租车：在 Dynamo CarRentals DB 表中插入一条带有 `transaction_status` a `pending` 的记录以预订租车。
- 确认租车：更新 Dynamo CarRentals DB 表中的记录，将其`transaction_status`设置为，`confirmed`以确认租车。
- 取消租车预订：从 Dynamo CarRentals DB 表中删除该记录，以取消待处理的租车。
- 处理付款：在 DynamoDB 付款表中插入付款记录。
- 取消付款：从 DynamoDB 付款表删除付款记录。

Amazon SNS

该示例应用程序创建了以下主题和订阅，用于发送 SMS 消息，并通知客户预订成功或失败。如果您想在测试示例应用程序时接收短信，请在状态机定义文件中使用有效电话号码更新 SMS 订阅。

AWS CDK 片段（在以下代码的第二行中添加电话号码）：

```
const topic = new  sns.Topic(this, 'Topic');
topic.addSubscription(new subscriptions.SmsSubscription('+11111111111'));
const snsNotificationFailure = new tasks.SnsPublish(this ,'SendingSMSFailure', {
topic:topic,
integrationPattern: sfn.IntegrationPattern.REQUEST_RESPONSE,
message: sfn.TaskInput.fromText('Your Travel Reservation Failed'),
});
```

```
const snsNotificationSuccess = new tasks.SnsPublish(this ,'SendingSMSSuccess', {
topic:topic,
integrationPattern: sfn.IntegrationPattern.REQUEST_RESPONSE,
message: sfn.TaskInput.fromText('Your Travel Reservation is Successful'),
});
```

AWS SAM 片段（用您的有效电话号码替换 +1111111111 字符串）：

```
  StateMachineTopic11111111111:
    Type: 'AWS::SNS::Subscription'
    Properties:
      Protocol: sms
      TopicArn:
        Ref: StateMachineTopic
      Endpoint: '+11111111111'
    Metadata:
      'aws:sam:path': SamServerlessSagaStack/StateMachine/Topic/+11111111111/Resource
```

Terraform 片段（用您的有效电话号码替换 +111111111 字符串）：

```
resource "aws_sns_topic_subscription" "sms-target" {
  topic_arn = aws_sns_topic.topic.arn
  protocol  = "sms"
  endpoint  = "+11111111111"
}
```

成功预订

以下流程说明了成功的预订ReserveFlight，“” 和 “ReserveCarRentalProcessPayment” 后面是 “ConfirmFlight” 和 “” ConfirmCarRental。通过发送给 SNS 主题订阅用户的短信告知客户预订成功。

预订失败

此流程是 saga 模式失败的一个例子。如果在预订航班和租车后，“ProcessPayment” 失败，则按相反的顺序取消步骤。 预订已解除，并通过发送至 SNS 主题订阅用户的 SMS 消息通知客户失败。

通过使用 AWS CDK 设置 Amazon ECS Anywhere 来管理本地容器应用程序

由 Rahul Sharad Gaikwad 博士 (AWS) 编写

摘要

Amazon ECS Anywhere 是 Amazon Elastic Container Service (Amazon ECS)的扩展。您可以使用 ECS Anywhere 在本地或客户托管环境中部署本机 Amazon ECS 任务。此功能有助于降低成本，并减少复杂的本地容器编排和操作。您可以使用 ECS Anywhere 在本地和云环境中部署和运行容器应用程序。它使您的团队无需学习多个域和技能组合，也无需自行管理复杂的软件。

此模式演示了使用 AWS Cloud Development Kit (AWS CDK)堆栈设置 ECS Anywhere 的步骤。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 已安装和配置 AWS 命令行界面（AWS CLI）。（请参阅 AWS CLI 文档中的安装、更新和卸载 AWS CLI。）
- AWS CDK Toolkit，已安装并配置。（请参阅 AWS CDK 文档中的 AWS CDK 工具包，并按照说明在全球范围内安装版本 2。）
- 节点包管理器 (npm)，已为中的 AWS CDK 安装和配置。 TypeScript（请参阅 npm 文档中的下载和安装 Node.js 和 npm。）

限制

- 有关限制和注意事项，请参阅 Amazon ECS 文档中的外部实例 (Amazon ECS Anywhere)。

产品版本

- AWS CDK 工具包第 2 版
- npm 版本 7.20.3 或更高版本
- Node.js 版本 16.6.1 或更高版本

架构

目标技术堆栈

- AWS CloudFormation
- AWS CDK
- Amazon ECS Anywhere
- AWS Identity and Access Management（AWS IAM）

目标架构

下图说明了使用 AWS CDK 和 ECS Anywhere 设置的高级系统架构 TypeScript，如该模式所实现的那样。

1. 当您部署 AWS CDK 堆栈时，它会在 AWS 上创建一个 CloudFormation 堆栈。
2. 该 CloudFormation 堆栈预配置 Amazon ECS 集群和相关的 AWS 资源。
3. 要将外部实例注册到 Amazon ECS 集群，您必须在虚拟机 (VM) 上安装 AWS Systems Manager Agent (SSM Agent)代理，并将该 VM 注册为 AWS Systems Manager 托管实例。
4. 对于您向 Amazon ECS 集群注册的每个外部实例，必须安装 SSM Agent、Amazon ECS 容器代理和 Docker。
5. 当外部实例注册并配置到 Amazon ECS 集群时，它可以在注册为外部实例的虚拟机上运行多个容器。

自动化和扩缩

此模式提供的GitHub 存储库使用 AWS CDK 作为基础设施即代码 (IaC) 工具来创建该架构的配置。AWS CDK 可帮助您编排资源，并设置 ECS Anywhere。

工具

- AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，可帮助您在代码中定义和预置 Amazon Web Services Cloud 基础设施。
- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。

代码

此模式的源代码可在 GitHub Amazon ECS Anywhere CDK 示例存储库中找到。若要克隆和使用存储库，请按照下一节中的说明进行操作。

操作说明

验证 AWS CDK 配置

Task	描述	所需技能
验证 AWS CDK 版本。	运行以下命令验证 AWS CDK Toolkit 的版本： `cdk --version` 此模式需要 AWS CDK 版本 2。如果您使用 AWS CDK 早期版本，请按照 AWS CDK 文档中的说明对其进行更新。	DevOps 工程师

设置 AWS 凭证。	要设置凭证，请运行 `aws configure` 命令并按照提示进行操作：
	`$aws configure`
	`AWS Access Key ID [None]: <your-access-key-ID>`
	`AWS Secret Access Key [None]: <your-secret-access-key>`
	`Default region name [None]: <your-Region-name>`
	`Default output format [None]:`
		DevOps 工程师

引导 AWS CDK 环境

Task	描述	所需技能
克隆 AWS CDK 代码存储库。	使用以下命令克隆此模式的 GitHub 代码存储库： `git clone https://github.com/aws-samples/amazon-ecs-anywhere-cdk-samples.git`	DevOps 工程师
引导环境。	要将 AWS CloudFormation 模板部署到您要使用的账户和 AWS 区域，请运行以下命令： `cdk bootstrap <account-number>/<Region>` 有关更多信息，请参阅 AWS CDK 文档中的引导。	DevOps 工程师

构建和部署项目

Task	描述	所需技能

安装软件包依赖关系并编译 TypeScript 文件。	通过运行以下命令安装软件包依赖关系并编译 TypeScript 文件：
	`$cd amazon-ecs-anywhere-cdk-samples`
	`$npm install`
	`$npm fund`
	这些命令安装示例存储库内的所有软件包。
		DevOps 工程师

Task	描述	所需技能

	Important
	如果您遇到任何关于丢失软件包的错误，请使用以下命令之一：
	`$npm ci`
	—或者—
	`$npm install -g @aws-cdk/<package_name>`
	有关更多信息，请参阅 npm 文档中的 npm ci 和 npm install。	

构建 项目。	若要生成项目代码，请运行以下命令： `npm run build` 有关构建和部署项目的更多信息，请参阅 AWS CDK 文档中的您的第一个 AWS CDK 应用程序。	DevOps 工程师
部署项目。	若要部署项目代码，请运行以下命令： `cdk deploy`	DevOps 工程师

Task	描述	所需技能
验证堆栈创建和输出。	在 https://console.aws.amazon.com/cloudformation 上打开 AWS CloudFormation 控制台，然后选择堆栈。`EcsAnywhereStack` “输出” 选项卡显示要在外部 VM 上运行的命令。	DevOps 工程师

设置本地计算机

Task	描述	所需技能
使用 Vagrant 设置您的 VM。	出于演示目的，你可以使用 V HashiCorp agrant 来创建虚拟机。Vagrant 是一个开源实用程序，用于构建和维护便携式虚拟软件开发环境。通过从放置 Vagrantfile 的根目录下运行 `vagrant up` 命令来创建 Vagrant VM。有关更多信息，请参阅 Packer 文档。	DevOps 工程师
将您的虚拟机注册为外部实例。	1. 使用 `vagrant ssh` 命令登录 Vagrant 虚拟机。有关更多信息，请参阅 Packer 文档。 2. 创建激活码和 ID，您可以使用该激活码和 ID 在 AWS Systems Manager 注册虚拟机并激活外部实例。此命令的输出包括 `ActivationId` 和 `ActivationCode` 值： `aws ssm create-activation --iam-role`	DevOps 工程师

Task	描述	所需技能
	EcsAnywhereInstanc eRole | tee ssm-activ ation.json 3. 导出激活 ID 和代码值： export ACTIVATIO N_ID=<activation-ID> export ACTIVATIO N_CODE=<activation- code> 4. 在本地服务器或虚拟机 (VM) 上下载安装脚本： curl -o "ecs-anywhere- install.sh" "https:// amazon-ecs-agent.s 3.amazonaws.com/ec s-anywhere-install -latest.sh" && sudo chmod +x ecs-anywhere- install.sh 5. 在本地服务器或虚拟机 (VM) 上下载运行脚本： sudo ./ecs-anywhere-ins tall.sh \ --cluster test-ecs- anywhere \ --activation-id $ACTIVATION_ID \ --activation-code $ACTIVATION_CODE \ --region <Region>	

Task	描述	所需技能
	有关设置和注册 VM 的更多信息，请参阅 Amazon ECS 文档中的将外部实例注册到集群。	

验证 ECS Anywhere 和外部虚拟机的状态。	要验证您的虚拟盒子是否已连接到 Amazon ECS 控制面板并正在运行，请使用以下命令：
	`aws ssm describe-instance-information`
	`aws ecs list-container-instances --cluster $CLUSTER_NAME`
		DevOps 工程师

清理

Task	描述	所需技能
清理和删除资源。	完成此模式后，应删除您创建的资源，以避免产生任何进一步的费用。若要进行清理，请运行以下命令： `cdk destroy`	DevOps 工程师

相关资源

- Amazon ECS Anywhere 文档
- Amazon ECS Anywhere 演示（视频）
- Amazon ECS Anywhere 研讨会示例

在 AWS 上实现 ASP.NET Web 表单应用程序的现代化

由 Vijai Anand Ramalingam (AWS) 和 Sreelaxmi Pai (AWS) 编写

摘要

此模式描述了通过将传统的整体式 ASP.NET Web Forms 应用程序移植到 AWS 上的 ASP.NET Core，对其进行现代化改造的步骤。

将 ASP.NET Web 窗体应用程序移植到 ASP.NET Core 可帮助您利用 Linux 的性能、成本节约和强大的生态系统。不过，这可能需要大量的人工操作。在此模式中，使用分阶段方法逐步对旧应用程序进行现代化改造，然后在 Amazon Web Services Cloud 中容器化。

考虑一个用于购物车的传统单体应用程序。假设它是作为 ASP.NET Web 窗体应用程序创建的，并且由带有代码隐藏 (`aspx.cs`) 文件的 .aspx 页组成。现代化过程包括以下步骤：

1. 使用适当的分解模式将单体分解为微服务。有关更多信息，请参阅 AWS Prescriptive Guidance 网站上的指南将单体架构分解为微服务。
2. 将传统 ASP.NET Web 窗体 (.NET Framework) 应用程序移植到 .NET 5 或更高版本中的 ASP.NET Core。在此模式中，使用 Porting Assistant for .NET 扫描 ASP.NET Web 窗体应用程序，并确定与 ASP.NET Core 的不兼容。这减少了手动移植的工作量。
3. 使用 React 重新开发 Web 表单用户界面层。此模式不涵盖 UI 重新开发。有关说明，请参阅 React 文档中的创建新的 React 应用程序。
4. 将 Web 窗体代码后置文件（业务接口）重新开发为 ASP.NET Core Web API。此模式使用 NDepend 报告来帮助识别所需的文件和依赖关系。
5. 使用 Porting Assistant for .NET 将旧应用程序中的共享/通用项目（例如业务逻辑和数据访问）升级到 .NET 5 或更高版本。
6. 添加 Amazon Web Services 来补充您的应用程序。例如，您可以使用 Amazon CloudWatch Logs 来监控、存储和访问应用程序的日志，使用 AWS Systems Manager 来存储您的应用程序设置。
7. 将现代化的 ASP.NET Core 应用程序容器化。此模式在 Visual Studio 中创建一个面向 Linux 的 Docker 文件，并使用 Docker Desktop 在本地对其进行测试。此步骤假设您的旧版应用程序已在本地或亚马逊弹性计算云 (Amazon EC2) Windows 实例上运行。有关更多信息，请参阅在亚马逊 EC2 Linux 实例上运行 ASP.NET Core Web API Docker 容器的模式。
8. 将现代化的 ASP.NET Core 应用程序部署到 Amazon Elastic Container Service (Amazon ECS)。此模式不涵盖部署步骤。有关说明，请参阅 Amazon ECS 研讨会。

> **Note**
>
> 这种模式不包括用户界面开发、数据库现代化或容器部署步骤。

先决条件和限制

先决条件

- Visual Studio 或 Visual Studio Code，已下载并安装。
- 使用 AWS 管理控制台和 AWS 命令行界面(AWS CLI) 版本 2 访问 Amazon Web Services account。（请参阅配置 AWS CLI 的说明。）
- Visual Studio 的 AWS Toolkit（请参阅设置说明）。
- Docker Desktop，已下载并安装。
- .NET SDK，已下载并安装。
- NDepend 工具，已下载并安装。要安装 Visual Studio 的 NDepend 扩展程序，请运行`NDepend.VisualStudioExtension.Installer`（参见说明书）。可以选择 Visual Studio 2019 或 2022，具体取决于你的要求。
- Porting Assistant for .NET，已下载并安装。

架构

实现购物车应用程序的现代化

下图演示了旧版 ASP.NET 购物车应用程序的现代化过程。

目标架构

下图说明了 AWS 上现代化购物车应用程序的架构。ASP.NET Cor APIs e Web 部署到亚马逊 ECS 集群。日志和配置服务由 Amazon CloudWatch Logs 和 AWS Systems Manager 提供。

工具

Amazon Web Services

- Amazon ECS - Amazon Elastic Container Service (Amazon ECS) 是一项高度可扩展的快速容器管理服务，可助您轻松运行、停止和管理集群上的容器。您可以在由 AWS Fargate 托管的无服务器基础设施上运行任务和服务。或者，为了更好地控制您的基础架构，您可以在自己管理的 EC2 实例集群上运行任务和服务。
- Amazon CloudWatch 日 CloudWatch 志 — Amazon Logs 集中您使用的所有系统、应用程序和 AWS 服务的日志。您可以查看和监控日志，搜索特定的错误代码或模式，根据特定字段过滤日志，或将日志安全存档以备将来分析。
- AWS Systems Manager - AWS Systems Manager 是一项 Amazon Web Services，可用于查看和控制 AWS 上的基础设施。使用 Systems Manager 控制台，您可查看来自多个 Amazon Web Services 的操作数据并在 AWS 资源之间自动执行操作任务。Systems Manager 通过扫描托管实例并报告其检测到的所有策略违规行为（或采取纠正措施）来帮助您维护安全性与合规性。

工具

- Visual Stud io 或 Visual Studio Code — 用于构建.NET 应用程序 APIs、Web 和其他程序的工具。
- AWS Toolkit for Visual Studio - Visual Studio 的扩展，可帮助开发、调试和部署使用 Amazon Web Services 的 .NET 应用程序。
- Docker Desktop - 一种简化构建和部署容器化应用程序的工具。
- NDepend— 一种用于监视.NET 代码的依赖关系、质量问题和代码更改的分析器。
- Porting Assistant for .NET - 一种分析工具，用于扫描 .NET 代码以识别与 .NET Core 的不兼容之处并估计迁移工作量。

操作说明

将旧版应用程序移植到 .NET 5 或更高版本

Task	描述	所需技能
将 .NET Framework 旧应用程序升级到 .NET 5。	您可以使用 Porting Assistant for .NET 将旧版 ASP.NET Web Forms 应用程序转换为 .NET 5 或更高版本。请按照 Porting Assistant for .NET 文档中的说明进行操作。	应用程序开发人员

Task	描述	所需技能
生成 NDepend 报告。	当您通过将 ASP.NET Web 窗体应用程序分解为微服务来实现现代化时，您可能不需要旧应用程序中的所有 .cs 文件。您可以使用 NDepend 为任何隐藏代码 (.cs) 的文件生成报告，以获取所有调用者和被调用者。此报告可帮助您仅识别和使用微服务中所需文件。 安装后 NDepend（参见 “先决条件” 部分），在 Visual Studio 中打开旧版应用程序的解决方案（.sln 文件），然后按照以下步骤操作： 1. 在 Visual Studio 中构建旧应用程序。 2. 在 Visual Studio 菜单栏上 NDepend，选择 “将新 NDepend 项目附加到当前 VS 解决方案”。 3. 选择分析.NET 程序集。 4. 分析完成后，导航到解决方案资源管理器中的项目。右键单击要为其生成报告的任何代码隐藏文件（例如 `listproducts.aspx.cs`），然后选择在依赖关系图上显示。 5. 在导航栏中，选择调用者和被调用者，然后选择编辑代码查询。	应用程序开发人员

Task	描述	所需技能
	6. 在查询和规则编辑窗格中，选择下载箭头，然后选择导出到 Excel。 此过程会生成代码隐藏文件的报告，其中列出了所有调用者和被调用者。有关依赖关系图的更多信息，请参阅NDepend 文档。	

Task	描述	所需技能
创建新的 .NET 5 解决方案。	要为现代化的 ASP.NET Core 网页创建新的.NET 5（或更高版本）结构，请执行以下操作：APIs 1. 打开 Visual Studio。 2. 创建一个新的空白解决方案。 3. 根据您的旧应用程序创建面向 .NET 5（或更高版本）的新项目。有关购物车应用程序的旧项目和新项目的示例，请参阅其他信息部分。 4. 使用上一步中的 NDepend 报告来识别所有必需的文件。从您之前升级的应用程序中复制这些文件，并将它们添加到新的解决方案中。 5. 构建解决方案并解决所有问题。 有关创建项目和解决方案的详细信息，请参阅 Visual Studio 文档。 **Note** 在构建解决方案和验证功能时，除了已识别的文件外，您可能会 NDepend 发现要添加到解决方案中的其他几个文件。	应用程序开发人员

更新您的应用程序代码。

Task	描述	所需技能
APIs 使用 ASP.NET 核心实现网络。	假设您在旧版单体购物车应用程序中确定的微服务之一为产品。你在上一个操作说明中为产品创建了一个新的 ASP.NET Core Web API 项目。在此步骤中，您将识别和现代化与产品相关的所有 Web 窗体（.aspx 页面）。假设产品由四个 Web 表单组成，如前面的架构部分所示： • 列出产品 • 查看产品 • 添加/编辑产品 • 删除产品 您应该分析每个 Web 表单，识别发送到数据库以执行某些逻辑的所有请求，并获取响应。您可以将每个请求实现为 Web API 端点。鉴于其 Web 表单，产品可以具有以下可能的端点： • `/api/products` • `/api/products/{id}` • `/api/products/add` • `/api/products/update/{id}` • `/api/products/delete/{id}`	应用程序开发人员

Task	描述	所需技能
	如前所述，您还可以重用升级到 .NET 5 的所有其他项目，包括业务逻辑、数据访问和共享/通用项目。	
配置 Amazon CloudWatch 日志。	您可以使用 Amazon CloudWatch Logs 来监控、存储和访问应用程序的日志。您可以使用 AWS 软件开发工具包将数据记录到 Amazon CloudWatch 日志中。您还可以使用流行的.NET CloudWatch 日志框架（例如 Log4 Net 和 ASP.NET C ore 日志框架）将.NET 应用程序与日志集成。NLog 有关此步骤的更多信息，请参阅博客文章 Amazon CloudWatch 日志和.NET 日志框架。	应用程序开发人员

Task	描述	所需技能
配置 AWS Systems Manager Parameter Store。	您可以使用 AWS Systems Manager Parameter Store 来存储应用程序设置，例如与应用程序代码分开的连接字符串。Amazon.extens.Configurat NuGet SystemsManager简化了应用程序将这些设置从 AWS Systems Manager Parameter Store 加载到.NET 核心配置系统的方式。 有关此步骤的更多信息，请参阅博客文章 AWS Systems Manager 的 .NET Core 配置提供程序。	应用程序开发人员

添加身份验证和授权

Task	描述	所需技能
使用共享 cookie 进行身份验证。	对旧版单体应用程序进行现代化改造是一个迭代过程，需要单体应用及其现代化版本共存。您可以使用共享 cookie 来实现两个版本之间的无缝身份验证。旧版 ASP.NET 应用程序继续验证用户凭据并颁发 cookie，而现代化的 ASP.NET Core 应用程序则验证 cookie。 有关说明和示例代码，请参阅示例 GitHub 项目。	应用程序开发人员

在本地生成并运行容器

Task	描述	所需技能
使用 Visual Studio 创建 Docker 映像。	在此步骤中，将使用 Visual Studio for .NET Core Web API 创建 Docker 文件。 1. 打开 Visual Studio。 2. 在解决方案资源管理器中，从项目的上下文（右键单击）菜单中，依次选择添加、Docker 支持。 3. 选择 Linux 作为目标操作系统。 Visual Studio 为您的项目创建一个 Docker 文件。有关示例 Docker 文件，请参阅 Microsoft 网站上的 Visual Studio Container Tools for Docker。	应用程序开发人员
使用 Docker Desktop 构建并运行容器。	现在您可以在 Docker Desktop 中构建、创建和运行容器。 1. 打开 Command Prompt (命令提示符窗口)。导航到 Docker 文件所在的解决方案文件夹。请运行以下命令，创建 Docker 映像： `docker build -t aspnetcorewebapiimage -f Dockerfile .` 2. 运行以下命令，查看所有 Docker 映像。	应用程序开发人员

Task	描述	所需技能

	`docker images`
	3. 运行以下命令，创建容器。
	`docker run -d -p 8080:80 --name aspnetcorewebapicontainer aspnetcorewebapiimage`
	4. 打开 Docker Desktop，然后选择容器/应用程序。您可以看到一个名为 `aspnetcorewebapicontainer` 的新容器正在运行。	

相关资源

- 在亚马逊 EC2 Linux 实例上运行 ASP.NET Core Web API Docker 容器（AWS Prescriptive Guidence）
- Amazon ECS 研讨会
- 使用 AW@@S 执行 ECS 蓝/绿部署（AW CodeDeploy CloudFormation S CloudFormation 文档）
- 入门 NDepend（NDepend 文档）
- Porting Assistant for .NET

其他信息

下表提供了旧版购物车应用程序的示例项目示例，以及新式 ASP.NET Core 应用程序中的等效项目。

传统解决方案：

项目名称	项目模板	目标架构
业务界面	类库	NET Framework。

BusinessLogic	类库	NET Framework。
WebApplication	ASP.NET Framework Web 应用程序	NET Framework。
UnitTests	NUnit 测试项目	NET Framework。
共享->通用	类库	NET Framework。
共享->框架	类库	NET Framework。

新解决方案：

项目名称	项目模板	目标架构
BusinessLogic	类库	.NET 5.0
<WebAPI>	ASP.NET 核心 Web API	.NET 5.0
<WebAPI>。 UnitTests	NUnit 3 测试项目	.NET 5.0
共享->通用	类库	.NET 5.0
共享->框架	类库	.NET 5.0

使用 AWS Fargate 大规模运行事件驱动型和计划性工作负载

创建者：HARI OHM PRASATH RAJAGOPAL（AWS）

摘要

注意： CodeCommit AWS 不再向新客户开放。AWS 的现有客户 CodeCommit 可以继续照常使用该服务。了解更多

此模式描述了如何使用 AWS Fargate 在 Amazon Web Services（AWS）Cloud 上大规模运行计划性和事件驱动型工作负载。

在此模式设置的用例中，每当提交拉取请求时，都会扫描代码中是否有 AWS 敏感信息，例如 Amazon Web Services account 和凭证。拉取请求会启动 Lambda 函数。Lambda 函数调用负责代码扫描的 Fargate 任务。每当提出新的拉取请求时，就会启动 Lambda。如果扫描发现任何敏感信息，亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 将通过电子邮件发送扫描结果。

这种模式在以下企业用例很有用：

- 如果您的企业必须运行许多计划性和事件驱动型工作负载，而这些工作负载由于运行时系统（15 分钟限制）或内存限制而无法由 AWS Lambda 运行
- 如果您希望 AWS 管理为这些工作负载预调配的实例

使用此模式时，您可以选择创建虚拟私有云（VPC）。这种模式也使用 AWS CodeCommit。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- AWS CodeCommit 用于托管代码库和创建拉取请求
- AWS 命令行界面（AWS CLI）版本 1.7 或更高版本，已在 macOS、Linux 或 Windows 上安装并配置
- 在容器中运行的工作负载
- 在类路径中设置的 Apache Maven 可执行文件

架构

整个过程包括以下步骤。

1. 每当提交新的拉取请求时 CodeCommit，就会启动 Lambda 函数。Lambda 函数通过亚马逊监听事件CodeCommit Pull Request State Change。 EventBridge
2. Lambda 函数提交了一个新的 Fargate 任务，其中包含以下环境参数，用于检出代码并对其进行扫描。

```
RUNNER # <<TaskARN>>
SNS_TOPIC # <<SNSTopicARN>>
SUBNET # <<Subnet in which Fargate task gets launched>>
```

   如果扫描过程发现代码中的敏感信息，Fargate 会向 Amazon SNS 主题发送新消息。
3. SNS 订阅用户阅读主题中的消息并发送电子邮件。

技术

- AWS CodeCommit
- Amazon Elastic Container Registry（Amazon ECR）
- Amazon Elastic Container Service（Amazon ECS）
- Amazon EventBridge
- AWS Fargate
- AWS Lambda
- Amazon SNS
- Docker

工具

工具

- AWS CLI – AWS 命令行界面（CLI）是用于管理 Amazon Web Services 的统一工具。
- AWS CodeCommit — AWS CodeCommit 是一项完全托管的源代码控制服务，可托管基于 Git 的安全存储库。使用 CodeCommit，团队可以在安全且高度可扩展的环境中协作处理代码。
- Amazon ECR – Amazon Elastic Container Registry (Amazon ECR) 是一个完全托管式注册表，可让开发人员存储、管理和部署 Docker 容器映像。

- Amazon ECS – Amazon Elastic Container Service（Amazon ECS）是一个可扩展性高的快速容器管理服务。您可以使用 Amazon ECS 来运行、停止和管理集群上的容器。
- AWS Fargate — AWS Fargate 是一项可以与 Amazon ECS 配合使用的技术，无需管理服务器或亚马逊实例集群即可运行容器。 EC2
- AWS Lambda – AWS Lambda 是一项计算服务，支持无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 是一项托管服务，提供从发布者到订阅用户（也称为创建者和使用者）的消息传输。发布者通过将消息发送至主题与订阅用户进行异步交流，主题是一个逻辑接入点和通信渠道。订阅 SNS 主题的客户端会使用受支持的协议接收已发布的消息，例如 Lambda、电子邮件、移动推送通知和移动短信（SMS）。
- Docker 允许您在名为容器的软件包中构建、测试和交付应用程序。
- Git 客户端 – 用于查看所需构件的命令行或桌面工具
- Maven – Apache Maven 是一款项目管理工具，用于集中管理项目的构建、报告和文档。

操作说明

设置本地存储库

Task	描述	所需技能
下载代码。	在附件部分中，下载 .zip 文件并解压文件。	开发人员、AWS 系统管理员
设置存储库。	在根文件夹上运行 `mvn clean install`。	开发人员、AWS 系统管理员

创建 Amazon ECR 映像并推送映像

Task	描述	所需技能
创建 Amazon ECR 存储库并登录。	打开 Amazon ECR 控制台。在导航窗格中，选择存储库，然后选择创建存储库。要获取有关此操作和其他操作的帮助，请参阅相关资源部分。	开发人员、AWS 系统管理员

Task	描述	所需技能
推送容器映像。	打开存储库并选择查看推送命令，然后登录 Docker。登录后，运行其他信息部分的推送容器映像下的命令，其中包含所需替换内容。这将上传用于执行代码扫描的 Docker 容器映像。上传完成后，将最新构建的 URL 复制到 Amazon ECR 存储库中。	开发人员、AWS 系统管理员

创建 CodeCommit 存储库

Task	描述	所需技能
创建 CodeCommit 存储库。	要创建新的 AWS CodeCommit 存储库，请运行其他信息部分中创建 CodeCommit 存储库下的命令。	开发人员、AWS 系统管理员

创建 VPC（可选）

Task	描述	所需技能
创建 VPC。	如果您想使用新的 VPC 而不是现有的 VPC，请运行其他信息部分的创建 VPC 下的命令。AWS Cloud Development Kit (AWS CDK) 脚本将输出已创建的 VPC 和子网的内容。IDs	开发人员、AWS 系统管理员

创建 Amazon ECS 集群和 Fargate 任务

Task	描述	所需技能
创建集群和任务。	要创建 Amazon ECS 集群和 Fargate 任务定义，请运行其他信息部分的创建集群和任务下的命令。在运行 Shell 脚本时，请确保将正确的 VPC ID 和 Amazon ECR 存储库 URI 作为参数传入。该脚本创建指向 Docker 映像（负责扫描）的 Fargate 任务定义。然后，该脚本会创建一个作业和一个相关的执行角色。	开发人员、AWS 系统管理员
验证 Amazon ECS 集群。	打开 Amazon ECS 控制台。在导航窗格中，选择集群，然后选择新创建的名为 Fargate-Job-Cluster 的 Amazon ECS 集群。之后，在导航窗格中选择任务定义，并确认有带有前缀 `awscdkfargateecsTaskDef` 的新任务定义。	开发人员、AWS 系统管理员

创建 SNS 主题和订阅用户

Task	描述	所需技能
创建 SNS 主题。	要创建 SNS 主题，请运行其他信息部分的创建 SNS 主题下的命令。成功创建后，请注意将在下一步中使用的 `SNS ARN`。	开发人员、AWS 系统管理员
创建 SNS 订阅用户。	要创建 SNS 主题的电子邮件订阅用户，请运行其他信息部分的创建 SNS 订阅用户下的命	开发人员、AWS 系统管理员

Task	描述	所需技能
	令。请务必替换在 CLI 命令中使用的 `TopicARN` 和 `Email address`。要接收电子邮件通知，请务必确认用作订阅用户的电子邮件地址。	

创建 Lambda 函数并触发器 CodeCommit

Task	描述	所需技能
创建函数和触发器。	要创建带有 CodeCommit 触发器的 Lambda 函数，请在 Lambda 函数下运行该命令，然后在 “其他信息” CodeCommit 部分中触发。在运行命令之前，请务必将参数替换为相应的值。该脚本创建 Lambda 函数，并将其配置为在发出新的拉取请求时调用。	开发人员、AWS 系统管理员

测试应用程序

Task	描述	所需技能
测试应用程序。	如果您将任何 AWS 敏感信息签入 CodeCommit 存储库，则应启动 Lambda 函数。Lambda 函数启动 Fargate 任务，该任务会扫描代码并通过电子邮件通知发送扫描结果。	开发人员、AWS 系统管理员

相关资源

- 创建新的 Amazon ECR 存储库
- 将 Docker 映像推送到 Amazon ECR

其他信息

推送容器映像

```
> cd 1-ecr-image-push
> ./run.sh <<ecr-repository>>
```

创建 CodeCommit 存储库

```
aws codecommit create-repository --repository-name test-repo --repository-description
 "My Test repository"
```

创建 VPC

```
> cd 2-create-vpc
> ./run.sh
```

输出

```
aws-batch-cdk-vpc-efs-launch-template.privatesubnet = subnet-<<id>>
aws-batch-cdk-vpc-efs-launch-template.publicsubnet = subnet-<<id>>
aws-batch-cdk-vpc-efs-launch-template.vpcid = vpc-<<id>>
```

创建集群和任务

```
> export CDK_DEFAULT_ACCOUNT = <<aws_account_id>>
> export CDK_DEFAULT_REGION = <<aws_region>>
> cd 3-create-ecs-task
> ./run.sh <<vpc-id>> <<ecr-repo-uri>>
```

输出

```
aws-cdk-fargate-ecs.CLUSTERNAME = Fargate-Job-Cluster
```

```
aws-cdk-fargate-ecs.ClusterARN = <<cluster_arn>>
aws-cdk-fargate-ecs.ContainerARN = Fargate-Container
aws-cdk-fargate-ecs.TaskARN = <<task_arn>>
aws-cdk-fargate-ecs.TaskExecutionRole = <<execution_role_arn>>
aws-cdk-fargate-ecs.TaskRole = <<task_role_arn>>
```

创建 SNS 主题

```
aws sns create-topic --name code-commit-topic
```

创建 SNS 订阅用户

```
aws sns subscribe \
    --topic-arn <<topic_arn>> \
    --protocol email \
    --notification-endpoint <<email_address>>
```

Lambda 函数和触发器 CodeCommit

```
> export CDK_DEFAULT_ACCOUNT = <<aws_account_id>>
> export CDK_DEFAULT_REGION = <<aws_region>>
> cd 5-Lambda-CodeCommit-Trigger
> ./run.sh <<taskarn>> <<snstopicarn>> subnet-<<id>> <<codecommitarn>>
```

输出

```
aws-cdk-fargate-lambda-event.Cloudwatchrule = <<cloudwatchrule>>
aws-cdk-fargate-lambda-event.CodeCommitLambda = AWS-Code-Scanner-Function
aws-cdk-fargate-lambda-event.LambdaRole = <<lambdaiamrole>>
```

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 C# 和 AWS CDK 在 SaaS 架构中为孤岛模型进行租户登录

创建者：Tabby Ward（AWS）、Susmitha Reddy Gankidi（AWS）和 Vijai Anand Ramalingam（AWS）

摘要

软件即服务（SaaS）应用程序可以使用各种不同的架构模型构建。孤岛模型是指向租户提供专用资源的架构。

SaaS 应用程序依靠无摩擦模型将新租户引入其环境。这通常需要编排多个组件，才能成功预调配和配置创建新租户所需所有元素。在 SaaS 架构中，此过程称为租户登录。在每个 SaaS 环境中，应通过在登录流程中使用基础设施即代码，实现登录完全自动化。

此模式将引导您完成在 Amazon Web Services（AWS）上创建租户并为租户预调配基本基础设施的示例。该模式使用 C# 和 AWS Cloud Development Kit（AWS CDK）。

因为这种模式会产生账单警报，所以我们建议在美国东部（弗吉尼亚州北部）或 us-east-1 AWS 区域部署堆栈。有关更多信息，请参阅 AWS 文档。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 要在这种模式下创建 AWS 资源，需要具有足够的 IAM 访问权限的 AWS Identity and Access Management（IAM）主体。有关更多信息，请参阅 IAM 角色。
- 安装 Amazon 命令行界面（AWS CLI）并配置 AWS CLI 以执行 AWS CDK 部署。
- 已下载并安装 Visual Studio 2022 或者已下载并安装 Visual Studio Code。
- 已设置 AWS Toolkit for Visual Studio。
- .NET Core 3.1 或更高版本（C# AWS CDK 应用程序需要此选项）
- 已安装 Amazon.Lambda.Tools。

限制

- AWS CDK 使用 AWS CloudFormation，因此 AWS CDK 应用程序受 CloudFormation 服务配额的限制。有关更多信息，请参阅 AWS CloudFormation 配额。

- 租户 CloudFormation 堆栈是使用 CloudFormation 服务角色创建的，操作上`infra-cloudformation-role`带有通配符（`sns*` 和`sqs*`），但资源锁定到`tenant-cluster`前缀。对于生产用例，请评估此设置并仅提供对该服务角色所需访问权限。`InfrastructureProvisionLambda` 函数还使用通配符 (`cloudformation*`) 来配置 CloudFormation 堆栈，但资源锁定到前缀。`tenant-cluster`
- 此示例代码的 docker 构建使用 `--platform=linux/amd64` 强制基于 `linux/amd64` 的映像。这是为了确保最终的映像构件适用于 Lambda，它默认使用 x86-64 架构。如果您需要更改目标 Lambda 架构，请务必同时更改 Dockerfiles 和 AWS CDK 代码。有关更多信息，请参阅博客文章：将 AWS Lambda 函数迁移到基于 ARM 的 AWS Graviton2 处理器。
- 堆栈删除过程不会清理堆栈生成的 CloudWatch 日志（日志组和日志）。您必须通过 AWS 管理控制台、Amazon 控制 CloudWatch 台或 API 手动清理日志。

此模式设置为示例。对于生产用途，请评估以下设置并根据您的业务需求进行更改：

- 为简单起见，本示例中的 AWS Simple Storage Service（Amazon S3）存储桶未启用版本控制。根据需要评估和更新设置。
- 为简单起见，此示例在没有身份验证、授权或节流的情况下设置 Amazon API Gateway REST API 端点。对于生产用途，我们建议将系统与业务安全基础设施集成。评估此设置并根据需要添加所需安全设置。
- 在此租户基础设施示例中，Amazon Simple Notification Service（Amazon SNS）和 Amazon Simple Queue Service（Amazon SQS）仅有最低设置。根据 AWS KMS 密钥政策，每个租户的 AWS 密钥管理服务 (AWS KMS) 向账户中的亚马逊 CloudWatch和亚马逊 SN S 服务开放，供其使用。该设置只是一个占位符示例。根据您的业务用例根据需要调整设置。
- 整个设置包括但不限于 API 终端节点以及使用 AWS 预配置和删除后端租户 CloudFormation，仅涵盖基本的成功路径案例。根据您的业务需求，使用必要的重试逻辑、额外的错误处理逻辑和安全逻辑来评估和更新设置。
- 在撰写本文时，示例代码使用 up-to-date cdk-nag 进行测试，以检查策略。将来可能会强制执行新的策略。这些新策略可能需要您根据建议手动修改堆栈，然后才能部署堆栈。查看现有代码，确保其符合您的业务需求。
- 该代码依赖 AWS CDK 生成随机后缀，而不是依赖静态分配的物理名称来创建大部分资源。此设置是为了确保这些资源是唯一的，并且不会与其他堆栈发生冲突。有关更多信息，请参阅 AWS CDK 文档。根据您的业务需求对此进行调整。
- 此示例代码将 .NET Lambda 构件打包成基于 Docker 的映像，并使用 Lambda 提供的容器映像运行时系统运行。容器映像运行时系统具有的优势包括：标准传输和存储机制（容器注册表）和更精确的本地测试环境（通过容器映像）。您可以将项目切换为使用 Lambda 提供的 .NET 运行时系统来缩短

Docker 映像的构建时间，但随后您需要设置传输和存储机制，并确保本地设置与 Lambda 设置相匹配。调整代码以适应用户的业务需求。

产品版本

- AWS CDK 版本 2.45.0 或更高版本
- Visual Studio 2022

架构

技术堆栈

- Amazon API Gateway
- AWS CloudFormation
- Amazon CloudWatch
- Amazon DynamoDB
- AWS Identity and Access Management (IAM)
- AWS KMS
- AWS Lambda
- Amazon S3
- Amazon SNS
- Amazon SQS

架构

下图显示了租户堆栈的创建流程。有关控制面板和租户技术堆栈的更多信息，请参阅其他信息部分。

租户堆栈创建流程

1. 用户向 Amazon API Gateway 托管的 REST API 发送带有 JSON 格式的新租户有效负载（租户名称、租户描述）的 POST API 请求。API 网关处理请求并将其转发到后端 Lambda 租户登录函数。在此示例中，没有授权或身份验证。在生产环境中，此 API 应与 SaaS 基础设施安全系统集成。
2. 租户登录功能会验证请求。然后，它会尝试将租户记录（包括租户名称、生成的租户通用唯一标识符（UUID）和租户描述）存储到 Amazon DynamoDB 租户登录表中。

3. 在 DynamoDB 存储记录后，DynamoDB 流会启动下游 Lambda 租户基础设施函数。
4. 租户基础设施 Lambda 函数基于收到的 DynamoDB 数据流进行操作。如果流用于 INSERT 事件，则该函数使用流的 NewImage 部分（最新更新记录，“租户名称” 字段）进行调用， CloudFormation 以使用存储在 S3 存储桶中的模板创建新的租户基础架构。 CloudFormation 模板需要租户名称参数。
5. AWS 根据 CloudFormation 模板和输入参数 CloudFormation 创建租户基础设施。
6. 每个租户基础设施设置都有一个 CloudWatch 警报、一个账单警报和一个警报事件。
7. 警报事件将变成 SNS 主题的消息，该消息由租户的 AWS KMS 密钥加密。
8. SNS 主题将收到的警报消息转发到 SQS 队列，该队列由租户的 AWS KMS 加密以获取加密密钥。

其他系统可以与 Amazon SQS 集成，根据队列中的消息执行操作。在此示例中，为了保持代码的通用性，传入的消息将保留在队列中，需要手动删除。

租户堆栈删除流程

1. 用户向 Amazon API Gateway 托管的 REST API 发送带有 JSON 格式的新租户有效负载（租户名称、租户描述）的 DELETE API 请求，REST API 将处理该请求并转发到租户登录功能。在此示例中，没有授权或身份验证。在生产环境中，此 API 将与 SaaS 基础设施安全系统集成。
2. 租户登录功能将验证请求，然后尝试从租户登录表中删除租户记录（租户名称）。
3. DynamoDB 成功删除记录（该记录存在于表中并已删除）后，DynamoDB 流将启动下游 Lambda 租户基础设施函数。
4. 租户基础设施 Lambda 函数基于收到的 DynamoDB 数据流记录进行操作。如果流是针对REMOVE事件，则该函数使用记录的 OldImage 部分（记录信息和租户名称字段，位于最新更改之前，即删除），根据该记录信息启动对现有堆栈的删除。
5. AWS 根据输入 CloudFormation 删除目标租户堆栈。

工具

Amazon Web Services

- Amazon API Gateway 可帮助您创建、发布、维护、监控和保护任何规模的 RES WebSocket APIs T、HTTP。
- AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，可帮助您在代码中定义和预调配 Amazon Web Services Cloud 基础设施。

- AWS CDK Toolkit 是命令行云开发套件，可帮助您与 AWS Cloud Development Kit（AWS CDK）应用程序进行交互。
- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。
- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥，以帮助保护您的数据。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Storage Service (Amazon S3)是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。
- Amazon Simple Queue Service (Amazon SQS) 提供了一个安全、持久且可用的托管队列，它可帮助您集成和分离分布式软件系统与组件。
- AWS Toolkit for Visual Studio 是 Visual Studio 集成式开发环境（IDE）的插件。Toolkit for Visual Studio 支持开发、调试和部署使用 Amazon Web Services 的 .NET 应用程序。

其他工具

- Visual Studio 是一个 IDE，包括编译器、代码完成工具、图形设计器和其他支持软件开发的功能。

代码

此模式的代码位于孤岛模型的 SaaS 架构中的租户登录 APG 示例存储库中。

操作说明

设置 AWS CDK

Task	描述	所需技能
验证 Node.js 的安装。	要验证 Node.js 是否已安装在本地计算机上，请运行以下命令。 `node --version`	AWS 管理员，AWS DevOps
安装 AWS CDK Toolkit。	要在本地计算机上安装 AWS CDK Toolkit，请运行以下命令。 `npm install -g aws-cdk` 如果未安装 npm，则可以从 Node.js 站点进行安装。	AWS 管理员，AWS DevOps
验证 AWS CDK Toolkit 的版本。	要验证 AWS CDK Toolkit 版本是否已在本机上正确安装，请运行以下命令。 `cdk --version`	AWS 管理员，AWS DevOps

查看租户登录控制面板的代码

Task	描述	所需技能
克隆存储库。	克隆存储库，然后导航到该 `\tenant-onboarding-in-saas-architecture-for-silo-model-apg-example` 文件夹。	AWS 管理员，AWS DevOps

Task	描述	所需技能
	在 Visual Studio 2022 中，打开 `\src\TenantOnboardingInfra.sln` 解决方案。打开 `TenantOnboardingInfraStack.cs` 文件并查看代码。 以下资源是作为此堆栈的一部分创建的： • DynamoDB 表 • S3 存储桶（将 CloudFormation 模板上传到 S3 存储桶。） • Lambda 执行角色 • Lambda 函数 • API 网关 CLI • Lambda 函数的事件源	
查看 CloudFormation 模板。	在 `\tenant-onboarding-in-saas-architecture-for-silo-model-apg-example\template` 文件夹中 `infra.yaml`，打开并查看 CloudFormation 模板。此模板将使用从租户登录 DynamoDB 表中检索到的租户名称注入数据。 该模板提供了租户特定的基础设施。在此示例中，它配置了 AWS KMS 密钥、亚马逊 SNS、亚马逊 SQS 和警报。CloudWatch	AWS 应用程序开发人员 DevOps

Task	描述	所需技能
查看租户登录函数。	打开 `Function.cs` 并查看租户登录函数的代码，该代码是使用带有 .NET 6（容器映像）蓝图的 Visual Studio AWS Lambda 项目 (.NET Core- C#) 模板创建的。 打开 `Dockerfile` 并查看代码。`Dockerfile` 是一个文本文件，包含构建 Lambda 容器映像的说明。 请注意，以下 NuGet 包已作为依赖项添加到`TenantOnboardingFunction` 项目中： • `Amazon.Lambda.APIGatewayEvents` • `AWSSDK.DynamoDBv2` • `Newtonsoft.Json`	AWS 应用程序开发人员 DevOps

Task	描述	所需技能
查看租户 InfraProvisioning 功能。	导航到 `\tenant-onboarding-in-saas-architecture-for-silo-model-apg-example\src\InfraProvisioningFunction` 。 打开 `Function.cs` 并查看租户基础设施预调配函数的代码，该代码是使用带有 .NET 6（容器映像）蓝图的 Visual Studio AWS Lambda 项目（.NET Core-C#）模板创建的。 打开 `Dockerfile` 并查看代码。 请注意，以下 NuGet 包已作为依赖项添加到`InfraProvisioningFunction` 项目中： • `Amazon.Lambda.DynamoDBEvents` • `AWSSDK.DynamoDBv2` • `AWSSDK.Cloudformation`	AWS 应用程序开发人员 DevOps

部署 AWS 资源

Task	描述	所需技能
构建解决方案。	要构建解决方案，请执行以下步骤：	应用程序开发人员

Task	描述	所需技能
	1. 在 Visual Studio 2022 中，打开 `\tenant-onboarding-in-saas-architecture-for-silo-model-apg-example\src\TenantOnboardingInfra.sln` 解决方案。 2. 打开解决方案的上下文（右键单击）菜单，然后选择构建解决方案。 **Note** 在生成解决方案之前，请确保将Amazon.CDK.Lib NuGet 软件包更新到`\tenant-onboarding-in-saas-architecture-for-silo-model-apg-example\src\TenantOnboardingInfra` 项目中的最新版本。	

Task	描述	所需技能

引导 AWS CDK 环境。	打开 Windows 命令提示符并导航到 `cdk.json` 文件所在的 AWS CDK 应用程序根文件夹（`\tenant-onboarding-in-saas-architecture-for-silo-model-apg-example` ）。运行以下命令进行引导。
	`cdk bootstrap`
	如果您已为凭证创建了 AWS 配置文件，请将命令与配置文件一起使用。
	`cdk bootstrap --profile <profile name>`
		AWS 管理员，AWS DevOps

列出 AWS CDK 堆栈。	要列出要作为本项目的一部分创建的所有堆栈，请运行以下命令。
	`cdk ls`
	`cdk ls --profile <profile name>`
	如果您已为凭证创建了 AWS 配置文件，请将命令与配置文件一起使用。
	`cdk ls --profile <profile name>`
		AWS 管理员，AWS DevOps

Task	描述	所需技能

查看将创建哪些 AWS 资源。	要查看将作为此项目的一部分创建的所有 AWS 资源，请运行以下命令。
	`cdk diff`
	如果您已为凭证创建了 AWS 配置文件，请将命令与配置文件一起使用。
	`cdk diff --profile <profile name>`
		AWS 管理员，AWS DevOps

Task	描述	所需技能

使用 AWS CDK 部署所有 AWS 资源。	要部署所有 AWS 资源，请运行以下命令。
	`cdk deploy --all --require-approval never`
	如果您已为凭证创建了 AWS 配置文件，请将命令与配置文件一起使用。
	`cdk deploy --all --require-approval never --profile <profile name>`
	部署完成后，从命令提示符的输出部分复制 API URL，如以下示例所示。
	`Outputs: TenantOnboardingInfraStack.TenantOnboardingAPIEndpoint42E526D7 = https://j2qmp8ds21i1i.execute-api.us-west-2.amazonaws.com/prod/`
		AWS 管理员，AWS DevOps

验证功能

Task	描述	所需技能
创建新租户。	要创建新租户，请发送以下 curl 请求。	应用程序开发人员、AWS 管理员、AWS DevOps

Task	描述	所需技能

	`curl -X POST <TenantOnboardingAPIEndpoint* from CDK Output>tenant -d '{"Name":"Tenant123", "Description":"Stack for Tenant123"}'`
	
	将占位符 `<TenantOnboardingAPIEndpoint* from CDK Output>` 更改为 AWS CDK 中的实际值，如以下示例所示。
	
	`curl -X POST https://j2qmp8ds21i1i.execute-api.us-west-2.amazonaws.com/prod/tenant -d '{"Name": "Tenant123", "Description":"test12"}'`
	
	以下示例显示了输出。
	
	`{"message": "A new tenant added - 5/4/2022 7:11:30 AM"}`	

Task	描述	所需技能
在 DynamoDB 中验证新创建租户的详细信息。	要验证 DynamoDB 中新创建租户的详细信息，请执行以下步骤。 1. 打开 AWS 管理控制台，然后导航到 Amazon DynamoDB 服务。 2. 在左侧导航栏中，选择探索项目，然后选择 `TenantOnboarding` 表格。 **Note** 租户名称前面将加上。`tenantcluster-` 有关更多信息，请参阅其他信息部分。 3. 验证是否使用租户详细信息创建了一个新项目。	应用程序开发人员、AWS 管理员、AWS DevOps

Task	描述	所需技能

验证为新租户创建的堆栈。	验证新堆栈是否已成功创建并根据 CloudFormation 模板为新创建的租户配置了基础架构。
	1. 打开控制 CloudFormation 台。
	2. 在左侧导航栏中，选择堆栈，然后验证是否已成功创建具有租户名称的堆栈。
	3. 选择新创建租户的堆栈，然后选择资源选项卡。记下警报资源和 Amazon SQS 资源。
	4. 打开配置了 AWS 凭证的新终端，然后指向正确的区域。要发出测试警报，请输入以下代码，<alarm resource name> 替换为步骤 3 中注明的警报资源名称。
	`aws cloudwatch set-alarm-state --alarm-name <alarm resource name> --state-value ALARM --state-reason 'Test setup'`
	以下示例显示了带有警报资源名称的代码。
	`aws cloudwatch set-alarm-state --alarm-name tenantcluster-tenant123-alarm --state-value ALARM --`
		应用程序开发人员、AWS 管理员、AWS DevOps

Task	描述	所需技能
	`state-reason 'Test setup'` 5. 打开控制台，然后导航到 Amazon SQS 控制台。选择步骤 3 中标识的 Amazon SQS 资源名称。按照 AWS 文档说明接收和删除步骤 4 中发出的警报中的测试消息。	

Task	描述	所需技能

删除租户堆栈。	要删除租户堆栈，请发送以下 curl 请求。
	
	`curl -X DELETE <TenantOnboardingAPIEndpoint* from CDK Output>tenant/<Tenant Name from previous step>`
	
	将占位符 `<TenantOnboardingAPIEndpoint* from CDK Output>` 更改为 AWS CDK 中的实际值，然后更改 `<Tenant Name from previous step>` 为上一个租户创建步骤中的实际值，如以下示例所示。
	
	`curl -X DELETE https://j2qmp8ds21i1i.execute-api.us-west-2.amazonaws.com/prod/tenant/Tenant123`
	
	以下示例显示了输出。
	
	`{"message": "Tenant destroyed - 5/4/2022 7:14:48 AM"}`
		应用程序开发人员、AWS DevOps、AWS 管理员

Task	描述	所需技能
验证现有租户的堆栈删除。	要验证现有租户的堆栈是否已删除，请执行以下步骤： 1. 打开控制台并导航到 CloudFormation 控制台。 2. 在左侧导航栏中，确认带有租户名称的现有堆栈已不在 CloudFormation 控制台中（如果控制台设置为仅显示活动堆栈）或正在被删除。如果堆栈已不在 CloudFormation 控制台中，请使用下拉列表将控制台的设置从 “活动” 更改为 “已删除”，以查看已删除的堆栈并验证堆栈是否已成功删除。	应用程序开发人员、AWS 管理员、AWS DevOps

清理

Task	描述	所需技能
摧毁环境。	在清理堆栈之前，请确保满足以下条件： • DynamoDB 中的所有记录均通过之前的租户删除操作或通过 DynamoDB 控制台或 API 删除。每次删除租户记录都将启动清理其 AWS CloudFormation 对应记录。 • 在 AWS 控制台上清理所有基于租户的 AWS CloudFormation 堆栈（以防万一 DynamoDB 触发器清	AWS 管理员，AWS DevOps

Task	描述	所需技能
	理逻辑失败）。 CloudFormation 测试完成后，可以通过运行以下命令使用 AWS CDK 销毁所有堆栈和相关资源。 `cdk destroy --all;` 如果您已为凭证创建了 AWS 配置文件，请使用配置文件。 确认堆栈删除提示以删除堆栈。	
清理 Amazon CloudWatch 日志。	堆栈删除过程不会清理堆栈生成的日 CloudWatch 志（日志组和日志）。使用 CloudWatch 控制台或 API 手动清理 CloudWatch 资源。	应用程序开发人员、AWS DevOps、AWS 管理员

相关资源

- AWS CDK .NET 研讨会
- 在 C# 中使用 AWS CDK
- CDK .NET 参考资料

其他信息

控制面板技术堆栈

用 .NET 编写的 CDK 代码用于预调配控制面板基础设施，该基础设施由以下资源组成：

1. API Gateway

用作控制面板堆栈的 REST API 入口点。

2. 租户登录 Lambda 函数

   此 Lambda 函数由 API 网关使用 m 方法启动。

   POST 方法 API 请求会导致（`tenant name`、`tenant description`）被插入到 DynamoDB 表 `Tenant Onboarding` 中。

   在此代码示例中，租户名称也用作租户堆栈名称的一部分以及该堆栈中资源的名称。这是为了使这些资源更易于识别。此租户名称在设置中必须是唯一的，以避免冲突或错误。详细的输入验证设置在 IAM 角色文档和限制部分中进行了说明。

   只有在表中的任何其他记录中未使用租户名称时，DynamoDB 表的持久化过程才会成功。

   在这种情况下，租户名称是该表的分区键，因为只有分区键可以用作 `PutItem` 条件表达式。

   如果以前从未记录过租户名称，则该记录将成功保存到表格中。

   但是，如果表中的现有记录已使用租户名称，则操作将失败并启动 DynamoDB `ConditionalCheckFailedException` 异常。该异常将用于返回一条失败消息（`HTTP BadRequest`），指示租户名称已存在。

   `DELETE` 方 法 API 请求将从 `Tenant Onboarding` 表中删除特定租户名称的记录。

   即使该记录不存在，本示例中的 DynamoDB 记录删除也会成功。

   如果目标记录存在并被删除，它将创建一个 DynamoDB 数据流记录。否则，将不会创建任何下游记录。

3. 在 Amazon DynamoDB Streams 启用时的租户登录 DynamoDB

   这会记录租户元数据信息，任何记录的保存或删除都会将数据流发送到下游 `Tenant InfrastructureLambda` 函数。

4. 租户基础设施 Lambda 函数

   此 Lambda 函数由上一步中的 DynamoDB 数据流记录启动。如果记录是针对某个`INSERT`事件的，则它会调用 AWS CloudFormation 以使用存储在 S3 存储桶中的 CloudFormation 模板创建新的租户基础设施。如果该记录用于 `REMOVE`，则它会根据流记录的 `Tenant Name` 字段启动对现有堆栈的删除。

5. S3 bucket

这是用来存储 CloudFormation 模板的。

6. 每个 Lambda 函数的 IAM 角色和一个服务角色 CloudFormation

   每个 Lambda 函数都有其唯一的 IAM 角色，该角色具有完成任务的最低权限许可。例如，`Tenant On-boarding` Lambda 函数具有对 DynamoDB 的读/写访问权限，而 `Tenant Infrastructure` Lambda 函数对 DynamoDB 流只有读取权限。

   为租户堆栈置备创建了自定义 CloudFormation 服务角色。此服务角色包含 CloudFormation 堆栈配置的其他权限（例如，AWS KMS 密钥）。这会在 Lambda 之间划分角色 CloudFormation ，以避免对单个角色（基础设施 Lambda 角色）拥有所有权限。

   允许强大操作（例如创建和删除 CloudFormation 堆栈）的权限被锁定，并且仅允许在以开头的资源上使用`tenantcluster-`。AWS KMS 是个例外，因为它的资源命名约定。从 API 摄取的租户名称将与其他验证检查一起添加到 `tenantcluster-` 前面（仅带破折号的字母数字，并且限制在 30 个字符以内，以适应大多数 AWS 资源命名）。这样可以确保租户名称不会意外导致核心基础设施堆栈或资源中断。

租户技术堆栈

CloudFormation 模板存储在 S3 存储桶中。该模板预置了租户特定的 AWS KMS 密钥、 CloudWatch 警报、SNS 主题、SQS 队列和 SQS 策略。

AWS KMS 密钥用于其消息的 Amazon SNS 和 Amazon SQS 的数据加密。SNS2 和 AwsSolutions AwsSolutions- 的安全实践SQS2建议您设置带加密功能的 Amazon SNS 和 Amazon SQS。但是，使用 AWS 托管密钥时， CloudWatch 警报不适用于 Amazon SNS，因此在这种情况下，您必须使用客户托管密钥。有关更多信息，请参阅 AWS Knowledge Center。

在 Amazon SQS 队列上使用 SQS 策略来允许创建的 SNS 主题将消息传送到队列。如果没有 SQS 策略，访问将被拒绝。有关更多信息，请参阅 Amazon SNS 文档。

使用 CQRS 和事件溯源将整体分解为微服务

由 Rodolfo Jr. Cerrada (AWS)、Dmitry Gulin (AWS) 和 Tabby Ward (AWS) 创建

摘要

此示例介绍了两种模式，使用命令查询责任分离 (CQRS) 模式和事件溯源模式。CQRS 模式将命令模型和查询模型职责分开。事件溯源模式利用异步事件驱动通信来改善整体用户体验。

您可使用 CQRS 和 Amazon Web Services (AWS) 服务独立维护和扩展每个数据模型，同时将您的整体应用程序重构为微服务架构。然后，您可使用事件溯源模式，将数据从命令数据库同步至查询数据库。

此示例使用包含解决方案 (*.sln) 文件的示例代码，您可以使用最新版本 Visual Studio 打开该文件。此示例包含奖励 API 代码，用于展示 CQRS 和事件溯源在 AWS 无服务器和传统或本地应用程序中的工作方式。

要了解有关 CQRS 和事件溯源的更多信息，请参阅其他信息部分。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- Amazon CloudWatch
- Amazon DynamoDB 表
- Amazon DynamoDB Streams
- AWS Identity and Access Management (IAM) 访问密钥。有关更多信息，请参阅相关资源部分中的视频
- AWS Lambda
- 熟悉 Visual Studio
- 熟悉 AWS Toolkit for Visual Studio；有关更多信息，请参阅相关资源部分的AWS Toolkit for Visual Studio 演示

产品版本

- Visual Studio 2019 Community Edition。
- AWS Toolkit for Visual Studio 2019。

- .NET Core 3.1。此组件是 Visual Studio 安装选项之一。若要在安装过程中纳入 .NET Core，请选择 NET Core 跨平台开发。

限制

- 传统本地应用程序 (ASP.NET Core Web API 和数据访问对象) 的示例代码不附带数据库。但是，它附带了 `CustomerData` 内存对象，该对象充当模拟数据库。所提供的代码适用于测试模式。

架构

源技术堆栈

- ASP.NET Core Web API 项目
- IIS Web 服务器
- 数据访问对象
- CRUD 模型

源架构

在源架构中，CRUD 模型在一个应用程序中同时包含命令与查询接口。有关示例代码，请参阅 `CustomerDAO.cs`（附后）。

目标技术堆栈

- Amazon DynamoDB
- Amazon DynamoDB Streams
- AWS Lambda
- （可选）Amazon API Gateway
- （可选）Amazon Simple Notification Service (Amazon SNS)

目标架构

在目标架构中，命令与查询接口是分开的。下图所示的架构可使用 API Gateway 和 Amazon SNS 进行扩展。有关更多信息，请参阅其他信息部分。

1. 命令 Lambda 函数对数据库执行写入操作，如创建、更新或删除。
2. 查询 Lambda 函数对数据库执行读取操作，如获取或选择。
3. 此 Lambda 函数处理来自命令数据库的 DynamoDB 数据流，并更新查询数据库以获取更改。

工具

工具

- Amazon DynamoDB – Amazon DynamoDB 是一种全托管 NoSQL 数据库服务，提供快速而可预测的性能，能够实现无缝扩展。
- Amazon DynamoDB Streams - DynamoDB Streams 捕获在任何 DynamoDB Streams 表中按时间排序的项目级修改序列。它还会将这类信息存储在日志中长达 24 个小时。静态加密会加密 DynamoDB 流中的数据。
- AWS Lambda – AWS Lambda 是一项计算服务，支持无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。您只需为消耗的计算时间付费 - 代码未运行时不产生费用。
- AWS 管理控制台 — AWS 管理控制台是一款 Web 应用程序，包含多种用于管理 Amazon Web Services 的服务控制台。
- Visual Studio 2019 Community Edition – Visual Studio 2019 是一个集成式开发环境（IDE）。开源参与者可免费使用社区版。在此示例中，您将使用 Visual Studio 2019 Community Edition 打开、编译和运行示例代码。仅供查看，您可以使用任何文本编辑器或 Visual Studio Code。
- AWS Toolkit for Visual Studio – AWS Toolkit for Visual Studio 是 Visual Studio IDE 的一个插件。AWS Toolkit for Visual Studio 可让您更轻松地开发、调试和部署使用 Amazon Web Services .NET 应用程序。

代码

示例代码附后。有关部署示例代码的说明，请参阅操作部分。

操作说明

打开和生成解决方案

Task	描述	所需技能
打开解决方案。	1. 从附件部分下载示例源代码 (`CQRS-ES Code.zip`)，然后提取文件。 2. 在 Visual Studio IDE 中，选择文件、打开、项目解决方案，然后导航至提取源代码的文件夹。 3. 选择 AWS.APG.CQRSES.sln，然后选择打开。整个解决方案已加载至 Visual Studio 中。	应用程序开发人员
构建解决方案。	打开解决方案的上下文（右键单击）菜单，然后选择生成解决方案。这将生成和编译解决方案的所有项目。它应该可成功编译。 Visual Studio 解决方案资源管理器应该显示目录结构。 • `CQRS On-Premises Code Sample` 包含在本地使用 CQRS 的示例。 • `CQRS AWS Serverless` 包含使用 AWS 无服务器服务的所有 CQRS 和事件溯源示例代码。	应用程序开发人员

构建 DynamoDB 表格

Task	描述	所需技能
提供凭证。	如果您还没有访问密钥，请查看相关资源部分中的视频。 1. 在 Solution Explorer 中，展开 CQRS AWS Serverless，然后展开生成解决方案文件夹。 2. 展开 AwS.APG.CQRSES.Build 项目并查看 `Program.cs` 文件。 3. 滚动到 `Program.cs` 顶部并查找 `Program()` 。 4. 将 `YOUR ACCESS KEY` 替换为您的账户访问密钥，然后将 `YOUR SECRET KEY` 替换为您的账户密钥。请注意，在生产环境中，您无法对密钥进行硬编码。相反，您可使用 AWS Secrets Manager 存储和检索证书。	应用程序开发人员、数据工程师、数据库管理员
构建 项目。	要生成项目，请打开 AwS.APG.CQRSES.Build 项目的上下文（右键单击）菜单，然后选择生成。	应用程序开发人员、数据工程师、数据库管理员
生成和填充表格。	若要生成表格并向其中填充种子数据，打开 AwS.APG.CQRSES.Build 项目的上下文（右键点击）菜单，然后选择调试、启动新实例。	应用程序开发人员、数据工程师、数据库管理员

Task	描述	所需技能
验证表结构与数据。	若要进行验证，请导航至 AWS 各区服务浏览器，然后展开 Amazon DynamoDB。它应该显示此表格。打开每个表，以显示示例数据。	应用程序开发人员、数据工程师、数据库管理员

运行本地测试

Task	描述	所需技能
构建 CQRS 项目。	1. 打开解决方案，然后导航到 CQRS AWS Services/CQRS/Tests 解决方案文件夹。 2. 在 AWS.APG.CQRSES 中。 CQRSLambda.Tests 项目，打开 BaseFunctionTest.cs，AccessKey然后SecretKey用您创建的 IAM 密钥替换和。 3. 保存更改。 4. 若要编译和生成测试项目，请打开项目的上下文（右键单击）菜单，然后选择生成。	应用程序开发人员、测试工程师
生成事件溯源项目。	1. 导航到 CQRS AWS Services/Event Source/Tests 解决方案文件夹。 2. 在 AWS.APG.CQRSES 中。 EventSourceLambda.Test s 项目，打开 BaseFunctionTest.c	应用程序开发人员、测试工程师

Task	描述	所需技能
	s，AccessKey然后SecretKey用您创建的 IAM 密钥替换和。 3. 保存更改。 4. 若要编译和生成测试项目，请打开项目的上下文（右键单击）菜单，然后选择生成。	
运行测试。	要运行所有测试，请选择查看、测试资源管理器，然后选择在视图中运行所有测试。所有测试都应通过，通过后以绿色复选标记图标表示。	应用程序开发人员、测试工程师

将 CQRS Lambda 函数发布至 AWS

Task	描述	所需技能
发布第一个 Lambda 函数。	1. 在解决方案资源管理器中，打开 AWS.APG.CQRSES 的上下文（右键单击）菜单。 CommandCreateLambda 项目，然后选择 “发布到 AWS Lambda”。 2. 选择您要使用的配置文件、要在其中部署 Lambda 函数的 AWS 区域 以及函数名称。 3. 在其余字段中，保留默认值，然后选择下一步。 4. 在 “角色名称” 下拉列表中，选择AWSLambdaFullAccess。	应用程序开发者、 DevOps 工程师

Task	描述	所需技能
	5. 若要提供您的账户密钥，请选择添加，然后输入 `AcessKey` 作为变量，输入您的访问密钥作为值。然后再次选择添加，输入 `SecretKey` 作为变量，输入您的密钥作为值。 6. 在其余字段中，保留默认值，然后选择上传。Lambda 测试函数上传之后，它会自动出现在 Visual Studio 中。 7. 对以下项目重复第 1 步至第 6 步： • AWS.APG.CQRSES。CommandDeleteLambda • AWS.APG.CQRSES。CommandUpdateLambda • AWS.APG.CQRSES。CommandAddRewardLambda • AWS.APG.CQRSES。CommandRedeemRewardLambda • AWS.APG.CQRSES。QueryCustomerListLambda • AWS.APG.CQRSES。QueryRewqardLambda	

Task	描述	所需技能
验证函数上传情况。	（可选）您可以通过导航到 AWS 各区服务浏览器并展开 AWS Lambda 来验证函数是否已成功加载。若要打开测试窗口，请选择 Lambda 函数（双击）。	应用程序开发者、 DevOps 工程师
测试 Lambda 函数。	1. 在其他信息部分输入请求数据，或从测试数据中复制示例请求数据。确保为正在测试的函数选择数据。 2. 要运行测试，请选择 Invoke (调用)。响应和所有错误显示在响应文本框中，日志显示在日志文本框或 CloudWatch 日志中。 3. 若要验证数据，请在 AWS 各区服务浏览器中选择 DynamoDB 表（双击）。 所有 CQRS Lambda 项目都位于 `CQRS AWS Serverless\CQRS\Command Microservice` 和 `CQRS AWS Serverless\CQRS\Command Microservice` 解决方案文件夹下。有关解决方案目录和项目，请参阅其他信息部分中的源代码目录。	应用程序开发者、 DevOps 工程师

Task	描述	所需技能
发布其余函数。	对以下项目重复之前的步骤： • AWS.APG.CQRSES。CommandDeleteLambda • AWS.APG.CQRSES。CommandUpdateLambda • AWS.APG.CQRSES。CommandAddRewardLambda • AWS.APG.CQRSES。CommandRedeemRewardLambda • AWS.APG.CQRSES。QueryCustomerListLambda • AWS.APG.CQRSES。QueryRewqardLambda	应用程序开发者、 DevOps 工程师

将 Lambda 函数设置为事件侦听器

Task	描述	所需技能
发布客户和奖励 Lambda 事件的处理程序。	若要发布每个事件处理程序，请按前述操作中的步骤进行操作。 这些项目位于 `CQRS AWS Serverless\Event Source\Customer Event` 和 `CQRS AWS Serverless\Event Source\Reward Event` 解决方案文件夹下。有关更多信息，请参阅其他信息中的源代码目录部分。	应用程序开发人员

Task	描述	所需技能
附加事件溯源 Lambda 事件侦听器。	1. 使用您在发布 Lambda 项目时相同的账户登录 AWS 管理控制台。 2. 对于该区域，选择美国东部 1 或您在上一篇操作说明中部署 Lambda 函数的 Region。 3. 导航至 Lambda 服务。 4. 选择 `EventSourceCustomer` Lambda 函数。 5. 选择添加触发器。 6. 在触发器配置下拉列表，选择 DynamoDB。 7. 在 DynamoDB 表下拉列表中，选择。cqrses-customer-cmd 8. 在起始位置下拉列表，选择裁剪范围始于 。“裁剪范围”指 DynamoDB 触发器将开始读取最后一个（未裁剪的）流数据记录，这是分片中最旧的记录。 9. 选中启用触发器复选框。 10. 在其余字段中，保留默认值，然后选择添加。 侦听器成功连接至 DynamoDB 表后，它将显示在 Lambda 设计器页面上。	应用程序开发人员

Task	描述	所需技能
发布并附加 EventSourceReward Lambda 函数。	要发布并附加 **EventSourceReward** Lambda 函数，请重复前两个故事中的步骤，cqrses-reward-cmd从 DynamoDB 表下拉列表中进行选择。	应用程序开发人员

测试和验证 DynamoDB 流与 Lambda 触发器

Task	描述	所需技能
测试流数据和 Lambda 触发器。	1. 在 Visual Studio 中，导航至 AWS 各区服务浏览器。 2. 展开 AWS Lambda，然后选择CommandRedeemReward函数（双击）。在打开的函数窗口中，您可测试函数。 3. 在请求文本框中，以 JavaScript 对象表示法 (JSON) 格式输入请求数据。有关示例请求，请参阅其他信息部分中的测试数据。 4. 选择 调用。	应用程序开发人员
使用 DynamodDB 奖励查询表验证。	1. 打开cqrses-reward-query桌子。 2. 查看兑换奖励的客户积分。应从客户总积分中减去已兑换的积分。	应用程序开发人员
使用 CloudWatch 日志进行验证。	1. 导航到日志组 CloudWatch 并选择该组。	应用程序开发人员

Task	描述	所需技能
	2. /aws/lambda/EventSourceReward日志组包含EventSourceReward 触发器的日志。所有 Lambda 调用都会被记录下来，包括你在 Lambda 代码中 context.Logger.LogLine 和 Console.Writeline 中输入的消息。	
验证 EventSourceCustomer 触发器。	要验证EventSourceCustomer 触发器，请使用EventSourceCustomer 触发器的相应客户表和 CloudWatch 日志，重复此长篇故事中的步骤。	应用程序开发人员

相关资源

参考

- Visual Studio 2019 Community Edition 下载
- AWS Toolkit for Visual Studio 下载
- AWS Toolkit for Visual Studio 用户指南
- Serverless on AWS
- DynamoDB 用例与设计模式
- Martin Fowler CQRS
- Martin Fowler 事件溯源

视频

- AWS Toolkit for Visual Studio 演示

- 如何为新 IAM 用户创建访问密钥 ID？

其他信息

CQRS 和事件溯源

CQRS

CQRS 模式将单个概念操作模型（例如数据访问对象单个 CRUD（创建、读取、更新、删除）模型）分离为命令和查询操作模型。命令模型是指任何更改状态的操作，如创建、更新或删除。查询模型是指任何返回值操作。

1. 客户 CRUD 模型包含以下接口：
   - `Create Customer()`
   - `UpdateCustomer()`
   - `DeleteCustomer()`
   - `AddPoints()`
   - `RedeemPoints()`
   - `GetVIPCustomers()`
   - `GetCustomerList()`
   - `GetCustomerPoints()`

随着您的需求变得更加复杂，您可放弃这种单一模型方法。CQRS 使用命令模型和查询模型分离写入和读取数据的职责。这样就可以独立维护和管理数据。通过明确职责分工，对每个模型的增强不会影响其他模型。这种分离可改善维护和性能，并随着应用程序的增长降低其复杂性。

1. 客户命令模型接口：
   - `Create Customer()`
   - `UpdateCustomer()`
   - `DeleteCustomer()`
   - `AddPoints()`
   - `RedeemPoints()`

2. 客户查询模型接口：

   - `GetVIPCustomers()`
   - `GetCustomerList()`
   - `GetCustomerPoints()`
   - `GetMonthlyStatement()`

有关示例代码，请参阅源代码目录。

然后，CQRS 模式可解耦数据库。这种解耦使每项服务能完全独立，这是微服务架构的主要组成部分。

在 Amazon Web Services Cloud 中使用 CQRS，您可进一步优化每项服务。例如，您可设置不同的计算设置，或者在无服务器或基于容器的微服务之间进行选择。您可以用 Amazon 替换您的本地缓存 ElastiCache。如果您本地有发布/订阅消息，则可将其替换为 Amazon Simple Notification Service (Amazon SNS)。此外，您还可以利用 pay-as-you-go定价和各种 AWS 服务，这些服务只需按实际用量付费。

CQRS 可提供以下优势：

- 独立扩展 — 每个模型都可调整其扩展策略，以满足服务要求和需求。与高性能应用程序类似的是，将读写分离可以使模型独立扩展，以满足每种需求。您还可以添加或减少计算资源，以满足模型的可扩展性需求，而不影响另一种模型。
- 独立维护 — 查询模型和命令模型的分离改进了模型的可维护性。您可在不影响另一个模型的情况下对一个模型进行代码更改和增强。
- 安全 - 可以更轻松地将权限和策略应用于不同的读取和写入模型。
- 优化读取 - 您可定义针对查询进行优化的架构。例如，您可为聚合数据定义一个架构，为事实表定义一个单独的架构。
- 集成 – CQRS 非常适合基于事件的编程模型。
- 托管复杂性 — 查询和命令模型的分离适合复杂的领域。

使用 CQRS 时，请记住以下注意事项：

- CQRS 模式仅适用于应用程序的特定部分，不适用于整个应用程序。如果在不适合该模式的领域实施，它会降低生产力、增加风险和引入复杂性。

- 该模式最适合具有不平衡读写操作的常用模型。
- 对于读取量大的应用程序（例如需要时间处理的大型报告），CQRS 使您可以选择正确的数据库，并创建一个架构以存储聚合数据。通过仅处理一次报告数据并将其转储到聚合表中，可以提高读取和查看报告的响应时间。
- 对于写入量大的应用程序，您可以配置数据库进行写入操作，并允许命令微服务在写入需求增加时独立扩展。有关示例，请参阅 `AWS.APG.CQRSES.CommandRedeemRewardLambda` 和 `AWS.APG.CQRSES.CommandAddRewardLambda` 微服务。

事件溯源

下一步，在运行命令时使用事件溯源同步查询数据库。例如，请考虑以下事件：

- 添加客户奖励积分，要求更新查询数据库中的客户总奖励积分或汇总奖励积分。
- 在命令数据库中更新客户姓氏，这要求更新查询数据库中的代理客户信息。

在传统 CRUD 模型中，您可通过锁定数据直到完成事务来确保数据的一致性。在事件溯源中，通过发布一系列事件来同步数据，订阅用户将使用这些事件来更新其各自的数据。

事件溯源模式可确保并记录一系列数据操作，并通过一系列事件将其发布。这些事件表示：该事件的订阅用户为保持记录更新而必须处理的一系列数据更改。这些事件由订阅用户使用，用于同步订阅用户数据库的数据。在本例中，就是查询数据库。

下图显示了 AWS 上与 CQRS 共用的事件溯源。

1. 命令 Lambda 函数对数据库执行写入操作，如创建、更新或删除。
2. 查询 Lambda 函数对数据库执行读取操作，如获取或选择。
3. 此 Lambda 函数处理来自命令数据库的 DynamoDB 数据流，并更新查询数据库以获取更改。您也可使用此功能向 Amazon SNS 发布消息，以便其订阅用户可以处理数据。
4. （可选）Lambda 事件订阅用户处理 Amazon SNS 发布的消息，并更新查询数据库。
5. （可选）Amazon SNS 会发送有关写入操作的电子邮件通知。

在 AWS 上，查询数据库可通过 DynamoDB Streams 进行同步。DynamoDB 可在 DynamoDB 表中捕获按时间排序的项目级修改序列，并在 24 个小时内持久存储信息。

激活 DynamoDB Streams 使数据库能发布一系列事件，从而使事件溯源模式成为可能。事件溯源模式添加事件订阅用户。事件订阅用户应用程序使用事件，并根据订阅用户的责任进行处理。在上图中，事件订阅用户将更改推送至查询 DynamoDB 数据库以保持数据同步。Amazon SNS、消息代理和事件订阅用户应用程序的使用，使架构保持独立。

事件溯源包括以下优势：

- 事务数据的一致性
- 可靠的审计跟踪记录和操作历史记录，可用于监控数据中采取的操作
- 允许微服务等分布式应用程序在整个环境中同步数据
- 当状态发生变化时，都能可靠地发布事件
- 重建或重现过去状态
- 松散耦合实体，用于交换事件以从单体应用程序迁移至微服务
- 减少由并发更新引起的冲突；事件溯源避免了直接在数据存储中更新对象的要求
- 通过将任务和事件脱钩，来实现灵活性和可扩展性
- 外部系统更新
- 单个事件中管理多项任务

使用事件溯源时，请记住以下注意事项：

- 由于源订阅用户数据库之间的数据更新会有延迟，因此撤消更改的唯一方法是向事件存储中添加补偿事件。
- 因为其编程风格不同，因此实现事件溯源需要学习曲线。

测试数据

成功部署后，使用以下数据测试 Lambda 函数。

CommandCreate Customer

```
{  "Id":1501,  "Firstname":"John",  "Lastname":"Done",  "CompanyName":"AnyCompany",
 "Address": "USA",  "VIP":true }
```

CommandUpdate Customer

```
{  "Id":1501,  "Firstname":"John",  "Lastname":"Doe",  "CompanyName":"Example Corp.",
 "Address": "Seattle, USA",  "VIP":true }
```

CommandDelete Customer

输入客户 ID 为请求数据。例如，如果客户 ID 为 151，则输入 151 为请求数据。

```
151
```

QueryCustomerList

此值为空。当它被调用时，将返回所有客户。

CommandAddReward

这将为身份为 1 的客户 (Richard) 增加 40 点积分。

```
{
  "Id":10101,
  "CustomerId":1,
  "Points":40
}
```

CommandRedeemReward

这将扣除 ID 为 1 的买家 (Richard) 的 15 点积分。

```
{
  "Id":10110,
  "CustomerId":1,
  "Points":15
}
```

QueryReward

输入客户 ID。例如，为 Richard 输入 1，为 Arnav 输入 2，为 Shirley 输入 3。

```
2
```

源代码目录

使用下表指导，以了解 Visual Studio 解决方案的目录结构。

CQRS 本地代码示例解决方案目录

客户 CRUD 模型

CQRS On-Premises Code Sample\CRUD Model\AWS.APG.CQRSES.DAL 项目

客户 CRUD 模型的 CQRS 版本

- 客户命令：CQRS On-Premises Code Sample\CQRS Model\Command Microservice \AWS.APG.CQRSES.Command 项目
- 客户查询：CQRS On-Premises Code Sample\CQRS Model\Query Microservice \AWS.APG.CQRSES.Query 项目

命令和查询微服务

Command 微服务位于解决方案文件夹 CQRS On-Premises Code Sample\CQRS Model \Command Microservice：

- AWS.APG.CQRSES.CommandMicroservice ASP.NET Core API 项目充当使用者与服务交互的入口。
- AWS.APG.CQRSES.Command .NET Core 项目是托管与命令相关的对象和接口的对象。

查询微服务位于解决方案文件夹 CQRS On-Premises Code Sample\CQRS Model\Query Microservice：

- AWS.APG.CQRSES.QueryMicroservice ASP.NET Core API 项目充当使用者与服务交互的入口。
- AWS.APG.CQRSES.Query .NET Core 项目是托管与查询相关的对象和接口的对象。

CQRS AWS 无服务器代码解决方案目录

此代码是使用 AWS 无服务器服务本地代码的 AWS 版本。

在 C# .NET Core 中，每个 Lambda 函数都由一个 .NET 核心项目表示。在此模式示例代码中，命令和查询模型中的每个接口都包含一个单独的项目。

使用 Amazon Web Services 的 CQRS

您可在 `CQRS AWS Serverless\CQRS` 文件夹中找到使用 AWS 无服务器服务的 CQRS 的根解决方案目录。该示例包括两个模型：即“客户” 和“奖励”。

客户和奖励的 Lambda 命令函数位于 `CQRS\Command Microservice\Customer` 和 `CQRS\Command Microservice\Reward` 文件夹。它们包含以下 Lambda 项目：

- 客户命令：`CommandCreateLambda`、`CommandDeleteLambda` 和 `CommandUpdateLambda`
- 奖励命令：`CommandAddRewardLambda` 和 `CommandRedeemRewardLambda`

“客户”和“奖励”的 Lambda 查询函数位于 `CQRS\Query Microservice\Customer` 和 `CQRS\QueryMicroservice\Reward` 文件夹下。它们包含 `QueryCustomerListLambda` 和 `QueryRewardLambda` Lambda 项目。

CQRS 测试项目

测试项目位于 `CQRS\Tests` 文件夹下。该项目包含用于自动测试 CQRS Lambda 函数的测试脚本。

使用 Amazon Web Services 的事件溯源

以下 Lambda 事件处理程序由客户和奖励 DynamoDB 流启动，旨在用于处理和同步查询表中的数据。

- `EventSourceCustomer` Lambda 函数映射到客户表 (`cqrses-customer-cmd`) DynamoDB 流。
- `EventSourceRewardLambda` 函数映射到奖励表 (`cqrses-reward-cmd`) DynamoDB 流。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

更多模式

- 使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon EKS 上私密访问容器应用程序
- 通过 AWS Systems Manager 自动添加或更新 Windows 注册表项
- 使用 DR Orchestrator 框架自动执行跨区域故障转移和故障恢复
- 使用 CI/CD 管道自动构建 Java 应用程序并将其部署到 Amazon EKS
- 自动使用 AWS CDK 为微服务构建 CI/CD 管道与 Amazon ECS 集群
- 使用 BMC AMI 云数据将大型机数据备份并存档到 Amazon S3
- 使用 Amazon A EC2 uto Scaling 和 Systems Manager 构建 Micro Focus 企业服务器 PAC
- 使用 Amazon 构建企业数据网格 DataZone AWS CDK，以及 AWS CloudFormation
- 使用无服务器方法将 Amazon Web Services 串在一起
- 对经过 Blu Age 现代化改造的大型机工作负载进行容器化
- 从 AWS 存储库持续部署现代 AWS Amplify 网络应用程序 CodeCommit
- 使用 Python 在 AWS 上将 EBCDIC 数据转换并解压为 ASCII
- 使用 Micro Focus 转换具有复杂记录布局的大型机数据文件
- 使用和 P HashiCorp acker 创建管道 CodePipeline 和 AMI
- 使用创建管道并将项目更新部署到本地 EC2 实例 CodePipeline
- 使用 AWS Amplify、Angular 和模块联合为微前端创建门户
- 部署和调试 Amazon EKS 集群
- 使用 Elastic Beanstalk 部署容器
- 通过使用兼容 PostgreSQL 的 Aurora 全局数据库来模拟 Oracle 灾难恢复
- 使用 AWS 大型机现代化和 Amazon Q 生成数据见解 QuickSight
- 使用和 AWS Mainframe Modernization Amazon Q 生成 Db2 z/OS 数据见解 QuickSight
- 迁移到 Amazon ECR 存储库时自动识别重复的容器映像
- 在 Amazon API Gateway 中使用自定义域名实现基于路径的 API 版本控制
- 使用 Oracle SQL Developer 和 AWS SCT 以增量方式从 Amazon RDS for Oracle 迁移至 Amazon RDS for PostgreSQL
- 将 Stonebranch Universal Controller 与 AWS Mainframe Modernization 集成
- 管理多个 Amazon Web Services account 和 AWS 区域中的 AWS Service Catalog 产品
- 将 AWS 成员账户从 AWS Organizations 迁移至 AWS Control Tower
- 使用 Connect from Precisely 将 VSAM 文件迁移和复制到 Amazon RDS 或 Amazon MSK

- 使用 AWS DMS 从 SAP ASE 迁移至 Amazon RDS for SQL Server
- 将 Oracle 外部表迁移到 Amazon Aurora PostgreSQL-Compatible
- 使用 Amazon Q 开发人员对 CardDemo 大型机应用程序进行现代化改造
- 使用 Rocket Enterprise Server 和 LRS V AWS PSX/MFI 实现大型机批量打印工作负载的现代化
- 使用 Micro Focus 企业服务器和 LRS VPSX/MFI 在 AWS 上实现大型机在线打印工作负载的现代化
- 使用 Rocket Enterprise Server 和 L PageCenter RS X 实现大型机输出管理的现代化 AWS
- 使用 Transfer Family 将大型机文件直接移动到 Amazon S3
- 优化 AWS App2Container 生成的 Docker 映像
- 使用 AWS CDK 和 GitHub 操作工作流程优化多账户无服务器部署
- 使用 IaC 原则自动部署 Amazon Aurora 全球数据库
- 使用 Precission Connect 将大型机数据库复制到 AWS
- 使用 Amazon ECS Anywhere 在亚马逊 WorkSpaces 上运行亚马逊 ECS 任务
- 将遥测数据从发送到 AWS Lambda 以 OpenSearch 进行实时分析和可视化
- 在 Amazon S3 中设置 Helm v3 图表存储库
- 在多区域、多账户组织中设置 AWS CloudFormation 偏差检测
- 使用 AWS Lambda 以六边形架构构建 Python 项目
- 使用 LocalStack 和 Terraform 测试来测试 AWS 基础架构
- 将 SAP Pacemaker 集群从升级到 ENSA1 ENSA2
- 使用 Amazon Q Developer 作为编码助手来提高工作效率
- 在本地验证 Account Factory for Terraform (AFT) 代码

主机

主题

- 通过安装， AWS 服务 从 IBM z/OS 进行访问 AWS CLI
- 使用 BMC AMI 云数据将大型机数据备份并存档到 Amazon S3
- 使用和构建 COBOL Db2 程序 AWS Mainframe ModernizationAWS CodeBuild
- 使用 Amazon A EC2 uto Scaling 和 Systems Manager 构建 Micro Focus 企业服务器 PAC
- 在 Amazon Web Services Cloud 中构建高级大型机文件查看器
- 对经过 Blu Age 现代化改造的大型机工作负载进行容器化
- 使用 Python 在 AWS 上将 EBCDIC 数据转换并解压为 ASCII
- 使用 AWS Lambda 在 Amazon S3 中将大型机文件从 EBCDIC 格式转换为字符分隔 ASCII 格式
- 使用 Micro Focus 转换具有复杂记录布局的大型机数据文件
- 使用 Terraform 为容器化 Blu Age 应用程序部署环境
- 使用和 AWS Mainframe Modernization Amazon Q 生成 Db2 z/OS 数据见解 QuickSight
- 使用 AWS 大型机现代化和 Amazon Q 生成数据见解 QuickSight
- 将 Stonebranch Universal Controller 与 AWS Mainframe Modernization 集成
- 使用 Connect from Precisely 将 VSAM 文件迁移和复制到 Amazon RDS 或 Amazon MSK
- 使用 Rocket Enterprise Server 和 L PageCenter RS X 实现大型机输出管理的现代化 AWS
- 使用 Amazon Q 开发人员对 CardDemo 大型机应用程序进行现代化改造
- 使用 Rocket Enterprise Server 和 LRS V AWS PSX/MFI 实现大型机批量打印工作负载的现代化
- 大型机现代化： AWS 使用 Rock DevOps et 软件企业套件开启
- 使用 Micro Focus 企业服务器和 LRS VPSX/MFI 在 AWS 上实现大型机在线打印工作负载的现代化
- 使用 Transfer Family 将大型机文件直接移动到 Amazon S3
- 使用可信上下文在 AWS 上的 Db2 联合身份验证数据库中保护和简化用户访问
- 以 CSV 文件形式将大规模 Db2 z/OS 数据传输到 Amazon S3
- 更多模式

通过安装， AWS 服务 从 IBM z/OS 进行访问 AWS CLI

由 Souma Ghosh (AWS)、Phil de Valence (AWS) 和 Paulo Vitor Pereira (AWS) 创作

摘要

AWS Command Line Interface (AWS CLI) 是一个开源工具，用于在命令行 shell 中使用命令来管理多个 AWS 服务。只需最少的配置，您就可以运行命令行会话（例如命令提示符、终端和 bash shell）中的命令，以实现与基于浏览 AWS Management Console器的功能相同的功能。

中的所有 AWS 基础设施即服务 (IaaS) 管理、管理和访问功能 AWS Management Console 均在 AWS API 和中提供。 AWS CLI您可以将安装在 IBM z/OS 大型机 AWS CLI 上，以便直接从 z/OS 进行访问、管理和交互 AWS 服务 。使用 AWS CLI 户和应用程序可以执行各种任务，例如：

- 在 z/OS 和亚马逊简单存储服务 (Amazon S3) Simple S3 对象存储之间传输文件或数据集并查看存储桶的内容
- 启动和停止不同的 AWS 资源；例如，在 AWS Mainframe Modernization 环境中启动批处理作业
- 调用 AWS Lambda 函数来实现常见的业务逻辑
- 与人工智能和机器学习 (AI/ML) 以及分析服务集成

此模式描述如何在 z/OS AWS CLI 上安装、配置和使用。您可以全局安装它，因此所有 z/OS 用户都可以使用它，也可以在用户级别上使用。该模式还详细说明了如何 AWS CLI 在 z/OS Unix 系统服务 (USS) 的交互式命令行会话中使用或将其用作批处理作业。

先决条件和限制

先决条件

- 从 z/OS 到 AWS

  默认情况下，在 TCP 端口 443 上使用 HTTPS AWS CLI 向发送请求。 AWS 服务 要 AWS CLI 成功使用，您必须能够在 TCP 端口 443 上建立出站连接。您可以使用以下任一 z/OS USS 命令（其中一些可能未安装在您的环境中）来测试从 z/OS 到以下的网络连接： AWS

```
ping amazonaws.com
dig amazonaws.com
traceroute amazonaws.com
curl -k https://docs.aws.amazon.com/cli/v1/userguide/cli-chap-welcome.html
```

- AWS 凭证

为了与 z/OS 中的 AWS Cloud 服务进行通信， AWS CLI 需要您配置一些具有访问目标权限的凭据。 AWS 账户对于编程命令 AWS，您可以使用访问密钥，其中包括访问密钥 ID 和私有访问密钥。如果没有访问密钥，您可以使用进行创建 AWS Management Console。最佳做法是，除非需要 root 用户，否则不要将 AWS 账户 root 用户的访问密钥用于任何任务。取而代之的是，创建一个新的管理员 IAM 用户，并准备好使用最低权限为该用户设置访问密钥。创建用户后，您可以为该用户创建访问密钥 ID 和私有访问密钥。

> ⚠ Warning
>
> AWS Identity and Access Management (IAM) 用户拥有存在安全风险的长期证书。为帮助减轻这种风险，我们建议仅向这些用户提供执行任务所需的权限，并在不再需要这些用户时将其移除。

- 适用于 z/OS 的 IBM Python

  AWS CLI 需要 Python 3.8 或更高版本。IBM 使用适用于 z/OS 的 IBM Open Enterprise Python 支持 Python 在 z/OS 上运行。IBM Open Enterprise Python 可通过 Shopz SMP/E 免费获得，也可以从 IBM 网站下载 PAX 文件。有关说明，请参阅适用于 z/OS 的 IBM Open Enterprise Python 的安装和配置文档。

限制

- 此模式中提供的安装说明仅适用于AWS CLI 版本 1。的最新版本 AWS CLI 是版本 2。但是，这种模式使用的是旧版本，因为版本 2 的安装方法不同，而且版本 2 中可用的二进制可执行文件与 z/OS 系统不兼容。

产品版本

- AWS CLI 第 1 版
- Python 3.8 或更高版本

架构

技术堆栈

- 运行 z/OS 的大型机

- 大型机 z/OS UNIX 系统服务 (USS)
- 大型机 Open MVS (OMVS) — z/OS UNIX 外壳环境命令界面
- 大型机磁盘，例如直接访问存储设备 (DASD)
- AWS CLI

目标架构

下图显示了 IBM z/OS 上的 AWS CLI 部署。您可以 AWS CLI 从交互式用户会话（例如 SSH）和 telnet 会话中调用。您也可以使用作业控制语言 (JCL) 从批处理作业中调用它，也可以从任何可以调用 z/OS Unix shell 命令的程序中调用它。

通过 TCP/IP 网络 AWS CLI 与 AWS 服务 端点通信。这种网络连接可以通过互联网进行，也可以通过从客户数据中心到 AWS Cloud 数据中心的私人 AWS Direct Connect 连接进行。通信使用 AWS 凭据进行身份验证并加密。

自动化和扩缩

您可以使用探索的功能 AWS CLI 并开发 USS shell 脚本来管理 z/OS 中的 AWS 资源。 AWS 服务 您还可以从 z/OS 批处理环境中运行 AWS CLI 命令和 shell 脚本，还可以通过与大型机调度器集成，自动执行批处理作业，使其按特定的计划运行。 AWS CLI 命令或脚本可以在参数 (PARMs) 和过程 (PROCs) 中编码，也可以按照标准方法进行扩展，即使用不同的参数从不同的批处理作业调用 PARM 或 PROC。

工具

- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行外壳中的命令进行交互。

最佳实践

- 出于安全考虑，请将访问权限限制为存储 AWS 访问密钥详细信息的 USS 目录。仅允许使用的用户或程序进行访问 AWS CLI。
- 请勿将 AWS 账户 root 用户访问密钥用于任何任务。取而代之的是，为自己创建一个新的管理员 IAM 用户，并使用访问密钥对其进行设置。

> **Warning**
>
> IAM 用户拥有存在安全风险的长期证书。为帮助减轻这种风险，我们建议仅向这些用户提供执行任务所需的权限，并在不再需要这些用户时将其移除。

操作说明

在 z/ AWS CLI OS USS 上安装版本 1

Task	描述	所需技能
安装 Python 3.8 或更高版本。	1. 使用以下方法之一登录 z/OS USS 命令提示符界面： • 使用交互式系统生产力工具 (ISPF) 面板中的分时选项 (TSO) OMVS 命令，或 • 使用 SSH 或 telnet 连接到大型机逻辑分区 (LPAR) 的 IP。 此模式假设`cliuser`这是用于登录 USS 环境的用户 ID，`/u/cliuser/` 也是用户的主目录。您可以根据安装要求在 z/OS 环境中以不同的方式设置用户主目录。 2. 如果尚未安装 Python 3.8 或更高版本，请按照适用于 z/OS 的 IBM Open Enterprise Python 安装指南进行安装。	大型机 z/OS 管理员
设置 USS 环境变量。	向配置文件添加环境变量。您可以将它们添加到单个用户的`/`	大型机 z/OS 管理员

Task	描述	所需技能

	u/cliuser/.profile 文件中 (cliuser)，也可以将其添加到所有用户的/etc/profile 文件中。
	
	Note
	此模式假定/u/awscli/python 目录中已安装 Python。如果您的安装目录不同，请相应地更新代码。
	
	`# Python configuration`
	`export BPXKAUTOCVT='ON'`
	`export CEERUNOPTS='FILETAG(AUTOCVT,AUTOTAG) POSIX(ON)'`
	`export TAGREDIR_ERR=txt`
	`export TAGREDIR_IN=txt`
	`export TAGREDIR_OUT=txt`
	
	`# AWS CLI configuration`
	`export PATH=/u/cliuser/python/bin:$PATH`
	`export PYTHONPATH=/u/cliuser/python:$PYTHONPATH`	

Task	描述	所需技能
测试 Python 的安装。	运行 python 命令： `python --version` 输出应确认您已正确安装了 Python 3.8 或更高版本。	大型机 z/OS 管理员

验证或安装 pip。	1. 当你从 IBM 网站安装 Python 时，pip 命令通常会自动安装。要进行验证，请运行以下命令：
	`pip —-version`
	如果安装了 pip，则此命令应显示已安装的版本。
	2. 如果找不到 pip 命令，请运行以下命令安装 pip：
	`python -m ensurepip --upgrade`
	有关更多安装选项，请参阅 pip 文档。
		大型机 z/OS 管理员

Task	描述	所需技能

安装 AWS CLI 版本 1。	1. 要安装 AWS CLI，请运行以下命令：
	`python -m pip install awscli`
	该输出值应该类似于以下内容。
	`Successfully installed PyYAML-6.0.1 awscli-1.32.23 botocore-1.34.23 colorama-0.4.4 docutils-0.16 jmespath-1.0.1 pyasn1-0.5.1 python-dateutil-2.8.2 rsa-4.7.2 s3transfer-0.10.0 urllib3-2.0.7`
	2. 通过运行以下命令更改 aws 可执行文件的权限。请务必使用您的 Python 安装路径更新占位<python_installation_dir> 符目录。
	`chmod 744 <python_installation_dir>/bin/aws`
	3. 运行以下命令来测试 AWS CLI 安装：
	`aws --version`
		大型机 z/OS 管理员

Task	描述	所需技能
	输出应显示 AWS CLI、Python 和 botocore 的版本，如下所示。 `aws-cli/1.32.3 Python/3.9.5 OS/390/27.00 botocore/1.34.3`	

配置来自 z/OS 的 AWS CLI 访问权限

Task	描述	所需技能

配置 AWS 访问密钥、默认区域和输出。	该AWS CLI 文档描述了设置 AWS 访问权限的不同选项。您可以根据组织的标准选择配置。此示例使用短期凭证配置。
	1. AWS CLI使用@@ 以下命令配置：
	`aws configure`
	2. 出现提示时，请提供以下项目的详细信息。访问密钥 ID 和私有访问密钥值来自您在先决条件步骤中设置 AWS 凭据时获得的密钥。
	`AWS Access Key ID [None]: ASIAIOSFO DNN7EXAMPLE AWS Secret Access Key [None]: wJalrXUtn`
		AWS 管理员、大型机 z/OS 管理员、大型机 z/OS 开发人员

Task	描述	所需技能
	FEMI/K7MDENG/bPxRf iCYEXAMPLEKEY Default region name [None]: us-east-1 Default output format [None]: aws configure set aws_session_token IQoJb3JpZ2luX2IQoJ b3JpZ2luX2IQoJb3Jp Z2luX2IQoJb3JpZ2lu X2IQoJb3JpZVERYLON GSTRINGEXAMPLE 此配置（包括访问密钥）存储在/u/cliuser/.aws 文件夹中。出于安全考虑，请将此文件夹限制为仅允许使用该文件夹的用户或程序进行访问 AWS CLI。	

Task	描述	所需技能
测试 AWS CLI.	1. 在命令提示符下运行以下命令，AWS CLI 使用简单的命令进行测试： `aws s3 ls` 输出应列出配置的所有 S3 存储桶，AWS 账户 且不会出现任何错误。 2. 按照接下来的两个长篇故事中的说明将数据从 USS 传输到 Amazon S3。您可以从这两个选项中选择其一： • 选项 1（下一个长篇故事）：以交互方式将 EBCDIC 逗号分隔值 (CSV) 文件传输到 Amazon S3，然后从亚马逊 Athena 查询该文件。 • 选项 2：将 EBCDIC 固定长度数据集作为批处理任务传输到 Amazon S3。	大型机 z/OS 管理员、大型机 z/OS 开发人员

选项 1-从 USS 会话以交互方式将数据从 USS 传输到 Amazon S3

Task	描述	所需技能
下载并传输示例 CSV 文件。	1. `sales-records.csv` 从“附件”部分下载。此文件提供了包含销售记录的 CSV 示例。 2. 将文件传输到 z/OS USS。	应用程序开发人员、大型机 z/OS 开发人员

Task	描述	所需技能
	3. 使用您选择的文本编辑器在 USS 中验证`/u/cliuser/sales-records.csv` 文件是否可读取 EBCDIC 格式。	

Task	描述	所需技能

创建 S3 存储桶并上传 CSV 文件。	1. 创建一个 S3 存储桶来存储 CSV 文件。
	`aws s3 mb s3://<s3_bucket_name>`
	其中<s3_bucket_name> 是存储桶的唯一名称；例如：
	`aws s3 mb s3://DOC-EXAMPLE-BUCKET1`
	2. 将 CSV 文件从 z/OS USS 上传到 S3 存储桶：
	`aws s3 cp <csv_file_path> s3://<s3_bucket_name>`
	例如：
	`aws s3 cp /u/cliuser/sales-records.csv s3://DOC-EXAMPLE-BUCKET1`
	3. 列出 S3 存储桶的内容并确认其中包含上传的文件：
	`aws s3 ls s3://<s3_bucket_name>`
	例如：
		应用程序开发人员、大型机 z/OS 开发人员

Task	描述	所需技能
	`aws s3 ls s3://DOC-EXAMPLE-BUCKET1`	
查看 S3 存储桶和上传的文件。	1. 登录 AWS Management Console 并打开 Amazon S3 控制台。 2. 导航以查看新的 S3 存储桶和上传的对象。 有关上传对象的更多信息，请参阅 Amazon S3 文档中的 Amazon S3 入门。	常规 AWS

Task	描述	所需技能

对亚马逊 Athena 表运行 SQL 查询。	1. 打开 Amazon Athena 控制台。
	2. 使用来自 Amazon S3 的 CSV 数据创建新表（例如 DOC-EXAMPLE-BUCKET ）。有关更多信息，请参阅亚马逊 S3 文档中的使用亚马逊 Athena 查询亚马逊 S3 库存。
	3. 对表运行SELECT查询以查看数据。
	`SELECT * FROM <table_name>;`
	例如：
	`SELECT * FROM DOC-EXAMPLE-BUCKET;`
	SQL 查询的输出将显示您的 CSV 文件的内容。
		常规 AWS，应用程序开发人员

选项 2-使用批处理 JCL 将数据从 USS 传输到 Amazon S3

Task	描述	所需技能
上传示例文件。	1. `sales-records-fixed.txt` 从“附件”部分下载。这是包含销售记录的示例文件。将文本文件重命名；例如，重命名为`USER.DATA.FIXED` 。	大型机 z/OS 开发人员

Task	描述	所需技能
	2. 将文件作为固定块 (FB)、记录长度为 256 (LRECL)、物理顺序 (PS) 的数据集传输到 z/OS。 3. 在 ISPF 选项 3.4 下，使用`USER.DATA.FIXED` 数据集列表实用程序验证数据集是否可读取 EBCDIC 格式。有关输出示例，请参阅“其他信息”部分。	

Task	描述	所需技能

创建批处理 JCL。	按如下方式对批处理 JCL 进行编码，以创建目标 S3 存储桶、上传数据集并列出存储桶内容。请务必将目录名称、文件名和存储桶名称替换为您自己的值。
	
	`//AWSCLICP JOB`
	` ACTINFO1,'IBMUSER'`
	`,CLASS=A,MSGCLASS=`
	`H,MSGLEVEL=(1,1),`
	`// NOTIFY=&SYSUID,TIM`
	`E=1440`
	`//*--------------`
	`------------------`
	`------------------`
	`-------`
	`//* Sample job for AWS`
	` CLI`
	`//*--------------`
	`------------------`
	`------------------`
	`-------`
	`//USSCMD EXEC PGM=BPXBA`
	`TCH`
	`//STDERR  DD SYSOUT=*`
	`//STDOUT  DD SYSOUT=*`
	`//STDENV  DD *`
	` export PATH=/u/c`
	`liuser/python/bin:`
	`$PATH`
	`//STDPARM DD *`
	`SH`
	` export _BPXK_AUT`
	`OCVT=ON;`
	` aws s3 mb s3://DOC-`
	`EXAMPLE-BUCKET2;`
	` cp "//'USER.DATA.FIXE`
	`D'" /tmp/tmpfile;`
		大型机 z/OS 开发人员

Task	描述	所需技能
	``` aws s3 cp /tmp/tmpf ile s3://DOC-EXAMPLE- BUCKET2/USER.DATA.F IXED; rm /tmp/tmpfile; aws s3 ls s3://DOC- EXAMPLE-BUCKET2; /* ```	
提交批处理 JCL 作业。	1. 提交您在上一步中编码的 JCL 作业。 2. 在“系统显示和搜索工具”(SDSF) 中检查作业的状态。如果成功，则任务应以返回码 0 结束。 3. 任务日志的标准输出 (STDOUT) 显示存储桶的成功创建、数据集上传和存储桶内容列表。有关屏幕插图示例，请参阅“其他信息”部分。	大型机 z/OS 开发人员
查看上传到 S3 存储桶的数据集。	1. 登录 AWS Management Console 并打开 Amazon S3 控制台。 2. 导航以在测试存储桶中查看上传的文件。 3. 您可以进一步处理`USER.DATA.FIXED` 文件或使用诸如 Amazon Redshift 之类的分析服务对其进行分析。	常规 AWS

相关资源

- AWS CLI 版本 1 文档
- AWS Mainframe Modernization CLI 命令参考
- AWS Mainframe Modernization

其他信息

ISPF 选项 3.4（数据集列表实用程序）中的 USER.DATA.FIXED

已提交的批处理作业的 SYSOUT

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 BMC AMI 云数据将大型机数据备份并存档到 Amazon S3

由 Santosh Kumar Singh (AWS)、Mikhael Liberman（Model9 大型机软件）、Gilberto Biondo (AWS) 和 Maggie Li (AWS) 创建

摘要

此模式演示了如何将大型机数据直接备份和存档到亚马逊简单存储服务 (Amazon S3) Service，然后使用 BMC AMI 云数据（以前称为 Model9 Manager）将这些数据调回并恢复到大型机。如果您正在寻找一种方法来实现备份和存档解决方案的现代化，以此作为大型机现代化项目的一部分，或者为了满足合规性要求，那么这种模式可以帮助实现这些目标。

通常，在大型机上运行核心业务应用程序的组织使用虚拟磁带库 (VTL) 来备份文件和日志等数据存储。这种方法可能很昂贵，因为它消耗了可计费的 MIPS，而且存储在大型机之外的磁带上的数据无法访问。为避免这些问题，您可以使用 BMC AMI Cloud Data 快速且经济高效地将大型机的运营和历史数据直接传输到 Amazon S3。您可以使用 BMC AMI Cloud Data 通过 TCP/IP 备份和存档数据， AWS 同时利用 IBM z 集成信息处理器 (ZiIP) 引擎来降低成本、并行性和传输时间。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 带有有效许可证密钥的 BMC AMI 云数据
- 大型机和 AWS 之间的 TCP/IP 连接
- 用于对 S3 存储桶进行读/写访问的 AWS Identity and Access Management (IAM) 角色
- 提供大型机安全产品 (RACF) 访问权限以运行 BMC AMI Cloud 流程
- 具有可用网络端口、允许访问 S3 存储桶的防火墙规则和专用 z/FS 文件系统的 BMC AMI Cloud z/OS 代理（Java 版本 8、64 位 SR5 FP16 或更高版本）
- 已满足 BMC AMI 云管理服务器的@@ 要求

限制

- BMC AMI Cloud Data 将其操作数据存储在 PostgreSQL 数据库中，该数据库作为 Docker 容器运行在与管理服务器相同的亚马逊弹性计算云 (Amazon) EC2 实例上。目前不支持将亚马逊关系数据库服务（Amazon RDS）用作 BMC AMI 云数据的后端。有关最新产品更新的更多信息，请参阅新增内容？ 在 BMC 文档中。

- 这种模式仅备份和存档 z/OS 大型机数据。BMC AMI 云数据仅备份和存档大型机文件。
- 此模式不会将数据转换为标准的开放格式，例如 JSON 或 CSV。使用其他转换服务，例如 BMC AMI Cloud Analytics（以前称为 Model9 Gravity），将数据转换为标准的开放格式。云原生应用程序和数据分析工具可以在数据写入云端后对其进行访问。

产品版本

- BMC AMI 云数据版本 2.x

架构

源技术堆栈

- 运行 z/OS 的大型机
- 大型机文件，例如数据集和 z/OS UNIX 系统服务 (USS) 文件
- 大型机磁盘，例如直接访问存储设备 (DASD)
- 大型机磁带（虚拟或物理磁带库）

目标技术堆栈

- Amazon S3
- 虚拟私有云 (VPC) 中的亚马逊 EC2 实例
- AWS Direct Connect
- Amazon Elastic File System (Amazon EFS)

目标架构

下图显示了一个参考架构，在该架构中，大型机上的 BMC AMI Cloud Data 软件代理驱动将数据存储在 Amazon S3 中的传统数据备份和存档流程。

图表显示了以下工作流：

1. BMC AMI 云数据软件代理在大型机逻辑分区上运行 () LPARs。软件代理通过 TCP/IP 将大型机数据从 DASD 或磁带直接读写到 Amazon S3。

2. AWS Direct Connect 在本地网络和之间建立物理隔离连接 AWS。为了增强安全性，请在上面运行 site-to-site VPN AWS Direct Connect 来加密传输中的数据。
3. S3 存储桶将大型机文件存储为对象存储数据，而 BMC AMI Cloud Data 代理直接与 S3 存储桶通信。证书用于对代理与 Amazon S3 之间的所有通信进行 HTTPS 加密。Amazon S3 数据加密功能用于加密和保护静态数据。
4. BMC AMI 云数据管理服务器在实例上 EC2 作为 Docker 容器运行。这些实例与在大型机 LPARs 和 S3 存储桶上运行的代理通信。
5. Amazon EFS 安装在主动和被动 EC2 实例上，用于共享网络文件系统 (NFS) 存储。这是为了确保与在管理服务器上创建的策略相关的元数据不会在故障转移时丢失。在主动服务器进行故障转移时，可以访问被动服务器而不会丢失任何数据。如果被动服务器出现故障，则可以在不丢失任何数据的情况下访问主动服务器。

工具

Amazon Web Services

- 亚马逊弹性计算云 (Amazon EC2) 在中提供可扩展的计算容量 AWS Cloud。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- Amazon Elastic File System（Amazon EFS）可帮助您在中创建和配置共享文件系统 AWS Cloud。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索几乎任何数量的数据。
- Amazon Virtual Private Cloud（亚马逊 VPC）可帮助您将 AWS 资源启动到您定义的虚拟网络中。该虚拟网络类似于您在数据中心中运行的传统网络，并具有使用 AWS的可扩展基础设施的优势。
- AWS Direct Connect通过标准以太网光纤电缆将您的内部网络链接到某个 AWS Direct Connect 位置。通过此连接，您可以直接创建通往公共 AWS 服务的虚拟接口，同时绕过网络路径中的互联网服务提供商。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。

BMC 工具

- BMC AMI 云管理服务器是一个 GUI 应用程序，作为 Docker 容器在适用于亚马逊的亚马逊 Linux 亚马逊系统映像 (AMI) 上运行。 EC2管理服务器提供管理 BMC AMI Cloud 活动的功能，例如报告、创建和管理策略、运行存档以及执行备份、召回和恢复。

- BMC AMI Cloud 代理在本地大型机 LPAR 上运行，该主机使用 TCP/IP 将文件直接读写到对象存储。已启动的任务在大型机 LPAR 上运行，负责在 Amazon S3 中读取和写入备份和存档数据。
- BMC AMI Cloud 大型机命令行界面 (M9CLI) 为您提供了一组命令，可直接从 TSO/E 或批量操作中执行 BMC AMI Cloud 操作，无需依赖管理服务器。

操作说明

创建 S3 存储桶和 IAM policy

Task	描述	所需技能
创建 S3 存储桶。	创建 S3 存储桶，以存储大型机环境中待备份和待存档的文件和卷。	常规 AWS
创建一个 IAM 策略。	所有 BMC AMI Cloud 管理服务器和代理都需要访问您在上一步中创建的 S3 存储桶。 若要授予所需访问权限，请创建以下 IAM policy： `{ "Version": "2012-10-17", "Statement": [ { "Sid": "Listfolder", "Action": [ "s3:ListBucket", "s3:GetBucketLocation", "s3:ListBucketVersions" ],`	常规 AWS

Task	描述	所需技能
	"Effect": "Allow", "Resource": ["arn:aws:s3:::<Bucket Name>"] }, { "Sid": "Objectaccess", "Effect": "Allow", "Action": ["s3:PutObject", "s3:GetObjectAcl", "s3:GetObject", "s3:DeleteObjectVersion", "s3:DeleteObject", "s3:PutObjectAcl", "s3:GetObjectVersion"], "Resource": ["arn:aws:s3:::<Bucket Name>/*"] }] }	

获取 BMC AMI Cloud 软件许可证并下载软件

Task	描述	所需技能
获取 BMC AMI Cloud 软件许可证。	要获取软件许可密钥，请联系 BMC AMI Cloud 团队。z/OS D M=CPU 命令的输出是生成许可证的必要条件。	构建 lead
下载 BMC AMI Cloud 软件和许可密钥。	按照 BMC 文档中的说明获取安装文件和许可证密钥。	大型机基础架构管理员

在大型机上安装 BMC AMI Cloud 软件代理

Task	描述	所需技能
安装 BMC AMI Cloud 软件代理。	1. 在开始安装过程之前，请验证是否已满足代理的最低软件和硬件要求。 2. 要安装代理，请按照 BMC 文档中的说明进行操作。 3. 代理开始在大型机 LPAR 上运行后，检查后台中的 `ZM91000I MODEL9 BACKUP AGENT INITIALIZED` 消息。通过在代理的 STDOUT 中查找`Object store connectivity has been established successfully` 消息，验证代理与 S3 存储桶之间是否成功建立了连接。	大型机基础架构管理员

在实例上设置 BMC AMI 云管理服务器 EC2

Task	描述	所需技能
创建 Amazon EC2 Linux 2 实例。	按照亚马逊 EC2 文档中的步骤 1：启动实例中的说明，在不同的可用区域中启动两个 Amazon EC2 Linux 2 实例。 该实例必须满足以下推荐的硬件和软件要求： • CPU – 至少 4 核 • RAM – 最低 8GB • 驱动器 – 40 GB • 推荐 EC2 实例 — c5.xLarge • 操作系统 – Linux • 软件 – Docker、unzip、vi/vim • 网络带宽-至少 1 GB 有关更多信息，请参阅 BMC 文档。	云架构师、云管理员
创建 Amazon EFS 文件系统。	按照 Amazon EFS 文档中的步骤 1：创建 Amazon EFS 文件系统中的说明创建 Amazon EFS 文件系统。 创建文件系统时，请执行以下操作： • 选择标准存储类。 • 选择您用于启动 EC2 实例的 VPC。	云管理员、云架构师

Task	描述	所需技能

安装 Docker 并配置管理服务器。	Connect 连接到您的 EC2 实例：
	
	按照亚马逊 EC2 文档中连接您 EC2 的 Linux 实例中的说明连接到您的实例。
	
	配置您的 EC2 实例：
	
	对于每个 EC2 实例，请执行以下操作：
	
	1. 要安装 Docker，请运行以下命令：
	
	`sudo yum install docker`
	
	2. 要启动 Docker，请运行以下命令：
	
	`sudo service docker start`
	
	3. 要验证 Docker 的状态，请运行以下命令：
	
	`sudo service docker status`
	
	4. 在 `/etc/selinux` 文件夹在，将 `config` 文件更改为 `SELINUX=permissive`。
	
	5. 将`model9-v2.x.y_build_build-id-server.zip` 和`Verificat`
		云架构师、云管理员

Task	描述	所需技能

	ionScripts.zip 文件（您之前下载的）上传到其中一个 EC2 实例中的临时文件夹（例如，上传到您的实例中的/var/tmp文件夹）。
	6. 要转到该tmp文件夹，请运行以下命令：
	`cd/var/tmp`
	7. 要解压缩验证脚本，请运行以下命令：
	`unzip Verificat ionScripts.zip`
	8. 要更改目录，请运行以下命令：
	`cd /var/tmp/ sysutils/PrereqsSc ripts`
	9. 要运行验证脚本，请运行以下命令：
	`./M9VerifyPrereqs. sh`
	10验证脚本提示输入后，输入 Amazon S3 网址和端口号。然后，输入 z/OS IP/DNS 与端口号。	

Task	描述	所需技能
	Note 该脚本运行检查以确认该 EC2 实例可以连接到大型机上运行的 S3 存储桶和代理。如果连接已建立，则会显示一条成功消息。	

Task	描述	所需技能

安装管理服务器软件。	1. 在您计划将服务器设为活动服务器的 EC2 实例的根目录（例如/data/model9 ）中创建文件夹和子文件夹。
	2. 要安装amazon-efs-utils软件包并挂载之前创建的 Amazon EFS 文件系统，请运行以下命令：
	`sudo yum install -y amazon-efs-utils`
	`sudo mount -t efs -o tls <File System ID>:/ /data/model9`
	3. 要使用 Amazon EFS /etc/fstab 文件系统的条目更新 EC2 实例的文件（以便在 Amazon 重 EC2 启时自动重新挂载 Amazon EFS），请运行以下命令：
	`<Amazon-EFS-file-system-id>:/ /data/model9 efs defaults, _netdev 0 0`
	4. 要定义 BMC AMI Cloud 安装文件的路径和目标安装位置，请运行以下命令导出变量：
	`export MODEL9_HOME=/data/model9`
	`export M9INSTALL=/var/tmp`
		云架构师、云管理员

Task	描述	所需技能

	Note
	我们建议您将这些 EXPORT 命令添加到.bashrc脚本中。
	5. 若要更改目录，请运行 cd $MODEL9_HOME 命令，然后通过运行 mkdir diag 命令创建另一个子目录。
	6. 要解压缩安装文件，请运行以下命令：
	`unzip $M9INSTALL/model9-<v2.x.y>_build_<build-id>-server.zip`
	Note
	用您的值替换x.y（版本）和build-id。
	7. 若要部署应用程序，请运行以下命令：
	`docker load -i $MODEL9_HOME/model9-<v2.x.y>_build_<build-id>.docker`
	`docker load -i $MODEL9_HOME/postg`	

Task	描述	所需技能

	`res-12.10-x86.docker.gz`
	
	Note
	用您的值替换`v2.x.y`（版本）和`build-id`。
	
	8. 在 `$MODEL9_HOME/conf`文件夹中，更新 `model9-local.yml` 文件。
	
	Note
	有些参数具有默认值，其他参数可以根据需要进行更新。有关更多信息，请参阅 `model9-local.yml` 文件中的说明。
	
	9. 创建一个名为的文件 **`$MODEL9_HOME/conf`** ，然后在该文件中添加以下参数：
	
	`TZ=America/New_York`
	`EXTRA_JVM_ARGS=-Xmx2048m`
	
	10要创建 Docker 网桥，请运行以下命令：	

Task	描述	所需技能

	`docker network create -d bridge model9network`
	11.要启动适用于 BMC AMI Cloud 的 PostgreSQL 数据库容器，请运行以下命令：
	`docker run -p 127.0.0.1:5432:5432 \ -v $MODEL9_HOME/db/data:/var/lib/postgresql/data:z \ --name model9db --restart unless-stopped \ --network model9network \ -e POSTGRES_PASSWORD=model9 -e POSTGRES_DB=model9 -d postgres:12.10`
	12.PostgreSQL 容器开始运行后，运行以下命令，以启动应用程序服务器：
	`docker run -d -p 0.0.0.0:443:443 -p 0.0.0.0:80:80 \ --sysctl net.ipv4.tcp_keepalive_time=600 \ --sysctl net.ipv4.tcp_keepalive_intvl=30 \`	

Task	描述	所需技能

	`--sysctl net.ipv4.tcp_keepalive_probes=10 \ -v $MODEL9_HOME:/model9:z -h $(hostname) --restart unless-stopped \ --env-file $MODEL9_HOME/conf/model9.env \ --network model9network \ --name model9-v2.x.y model9:<v2.x.y>.<build-id>`
	
	Note
	用您的值替换v2.x.y（版本）和build-id。
	
	13要检查两个容器的运行状况，请运行以下命令：
	
	`docker ps -a`
	
	14要在被动 EC2 实例上安装管理服务器，请重复步骤 1—4、7 和 10—13。
	
	Note
	要解决问题，请访问存储在/data/model9/logs/ 文件夹中的日志。有关更多	

Task	描述	所需技能
	信息，请参阅 BMC 文档。	

在 BMC AMI 云管理服务器上添加代理并定义备份或存档策略

Task	描述	所需技能
添加新代理。	在添加新代理之前，请确认以下事项： • BMC AMI Cloud 代理正在大型机 LPAR 上运行，并且已完全初始化。通过在缓冲池中查找`ZM91000I MODEL9 BACKUP AGENT INITIALIZED` 初始化消息来识别代理。 • 管理服务器的 Docker 容器已完全初始化并正在运行。 在定义任何备份和存档策略之前，必须在管理服务器上创建代理。若要创建代理，请执行以下操作： 1. 使用网络浏览器访问部署在您的 Amazon EC2 计算机上的管理服务器，然后使用您的大型机凭证登录。 2. 选择代理选项卡，然后选择添加新代理。 3. 对于名称，请输入代理名称。	大型机存储管理员或开发人员

Task	描述	所需技能
	4. 在“主机名/IP 地址”中，输入您的大型机的主机名或 IP 地址。 5. 对于端口，请输入端口号。 6. 选择测试链接。如果连接成功建立，则会看到一条成功消息。 7. 选择 CREATE（创建）。 创建代理后，您将在表格中出现的新窗口中看到对象存储和大型机代理的连接状态。	
创建备份或存档策略。	1. 选择策略。 2. 选择创建策略。 3. 在创建新策略页面上，输入您的策略规范。 **Note** 有关可用规范的更多信息，请参阅 BMC 文档中的创建新策略。 4. 选择 Finish (结束)。 5. 现在以表格形式列出新策略。若要查看此表，请选择策略选项卡。	大型机存储管理员或开发人员

从管理服务器运行备份或存档策略

Task	描述	所需技能
运行备份或存档策略。	手动或自动（根据计划）运行您之前在管理服务器上创建的数据备份或存档策略。要手动运行策略，请执行以下操作： 1. 从导航菜单中，选择策略。 2. 在要运行的策略的表格右侧，选择三点菜单。 3. 选择立即运行。 4. 在弹出的确认窗口中，选择是，立即运行策略。 5. 策略运行后，在“策略活动”部分验证运行状态。 6. 对于已运行的策略，选择三点菜单，然后选择查看运行日志以查看日志。 7. 要验证备份是否已创建，请检查 S3 存储桶。	大型机存储管理员或开发人员
恢复备份或存档策略。	1. 在导航菜单上，选择策略选项卡。 2. 选择用于运行还原过程的策略。这将列出过去针对该特定策略运行的所有备份或存档活动。 3. 若要选择待还原备份，请选择 Date-time 列。`file/Volume/Storage` 组名显示了策略的运行详细信息。	大型机存储管理员或开发人员

Task	描述	所需技能
	4. 在表格右侧，选择三点菜单，然后选择 “恢复”。 5. 在弹出窗口中，输入目标名称、卷和存储组，然后选择还原。 6. 输入大型机凭证，然后再次选择恢复。 7. 要验证恢复是否成功，请检查日志或大型机。	

从大型机运行备份或存档策略

Task	描述	所需技能
使用 M9CLI 运行备份或存档策略。	使用 M9CLI 从 TSO/E、REXX 执行备份和还原过程， JCLs 无需在 BMC AMI Cloud 管理服务器上设置规则。 使用 TSO/E： 如果您使用 TSO/E，请确保将其连接M9CLI REXX到。TSO要通过 TSO/E 备份数据集，请使用命令。`TSO M9CLI BACKDSN <DSNAME>` **Note** 有关 M9CLI 命令的更多信息，请参阅 BMC 文档中的 CLI 参考。	大型机存储管理员或开发人员

Task	描述	所需技能

	使用 JCLs：
	
	要使用运行备份和存档策略 JCLs，请运行M9CLI命令。
	
	使用批量操作：
	
	以下示例说明如何通过批量运行M9CLI命令来存档数据集：
	
	`//JOBNAME JOB …`
	`//M9CLI EXEC PGM=IKJEF`
	`T01`
	`//STEPLIB DD DISP=SHR,`
	`DSN=<MODEL9 LOADLIB>`
	`//SYSEXEC DD DISP=SHR,`
	`DSN=<MODEL9 EXEC LIB>`
	`//SYSTSPRT DD SYSOUT=*`
	`//SYSPRINT DD SYSOUT=*`
	`//SYSTSIN DD TSO M9CLI`
	` ARCHIVE`
	` M9CLI ARCHIVE <DSNNAME`
	` OR DSN PATTERN>`
	`/`	

Task	描述	所需技能
在 JCL 批处理中运行备份或存档策略。	BMC AMI Cloud 提供了一个名为 M9SAPIJ 的 JCL 例程示例。 您可以自定义 M9SAPIJ 以运行使用 JCL 在管理服务器上创建的特定策略。此任务亦可作为批处理计划程序的一部分，用于自动运行备份和还原过程。 批处理任务需要提供以下必需值： • 管理服务器 IP 地址/主机名 • 端口号 • 策略 ID 或策略名称（在管理服务器上创建） **Note** 您也可以按照示例作业中的说明更改其他值。	大型机存储管理员或开发人员

相关资源

- 使用 AWS 实现大型机现代化（AWS 文档）
- 适用于大型机的 Cloud Backup 如何使用 Model9 和 AWS 降低成本（Amazon Web Services Partner Network 博客）
- 如何使用 Model9 在 AWS 上启用大型机数据分析（Amazon Web Services Partner Network 博客）
- AWS Direct Connect 弹性建议（AWS 文档）
- BMC AMI Cloud 文档（BMC 网站）

使用和构建 COBOL Db2 程序 AWS Mainframe ModernizationAWS CodeBuild

由 Luis Gustavo Dantas (AWS) 和 Eduardo Zimelewicz (AWS) 创作

摘要

此模式说明了如何使用 Replatform 工具创建一个用于预编译和绑定 COBOL Db2 程序的简单 AWS CodeBuild 项目。 AWS Mainframe Modernization 这允许在 AWS Mainframe Modernization Replatform 运行时环境中部署和执行这些程序。

COBOL 是一种面向业务的编程语言，由于其可靠性和可读性，它为许多关键应用程序提供了支持。IBM Db2 是一种关系数据库管理系统，可高效管理大量数据，并通过 SQL 与 COBOL 程序集成。尽管出现了新技术，但COBOL和Db2共同构成了金融和政府等行业关键任务运营的支柱。

将 COBOL 和 Db2 组件从大型机环境迁移到其他平台会带来诸如平台兼容性、集成复杂性、数据迁移和性能优化之类的挑战。移动这些关键组件需要仔细的规划、技术专业知识和资源，以确保顺利迁移，同时保持可靠性和功能性。

该 AWS Mainframe Modernization 服务提供工具和资源，用于将大型机应用程序和数据库改为在 AWS 基础设施（例如亚马逊弹性计算云 (Amazon) 实例 EC2）上运行。这涉及在不进行重大代码更改的情况下将大型机工作负载迁移到云端。

Db2 预编译和绑定过程对于优化数据库应用程序的性能和可靠性至关重要。预编译可将嵌入式 SQL 语句转换为可执行代码，从而减少运行时开销并提高效率。绑定过程将预编译的代码与数据库结构联系起来，从而便于访问路径和查询优化。此过程可确保数据完整性，提高应用程序响应能力，并防范安全漏洞。正确预编译和绑定应用程序可最大限度地减少资源消耗、增强可扩展性并降低 SQL 注入攻击的风险。

先决条件和限制

先决条件

- AWS 账户 和管理员级别的控制台访问权限。
- 一个 IBM Db2 数据库系统，例如适用于 z/OS 的 IBM Db2 或适用于 Linux、Unix 和 Windows 的 Db2 (LUW)。
- IBM 数据服务器客户机软件，可从 IBM 网站下载。有关更多信息，请参阅 IBM 数据服务器客户机和数据服务器驱动程序类型。

- 一个要编译和绑定的 COBOL Db2 程序。或者，此模式提供了一个基本的示例程序供您使用。
- AWS 带有私有网络的虚拟私有云 (VPC)。有关创建 VPC 的信息，请参阅亚马逊虚拟私有云（亚马逊 VPC）文档。
- 源代码控制存储库，例如 GitHub 或 GitLab。

限制

- 有关 AWS CodeBuild 配额，请参阅配额 AWS CodeBuild。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额页面，然后选择服务的链接。

架构

源技术堆栈

源堆栈包括：

- 使用 Db2 数据库存储数据的 COBOL 程序
- IBM COBOL 编译器和适用于 z/OS 的 Db2 预编译器
- 大型机设置的其他部分，例如文件系统、事务管理器和缓冲池

目标技术堆栈

这种模式的方法适用于两个选项：将数据从适用于 z/OS 的 Db2 移动到 LUW 的 Db2，或者在 z/OS 上停留在 Db2 上。目标架构包括：

- 使用 Db2 数据库存储数据的 COBOL 程序
- AWS Mainframe Modernization 平台重组编译工具
- AWS CodeBuild 作为构建应用程序的基础架构
- 其他 AWS Cloud 资源，例如亚马逊 Linux

目标架构

该图阐释了以下内容：

1. 用户将其代码上传到源代码控制存储库，例如 GitHub 或 GitLab。
2. AWS CodePipeline 注意到更改并从存储库中获取代码。
3. CodePipeline 启动 AWS CodeBuild 并发送代码。
4. CodeBuild 按照`buildspec.yml`模板（在“其他信息”部分中提供）中的说明执行以下操作：
   a. 从亚马逊简单存储服务 (Amazon S3) 存储桶获取 IBM 数据服务器客户端。
   b. 安装和设置 IBM 数据服务器客户机。
   c. 从中检索 Db2 凭证。 AWS Secrets Manager
   d. 连接到 Db2 服务器。
   e. 预编译、编译和绑定 COBOL 程序。
   f. 将成品保存在 S3 存储桶中 AWS CodeDeploy 以备使用。
5. CodePipeline 开始 CodeDeploy。
6. CodeDeploy 协调其代理，这些代理已经安装在运行时环境中。代理从 Amazon S3 获取应用程序并根据中的说明进行安装`appspec.yml`。

为了使事情变得简单并专注于构建，此模式中的说明涵盖了步骤 1 到 4，但不包括 COBOL Db2 程序的部署。

自动化和扩缩

为简单起见，此模式描述了如何手动配置资源。但是，有许多自动化选项可用，例如 AWS CloudFormation AWS Cloud Development Kit (AWS CDK)、和 HashiCorp Terraform，它们可以自动执行这些任务。有关更多信息，请参阅AWS CloudFormation和AWS CDK文档。

工具

AWS 服务

- AWS CodeBuild是一项完全托管的生成服务，可帮助您编译源代码、运行单元测试和生成可随时部署的工件。
- AWS CodeDeploy自动部署到亚马逊 EC2 或本地实例、 AWS Lambda 函数或亚马逊弹性容器服务 (Amazon ECS) Container Service 服务。
- AWS CodePipeline帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件更改所需的步骤。
- AWS Mainframe Modernization提供工具和资源，帮助您规划和实施从大型机到 AWS 托管运行时环境的迁移和现代化。

其他工具

- 适用于 AWS Mainframe Modernization 平台重构工具的 Amazon ECR 图片。要编译 COBOL 应用程序，您需要使用包含重新平台工具的亚马逊弹性容器注册表 (Amazon ECR) Container Registr CodeBuild y 镜像启动： AWS Mainframe Modernization

  ```
  673918848628.dkr.ecr.<your-region>.amazonaws.com/m2-enterprise-build-tools:9.0.7.R1
  ```

  有关可用的 ECR 映像的更多信息，请参阅《AWS Mainframe Modernization 用户指南》中的教程。
- IBM 数据服务器客户机软件对于预编译和绑定 COBOL Db2 程序至关重要。 CodeBuild它充当 COBOL 编译器和 Db2 之间的桥梁。

最佳实践

- 并非每个 COBOL 程序都依赖 Db2 作为其数据持久层。确保用于访问 Db2 的编译指令仅应用于专门设计用于与 Db2 交互的 COBOL 程序。实现一种逻辑来区分 COBOL Db2 程序和不使用 Db2 的 COBOL 程序。
- 我们建议您避免编译尚未修改的程序。实施一个流程来确定哪些程序需要编译。

操作说明

创建云基础架构

Task	描述	所需技能
创建 S3 存储桶来托管 IBM 数据服务器客户端和管道项目。	您需要设置 S3 存储桶，以 (a) 上传 IBM 数据服务器客户端，(b) 存储存储库中的代码，以及 (c) 存储构建过程的结果。 1. 登录并打开 Amazon S3 控制台。 AWS Management Console 2. 选择现有的 S3 存储桶或创建新的存储桶。记下存储桶	常规 AWS

Task	描述	所需技能
	的亚马逊资源名称 (ARN)，以备将来使用。 有关创建 S3 存储桶的方法，请参阅 Amazon S3 文档。	
将 IBM 数据服务器客户端上传到 S3 存储桶。	1. 在 Amazon S3 控制台上，选择要将其打开的存储桶。 2. 选择 “创建文件夹”，将其名称指定为 “客户端”，然后选择 “创建文件夹”。 3. 打开客户端文件夹，选择上传，添加文件。 4. 选择您之前从 IBM 网站下载到本地文件系统的 IBM 数据服务器客户机文件。 文件名应类似于`v11.5.8_linuxx64_client.tar.gz` 或`v11.5.9_linuxx64_client.tar.gz` 。 5. 选择 “打开”、“上传”，然后等待上传完成。 6. 在 “文件和文件夹” 选项卡上，选择数据服务器客户端，并记下其 S3 URI。	常规 AWS

Task	描述	所需技能
为您的 Db2 凭证创建一个 AWS Secrets Manager 密钥。	要创建用于安全存储您的 DB2 凭据的密钥，请执行以下操作： 1. 在 S ecrets Manager 控制台上，选择存储新密钥。 2. 在“选择密钥类型”窗格中，选择“另一种密钥类型”和“纯文本”。 3. 在纯文本框中，使用以下 JSON 结构键入您的 Db2 凭证。 `{ "username": "<your-db2-user-name>", "password": "<your-db2-password>", "db2node": "db2dev", "db2host": "<your-db2-hostname-or-IP>", "db2port": <your-db2-port>, "db2name": "<your-db2-location>", "qualifier": "<your-db2-qualifier>" }` 4. 选择“下一步”，然后为密钥命名，例如`dev-db2-cred` 。	常规 AWS

Task	描述	所需技能
	5. 选择 “下一步”、“下一步” 和 “存储”。 有关创建密钥的更多信息，请参阅 Secr ets Manager 文档。	
验证是否可以从 VPC 子网访问 Db2。	AWS CodeBuild 需要连接到 Db2 服务器，以便数据服务器客户端可以执行预编译和绑定操作。确保它 CodeBuild 可以通过安全连接到 Db2 服务器。 1. 打开 Amazon VPC 控制台。 2. 在导航窗格上，选择 Subnet s，然后记下私有子网IPv4 CIDRs CodeBuild 的 IDs 和。 3. 通过引入入站规则来更新 Db2 系统的当前网络访问控制设置。此规则应允许仅从与您的 CodeBuild 项目关联的子网 CIDRs 中对 Db2 端口进行自定义 TCP 访问。	网络管理员，常规 AWS

创建应用程序构件

Task	描述	所需技能
创建 COBOL Db2 资产。	1. 如果要使用简单的 COBOL Db2 示例，请将以下源代码另存为。`CDB2SMP.cbl` 或者，你可以用你已经拥有的程序替换此示例。	应用程序开发人员

Task	描述	所需技能

	<pre>IDENTIFICATION
	DIVISION.
	PROGRAM-ID.
	CDB2SMP.
	DATA
	DIVISION.
	WORKING-S
	TORAGE SECTION.
	01 WS-NAME
	PIC
	X(100).
	PROCEDURE
	DIVISION.
	EXEC SQL
	SELECT
	NAME INTO :WS-NAME
	FROM
	SYSIBM.SYSTABLES
	END-EXEC
	GOBACK.</pre> 2. 提交更改并将文件推送到您的存储库。	

创建 buildspec.yml 文件。	1. 根据 “其他信息” 部分中提供的示例生成buildspec.yml 文件。 2. 提交更改并将文件推送到您的存储库。	AWS DevOps

Task	描述	所需技能
将您的存储库连接到 CodePipeline。	1. 打开AWS 开发人员工具控制台。 2. 在导航窗格中，选择 “设置”、“连接”。 3. 按照您选择的源提供商的开发者工具控制台文档中的说明进行操作。 在后续步骤中创建 (IAM) 策略时，您将需要连接的 Amazon 资源名称 AWS Identity and Access Management (ARN)。CodePipeline	AWS DevOps

配置 权限

Task	描述	所需技能
为创建 IAM 策略 CodeBuild。	该 CodeBuild 项目需要访问某些资源，包括 Secrets Manager 和 Amazon S3。 设置必要权限： 1. 打开 IAM 管理控制台。 2. 在导航窗格中，选择策略，创建策略，然后选择 CodeBuild 服务。 3. 将格式从 Visual 切换为 JSON，然后将 “其他信息” 部分中提供的CodeBuild 策略复制到策略编辑器字段。	常规 AWS

Task	描述	所需技能
	4. 命名并保存此政策，以备将来在下一步中参考。 有关创建 IAM 策略的更多信息，请参阅 IAM 文档。	
为创建 IAM 角色 CodeBuild。	要使安全策略可用 CodeBuild，您需要配置 IAM 角色。 要创建此角色，请执行以下操作： 1. 在 IAM 控制台的导航窗格中，选择角色，创建角色。 3. 对于 “可信实体” 类型，请保留默认AWS 服务设置。 4. 在 “用例” 中，选择 CodeBuild 服务，然后选择 “下一步”。 4. 在可用 IAM 策略列表中，找到您为其创建的策略 CodeBuild，然后选择下一步将其附加到该角色。 5. 为角色指定名称，然后选择 “创建角色” 将其保存以备将来参考 CodeBuild。 有关为创建 IAM 角色的更多信息 AWS 服务，请参阅 IAM 文档。	常规 AWS

Task	描述	所需技能
为创建 IAM 策略 CodePipeline。	该 AWS CodePipeline 管道需要访问某些资源，包括您的代码存储库和 Amazon S3。 重复之前提供的步骤，CodeBuild 为创建 IAM 策略 CodePipeline（在步骤 2 中，选择CodePipeline而不是CodeBuild）。	AWS DevOps

Task	描述	所需技能

为创建 IAM 角色 CodePipeline。	要使安全策略可用 CodePipeline，您需要配置 IAM 角色。
	
	要创建此角色，请执行以下操作：
	
	1. 在 IAM 控制台上，选择角色，创建角色。
	2. 对于 Trusted entity type（可信实体类型），选择 Custom trust policy（自定义信任策略）。
	
	将显示一个带有空`Principal`元素的策略。
	3. 在这`Principal`行的大括号之间，添加：
	
	`"Service": "codepipeline.amazonaws.com"`
	
	信任策略将如下所示：
	
	`{ "Version": "2012-10-17", "Statement": [ { "Sid": "Statement1", "Effect": "Allow", "Principal": { "Service": "codepipeline.amazonaws.com" },`
		AWS DevOps

Task	描述	所需技能

	<pre>"Action":
	"sts:AssumeRole"
	}
	]
	}</pre>
	4. 选择下一步。
	5. 在可用 IAM 策略列表中，找到您为其创建的策略 CodePipeline，然后选择下一步将其附加到该角色。
	6. 为角色指定名称，然后选择“创建角色” 将其保存以备将来参考 CodePipeline。	

编译并绑定 COBOL Db2 程序

Task	描述	所需技能
创建 CodePipeline 管道和 CodeBuild 项目。	要创建 CodePipeline 管道以及编译和绑定 COBOL Db2 程序的 CodeBuild 项目，请执行以下操作： 1. 打开CodePipeline 控制台，然后选择 “创建管道”、“生成自定义管道”。 2. 为管道指定名称。 3. 对于服务角色，选择现有服务角色，然后选择为您创建的 IAM 角色指定 ARN。CodePipeline 4. 展开高级设置，选择自定义位置，选择您之前创建的 S3 存储桶，然后选择下一步。	AWS DevOps

Task	描述	所需技能
	5. 对于来源提供商，请选择您的第三方来源提供商，并为该提供商提供相关信息：。 a. 在 “连接” 中，选择为源提供商创建的连接。 b. 在存储库名称中，选择您的存储库。 c. 对于默认分支，请选择存储 COBOL 程序的分支和。`buildspec.yml` d. 选择下一步。 6. 对于生成提供商，请选择其他生成提供商AWS CodeBuild。 7. 在 “项目名称” 中，选择 “创建项目”。 控制台会显示一个 CodeBuild 窗口，您可以在其中创建构建项目。在此窗口中： a. 输入项目的名称。 b. 对于 Environment image (环境映像)，选择 Custom image (自定义映像)。 c. 对于环境类型，请选择 Linux 容器。 d. 对于 ECR 账户，请选择其他 ECR 账户。 e. 对于 Amazon ECR 存储库 URI，请输入：673918848	

Task	描述	所需技能
	628.dkr.ecr.<your-region>.amazonaws.com/m2-enterprise-build-tools:8.0.9.R1 。 f. 对于服务角色，选择现有服务角色并选择您为其创建的角色 CodeBuild。 g. 展开其他配置部分，然后选择此项目的 VPC、私有子网和安全组。 h. 在构建规范部分中，选择使用构建规范文件。 i. 在窗口末尾，选择 “继续” CodePipeline。CodeBuild 窗口将关闭，因此您可以返回 CodePipeline 控制台。 8. 返回 CodePipeline 控制台，选择 “下一步”。 9. 在 “添加部署阶段” 窗格中，选择 “跳过部署阶段”，然后确认。 10查看管道参数，然后选择创建管道。	
检查输出。	通过查看构建日志来验证 CodePipeline 构建是否成功。	AWS DevOps

Task	描述	所需技能

在 Db2 中查看结果。	验证 SYSPLAN 表中的软件包版本。
	
	`select CAST(NAME AS VARCHAR(10)) as name, VALIDATE, LAST_BIND_TIME, LASTUSED, CAST(PKGVERSION AS VARCHAR(10)) as PKGVERSION from SYSIBM.SYSPLAN where NAME = 'CDB2SMP' order by LAST_BIND_TIME desc`
	
	版本必须与 CodeBuild 构建 ID 相匹配，在我们的示例CDB2SMP中：
	
	`NAME VALIDATE LAST_BIND_TIME LASTUSED PKGVERSION ---------- -------- ---------- -------- CDB2SMP B 2024-05-18-11.53.11.503738 01/01/0001 19`	

故障排除

事务	解决方案
有时，当您在服务之间切换时， AWS 控制台会切换区域。	AWS 区域 每当您在服务之间切换时，请务必验证所选内容。

事务	解决方案
	AWS 区域 选择器位于控制台窗口的右上角。
可能很难从 CodeBuild中识别 Db2 连接问题。	要解决连接问题，请在`buildspec.yml` 文件中添加以下 c DB2 onnect 命令。此新增功能可帮助您调试和解决连接问题。 `db2 connect to $DB_NAME user $DB2USER using $DB2PASS`
有时，IAM 控制台中的角色窗格不会立即显示您创建的 IAM 策略。	如果遇到延迟，请刷新屏幕以显示最新信息。

相关资源

IBM 文档

- IBM 数据服务器客户机和驱动程序类型
- 下载 IBM 数据服务器客户机和驱动程序类型

AWS 文档

- Amazon S3 用户指南
- AWS CodeBuild 用户指南
- AWS Mainframe Modernization 用户指南
- AWS Secrets Manager 用户指南
- AWS CodePipeline 用户指南
- AWS CodeDeploy 用户指南

其他信息

CodeBuild 策略

用您的值替换占位符`<RegionID>`、`<AccountID>`、`<SubnetARN>`、`<BucketARN>`、和`<DB2CredSecretARN>`。

```
{"Version": "2012-10-17",
    "Statement": [
        {"Action": "ecr:GetAuthorizationToken", "Effect": "Allow", "Resource": "*" },
        {"Action": ["ecr:GetDownloadUrlForLayer", "ecr:BatchGetImage",
                    "ecr:BatchCheckLayerAvailability"],
         "Effect": "Allow",
         "Resource": "arn:aws:ecr:*:673918848628:repository/m2-enterprise-build-
tools"},
        {"Action": "s3:PutObject", "Effect": "Allow", "Resource": "arn:aws:s3:::aws-m2-
repo-*/*"},
        {"Action": ["logs:PutLogEvents", "logs:CreateLogStream",
 "logs:CreateLogGroup"],
         "Effect": "Allow", "Resource": "arn:aws:logs:<RegionId>:<AccountId>:*"},
        {"Action": ["ec2:DescribeVpcs", "ec2:DescribeSubnets",
                    "ec2:DescribeSecurityGroups", "ec2:DescribeNetworkInterfaces",
                    "ec2:DescribeDhcpOptions", "ec2:DeleteNetworkInterface",
                    "ec2:CreateNetworkInterface"],
         "Effect": "Allow", "Resource": "*"},
        {"Action": "ec2:CreateNetworkInterfacePermission",
         "Effect": "Allow", "Resource": ["<SubnetARN>"]},
        {"Action": "s3:*", "Effect": "Allow", "Resource": ["<BucketARN>/
*","<BucketARN>"]},
        {"Action": "secretsmanager:GetSecretValue",
         "Effect": "Allow", "Resource": "<DB2CredSecretARN>"}
    ]
}
```

CodePipeline 策略

用您的值替换占位<ConnectionARN>符<BucketARN>和。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {"Action": ["s3:List*", "s3:GetObjectVersion", "s3:GetObject",
 "s3:GetBucketVersioning" ],
        "Effect": "Allow",
        "Resource": ["<BucketARN>/*", "<BucketARN>"]},
        {"Action": ["codebuild:StartBuild", "codebuild:BatchGetBuilds"],
         "Effect": "Allow", "Resource": "*"},
        {"Action": ["codestar-connections:UseConnection"],
         "Effect": "Allow", "Resource": "<ConnectionARN>"}
        ]
```

```
}
```

buildspec.yml

将<your-bucket-name>占位符替换为您实际的 S3 存储桶名称。

```
version: 0.2
phases:
  pre_build:
    commands:
      - /var/microfocuslicensing/bin/mfcesd -no > /var/microfocuslicensing/logs/
mfcesd_startup.log 2>&1 &
      - |
        mkdir $CODEBUILD_SRC_DIR/db2client
        aws s3 cp s3://<your-bucket-name>/v11.5.8_linuxx64_client.tar.gz
 $CODEBUILD_SRC_DIR/db2client/ >> /dev/null 2>&1
        tar -xf $CODEBUILD_SRC_DIR/db2client/v11.5.8_linuxx64_client.tar.gz -C
 $CODEBUILD_SRC_DIR/db2client/
        cd $CODEBUILD_SRC_DIR/db2client/
        ./client/db2_install -f sysreq -y -b /opt/ibm/db2/V11.5 >> /dev/null 2>&1

        useradd db2cli
        /opt/ibm/db2/V11.5/instance/db2icrt -s client -u db2cli db2cli
        DB2CRED=$(aws secretsmanager get-secret-value --secret-id dev-db2-cred | jq -r
 '.SecretString | fromjson')
        read -r DB2USER DB2PASS DB_NODE DB_HOST DB_PORT DB_NAME DB_QUAL <<<$(echo
 $DB2CRED | jq -r
 '.username, .password, .db2node, .db2host, .db2port, .db2name, .qualifier')
        . /home/db2cli/sqllib/db2profile
        db2 catalog tcpip node $DB_NODE remote $DB_HOST server $DB_PORT
        db2 catalog db $DB_NAME as $DB_NAME at node $DB_NODE authentication server
  build:
    commands:
      - |
        revision=$CODEBUILD_SRC_DIR/loadlib
        mkdir -p $revision; cd $revision
        . /opt/microfocus/EnterpriseDeveloper/bin/cobsetenv
        cob -zU $CODEBUILD_SRC_DIR/CDB2SMP.cbl -C "DB2(DB==${DB_NAME} PASS==${DB2USER}.
${DB2PASS} VERSION==${CODEBUILD_BUILD_NUMBER} COLLECTION==DB2AWSDB"
artifacts:
  files:
    - "**/*"
  base-directory: $revision
```

使用 Amazon A EC2 uto Scaling 和 Systems Manager 构建 Micro Focus 企业服务器 PAC

由 Kevin Yung（AWS）、彼得·伍兹、亚伯拉罕·朗登（Micro Focus）和 Krithika Palani Selvam（AWS）创作

摘要

这种模式为大型机应用程序引入了可扩展架构，在横向扩展性能和可用性集群 (PAC) 中使用 Micro Focus 企业服务器，在亚马逊网络服务 () 上使用亚马逊弹性计算云 (Amazon) A EC2 uto Scaling 组 ()。AWS该解决方案通过 Amazon A EC2 uto Scaling 生命周期挂钩实现了全自动化。 AWS Systems Manager 通过使用这种模式，您可将大型机设置为在线和批处理应用程序，从而根据容量需求自动横向缩减和横向扩展，从而实现高弹性。

> **Note**
>
> 此模式已在 Micro Focus 企业服务器版本 6.0 中进行了测试。有关版本 8，请参阅设置 Micro Focus 运行时（在亚马逊上 EC2）。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- Micro Focus Enterprise Server 软件和许可证。有关详细信息，请联系 Micro Focus 销售。
- 了解重建和交付大型机应用程序以在 Micro Focus Enterprise Server 中运行。有关高级概述，请参阅 Micro Focus Enterprise Server 数据表。
- 了解 Micro Focus Enterprise Server 横向扩展性能与可用性集群中的概念。有关更多信息，请参阅 Micro Focus Enterprise Server 文档。
- 了解具有持续集成 (CI) 功能的大型机应用程序 DevOps 的总体概念。有关由 AWS and Micro Focus 开发的 AWS 规范性指导模式，请参阅大型机现代化：开 DevOps 启 Micro Foc AWS u s。

> **Note**
>
> 此模式已在 Micro Focus 企业服务器版本 6 中进行了测试。有关版本 8，请参阅设置 Micro Focus 运行时（在亚马逊上 EC2）。

限制

- 有关 Micro Focus Enterprise Server 支持的平台列表，请参阅 Micro Focus Enterprise Server 数据表。
- 此模式中使用的脚本和测试基于亚马逊 EC2 Windows Server 2019；其他 Windows Server 版本和操作系统未针对此模式进行测试。
- 该模式基于 Micro Focus Enterprise Server 6.0 for Windows；在此模式的开发中没有测试早期版本或更高版本。

产品版本

- Micro Focus Enterprise Server 6.0
- Windows Server 2019

架构

在传统大型机环境中，必须预调配硬件来托管应用程序和公司数据。为了满足和满足季节性、每月、每季度，甚至是前所未有的或意想不到的需求，大型机用户必须通过购买额外的存储和计算容量来横向扩展。增加存储和计算容量资源的数量可提高整体性能，但扩展不是线性的。

当您开始使用 Amazon A EC2 uto Scaling 和 Micro Focus 企业服务器在 AWS 上采用按需消费模式时，情况并非如此。以下各节详细介绍了如何使用 Micro Focus 企业服务器横向扩展性能和可用性集群 (PAC) 和 Amazon Auto Scaling 组构建全 EC2 自动、可扩展的大型机应用程序架构。

Micro Focus Enterprise Server 动缩放架构

首先，了解 Micro Focus Enterprise Server 的基本概念很重要。该环境为传统上在 IBM 大型机上运行的应用程序提供了一个与大型机兼容的 x86 部署环境。它提供在线和批处理运行以及支持以下功能的事务环境：

- IBM COBOL
- IBM PL/I

- IBM JCL 批处理作业
- IBM CICS 和 IMS TM 事务
- Web 服务
- 常用批处理实用程序，包括 SORT

Micro Focus Enterprise Server 使大型机应用程序能够以最少的更改运行。现有的大型机工作负载可以转移到 x86 平台并进行现代化改造，以利用 Amazon Web Services Cloud 原生扩展快速扩展至新的市场或地区。

AWS Prescriptive Guidence 模式大型机现代化：在 AWS DevOps 上使用 Micro Focus 引入了该架构，使用带有 AWS 和 AWS 的 Micro Focus 企业开发人员和企业测试服务器在 AWS CodePipeline 上加速大型机应用程序的开发和测试。 CodeBuild此模式侧重于将大型机应用程序部署到 AWS 生产环境以实现高可用性和弹性。

在大型机生产环境中，您可能已在大型机中设置 IBM Parallel Sysplex 以实现高性能和高可用性。为了创建类似于 Sysplex 的横向扩展架构，Micro Focus 在企业服务器中引入了性能和可用性群集 (PAC)。PACs 支持将大型机应用程序部署到多个企业服务器区域，作为单个映像进行管理，并在 Amazon EC2 实例中横向扩展。 PACs 还可根据需要支持可预测的应用程序性能和系统吞吐量。

在 PAC 中，多个 Enterprise Server 实例作为一个逻辑实体协同工作。因此，一个企业服务器实例的故障不会中断业务连续性，因为容量与其他地区共享，而新实例则使用行业标准功能（例如 Amazon A EC2 uto Scaling 组）自动启动。这消除了单点故障，提高了对硬件、网络以及应用程序问题的恢复能力。使用企业服务器通用网络管理 (ESCAD) 可以操作和管理扩展的企业服务器实例 APIs，从而简化了企业服务器的操作维护和可维护性。

> **Note**
>
> Micro Focus 建议性能和可用性群集 (PAC) 应至少包含三个企业服务器区域，这样在企业服务器区域出现故障或需要维护时可用性不会受到影响。

PAC 配置需要支持的关系数据库管理服务 (RDBMS) 管理区域数据库、跨区域数据库和可选的数据存储数据库。应使用 Micro Focus 数据库文件处理器支持使用数据存储数据库来管理虚拟存储访问方法 (VSAM) 文件，以提高可用性以及可扩展性。支持的 RDBMSs 内容包括：

- Microsoft SQL Server 2009 R2 及更高版本
- PostgreSQL 10.x，包括 Amazon Aurora PostgreSQL-Compatible Edition

- DB2 10.4 及更高版本

有关支持的 RDBMS 和 PAC 要求的详细信息，请参阅 Micro Focus Enterprise Server - 先决条件和 Micro Focus Enterprise Server - 建议的 PAC 配置。

下图介绍了 Micro Focus PAC 的典型 AWS 架构设置。

	组件	描述
1	Enterprise Server 实例自动扩缩组	在 PAC 中设置使用 Enterprise Server 实例部署的自动扩缩组。实例数量可以按比例扩展，也可以通过 Amazon CloudWatch 警报使用 CloudWatch 指标启动。
2	Enterprise Server ESCAW 实例自动扩缩组	设置使用 Enterprise Server Common Web Administration (ESCWA) (ESCAW) 部署的自动扩缩组。西亚经社会提供群组管理 APIs。 在 Enterprise Server 实例自动扩缩事件期间，ESCWA服务器充当控制面板，用于添加或删除 Enterprise Server，并在 PAC 中启动或停止 PAC 中的 Enterprise Server 区域。 由于 ESCWA 实例仅用于 PAC 管理，因此其流量模式是可预测的，并且其自动扩缩所需容量要求可设置为 1。
3	多可用区设置中的 Amazon Aurora 实例	设置关系数据库管理系统 (RDBMS)，以托管要在 Enterprise Server 实例之间共享的用户和系统数据文件。

4	亚马逊 ElastiCache (Redis OSS) 实例和副本	设置一个 ElastiCache （Redis OSS）主实例和至少一个副本来托管用户数据并充当企业服务器实例的扩展存储库 (SOR)。您可配置一个或多个横向扩展存储库来存储特定类型的用户数据。 Enterprise Server 使用 Redis NoSQL 数据库作为 SOR，这是维护 PAC 完整性的必要条件。
5	网络负载均衡器	设置负载均衡器，为应用程序提供主机名以连接到 Enterprise Server 实例提供的服务（例如，通过 3270 仿真器访问应用程序）。

这些组件构成了 Micro Focus Enterprise Server PAC 集群的最低要求。下一部分介绍集群管理自动化。

使用 AWS Systems Manager Automation 扩展

在 AWS 上部署 PAC 集群后，PAC 将通过企业服务器通用网络管理机构 (ESCAW) 进行管理 APIs。

要在自动扩展事件期间自动执行集群管理任务，您可以使用 Systems Manager Automation 运行手册和带有 Amazon Auto Scaling 的亚马逊 A EC2 uto Scaling。 EventBridge这些自动化架构如下图所示。

	组件	描述
1	自动扩缩生命周期挂钩	设置自动扩展生命周期挂钩，并在自动扩展组中启动新实例和终止现有实例 EventBridge 时向 Amazon 发送通知。
2	亚马逊 EventBridge	设置 Amazon EventBridge 规则，将自动扩展事件路由到

		Systems Manager 自动化运行手册目标。
3	自动化运行手册	设置 Systems Manager Automation 运行手册以运行 Windows PowerShell 脚本并调用 ESCA APIs W 来管理 PAC。有关示例，请参阅其他信息部分。
4	自动扩缩组中的 Enterprise Server ESCAP 实例	在自动扩缩组中设置 Enterpris e Server ESCAM 实例。西亚经社会的例子是 APIs 用来管理 PAC。

工具

- Micro Focus Enterprise Server – Micro Focus Enterprise Server 为使用企业开发人员的任何集成式开发环境（IDE）变体创建的应用程序提供运行环境。
- Amazon A EC2 ut EC2 o Scaling — Amazon Auto Scaling 可帮助您确保有正确数量的亚马逊 EC2 实例可用来处理应用程序的负载。您可以创建名为 Auto Scaling 组的 EC2 实例集合，并指定最小和最大实例数。
- 亚马逊 ElastiCache （Redis OSS）— Amazon ElastiCache 是一项网络服务，用于在云中设置、管理和扩展分布式内存数据存储或缓存环境。它可以提供高性能、可扩展且具有成本效益的缓存解决方案。
- Amazon RDS – Amazon Relational Database Service(Amazon RDS) 是一项 Web 服务，让用户能够在 Amazon Web Services Cloud 中轻松设置、操作和扩展关系数据库。它为行业标准的关系数据库提供了经济高效、可调整大小的容量，并管理常见的数据库管理任务。
- AWS Systems Manager – AWS Systems Manager 是一项 Amazon Web Services，您可用它在 AWS 上查看和控制您的基础设施。通过使用 Systems Manager 控制台，您可以查看来自多个 Amazon Web Services 的操作数据并在 AWS 资源之间自动执行操作任务。Systems Manager 通过扫描托管实例并报告其检测到的任何策略违规行为（或采取纠正措施）来帮助您维护安全性和合规性。

操作说明

创建 Amazon Aurora 实例

Task	描述	所需技能
为亚马逊 Aurora 实例创建 AWS CloudFormation 模板。	使用 AWS 示例代码片段制作一个用于创建兼容 Amazon Aurora PostgreSQL 的版本实例的 CloudFormation 模板。	云架构师
部署 CloudFormation 堆栈以创建 Amazon Aurora 实例。	使用该 CloudFormation 模板创建兼容 Aurora PostgreSQL 的实例，该实例已为生产工作负载启用多可用区复制。	云架构师
为 Enterprise Server 配置数据库连接设置。	按照 Micro Focus 文档中的说明为 Micro Focus Enterprise Server 准备连接字符串和数据库配置。	数据工程师、 DevOps 工程师

为 Redis 实例创建一个 Amazon ElastiCache 集群

Task	描述	所需技能
为 Redis 实例的 Amazon ElastiCache 集群创建 CloudFormation 模板。	使用 AWS 示例代码片段制作一个用于为 Redis 实例创建 Amazon ElastiCache 集群的 CloudFormation 模板。	云架构师
部署 CloudFormation 堆栈以为 Redis 实例创建亚马逊 ElastiCache 集群。	为已为生产工作负载启用多可用区复制的 Redis 实例创建 Amazon ElastiCache 集群。	云架构师
配置 Enterprise Server PSOR 连接设置。	按照 Micro Focus 文档中的说明为 Micro Focus Enterprise	DevOps 工程师

Task	描述	所需技能
	Server PAC 准备 PAC 横向扩展存储库 (PSOR) 连接配置。	

创建 Micro Focus Enterprise Server SECA 自动扩缩组

Task	描述	所需技能
创建 Micro Focus Enterprise Server AMI。	创建 Amazon EC2 Windows Server 实例并在该 EC2 实例中安装 Micro Focus 企业服务器二进制文件。创建 EC2 实例的亚马逊系统映像 (AMI)。有关更多信息，请参阅 Enterprise Server 安装文档。	云架构师
为企业服务器西亚经社会创建 CloudFormation 模板。	使用 AWS 示例代码片段制作一个模板，用于在自动扩缩组中创建 Enterprise Server ESCAM 的自定义堆栈。	云架构师
部署 CloudFormation 堆栈以为企业服务器 ESCAW 创建 Amazon EC2 扩展组。	使用 CloudFormation 模板部署自动扩展组，使用上一篇文章中创建的 Micro Focus Enterprise Server Server Server Server Server	云架构师

创建 AWS Systems Manager Automation 运行手册

Task	描述	所需技能
为 Syst CloudFormation ems Manager 自动化运行手册创建模板。	使用 “其他信息” 部分中的示例代码片段制作一个 CloudFormation 模板，该模板将创建 Systems Manager Automatio	云架构师

Task	描述	所需技能
	n 运行手册，用于自动创建 PAC、Enterprise Server 向内扩展和企业服务器横向扩展。	
部署包含 Systems Manager 自动化运行手册的 CloudFormation 堆栈。	使用该 CloudFormation 模板部署包含自动化运行手册的堆栈，用于创建 PAC、企业服务器扩展和企业服务器横向扩展。	云架构师

为 Micro Focus Enterprise Server 创建自动扩缩组

Task	描述	所需技能
创建用于为 Micro Focus 企业服务器设置自动伸缩组的 CloudFormation 模板。	使用 A WS 示例代码片段制作用于创建自动扩展组的 CloudFormation 模板。此模板将重复使用为 Micro Focus Enterprise Server ESCWA 实例创建的 AMI。 然后使用 A WS 示例代码片段创建自动扩展生命周期事件，并将 Amazon 设置为在同一个模板中筛选横 EventBridge 向扩展和缩小事件。 CloudFormation	云架构师
为 Micro Focus 企业服务器的自动扩展组部署 CloudFormation 堆栈。	部署包含 Micro Focus 企业服务器自动扩展组的 CloudFormation 堆栈。	云架构师

相关资源

- Micro Focus Enterprise Server 性能和可用性集群 (PAC)

- Amazon A EC2 uto Scaling 生命周期挂钩
- 使用触发器运行自动化 EventBridge

其他信息

必须自动执行以下场景才可横向缩减或横向扩展 PAC 集群。

用于启动或重新创建 PAC 自动化

在 PAC 群集启动时，企业服务器要求西亚经社会调用 APIs 以创建 PAC 配置。这将启动并将 Enterprise Server 区域添加至 PAC 中。要创建或重新创建 PAC，请使用以下步骤进行操作：

1. 在 ESCWA 中使用给定名称配置 PAC 横向扩展存储库 (PSOR)。

```
POST /server/v1/config/groups/sors
```

2. 创建具有给定名称的 PAC，并在其上随附 PSOR。

```
POST /server/v1/config/groups/pacs
```

3. 如果这是您第一次设置 PAC，请配置区域数据库与跨区域数据库。

> **Note**
>
> 此步骤使用 SQL 查询和 Micro Focus Enterprise Suit e 命令行 dbhfhadmin 工具来创建数据库并导入初始数据。

4. 将 PAC 定义安装到 Enterprise Server 区域。

```
POST /server/v1/config/mfds
POST /native/v1/config/groups/pacs/${pac_uid}/install
```

5. 在 PAC 中启动 Enterprise Server 区域。

```
POST /native/v1/regions/${host_ip}/${port}/${region_name}/start
```

前面的步骤可以通过使用 Windows PowerShell 脚本来实现。

以下步骤说明了如何通过重复使用 Windows PowerShell 脚本来构建用于创建 PAC 的自动化。

1. 创建亚马逊 EC2 启动模板，在引导过程中下载或创建 Windows PowerShell 脚本。例如，您可以使用 EC2 用户数据从亚马逊简单存储服务 (Amazon S3) 存储桶下载脚本。
2. 创建 AWS Systems Manager Automation 运行手册来调用 Windows PowerShell 脚本。
3. 使用实例标签将运行手册与 ESCWA 实例关联。
4. 通过使用启动模板创建 ESCWA 自动扩缩组。

您可以使用以下 AWS CloudFormation 代码段示例，创建自动化运行手册。

用于创建 CloudFormation PAC 的 Systems Manager 自动化运行手册的示例片段

```
PACInitDocument:
   Type: AWS::SSM::Document
   Properties:
     DocumentType: Command
     Content:
       schemaVersion: '2.2'
       description: Operation Runbook to create Enterprise Server PAC
       mainSteps:
       - action: aws:runPowerShellScript
         name: CreatePAC
         inputs:
           onFailure: Abort
           timeoutSeconds: "1200"
           runCommand:
           - |
             C:\Scripts\PAC-Init.ps1
PacInitAutomation:
   Type: AWS::SSM::Document
   Properties:
     DocumentType: Automation
     Content:
       description: Prepare Micro Focus PAC Cluster via ESCWA Server
       schemaVersion: '0.3'
       assumeRole: !GetAtt SsmAssumeRole.Arn
       mainSteps:
         - name: RunPACInitDocument
           action: aws:runCommand
           timeoutSeconds: 300
           onFailure: Abort
           inputs:
             DocumentName: !Ref PACInitDocument
```

```
              Targets:
                - Key: tag:Enterprise Server - ESCWA
                  Values:
                    - "true"
  PacInitDocumentAssociation:
    Type: AWS::SSM::Association
    Properties:
      DocumentVersion: "$LATEST"
      Name: !Ref PACInitDocument
      Targets:
        - Key: tag:Enterprise Server - ESCWA
          Values:
            - "true"
```

有关更多信息，请参阅 Micro Focus Enterprise Server - 配置 PAC。

使用新的 Enterprise Server 实例自动进行横向扩展

横向扩展 Enterprise Server 实例时，必须将其 Enterprise Server 区域添加到 PAC 中。以下步骤说明如何调用 ESCA APIs W 并将企业服务器区域添加到 PAC 中。

1. 将 PAC 定义安装到 Enterprise Server 区域。

```
POST '/server/v1/config/mfds'
POST /native/v1/config/groups/pacs/${pac_uid}/install
```

2. 在 PAC 中对区域执行热启动。

```
POST /native/v1/regions/${host_ip}/${port}/${region_name}/start
```

3. 通过将自动扩缩组与负载均衡器关联，将 Enterprise Server 实例添加至负载均衡器。

前面的步骤可以通过使用 Windows PowerShell 脚本来实现。有关更多信息，请参阅 Micro Focus Enterprise Server - 配置 PAC。

以下步骤可用于构建事件驱动的自动化，以便通过重复使用 Windows PowerShell 脚本将新启动的企业服务器实例添加到 PAC 中。

1. 为企业服务器实例创建 Amazon EC2 启动模板，以便在引导期间配置企业服务器区域。例如，您可使用 Micro Focus Enterprise Server 命令 mfds 导入区域配置。有关此命令的更多详细信息和可用选项，请参阅 Enterprise Server 参考。

2. 使用在上一步中创建的启动模板创建 Enterprise Server 自动扩缩组。
3. 创建 Systems Manager 自动化运行手册来调用 Windows PowerShell 脚本。
4. 使用实例标签将运行手册与 ESCWA 实例关联。
5. 创建 Amazon EventBridge 规则以筛选企业服务器自动扩展组的 “EC2 实例启动成功” 事件，并创建使用自动化运行手册的目标。

您可以使用以下示例 CloudFormation 片段来创建自动化运行手册和规则。 EventBridge

用于扩展企业服务器实例的 Systems Manager 示例 CloudFormation 片段

```
 ScaleOutDocument:
    Type: AWS::SSM::Document
    Properties:
      DocumentType: Command
      Content:
        schemaVersion: '2.2'
        description: Operation Runbook to Adding MFDS Server into an existing PAC
        parameters:
          MfdsPort:
            type: String
          InstanceIpAddress:
            type: String
            default: "Not-Available"
          InstanceId:
            type: String
            default: "Not-Available"
        mainSteps:
        - action: aws:runPowerShellScript
          name: Add_MFDS
          inputs:
            onFailure: Abort
            timeoutSeconds: "300"
            runCommand:
            - |
              $ip = "{{InstanceIpAddress}}"
              if ( ${ip} -eq "Not-Available" ) {
                $ip = aws ec2 describe-instances --instance-id {{InstanceId}} --output
text --query "Reservations[0].Instances[0].PrivateIpAddress"
              }
              C:\Scripts\Scale-Out.ps1 -host_ip ${ip} -port {{MfdsPort}}
```

```
  PacScaleOutAutomation:
    Type: AWS::SSM::Document
    Properties:
      DocumentType: Automation
      Content:
        parameters:
          MfdsPort:
            type: String
          InstanceIpAddress:
            type: String
            default: "Not-Available"
          InstanceId:
            type: String
            default: "Not-Available"
        description: Scale Out 1 New Server in Micro Focus PAC Cluster via ESCWA
Server
        schemaVersion: '0.3'
        assumeRole: !GetAtt SsmAssumeRole.Arn
        mainSteps:
          - name: RunScaleOutCommand
            action: aws:runCommand
            timeoutSeconds: 300
            onFailure: Abort
            inputs:
              DocumentName: !Ref ScaleOutDocument
              Parameters:
                InstanceIpAddress: "{{InstanceIpAddress}}"
                InstanceId: "{{InstanceId}}"
                MfdsPort: "{{MfdsPort}}"
              Targets:
                - Key: tag:Enterprise Server - ESCWA
                  Values:
                    - "true"
```

用于横向缩减 Enterprise Server 实例的自动化

与向外扩展类似，当企业服务器实例缩减时，会启动事件 EC2 实例终止生命周期操作，并且需要以下过程和 API 调用才能从 PAC 中删除 Micro Focus 企业服务器实例。

1. 在终止的 Enterprise Server 实例中停止该区域。

```
POST "/native/v1/regions/${host_ip}/${port}/${region_name}/stop"
```

2. 从 PAC 中删除 Enterprise Server 实例。

```
DELETE "/server/v1/config/mfds/${uid}"
```

3. 发送信号以继续终止 Enterprise Server 实例。

前面的步骤可以在 Windows PowerShell 脚本中实现。有关此过程的更多详细信息，请参阅 Micro Focus Enterprise Server 文档 – 管理 PAC。

以下步骤说明如何构建事件驱动的自动化，以便通过重复使用 Windows 脚本从 PAC 终止企业服务器实例。 PowerShell

1. 创建 Systems Manager 自动化运行手册来调用 Windows PowerShell 脚本。
2. 使用实例标签将运行手册与 ESCWA 实例关联。
3. 为 EC2 实例终止创建自动伸缩组生命周期挂钩。
4. 创建 Amazon EventBridge 规则以筛选企业服务器自动扩展组的 EC2 实例终止生命周期操作事件，并创建使用自动化运行手册的目标。

您可以使用以下示例 CloudFormation 模板来创建 Systems Manager Automation 运行手册、生命周期挂钩和 EventBridge 规则。

用于在企业 CloudFormation 服务器实例中进行扩展的 Systems Manager 自动化运行手册的示例片段

```
ScaleInDocument:
   Type: AWS::SSM::Document
   Properties:
     DocumentType: Command
     Content:
       schemaVersion: '2.2'
       description: Operation Runbook to Remove MFDS Server from PAC
       parameters:
         MfdsPort:
           type: String
         InstanceIpAddress:
           type: String
           default: "Not-Available"
         InstanceId:
           type: String
           default: "Not-Available"
       mainSteps:
```

```
        - action: aws:runPowerShellScript
          name: Remove_MFDS
          inputs:
            onFailure: Abort
            runCommand:
            - |
              $ip = "{{InstanceIpAddress}}"
              if ( ${ip} -eq "Not-Available" ) {
                $ip = aws ec2 describe-instances --instance-id {{InstanceId}} --output
text --query "Reservations[0].Instances[0].PrivateIpAddress"
              }
              C:\Scripts\Scale-In.ps1 -host_ip ${ip} -port {{MfdsPort}}

  PacScaleInAutomation:
    Type: AWS::SSM::Document
    Properties:
      DocumentType: Automation
      Content:
        parameters:
          MfdsPort:
            type: String
          InstanceIpAddress:
            type: String
            default: "Not-Available"
          InstanceId:
            type: String
            default: "Not-Available"
        description: Scale In 1 New Server in Micro Focus PAC Cluster via ESCWA Server
        schemaVersion: '0.3'
        assumeRole: !GetAtt SsmAssumeRole.Arn
        mainSteps:
          - name: RunScaleInCommand
            action: aws:runCommand
            timeoutSeconds: "600"
            onFailure: Abort
            inputs:
              DocumentName: !Ref ScaleInDocument
              Parameters:
                InstanceIpAddress: "{{InstanceIpAddress}}"
                MfdsPort: "{{MfdsPort}}"
                InstanceId: "{{InstanceId}}"
              Targets:
                - Key: tag:Enterprise Server - ESCWA
                  Values:
```

```
                - "true"
          - name: TerminateTheInstance
            action: aws:executeAwsApi
            inputs:
              Service: autoscaling
              Api: CompleteLifecycleAction
              AutoScalingGroupName: !Ref AutoScalingGroup
              InstanceId: "{{ InstanceId }}"
              LifecycleActionResult: CONTINUE
              LifecycleHookName: !Ref ScaleInLifeCycleHook
```

Amazon EC2 自动扩展触发器的自动化

为 Enterprise Server 实例设置扩展策略的过程需要了解应用程序的行为。多数情况下，您可设置目标跟踪扩展策略。例如，您可以使用平均 CPU 使用率作为自动扩展策略设置的 Amazon CloudWatch 指标。有关更多信息，请参阅 Amazon A EC2 uto Scaling 的目标跟踪扩展政策。对于具有常规流量模式的应用程序，请考虑使用预测扩展策略。有关更多信息，请参阅 Amazon A EC2 uto Scaling 的预测性扩展。

在 Amazon Web Services Cloud 中构建高级大型机文件查看器

由 Boopathy GOPALSAMY (AWS) 和 Jeremiah O'Connor (AWS) 编写

摘要

此模式提供代码示例和步骤，帮助您构建高级工具，以使用 AWS 无服务器服务浏览和查看大型机固定格式文件。该模式提供了如何将大型机输入文件转换为用于浏览和搜索的 Amazon S OpenSearch ervice 文档的示例。文件查看器工具可以帮助您实现以下目标：

- 保持相同的大型机文件结构和布局以保持您的 AWS 目标迁移环境中的一致性（例如，在将文件传输给外部方的批处理应用程序中，您可为文件保持相同的布局）
- 在大型机迁移过程中加快开发和测试速度
- 支持迁移后的维护活动

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 具有可由您的旧平台访问的子网的虚拟私有云（VPC）
-

> **Note**
>
> 输入文件及其相应的面向业务的通用语言 (COBOL) 字帖 (: 有关输入文件和 COBOL 字帖的示例，请参阅存储库中的。gfs-mainframe-solutions GitHub 有关 COBOL 字帖的更多信息，请参阅 IBM 网站上的《适用于 z/OS 6.3 的企业 COB OL 编程指南》。)

限制

- 副本解析限制为不超过两个嵌套级别（出现）

架构

源技术堆栈

- 以 FB（固定屏蔽）格式输入文件

- COBOL 副本布局

目标技术堆栈

- Amazon Athena
- 亚马逊 OpenSearch 服务
- Amazon Simple Storage Service（Amazon S3）
- AWS Lambda
- AWS Step Functions

目标架构

下图显示了解析大型机输入文件并将其转换为 OpenSearch 服务文档以供浏览和搜索的过程。

图表显示了以下工作流：

1. 管理员用户或应用程序将输入文件推送到一个 S3 存储桶，并将 COBOL copybook 推送到另一个 S3 存储桶。
2.

> **Note**
>
> 包含输入文件的 S3 存储桶调用 Lambda 函数，启动无服务器 Step Functions 工作流。：在此模式中使用 S3 事件触发器和 Lambda 函数来驱动 Step Functions 工作流程是可选的。此模式中的 GitHub 代码示例不包括对这些服务的使用，但您可以根据自己的要求使用这些服务。

3. Step Functions 工作流协调来自以下 Lambda 函数的所有批处理：
   - 该 `s3copybookparser.py` 函数解析副本布局并提取字段属性、数据类型和偏移量（输入数据处理的必要条件）。
   - `s3toathena.py` 函数创建 Athena 表格布局。Athena 解析 `s3toathena.py` 函数处理的输入数据，并将这些数据转换为 CSV 文件。
   - 该`s3toelasticsearch.py`函数从 S3 存储桶中提取结果文件并将该文件推送到 OpenSearch 服务。
4. 用户访问带有 S OpenSearch ervice 的 OpenSearch 仪表板以各种表和列格式检索数据，然后对索引数据运行查询。

工具

Amazon Web Services

- Amazon Athena 是一种交互式查询服务，使您可使用标准 SQL 直接分析 Amazon Simple Storage Service (Amazon S3) 中的数据。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。在这种模式中，您可以使用 Lambda 实现核心逻辑，例如解析文件、转换数据以及将数据加载到 S OpenSearch ervice 中以进行交互式文件访问。
- Amazon S OpenSearch er vice 是一项托管服务，可帮助您在 AWS 云中部署、运营和扩展 OpenSearch 服务集群。在这种模式中，您可以使用 S OpenSearch ervice 为转换后的文件编制索引，并为用户提供交互式搜索功能。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Step Functions 是一项无服务器编排服务，可帮助您搭配使用 Lambda 函数和其他 Amazon Web Services 来构建业务关键型应用程序。在这种模式中，您可使用 Step Functions 编排 Lambda 函数。

其他工具

- GitHub是一项代码托管服务，提供协作工具和版本控制。
- Python 是高级编程语言。

代码

此模式的代码可在 GitHub gfs-mainframe-patterns存储库中找到。

操作说明

准备目标环境

Task	描述	所需技能
创建 S3 存储桶。	创建 S3 存储桶，用于存储副本、输入文件和输出文件。我们建议您的 S3 存储桶采用以下文件夹结构： • `copybook/` • `input/` • `output/` • `query/` • `results/`	常规 AWS
创建 s3copybookparser 函数。	1. 创建名为的 Lambda 函数，`s3copybookparser` 然后从存储库上传源代码（`s3copybookparser.py` 和`copybook.py` ）。GitHub 2. 向 Lambda 函数附加 IAM policy `S3ReadOnly` 。	常规 AWS
创建 s3toathena 函数。	1. 创建名为的 Lambda 函数，`s3toathena` 然后从存储库上传源代码 (`s3toathena.py`)。GitHub将 Lambda 超时配置为大于 60 秒。 2. 若要提供对所需资源的访问权限，请对 Lambda 函数附加 IAM policy `AmazonAth`	常规 AWS

Task	描述	所需技能
	enaFullAccess 和 S3FullAccess 。	
创建 s3toelasticsearch 函数。	1. **Important** 在您的 Lambda 环境中添加 Python 依赖项。：要使用该s3toelasticsearch 函数，必须添加 Python 依赖项，因为 Lambda 函数使用 Python Elasticsearch 客户端依赖项（和）。Elasticsearch==7.9.0 requests_aws4auth 2. 创建名为的 Lambda 函数，s3toelasticsearch 然后从存储库上传源代码 (s3toelasticsearch.py)。GitHub 3. 将 Python 依赖项以 Lambda 层导入。 4. 将 IAM policy S3ReadOnly 和AmazonOpenSearchServiceReadOnlyAccess 附加到 Lambda 函数。	常规 AWS

Task	描述	所需技能
创建 OpenSearch 服务集群。	创建集群 1. 创建OpenSearch 服务集群。创建集群时，请执行以下操作: • **Note** 为集群创建可用于登录OpenSearch 仪表板的主用户和密码。：如果您通过 Amazon Cognito 使用身份验证，则无需执行此步骤。 • 选择精细访问权限控制。这为您提供了更多控制 OpenSearch 服务中数据访问权限的方法。 2. 复制域名 URL 并将其作为环境变量 HOST 传递至 Lambda 函数 `s3toelasticsearch` 。 向 IAM 角色授予访问权限 要提供 Lambda 函数的 IAM 角色 (`arn:aws:iam::**:role/service-role/s3toelasticsearch-role-**`) 的精细访问权限，请执行以下操作：	常规 AWS

Task	描述	所需技能
	1. 以主用户身份登录 OpenSearch 控制面板。 2. 选择安全选项卡，然后选择角色、all_access、映射用户、后端角色。 3. 添加 Lambda 函数的 IAM 角色的 Amazon 资源名称（ARN），然后选择保存。有关更多信息，请参阅 OpenSearch 服务文档中的将角色映射到用户。	
为编排创建 Step Functions。	1. 使用标准流程创建 Step Functions 状态机。该定义包含在GitHub 存储库中。 2. 在 JSON 脚本中，将 Lambda 函数替换 ARNs 为您环境 ARNs 中的 Lambda 函数中的。	常规 AWS

部署并运行

Task	描述	所需技能
将输入文件和副本上传到 S3 存储桶。	从GitHub 存储库示例文件夹下载示例文件，然后将文件上传到您之前创建的 S3 存储桶。 1. 将 `Mockedcopy.cpy` 和 `acctix.cpy` 上传到 `<S3_Bucket>/copybook` 文件夹。 2. 将 `Modedupdate.txt` 和 `acctindex.cpy` 示例输	常规 AWS

Task	描述	所需技能
	入文件上传至 <S3_Bucket>/input 夹。	

Task	描述	所需技能

调用 Step Functions。	1. 登录 AWS 管理控制台，然后打开 Step Functions 控制台。
	2. 在导航面板中，选择状态机。
	3. 选择您的状态机，然后选择开始执行。
	4. 在输入框内，输入以下副本/文件路径作为 S3 存储桶的 JSON 变量，然后选择开始执行。
	
	`{ "s3_copybook_bucket_name": "<BUCKET NAME>", "s3_copybook_bucket_key": "<COPYBOOK PATH>", "s3_source_bucket_name": "<BUCKET NAME", "s3_source_bucket_key": "INPUT FILE PATH" }`
	
	例如：
	
	`{ "s3_copybook_bucket_name": "fileaidtest", "s3_copybook_bucket_key": "copybook/acctix.cpy", "s3_source_bucket_name": "fileaidtest",`
		常规 AWS

Task	描述	所需技能

	<pre>"s3_source_bucket_
	key": "input/ac
	ctindex"
	}</pre>	

在 Step Functions 中验证工作流执行。	在 Step Functions 控制台，在图表检查器中查看工作流执行情况。执行运行状态用颜色编码来表示执行状态。例如，蓝色表示进行中，绿色表示成功，红色表示失败。您也可以查看执行事件历史记录部分的表格，了解有关执行事件的更多详细信息。 有关图形化工作流执行的示例，请参阅此模式的其他信息部分中的 Step Functions 图。	常规 AWS
在 Amazon 中验证配送日志 CloudWatch。	1. 登录 AWS 管理控制台并打开CloudWatch 控制台。 2. 在导航窗格中，选择日志，然后选择日志组。 3. 在搜索框中，搜索 `s3toelasticsearch` 函数的日志组。 有关成功传送日志的示例，请参阅此模式的 “其他信息” 部分中的CloudWatch 传送日志。	常规 AWS

Task	描述	所需技能
在 OpenSearch 仪表板中验证格式化后的文件并执行文件操作。	1. 登录 AWS 管理控制台。在 “分析” 下，选择 “亚马逊 OpenSearch 服务”。 2. 在导航窗格中，选择域。 3. 在搜索框中，在OpenSearch 仪表板中输入您的域名的 URL。 4. 选择您的控制面板，然后以主用户身份登录。 5. 以表格形式浏览索引数据。 6. 将输入文件与 OpenSearch 仪表板中格式化的输出文件（索引文档）进行比较。控制面板视图显示为格式化文件添加的列标题。确认未格式化输入文件中的源数据与控制面板视图中的目标数据匹配。 7. 对索引文件执行诸如搜索(例如：使用字段名称、值或表达式)、筛选和 DQL（控制面板查询语言）操作之类的操作。	常规 AWS

相关资源

参考

- COBOL 副本示例（IBM 文档）
- BMC Compuware File-Aid（BMC 文档）

教程

- 教程：使用 Amazon S3 触发器调用 Lambda 函数（AWS Lambda 文档）
- 如何使用 AWS Step Functions 和 AWS Lambda 创建无服务器工作流（AWS 文档）
- 在@@ 亚马逊 OpenSearch 服务中使用 OpenSearch 控制面板（AWS 文档）

其他信息

Step Functions 图

以下示例显示了 Step Functions 图表。该图显示了在此模式中使用的 Lambda 函数的执行运行状态。

CloudWatch 传送日志

以下示例显示了 `s3toelasticsearch` 执行执行的成功传送日志。

2022-08-10T15:53:33.033-05:00	处理文档数量：100	
	2022-08-10T15:53:33.171-05:00	[信息] 2022-08-10T20：53:33.171 Z a1b2c3d4-5678-90ab-cdef-: 443/_Bulk [状态:200 请求:0.100s] EXAMPLE11111 https://search-essearch-3h4uqclifeqaj2vg4mphe7ffle.us-east-2.es.amazonaws.com
	2022-08-10T15:53:33.172-05:00	批量写入成功：100 个文档

对经过 Blu Age 现代化改造的大型机工作负载进行容器化

由 Richard Milner-Watts (AWS) 编写

摘要

此模式为运行已使用 Blu Age 工具实现现代化的大型机工作负载提供了一个示例容器环境。Blu Age 将传统的大型机工作负载转换至现代 Java 代码。此模式为 Java 应用程序提供了包装，因此您可使用 Amazon Elastic Container Service (Amazon ECS) 或 Amazon Elastic Kubernetes Service (Amazon EKS) 等运行容器编排服务。

有关使用 Blu Age 和 Amazon Web Services 实现工作负载现代化的更多信息，请参见以下 AWS Prescriptive Guidance 出版物：

- 在无服务器 AWS 基础设施上运行现代化 Blu Age 大型机工作负载
- 使用 Terraform 为容器化 Blu Age 应用程序部署环境

如需有关使用 Blu Age 对大型机工作负载进行现代化改造的帮助，请在 Blu Age 网站上选择联系我们的专家。要获得有关将现代化工作负载迁移到 AWS、将其与 Amazon Web Services 集成以及将其投入生产的帮助，请联系您的 AWS 客户经理或填写 AWS Professional Services 表。

先决条件和限制

先决条件

- Blu Age 创建的现代化 Java 应用程序。出于测试目的，此模式提供了示例 Java 应用程序，您可将其用作概念验证。
- Docker 环境，您可以用它来构建容器。

限制

根据您使用的容器编排平台，可供容器使用的资源（例如 CPU、RAM 和存储）可能会受到限制。例如，如果您将 Amazon ECS 与 AWS Fargate 配合使用，请参阅 Amazon ECS 文档了解限制和注意事项。

架构

源技术堆栈

- Blu Age

- Java

目标技术堆栈

- Docker

目标架构

下图显示了 Docker 容器中 Blu Age 应用程序的架构。

1. 容器的入口点是包装器脚本。该 bash 脚本负责为 Blu Age 应用程序准备运行时环境，并处理输出。
2. 容器内的环境变量用于配置包装器脚本中的变量，例如 Amazon Simple Storage Service (Amazon S3) 存储桶名称和数据库凭证。环境变量由 AWS Secrets Manager 或 Parameter Store（AWS Systems Manager 的一项功能）提供。如果您使用 Amazon ECS 作为容器编排服务，也可在 Amazon ECS 任务定义中对环境变量进行硬编码。
3. 在运行 Blu Age 应用程序前，包装脚本负责将所有输入文件从 S3 存储桶提取到容器中。AWS 命令行界面（AWS CLI）已安装在容器中。这提供了一种通过网关虚拟私有云（VPC）端点访问 Amazon S3 中存储的数据元的机制。
4. Blu Age 应用程序的 Java 档案 (JAR) 文件可能需要与其他数据来源（例如 Amazon Aurora）进行通信。
5. 完成后，包装脚本将生成的输出文件传送到 S3 存储桶中以供进一步处理（例如，由 Amazon CloudWatch 日志服务处理）。如果您使用的是标准 CloudWatch 日志记录的替代方案，则该模式还支持将压缩的日志文件传送到 Amazon S3。

工具

Amazon Web Services

- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- Amazon Elastic Container Service (Amazon ECS)是一项快速且可扩展的容器管理服务，可帮助运行、停止和管理集群上的容器。

工具

- Docker是软件平台，用于快速构建、测试和部署应用程序。Docker 将软件打包成称为容器的标准化单元，容器拥有软件运行所需一切，包括库、系统工具、代码和运行时系统。您可使用 Docker 将应用程序部署和扩缩到任何环境中。
- Bash 是 GNU 操作系统的命令语言接口 (Shell)。
- Java 是这种模式中使用的编程语言和开发环境。
- Blu Age 是 AWS 大型机现代化工具，可将传统的大型机工作负载（包括应用程序代码、依赖项和基础设施）转换为现代云工作负载。

代码存储库

此模式的代码可在 GitHub Blu Age 示例容器存储库中找到。

最佳实践

- 使用环境变量，将变量外部化以改变应用程序的行为。这些变量使容器编排解决方案能够更改运行时环境，而无需重建容器。此模式包括对 Blu Age 应用程序有用的环境变量示例。
- 在运行 Blu Age 应用程序之前，请验证所有应用程序依赖项。例如，验证数据库是否可用以及凭证是否有效。在包装脚本中编写测试以验证依赖项，如果不满足则提前失败。
- 在包装器脚本中使用详细日志记录。直接与正在运行的容器交互可能具有挑战性，具体取决于编排平台以及作业所需时间。确保将有用的输出写入到 STDOUT，以帮助诊断任何问题。例如，输出可能包括运行应用程序之前和之后应用程序工作目录的内容。

操作说明

获取 Blu Age 应用程序 JAR 文件

Task	描述	所需技能
选项 1 - 使用 Blu Age 获取应用程序 JAR 文件。	这种模式中的容器需要 Blu Age 应用程序。或者，您可使用随此模式提供的示例 Java 应用程序作为原型。 与 Blu Age 团队合作，为您的应用程序获取 JAR 文件，该文件可以烘焙到容器中。如果	云架构师

Task	描述	所需技能
	JAR 文件不可用，请参见下一个任务以改用示例应用程序。	
选项 2 - 生成或使用提供的示例应用程序 JAR 文件。	此模式提供了预先构建示例 JAR 文件。此文件将应用程序的环境变量输出到 STDOUT，然后休眠 30 秒后退出。 此文件名为`bluAgeSample.jar` ，位于存储库的 docker 文件夹中。 GitHub 如果要修改代码并构建自己的 JAR 文件版本，请使用位于的源代码。 /java_ sample/src/sample _java_app.java 在存储库中。 GitHub 您可在 ./java_sample/build.sh 上使用构建脚本，编译 Java 源代码并生成一个新的 JAR 文件。	应用程序开发人员

构建 Blu Age 容器

Task	描述	所需技能
克隆 GitHub 存储库。	使用以下命令克隆代码存储库示例： `git clone https://github.com/aws-samples/aws-blu-age-sample-container`	AWS DevOps
使用 Docker 构建容器。	在将容器推送至 Docker 注册表（例如 Amazon ECR）之前，使用 Docker 构建容器：	AWS DevOps

Task	描述	所需技能
	1. 在您选择的终端上，导航到本地 GitHub 存储库中的`docker`文件夹。 2. 使用以下命令构建容器： `docker build -t <tag> .` 其中 `<tag>` 是您要使用的容器名称。	
测试 Blu Age 容器。	（可选）如有必要，请使用以下命令在本地测试容器： `docker run -it <tag> /bin/bash`	AWS DevOps

Task	描述	所需技能
向您的 Docker 存储库进行身份验证。	如果您计划使用 Amazon ECR，请按照 Amazon ECR 文档中的说明安装和配置 AWS CLI，并使用您的默认注册表对 Docker CLI 进行身份验证。 我们建议您使用get-login-password 命令进行身份验证。 **Note** 如果您使用查看推送命令按钮，Amazon ECR 控制台会提供此命令的预填充版本。有关更多信息，请参阅 Amazon ECR 文档。 `aws ecr get-login -password --region <region> | docker login --username AWS --password-stdin <account>.dkr.ecr.<region>.amazonaws.com` 如您不打算使用 Amazon ECR，请按照为您的容器注册系统提供的说明进行操作。	AWS DevOps

Task	描述	所需技能
创建容器存储库。	在 Amazon ECR 中创建存储库。有关说明，请参见示例使用 Terraform 为容器化 Blu Age 应用程序部署环境。 如果您使用的是其他容器注册表系统，请按为该系统提供的说明进行操作。	AWS DevOps

标记您的容器，并将其推送到目标存储库。	如果您使用的是Amazon ECR：
	
	1. 通过 Amazon ECR 注册表和存储库标记本地 Docker 映像，这样您就可以将其推送知远程存储库：
	
	`docker tag <tag>:latest <account>.dkr.ecr.<region>.amazonaws.com/<repository>:<versionNumber>`
	
	2. 将映象推送至远程存储库：
	
	`docker push <account>.dkr.ecr.<region>.amazonaws.com/<repository>:<versionNumber>`
	
	有关更多信息，请参阅《Amazon ECR 用户指南》中的推送 Docker 映像。
		AWS DevOps

相关资源

AWS 资源

- AWS Blu Age 示例容器存储库
- 在无服务器 AWS 基础设施上运行现代化 Blu Age 大型机工作负载
- 使用 Terraform 为容器化 Blu Age 应用程序部署环境
- 将 Amazon ECR 与 AWS CLI 配合使用（Amazon ECR 用户指南）
- 私有注册表身份验证（Amazon ECR 用户指南）
- Amazon ECS 文档
- Amazon EKS 文档

其他资源

- Blu Age 网站
- Docker 网站

使用 Python 在 AWS 上将 EBCDIC 数据转换并解压为 ASCII

由 Luis Gustavo Dantas (AWS) 编写

摘要

由于大型机通常托管关键业务数据，因此在将数据迁移置 Amazon Web Services (AWS) 云或其他美国信息交换标准规范 (ASCII) 环境时，实现数据现代化很重要。在大型机上，数据通常以扩展二进制编码十进制交换码 (EBCDIC) 格式编码。导出数据库、Virtual Storage Access Method (VSAM) 或平面文件通常会生成打包的二进制 EBCDIC 文件，这些文件迁移比较复杂。最常用的数据库迁移解决方案是变更数据捕获 (CDC)，在大多数情况下，其会自动转换数据编码。但是，CDC 机制可能不适用于此数据库、VSAM 或平面文件。对于此文件，需要另一种方法来实现数据现代化。

此模式介绍了如何通过将 EBCDIC 数据转换为 ASCII 格式来实现其现代化。转换后，您可将数据加载至分布式数据库中，也可以让云中的应用程序直接处理数据。该模式使用mainframe-data-utilities GitHub 存储库中的转换脚本和示例文件。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- EBCDIC 输入文件及其相应的通用面向业务的语言 (COBOL) 副本。存储库中包含一个 EBCDIC 文件和 COBOL 字帖样本。mainframe-data-utilities GitHub 有关 COBOL 副本的更多信息，请参阅 IBM 网站上的 Enterprise COBOL for z/OS 6.4 编程指南

限制

- 不支持 COBOL 程序定义的文件布局。它们必须单独提供。

产品版本

- Python 版本 3.8 或更高版本。

架构

源技术堆栈

- 大型机的 EBCDIC 数据

- COBOL 副本

目标技术堆栈

- 虚拟私有云 (VPC EC2) 中的亚马逊弹性计算云 (Amazon) 实例
- Amazon Elastic Block Store（Amazon EBS）
- Python 及其必需的包、 JavaScript 对象表示法 (JSON)、系统和日期时间
- ASCII 平面文件可供现代应用程序读取或加载至关系数据库表中

目标架构

架构图显示了在实例上将 EBCDIC 文件转换为 ASCII 文件的过程： EC2

1. 使用 parse_copybook_to_json.py 脚本，您可将 COBOL 副本转换为 JSON 文件。
2. 使用 JSON 文件和 extract_ebcdic_to_ascii.py 脚本，您可将 EBCDIC 数据转换为 ASCII 文件。

自动化和扩缩

在首次手动文件转换所需资源到位后，您可自动进行文件转换。此示例不包括自动化说明。有多种方法可自动转换。下文概述了一种可能使用的方法：

1. 将 AWS 命令行界面（AWS CLI）和 Python 脚本命令封装至 shell 脚本中。
2. 创建一个 AWS Lambda 函数，该函数将外壳脚本任务异步提交到实例。 EC2 有关更多信息，请参阅使用 AWS Lambda 安排 SSH 作业。
3. 创建 Amazon Simple Storage Service (Amazon S3) 触发器，该触发器在每次上传旧文件时调用 Lambda 函数。有关更多信息，请参见使用 Amazon S3 触发器调用 Lambda函数。

工具

Amazon Web Services

- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- 亚马逊弹性区块存储 (Amazon EBS) Block Store 提供块级存储卷，用于亚马逊弹性计算云 (Amazon) 实例。 EC2

- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。

其他工具

- GitHub是一项代码托管服务，提供协作工具和版本控制。
- Python 是高级编程语言。

代码存储库

此模式的代码可在mainframe-data-utilities GitHub 存储库中找到。

操作说明

准备实 EC2 例

Task	描述	所需技能
启动实 EC2 例。	该 EC2 实例必须具有出站互联网访问权限。这允许实例访问上可用的 Python 源代码 GitHub。创建实例： 1. 在 https://console.aws.amazon.com/ec2 上打开亚马逊 EC2 控制台。 2. 启动一个 EC2 Linux 实例。使用公有 IP 地址，允许通过端口 22 进行入站访问。确保实例的存储大小至少为 EBCDIC 数据文件大小的两倍。有关说明，请参阅 Amazon EC2 文档。	常规 AWS
安装 Git。	1. 使用安全外壳 (SSH) 客户端，连接到您刚刚启动的	常规 AWS、Linux

Task	描述	所需技能

	EC2 实例。有关更多信息，请参阅连接到您的 Linux 实例。
	2. 在 Amazon EC2 控制台中，运行以下命令。这会在 EC2 实例上安装 Git。
	`sudo yum install git`
	3. 运行以下命令并确认 Git 已成功安装。
	`git --version`	

Task	描述	所需技能

安装 Python。	1. 在 Amazon EC2 控制台中，运行以下命令。这将在 EC2 实例上安装 Python。
	`sudo yum install python3`
	2. 在 Amazon EC2 控制台中，运行以下命令。这会在 EC2 实例上安装 Pip3。
	`sudo yum install python3-pip`
	3. 在 Amazon EC2 控制台中，运行以下命令。这将在实例上安装适用于 Python 的 AWS 开发工具包 (Boto3)。EC2
	`sudo pip3 install boto3`
	4. 在 Amazon EC2 控制台中，运行以下命令，其中<us-east-1> 是您的 AWS 区域的代码。有关区域代码的完整列表，请参阅 Amazon EC2 文档中的可用区域。
	`export AWS_DEFAULT_REGION=<us-east-1>`
		常规 AWS、Linux

Task	描述	所需技能
克隆 GitHub 存储库。	1. 在 Amazon EC2 控制台中，运行以下命令。这将从中克隆mainframe-data-utilities存储库 GitHub 并打开默认的副本位置，即home文件夹。 `git clone https://github.com/aws-samples/mainframe-data-utilities.git` 2. 在 home 文件夹，确认该 `mainframe-data-utilities` 文件夹存在。	通用 AWS， GitHub

通过 EBCDIC 数据创建 ASCII 文件

Task	描述	所需技能
将 COBOL 副本解析至 JSON 布局文件。	在 `mainframe-data-utilities` 文件夹中，运行 parse_copybook_to_json.py 脚本。此自动化模块从 COBOL 副本读取文件布局并创建 JSON 文件。JSON 文件包含解释和提取源文件中数据所需信息。这将通过 COBOL 副本创建 JSON 元数据。 以下命令将 COBOL 副本转换为 JSON 文件。 `python3 parse_copybook_to_json.py \ -copybook LegacyReference/COBPACK2.cpy \`	常规 AWS、Linux

Task	描述	所需技能

```
-output sample-data/
cobpack2-list.json \
-dict sample-data/
cobpack2-dict.json \
-ebcdic sample-data/
COBPACK.OUTFILE.txt \
-ascii sample-data/
COBPACK.ASCII.txt \
-print 10000
```

该脚本打印所接收到的参数。

```
------------------
------------------
------------------
-----------------
Copybook file.....
.........| LegacyRef
erence/COBPACK2.cpy
Parsed copybook (JSON
 List).| sample-data/
cobpack2-list.json
JSON Dict (document
ation)...| sample-da
ta/cobpack2-dict.json
ASCII file.....
............| sample-
data/COBPACK.ASCII.t
xt
EBCDIC file.....
............| sample-
data/COBPACK.OUTFILE
.txt
Print each.....
............| 10000
---------------------
------------------
------------------
--------------
```

Task	描述	所需技能
	有关参数的更多信息，请参阅 GitHub 存储库中的 README 文件。	

Task	描述	所需技能
检查 JSON 布局文件。	1. 导航到 parse_copybook_to_json.py 脚本中定义的输出路径。 2. 检查 sample-data/cobpack2-list.json 文件的创建时间，确认您选择了相应的 JSON 布局文件。 3. 检查 JSON 文件并确认其内容是否与以下内容相似。 `"input": "extract-ebcdic-to-ascii/COBPACK.OUTFILE.txt", "output": "extract-ebcdic-to-ascii/COBPACK.ASCII.txt", "max": 0, "skip": 0, "print": 10000, "lrecl": 150, "rem-low-values": true, "separator": "|", "transf": [ { "type": "ch", "bytes": 19, "name": "OUTFILE-TEXT" }` JSON 布局文件最重要属性是： • input — 包含要转换的 EBCDIC 文件的路径	常规 AWS、JSON

Task	描述	所需技能
	• `output` — 定义生成 ASCII 文件的路径 • `lrecl` — 指定逻辑记录长度的大小（以字节为单位） • `transf` — 列出所有字段及其大小（以字节为单位） 有关 JSON 布局文件的更多信息，请参阅 GitHub 存储库中的自述文件。	

Task	描述	所需技能

创建 ASCII 文件。	运行 extract_ebcdic_to_ascii.py 脚本，该脚本包含在克隆的 GitHub 存储库中。此脚本读取 EBCDIC 文件，并写入转换后且可读的 ASCII 文件。
	`python3 extract_ebcdic_to_ascii.py -local-json sample-data/cobpack2-list.json`
	当脚本处理 EBCDIC 数据，它会为每批 10,000 条记录打印一条消息。请参阅以下示例。
	`---------- 2023-05-15 21:21:46.322253 | Local Json file | -local-json | sample-data/cobpack2-list.json 2023-05-15 21:21:47.034556 | Records processed | 10000 2023-05-15 21:21:47.736434 | Records processed | 20000 2023-05-15 21:21:48.441696 | Records processed | 30000 2023-05-15 21:21:49.173781 | Records processed | 40000 2023-05-15 21:21:49.874779 | Records processed | 50000`
		常规 AWS

Task	描述	所需技能
	2023-05-15 21:21:50. 705873 | Records processed | 60000 2023-05-15 21:21:51. 609335 | Records processed | 70000 2023-05-15 21:21:52. 292989 | Records processed | 80000 2023-05-15 21:21:52. 938366 | Records processed | 89280 2023-05-15 21:21:52. 938448 Seconds 6.616232 有关如何更改打印频率的信息，请参阅 GitHub 存储库中的 README 文件。	

Task	描述	所需技能

检查 ASCII 文件。	1. 检查 extract-ebcdic-to-ascii/cobpack.ascii.txt 文件的创建时间，以验证该文件是否是最近创建的。
	2. 在 Amazon EC2 控制台中，输入以下命令。这将会打开 ASCII 文件的第一条记录。
	`head sample-data/COBPACK.ASCII.txt -n 1| xxd`
	3. 检查第一条记录内容。由于 EBCDIC 文件通常是二进制文件，因此它们并没有回车符和换行符 (CRLF) 特殊字符。extract_ebcdic_to_ascii.py 脚本添加了竖线字符作为列分隔符，该分隔符在脚本参数中定义。
	
	如果您使用了提供的示例 EBCDIC 文件，以下是 ASCII 文件中的第一条记录。
	`00000000: 2d30 3030 3030 3030 3030 3130 3030 3030  -000000000100000`
	`00000010: 3030 307c 3030 3030 3030 3030 3031 3030  000|00000000100`
	`00000020: 3030 3030 3030 7c2d 3030 3030 3030 3030  000000|-00000000`
		常规 AWS、Linux

Task	描述	所需技能
	00000030: 3031 3030 3030 3030 3030 7c30 7c30 7c31 0100000000| 0|0|1 00000040: 3030 3030 3030 3030 7c2d 3130 3030 3030 00000000| -100000 00000050: 3030 307c 3130 3030 3030 3030 307c 2d31 000|10000 0000|-1 00000060: 3030 3030 3030 3030 7c30 3030 3030 7c30 00000000| 00000|0 00000070: 3030 3030 7c31 3030 3030 3030 3030 7c2d 0000|1000 00000|- 00000080: 3130 3030 3030 3030 307c 3030 3030 3030 100000000| 000000 00000090: 3030 3030 3130 3030 3030 3030 307c 2d30 000010000 0000|-0 000000a0: 3030 3030 3030 3030 3031 3030 3030 3030 000000000 1000000 000000b0: 3030 7c41 7c41 7c0a 00|A|A|.	

Task	描述	所需技能
评估 EBCDIC 文件。	在 Amazon EC2 控制台中，输入以下命令。这将会打开 EBCDIC 文件的第一条记录。 `head sample-data/COBPACK.OUTFILE.txt -c 150 | xxd` 如果您使用 EBCDIC 示例文件，则结果如下所示。 `00000000: 60f0 f0f0 f0f0 f0f0 f0f0 f0f0 f1f0 f0f0 f0f0 `........ .... 00000010: f0f0 f0f0 f0f0 f0f0 f0f0 f0f0 f1f0 f0f0 .... 00000020: f0f0 f0f0 f0f0 f0f0 f0f0 f0f0 f0f0 f1f0 .... 00000030: f0f0 f0f0 f0f0 d000 0000 0005 f5e1 00fa .... 00000040: 0a1f 0000 0000 0005 f5e1 00ff ffff fffa .... 00000050: 0a1f 0000 000f 0000 0c10 0000 000f 1000 .... 00000060: 0000 0d00 0000 0000 1000 0000 0f00 0000 ....`	常规 AWS、Linux、EBCDIC

Task	描述	所需技能
	00000070: 0000 1000 0000 0dc1 c100 0000 0000 0000 .... 00000080: 0000 0000 0000 0000 0000 0000 0000 0000 .... 00000090: 0000 0000 0000	
	要评估源文件和目标文件之间的等效性，则需要全面了解 EBCDIC。例如，示例 EBCDIC 文件的第一个字符是连字符 (-)。在 EBCDIC 文件的十六进制表示法中，此字符由 60 表示，在 ASCII 文件的十六进制表示法中，该字符由 2D 表示。有关 EBCDIC-to-ASCII转换表，请参阅 IBM 网站上的 EBCDIC 到 ASCII。	

相关资源

参考

- EBCDIC 字符集（IBM 文档）
- EBCDIC 到 ASCII（IBM 文档）
- COBOL（IBM 文档）
- JCL 的基本概念（IBM 文档）
- 连接到你的 Linux 实例（亚马逊 EC2 文档）

教程

- 使用 AWS Lambda 安排 SSH 多页（AWS Blog 文章）
- 使用 Amazon S3 触发器调用 Lambda 函数（AWS Lambda 文档）

使用 AWS Lambda 在 Amazon S3 中将大型机文件从 EBCDIC 格式转换为字符分隔 ASCII 格式

由 Luis Gustavo Dantas (AWS) 编写

摘要

此模式向您介绍如何启动 AWS Lambda 函数，该函数可自动将大型机 EBCDIC（扩展二进制编码的十进制交换代码）文件转换为字符分隔的 ASCII（美国信息交换标准代码）文件。在 ASCII 文件上传至 Amazon Simple Storage Service (Amazon S3) 存储桶之后，Lamba 函数运行。文件转换后，可以在基于 x86 工作负载上读取 ASCII 文件或将文件加载到现代数据库中。

此模式中演示的文件转换方法可帮助您克服在现代环境中处理 EBCDIC 文件所面临的挑战。以 EBCDIC 编码的文件通常包含二进制或压缩十进制格式表示的数据，并且字段的长度是固定的。这些特征造成了障碍，因为基于 x86 的现代工作负载或分布式环境通常使用的是 ASCII 编码数据，无法处理 EBCDIC 文件。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 一个 S3 存储桶
- 具有管理权限的 AWS Identity and Access Management (IAM) 用户
- AWS CloudShell
- Python 3.8.0 或更高版本
- 以 EBCDIC 编码的平面文件及其相应的数据结构，采用面向业务的通用语言 (COBOL) 副本

> **Note**
>
> 此模式使用示例 EBCDIC 文件（c lient.ebcdic.txt）及其相应的 COBOL 字帖（COBKS05 .cpy）。这两个文件都在 GitHub mainframe-data-utilities存储库中可用。

限制

- COBOL 副本通常包含多个布局定义。该mainframe-data-utilities项目可以解析这种抄本，但无法推断出在数据转换时要考虑哪种布局。这是因为副本不包含这种逻辑（改为保留在 COBOL 程序中）。因此，解析副本后，必须要手动配置布局选择规则。
- 这种模式受 Lambda 配额约束。

架构

源技术堆栈

- IBM z/OS、IBM i 和其他 EBCDIC 系统
- 数据以 EBCDIC 编码的顺序文件（例如 IBM Db2 卸载）
- COBOL 副本

目标技术堆栈

- Amazon S3
- Amazon S3 事件通知
- IAM
- Lambda 函数
- Python 3.8 或更高版本
- 大型机数据实用程序
- JSN 元数据
- 以字符分隔的 ASCII 文件

目标架构

下图介绍了将大型机 EBCDIC 文件转换为 ASCII 文件的架构。

图表显示了以下工作流：

1. 用户运行副本解析器脚本将 COBOL 副本转换至 JSON 文件。
2. 用户将 JSN 元数据上传至 S3 存储桶。这使得数据转换 Lambda 函数可读取元数据。

3. 用户或自动流程将 EBCDIC 文件上传至 S3 存储桶。
4. S3 通知事件将会触发数据转换 Lambda 函数。
5. AWS 会验证 Lambda 函数的 S3 存储桶读写权限。
6. Lambda 从 S3 存储桶读取文件，然后在本地将文件从 EBCDIC 转换至 ASCII。
7. Lambda 在亚马逊上记录流程状态。 CloudWatch
8. Lamba 将 ASCII 文件重新写入子 Amazon S3。

> **Note**
>
> Copybook 解析器脚本在将元数据转换为 JSON 然后将该数据上传到 S3 存储桶之后，仅运行一次。初始转换后，任何使用上传至 S3 存储桶的相同 JSON 文件的 EBCDIC 文件都将使用相同的元数据。

工具

AWS 工具

- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和在 AWS 上运行的应用程序的指标。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS CloudShell 是一个基于浏览器的外壳，您可以使用 AWS 命令行界面 (AWS CLI) Line CLI 和一系列预装的开发工具来管理 AWS 服务。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。Lambda 仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。

其他工具

- GitHub是一项代码托管服务，提供协作工具和版本控制。
- Python 是高级编程语言。

代码

此模式的代码可在 GitHub mainframe-data-utilities存储库中找到。

最佳实践

考虑下面的最佳实践：

- 在 Amazon 资源名称（ARN）级别设置所需权限。
- 始终为 IAM policy 授予最低权限。有关更多信息，请参阅 IAM 文档中的 IAM 安全最佳实践。

操作说明

创建环境变量与工作文件夹

Task	描述	所需技能

创建环境变量。	将以下环境变量复制至文本编辑器，然后将<placeholder>以下示例中的值替换为您的资源值：
	`bucket=<your_bucket_name>`
	`account=<your_account_number>`
	`region=<your_region_code>`
	Note
	稍后，您将创建对您的 S3 存储桶、AWS 账户和 AWS 区域的引用。
	要定义环境变量，请打开CloudShell 控制台，然后将更新的环境变量复制并粘贴到命令行中。
		常规 AWS

Task	描述	所需技能
	Note 每次会 CloudShell 话重新启动时都必须重复此步骤。	
创建工作文件夹。	要简化以后的资源清理过程，请运行以下命令在中 CloudShell 创建一个工作文件夹： `mkdir workdir; cd workdir` **Note** 每次断开与会 CloudShell 话的连接时，都必须将目录更改为工作目录 (`workdir`)。	常规 AWS

定义 IAM 角色和策略

Task	描述	所需技能
创建 Lambda 函数的信任策略。	EBCDIC 转换器在 Lambda 函数中运行。该函数必须具有 IAM 角色。在创建 IAM 角色前，您必须定义信任策略文档，使资源能够承担该策略。	常规 AWS

Task	描述	所需技能

	在 CloudShell 工作文件夹中，通过运行以下命令创建策略文档：
	
	`E2ATrustPol=$(cat <<EOF`
	`{`
	`  "Version": "2012-10-17",`
	`  "Statement": [`
	`  {`
	`  "Effect": "Allow",`
	`  "Principal": {`
	`  "Service": "lambda.amazonaws.com"`
	`  },`
	`  "Action": "sts:AssumeRole"`
	`  }`
	`  ]`
	`}`
	`EOF`
	`)`
	`printf "$E2ATrustPol" > E2ATrustPol.json`	

为 Lambda 转换用于创建 IAM 角色。	要创建 IAM 角色，请在 CloudShell 工作文件夹中运行以下 AWS CLI 命令： `aws iam create-role --role-name E2AConvLambdaRole --assume-role-policy-document file://E2ATrustPol.json`	常规 AWS

Task	描述	所需技能

为 Lamba 函数创建 IAM policy 文档。	Lambda 函数必须具有对 S3 存储桶的读写访问权限以及对 Amazon 日志的写入权限。CloudWatch
	
	要创建 IAM 策略，请在 CloudShell 工作文件夹中运行以下命令：
	
	`E2APolicy=$(cat <<EOF`
	`{`
	`"Version": "2012-10-17",`
	`"Statement": [`
	`{`
	`"Sid": "Logs",`
	`"Effect": "Allow",`
	`"Action": [`
	`"logs:PutLogEvents",`
	`"logs:CreateLogStream",`
	`"logs:CreateLogGroup"`
	`],`
	`"Resource": [`
	`"arn:aws:logs:*:*:log-group:*",`
	`"arn:aws:logs:*:*:log-group:*:log-stream:*"`
	`]`
	`},`
	`{`
		常规 AWS

Task	描述	所需技能

	<pre>"Sid": "S3",
	"Effect":
	"Allow",
	"Action": [
	
	"s3:GetObject",
	
	"s3:PutObject",
	
	"s3:GetObjectVersion"
	],
	"Resource":
	[
	
	"arn:aws:s3:::%s/*",
	
	"arn:aws:s3:::%s"
	]
	}
	]
	}
	EOF
	)
	printf "$E2APolicy"
	"$bucket" "$bucket"
	> E2AConvLambdaPolic
	y.json</pre>	

将 IAM policy 文档附加到 IAM 角色。	要将 IAM 策略附加到 IAM 角色，请在您的 CloudShell 工作文件夹中运行以下命令：
	
	<pre>aws iam put-role-policy
	--role-name E2AConvLa
	mbdaRole --policy-name
	E2AConvLambdaPolic
	y --policy-document
	file://E2AConvLamb
	daPolicy.json</pre>
		常规 AWS

创建 Lambda 函数以用于 EBCDIC 转换

Task	描述	所需技能
下载 EBCDIC 转换源代码。	在 CloudShell 工作文件夹中，运行以下命令从中下载 mainframe-data-utilities源代码 GitHub： `git clone https://github.com/aws-samples/mainframe-data-utilities.git mdu`	常规 AWS
创建 ZIP 压缩包。	在 CloudShell 工作文件夹中，运行以下命令创建 ZIP 包，该压缩包创建用于 EBCDIC 转换的 Lambda 函数： `cd mdu; zip ../mdu.zip *.py; cd ..`	常规 AWS

创建 Lambda 函数。	在 CloudShell 工作文件夹中，运行以下命令来创建用于 EBCDIC 转换的 Lambda 函数：
	`aws lambda create-function \`
	`--function-name E2A \`
	`--runtime python3.9 \`
	`--zip-file fileb://mdu.zip \`
	`--handler extract_ebcdic_to_ascii.lambda_handler \`
	`--role arn:aws:iam::$account:role/E2AConvLambdaRole \`
		常规 AWS

Task	描述	所需技能

	`--timeout 10 \`
	`--environment "Variable s={layout=$bucket/ layout/}"`
	
	Note
	环境变量布局告知 Lambda 函数 JSON 元数据所在的位置。	

为 Lamba 函数创建基于资源的策略。	在 CloudShell 工作文件夹中，运行以下命令以允许您的 Amazon S3 事件通知触发 Lambda 函数进行 EBCDIC 转换：
	
	`aws lambda add-permi ssion \`
	`--function-name E2A \`
	`--action lambda:In vokeFunction \`
	`--principal s3.amazon aws.com \`
	`--source-arn arn:aws:s 3:::$bucket \`
	`--source-account $account \`
	`--statement-id 1`
		常规 AWS

创建 Amazon S3 事件通知

Task	描述	所需技能
为 Amazon S3 的事件通知创建配置文档。	当文件置于输入文件夹时，Amazon S3 事件通知会启	常规 AWS

Task	描述	所需技能
	动 EBCDIC 转换 Lambda 函数。 在 CloudShell 工作文件夹中，运行以下命令为 Amazon S3 事件通知创建 JSON 文档： `{ "LambdaFunctionConfigurations": [{ "Id": "E2A", "LambdaFunctionArn": "arn:aws:lambda:%s:%s:function:E2A", "Events": ["s3:ObjectCreated:Put"], "Filter": { "Key": { "FilterRules": [{ "Name": "prefix", "Value": "input/" }] } } }] } EOF) printf "$S3E2AEvent" "$region" "$account" > S3E2AEvent.json`	

Task	描述	所需技能
创建 Amazon S3 事件通知。	在 CloudShell 工作文件夹中，运行以下命令来创建 Amazon S3 事件通知： `aws s3api put-bucket-notification-configuration --bucket $bucket --notification-configuration file://S3E2AEvent.json`	常规 AWS

创建和上载 JSON 元数据

Task	描述	所需技能

解析 COBOL 副本。	在 CloudShell 工作文件夹中，运行以下命令将示例 COBOL 抄本解析为 JSON 文件（该文件定义了如何正确读取和切片数据文件）：
	`python3 mdu/parse_copybook_to_json.py \`
	`-copybook mdu/LegacyReference/COBKS05.cpy \`
	`-output CLIENT.json \`
	`-output-s3key CLIENT.ASCII.txt \`
	`-output-s3bkt $bucket \`
	`-output-type s3 \`
	`-print 25`
		常规 AWS

Task	描述	所需技能

添加转换规则。	样本数据文件及其相应 COBOL 副本是一个多布局文件。这意味着转换必须按某些规则对数据进行切片。在这种情况下，每行位置 3 和 4 的字节定义了布局。
	
	在 CloudShell 工作文件夹中，编辑`CLIENT.json` 文件并将内容从更改`"transf-rule": [],` 为以下内容：
	
	`"transf-rule": [`
	`{`
	`"offset": 4,`
	`"size": 2,`
	`"hex": "0002",`
	`"transf": "transf1"`
	`},`
	`{`
	`"offset": 4,`
	`"size": 2,`
	`"hex": "0000",`
	`"transf": "transf2"`
	`}`
	`],`
		常规 AWS、IBM 大型机、Cobol

将 JSN 元数据上传至 S3 存储桶。	在 CloudShell 工作文件夹中，运行以下 AWS CLI 命令将 JSON 元数据上传到您的 S3 存储桶：
	
	`aws s3 cp CLIENT.json`
	`s3://$bucket/layout/`
	`CLIENT.json`
		常规 AWS

转换 EBCDIC 文件

Task	描述	所需技能
将 EBCDIC 文件发送至 S3 存储桶。	在 CloudShell 工作文件夹中，运行以下命令将 EBCDIC 文件发送到 S3 存储桶： `aws s3 cp mdu/sample-data/CLIENT.EBCDIC.txt s3://$bucket/input/` **Note** 我们建议您为输入 (EBCDIC) 和输出 (ASCII) 文件设置不同的文件夹，以避免在 ASCII 文件上传到 S3 存储桶时再次调用 Lambda 转换函数。	常规 AWS
检查输出情况。	在 CloudShell 工作文件夹中，运行以下命令以检查您的 S3 存储桶中是否生成了 ASCII 文件： `awss3 ls s3://$bucket/` **Note** 数据转换可能需要几秒钟才能完成。我们建议您查看 ASCII 文件。	常规 AWS

Task	描述	所需技能

	ASCII 文件可用后，运行以下命令，将文件从 S3 存储桶下载至当前文件夹：
	`aws s3 cp s3://$bucket/CLIENT.ASCII.txt .`
	检查 ASCII 文件内容：
	`head CLIENT.ASCII.txt`	

清除环境

Task	描述	所需技能

（可选）准备变量与文件夹。	如果与断开连接 CloudShell，请重新连接，然后运行以下命令将目录更改为工作文件夹：
	`cd workdir`
	确保您已定义环境变量：
	`bucket=<your_bucket_name>`
	`account=<your_account_number>`
	`region=<your_region_code>`
		常规 AWS

移除存储桶的通知配置。	在 CloudShell 工作文件夹中，运行以下命令以删除 Amazon S3 事件通知配置：	常规 AWS

Task	描述	所需技能
	`aws s3api put-bucket-notification-configuration \ --bucket=$bucket \ --notification-configuration="{}"`	
删除 Lambda 函数。	在 CloudShell 工作文件夹中，运行以下命令删除 EBCDIC 转换器的 Lambda 函数： `awslambdadelete-function--function-nameE2A`	常规 AWS

删除 IAM 角色和策略。	在 CloudShell 工作文件夹中，运行以下命令以删除 EBCDIC 转换器角色和策略：
	`aws iam delete-role-policy --role-name E2AConvLambdaRole --policy-name E2AConvLambdaPolicy`
	
	`aws iam delete-role --role-name E2AConvLambdaRole`
		常规 AWS

Task	描述	所需技能

删除 S3 存储桶内生成的文件。	在 CloudShell 工作文件夹中，运行以下命令删除 S3 存储桶中生成的文件：
	`aws s3 rm s3://$bucket/layout --recursive`
	`aws s3 rm s3://$bucket/input --recursive`
	`aws s3 rm s3://$bucket/CLIENT.ASCII.txt`
		常规 AWS

删除工作文件夹。	在 CloudShell 工作文件夹中，运行以下命令以删除`workdir`及其内容： `cd ..; rm -Rf workdir`	常规 AWS

相关资源

- 大型机数据实用程序自述文件 () GitHub
- EBCDIC 字符集（IBM 文档）
- EBCDIC 到 ASCII（IBM 文档）
- COBOL（IBM 文档）
- 使用 Amazon S3 触发器调用 Lambda 函数 (AWS Lambda 文档)

使用 Micro Focus 转换具有复杂记录布局的大型机数据文件

由彼得·韦斯特编写

摘要

此模式向您展示如何使用 Micro Focus 结构文件将包含非文本数据和复杂记录布局的大型机数据文件从 EBCDIC（扩展二进制编码十进制交换码）字符编码转换为 ASCII（美国信息交换标准代码）字符编码。若要完成文件转换，您必须执行以下操作：

1. 准备描述大型机环境中所有数据项和记录布局的单一源文件。
2. 使用 Micro Focus 数据文件编辑器作为 Micro Focus Classic Data File Tools 或 Data File Tools，创建包含数据记录布局的结构文件。结构文件可识别非文本数据，以便您可正确地将大型机文件从 EBCDIC 转换为 ASCII。
3. 通过Classic Data File Tools 或 Data File Tools 测试结构文件。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- Micro Focus Enterprise Developer for Windows，可通过 AWS Mainframe Modernization 获得

产品版本

- Micro Focus Enterprise Server 7.0 和更高版本

工具

- Micro Focus Enterprise Server 为使用企业开发人员的任何集成式开发环境（IDE）变体创建的应用程序提供运行环境。
- Micro Focus Classic Data File Tools 可帮您转换、导航、编辑和创建数据文件。Classic Data File Tools 包括数据文件转换器、记录布局编辑器以及数据文件编辑器。
- Micro Focus Data File Tools 可创建、编辑和移动数据文件。Data File Tools 包括数据文件编辑器、文件转换实用程序和数据文件结构命令行实用程序。

操作说明

准备源文件

Task	描述	所需技能
确定源组件。	确定文件的所有可能的记录布局，包含任何包含非文本数据的重新定义。 如果您的布局包含重新定义，则必须将这些布局分解为描述数据结构每种可能排列的独特布局。通常，数据文件记录布局可以用以下原型来描述： • 仅包含文本数据记录布局 • 使用非文本数据记录布局 • 使用从属于 REDEFINES 子句的非文本数据的记录布局 有关为包含复杂记录布局的文件创建扁平化记录布局的更多信息，请参阅在 ASCII 环境中重新托管 EBCDIC 应用程序以进行大型机迁移。	应用程序开发人员
确定记录布局条件。	对于具有多个记录布局的文件或包含带有 REDEFINES 子句的复杂布局的文件，请标识记录中的数据和条件，您可使用这些数据和条件来定义转换期间要使用的布局。我们建议您与了解处理此文件的程序的主题专家 (SME) 讨论此任务。	应用程序开发人员

Task	描述	所需技能

	例如，文件可能包含两种包含非文本数据的记录类型。您可检查源代码，并可能找到类似以下代码的代码：
	
	`MOVE "M" TO PART-TYPE`
	`MOVE "MAIN ASSEMBLY" TO PART-NAME`
	`MOVE "S" TO PART-TYPE`
	`MOVE "SUB ASSEMBLY 1" TO PART-NAME`
	
	代码可帮您识别以下内容：
	
	• “PART-TYPE” 字段用于确定以下记录类型
	• 值 “M” 用于 “M-PART-RECORD”
	• 值 “S” 用于 “S-PART-RECORD”
	
	您可记录此字段用于将记录布局与文件中的正确数据记录关联的值。	

Task	描述	所需技能
生成源文件。	如果文件通过多个源文件描述，或者如果记录布局包含从属于 REDEFINES 子句的非文本数据，则创建一个包含记录布局的新源文件。新程序不需要使用 SELECT 和 FD 语句描述文件。该程序可以简单地将记录描述含为 Working-Storage 中的 01 个级别。 **Note** 您可以为每个数据文件创建一个源文件，也可以创建一个描述所有数据文件的主源文件。	应用程序开发人员

Task	描述	所需技能
编译源文件。	编译源文件，以构建数据目录。我们建议您使用 EBCDIC 字符集来编译源文件。如果使用 IBMCOMP 指令或 ODOSLIDE 指令，则也必须在源文件中使用这些指令。 **Note** IBMCOMP 会影响 COMP 字段的字节存储，而 ODOSLIDE 会影响发生变化结构的填充。如果此指令设置不正确，则转换工具将无法正确读取数据记录。这会导致转换后的文件中显示错误数据。	应用程序开发人员

(选项 A) 使用 Classic Data File Tools 创建结构文件

Task	描述	所需技能
启动该工具并加载目录。	1. 选择 Windows “开始” 菜单图标，搜索并选择 Micro Focus Enterprise Developer，然后选择 Classic Data File Tools 2. 选择文件，然后选择记录布局。 3. 在选择要从中构造布局的文件对话框，在文件名中，选择之前编译源文件时创建的	应用程序开发人员

Task	描述	所需技能
	IDY (.idy) 文件。然后选择 Open (打开)。 4. 若要确认 Classic Data File Tools 正在使用 EBCDIC，则如果 IDY 文件设置为 EBCDIC 且 Datatools 设置为 ANSI，可在 Data File Tools 对话框选择是。	
创建默认记录布局。	对所有与任何条件布局不匹配的记录使用默认记录布局。 1. 在布局窗口中，展开数据结构，然后找到用于默认布局的 01 级别。 2. 右键点击 01 项目，然后选择新建布局。 3. 在新建记录布局向导对话框，选择默认布局，然后选择下一步。 4. 选择完成。 默认布局显示在布局窗格中，可以通过红色文件夹图标进行识别。	应用程序开发人员

Task	描述	所需技能
创建有条件记录布局。	当文件中包含多个记录布局时，请使用条件记录布局。 1. 在布局窗格中，展开数据结构，然后找到用于条件布局的 01 级别。 2. 右键点击 01 项目，然后选择新建布局。 3. 在新建记录布局向导对话框，选择条件布局，然后选择下一步。 4. 选择完成。条件布局显示在布局窗格中，可以通过黄色文件夹图标进行识别。 5. 展开条件布局，右键单击必须放置条件的字段，然后选择属性。 6. 在字段属性对话框中，输入条件。确认字符集已设置为 EBCDIC，然后选择确定。已设置条件的字段旁边会出现一个复选标记。 7. 对于需要此布局条件的任何其他字段，重复第 5—6 步。 8. 对必须添加的任何其他条件布局重复步骤第 1 至 6 步。 9. 选择 文件，选择 另存为，然后将结构文件保存到磁盘。	应用程序开发人员

（选项 B）使用 Data File Tools 创建结构文件

Task	描述	所需技能
启动该工具并加载目录。	1. 选择 Windows “开始” 菜单图标，搜索并选择 Micro Focus Enterprise Developer，然后选择 Data File Tools。 2. 选择文件、新建、结构、文件。 3. 在打开对话框中，在文件名中，选择之前编译源文件时创建的 IDY (`.idy`) 文件。然后选择 Open (打开)。 4. 要确认 Data File Tools 正在使用 EBCDIC，请确认调试文件部分的下拉菜单已设置为 EBCDIC。	应用程序开发人员
创建默认记录布局。	对所有不匹配任何条件布局的记录使用默认记录布局。 1. 在左边窗格的可用布局部分中，展开数据结构，然后找到用于默认布局的 01 级别。 2. 右键点击 01 项目，然后选择创建默认布局。 默认布局显示在布局窗格中，可以通过蓝色“D”图标进行识别。	应用程序开发人员

Task	描述	所需技能
创建有条件记录布局。	当文件中包含多个记录布局时，请使用条件记录布局。 1. 在右边窗格的选定布局部分中，展开数据结构，然后找到用于条件布局的 01 级别。 2. 右键单击 01 项目，然后选择创建条件布局。条件布局显示在右侧的布局窗格中，可以通过绿色的 “C” 图标识出。 3. 展开条件布局，右键单击必须放置条件的字段，然后选择属性。 4. 在字段属性 对话框中，输入条件。确认字符集已设置为EBCDIC，然后选择确定。已设置条件的字段旁边会出现一个红色“IF”图标。 5. 对于需要此布局条件的任何其他字段，重复第 3—4 步。 6. 对必须添加的任何其他条件布局重复步骤 1 至 4。 7. 选择文件，选择另存为，然后将结构文件保存到磁盘。	应用程序开发人员

(选项 A)使用 Classic Data File Tools 测试结构文件

Task	描述	所需技能
测试 EBCDIC 数据文件。	确认您可使用结构文件正确查看 EBCDIC 测试数据文件。 1. 选择 Windows “开始” 菜单图标，找到并选择 Micro Focus Enterprise Developer ，然后选择 Classic Data Tools。 2. 选择文件，然后选择打开。 3. 在打开对话框中，在文件名中选择 EBCDIC 数据集，然后选择打开。 4. 选择文件、数据文件编辑器、加载记录布局。 5. 在打开对话框中，在文件名中，选择结构文件，然后选择打开。 6. 要确认字符集模式已设置为 EBCDIC，确认下拉菜单已设置为 EBCDIC。您可在左窗格中看到原始记录数据，在右窗格中看到格式化的数据。 7. 选择各种记录，以确保所有格式都以正确的布局呈现。	应用程序开发人员

(选项 B)使用 Data File Tools 测试结构文件

Task	描述	所需技能
测试 EBCDIC 数据文件。	确认您可使用结构文件正确查看 EBCDIC 测试数据文件。 1. 选择 Windows “开始” 菜单图标，找到并选择 Micro Focus Enterprise Developer，然后选择 Data File Tools。 2. 选择文件、打开、数据文件。 3. 在打开数据文件对话框的本地选项卡，为文件名选择浏览以查找 EBCDIC 测试文件的位置。 4. 对于结构文件（可选），选择浏览以查找结构文件的位置。 5. 在文件详细信息部分，输入文件的详细信息，并确认编码设置为 EBCDIC。 6. 根据您的要求选择打开共享或打开单独模式。 7. 确认工具栏外观部分的下拉菜单设置为 EBCDIC。您将在左窗格中看到原始记录数据，在右窗格中看到格式化的数据。 8. 选择各种记录，以确保所有格式都以正确的布局呈现。	应用程序开发人员

测试数据文件转换

Task	描述	所需技能
测试 EBCDIC 文件转换。	1. 选择 Windows “开始” 菜单图标，找到并选择 Micro Focus Enterprise Developer，然后选择 Classic Data Tools。 2. 选择工具，然后选择转换。 3. 在数据文件转换对话框的输入文件 部分，为文件名选择浏览，以查找并选择 EBCDIC 输入文件。确认字符集已设置为 EBCDIC。 4. 在字符集转换部分，选中转换字符集和包含非文本数据项的记录复选框。选择选择要转换的布局，然后选择浏览，以查找并选择结构文件。 5. 在新建文件部分的文件名中，输入要创建的 ASCII 输出文件的路径和文件名。默认情况下，转换工具与输入文件的格式相同。测试时，将选项设置为默认值。 6. 选择转换。 7. 按照（选项 A）使用 Classic Data File Tools 测试结构文件或（选项 B）使用 Data File Tools 测试结构文件部分中的步骤进行操作，但要加载 ASCII 输出文件而不是 EBCDIC 文件。	应用程序开发人员

Task	描述	所需技能
	8. 将 EBCDIC 和 ASCII 文件加载至数据文件编辑器中，然后并排比较这些文件以检查转换的准确性。	

相关资源

- Micro Focus（Micro 文档）
- 大型机和遗留代码（AWS Blog 文章）
- AWS Prescriptive Guidance（AWS 文档）
- AWS 文档（AWS 文档）
- AWS 一般参考（AWS 文档）
- AWS 词汇表（AWS 文档）

使用 Terraform 为容器化 Blu Age 应用程序部署环境

由 Richard Milner-Watts (AWS) 编写

摘要

将传统的大型机工作负载迁移至现代云架构可以消除维护大型机的成本，而这些成本只会随着环境的老化而增加。然而，从大型机迁移作业可能会带来独特挑战。内部资源可能不熟悉工作逻辑，与一般CPUs商用相比，大型机在这些特殊任务中的高性能可能很难复制。重写这些工作可能是一项艰巨的任务，因此需要付出巨大的努力。

Blu Age 将传统的大型机工作负载转换为现代 Java 代码，然后您可将其作为容器运行。

此模式提供了示例无服务器架构，用于运行已使用 Blu Age 工具进行现代化改造的容器化应用程序。随附的 HashiCorp Terraform 文件将为编排 Blu Age 容器构建安全的架构，同时支持批处理任务和实时服务。

有关使用 Blu Age 和 Amazon Web Services 实现工作负载现代化的更多信息，请参见以下 AWS Prescriptive Guidance 出版物：

- 运行已经通过 AWS 无服务器基础设施上 Blu Age 现代化的大型机工作负载
- 对经过 Blu Age 现代化改造的大型机工作负载进行容器化

如需有关使用 Blu Age 对大型机工作负载进行现代化改造的帮助，请在 Blu Age 网站上选择联系我们的专家。要获得有关将现代化工作负载迁移到 AWS、将其与 Amazon Web Services 集成以及将其投入生产的帮助，请联系您的 AWS 客户经理或填写 AWS Professional Services 表。

先决条件和限制

先决条件

- 该示例容器化了 Blu Age 应用程序，其由对经过 Blu Age 现代化改造的大型机工作负载进行容器化模式提供。示例应用程序提供了处理现代化应用程序的输入和输出的逻辑，并且它可以与此体系结构集成。
- 部署此资源需要 Terraform。

限制

- Amazon Elastic Container Service (Amazon ECS) 对容器可用的任务资源设定了限制。这些资源包括 CPU、RAM 和存储。例如将 Amazon ECS 与 AWS Fargate 配合使用时，任务资源限制适用。

产品版本

此解决方案已使用以下版本进行测试：

- Terraform 1.3.6
- Terraform AWS Provider 4.46.0

架构

源技术堆栈

- Blu Age
- Terraform

目标技术堆栈

- Amazon Aurora PostgreSQL 兼容版
- AWS Backup
- Amazon Elastic Container Registry(Amazon ECR)
- Amazon ECS
- AWS Identity and Access Management Service (IAM)
- AWS Key Management Server (AWS KMS)
- AWS Secrets Manager
- Amazon Simple Notification Service(Amazon SNS)
- Amazon Simple Storage Service(Amazon S3)
- AWS Step Functions
- AWS Systems Manager

目标架构

下图显示了解决方案架构。

1. 解决方案部署了以下 IAM 角色：
   - 批处理任务角色
   - 批处理任务执行角色
   - 服务任务角色
   - 服务任务执行角色
   - Step Functions 角色
   - AWS Backup 角色
   - RDS 增强型监控角色。

   这些角色符合最低权限访问原则。
2. Amazon ECR 用于存储由此模式编排容器映像。
3. AWS Systems Manager Parameter Store 在运行时向 Amazon ECS 任务定义提供关于每个环境的配置数据。
4. AWS Secrets Manager 在运行时向 Amazon ECS 任务定义提供关于环境的敏感配置数据。这些数据通过 AWS KMS 加密。
5. Terraform 模块为所有实时和批处理任务创建 Amazon ECS 任务定义。
6. Amazon ECS 使用 AWS Fargate 作为计算引擎运行批处理任务。这是一项短暂任务，由 AWS Step Functions 按要求启动。
7. Amazon Aurora PostgreSQL-Compatible 提供了一个支持现代化应用程序的数据库。这取代了大型机数据库，例如 IBM Db2 或者 IBM IMS 数据库。
8. Amazon ECS 运行长期服务，以提供现代化的实时工作负载。这些无状态应用程序永久运行，容器分布至可用区中。
9. 网络负载均衡器用于授予对实时工作负载访问权限。网络负载均衡器支持较早协议，例如 IBM CICS。或者，您可以将应用程序负载均衡器与基于 HTTP 的工作负载结合使用。
10. Amazon S3 为任务输入和输出提供对象存储。容器应处理 Amazon S3 中的拉取和推送操作，以为 Blu Age 应用程序准备工作目录。
11. AWS Step Functions 服务用于编排运行 Amazon ECS 任务以处理批量工作负载。
12. 每个批处理工作负载的 SNS 主题用于将现代化应用程序与其他系统（例如电子邮件）集成，或启动其他操作，例如将输出对象从 Amazon S3 传送到 FTP。

> **Note**
>
> 默认情况下，该解决方案无法访问互联网。此模式假设虚拟私有云（VPC）将使 AWS Transit Gateway 等服务连接到其他网络。因此，部署了多个接口 VPC 端点，授予对解决方案使用的 Amazon Web Services 的访问权限。要开启直接互联网接入，您可以使用 Terraform 模块中的切换开关将 VPC 端点替换为互联网网关和相关资源。

自动化和扩缩

在整个模式中使用无服务器资源有助于确保，通过横向扩展，此设计的规模几乎没有限制。这样可以减少邻居噪音担忧，例如在原始大型机上可能遇到的对计算资源的竞争。可以根据需要安排批处理任务同时运行。

单个容器受到 Fargate 支持的最大大小限制。有关更多信息，请参阅 Amazon ECS 文档中的任务 CPU 和内存部分。

若要水平扩展实时工作负载，您可添加容器。

工具

AWS 服务

- Amazon Aurora PostgreSQL 兼容版是一个完全托管的、与 ACID 兼容的关系数据库引擎，可帮助您建立、运行和扩展 PostgreSQL 部署。
- AWS Backup 是一项完全托管的服务 AWS 服务，可帮助您在云端和内部集中和自动进行数据保护。
- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- Amazon Elastic Container Service (Amazon ECS)是一项快速且可扩展的容器管理服务，可帮助运行、停止和管理集群上的容器。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥，以帮助保护您的数据。
- AWS Secrets Manager 帮助您将代码中的硬编码凭证（包括密码）替换为对 Secrets Manager 的 API 调用，以便以编程方式检索密钥。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。

- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS St ep Functions 是一项无服务器编排服务，可帮助您组合 AWS Lambda 函数和其他功能 AWS 服务 来构建关键业务应用程序。
- AWS Systems Manager Parameter Store 提供安全的分层存储，用于配置数据管理和密钥管理。

其他服务

- HashiCorp Terraform 是一款基础设施即代码 (IaC) 工具，可帮助您使用代码来配置和管理云基础架构和资源。此模式使用 Terraform 创建示例架构。

代码存储库

此模式的源代码可在 GitHub Blu Age 示例 ECS 基础架构 (Terraform) 存储库中找到。

最佳实践

- 对于测试环境，请使用诸如配置现代化应用程序的 `forceDate` 选项之类的功能，以通过始终运行已知时间段来生成一致的测试结果。
- 单独调整每个任务以消耗最佳数量的资源。您可以使用 Amazon Conta CloudWatch iner Insights 获取有关潜在瓶颈的指导。

操作说明

为部署做好环境准备

Task	描述	所需技能
克隆解决方案源代码。	从GitHub 项目中克隆解决方案代码。	DevOps 工程师
通过部署资源存储 Terraform 状态来引导环境。	1. 打开终端窗口，确认 Terraform已安装且 AWS 凭据可用。 2. 导航到 `bootstrap-terraform` 文件夹。	DevOps 工程师

Task	描述	所需技能
	3. 如果您想更改 S3 存储桶 (<accountId>-terraform-backend) 和 Amazon DynamoDB 表 (terraform-lock) 的名称，请编辑 main.tf 文件。 4. 运行 terraform apply命令以部署资源。记下 S3 存储桶与 DynamoDB 表名称。	

部署解决方案基础设施

Task	描述	所需技能
查看和更新 Terraform 配置。	在根目录中，打开文件 main.tf,，查看内容，然后考虑进行以下更新： 1. AWS 区域 通过搜索字符串eu-west-1 并将其替换为要使用的所需区域来更新。 2. 如果在之前的操作说明中更改了默认存储桶名称，请更新 Terraform Backend 块中的存储桶名称。 3. 如果在之前的操作说明中更改了默认 dynamodb_table 值，请更新该值。 4. 将 stack_prefix 变量的值更新为所需字符串。此字	DevOps 工程师

Task	描述	所需技能
	符串将放在由此模式创建的所有资源名称之前。 5. 更新vpc_cidr值。这应该至少是一个 /24地址范围的值。 6. 查看 Locals部分。这用于定义将要部署的 Blu Age 任务。该解决方案将遍历列表对象 bluage_batch_modules ，为列表的每个元素创建关联的资源 (Step Functions 状态机、任务定义和 SNS 主题)。在某些情况下，您可能需要针对不同环境调整变量。例如，要在测试环境中强制运行时系统，可更改 force_execution_time 变量的值。 7. 要开启互联网接入，请将 direct_internet_access_required 值从 false 更改为 true。这将部署互联网网关，以及为基础设施开启公共互联网访问的 NAT 网关和路由表。默认情况下，该解决方案会将接口 VPC 端点部署到无需直接访问互联网的 VPC 中。 8. 要授予对通过 Elastic Load Balancing 提供的任何客户端-服务器工作负载的访问权限，请使用应允许的	

Task	描述	所需技能
	CIDR 网络更新 additional_nlb_igress_cidrs 值。	
部署 Terraform 文件。	请从终端运行 terraform apply 命令部署所有资源。查看 Terraform 生成的更改，然后输入是，以启动构建。 请注意，此基础设施部署时间可能为 15 分钟以上。	DevOps 工程师

（可选）部署有效的 Blu Age 容器化应用程序

Task	描述	所需技能
将 Blu Age 容器映像推送到 Amazon ECR。	将容器推送到上一篇操作说明中创建的 Amazon ECR 存储库。有关说明，请参阅 Amazon ECR 文档。 记下容器映像 URI。	DevOps 工程师
更新 Terraform，以引用 Blu Age 容器映像。	更新文件 main.tf ，以引用您上传的容器映像。	DevOps 工程师
重新部署 Terraform 文件。	从您的终端运行 terraform apply 以部署所有资源。查看来自 Terraform 的建议更新，然后输入 是继续部署。	DevOps 工程师

相关资源

- Blu Age
- 运行已经通过 AWS 无服务器基础设施上 Blu Age 现代化的大型机工作负载

- 对经过 Blu Age 现代化改造的大型机工作负载进行容器化

使用和 AWS Mainframe Modernization Amazon Q 生成 Db2 z/OS 数据见解 QuickSight

由 Shubham Roy (AWS)、Roshna Razack (AWS) 和 Santosh Kumar Singh (AWS) 创作

摘要

如果您的组织在 IBM Db2 大型机环境中托管关键业务数据，那么从这些数据中获取见解对于推动增长和创新至关重要。通过解锁大型机数据，您可以构建更快、更安全、更可扩展的商业智能，从而加快 Amazon Web Services ()AWS云中数据驱动的决策、增长和创新。

这种模式提供了一种解决方案，用于生成业务见解，并根据 IBM Db2 for z/OS 表中的大型机数据创建可共享的叙述。使用精确数据复制功能，将大型机数据更改流式传输到亚马逊托管流媒体for Apache Kafka（亚马逊 MSK）主题AWS Mainframe Modernization 。使用亚马逊 Redshift 流媒体提取，亚马逊 MSK 主题数据存储在亚马逊 Redshift 无服务器数据仓库表中，以便在亚马逊进行分析。QuickSight

在中提供数据后 QuickSight，您可以使用带有 Amazon Q 的自然语言提示 QuickSight来创建数据摘要、提问和生成数据故事。您不必编写 SQL 查询或学习商业智能 (BI) 工具。

业务背景

这种模式为大型机数据分析和数据洞察用例提供了解决方案。使用该模式，您可以为公司的数据构建可视化仪表板。为了演示解决方案，这种模式使用了一家为其在美国的成员提供医疗、牙科和视力计划的医疗保健公司。在此示例中，成员人口统计和计划信息存储在 IBM Db2 for z/OS 数据表中。可视化仪表板显示以下内容：

- 按地区划分的成员分布
- 按性别分列的成员分布
- 按年龄划分的会员分布
- 按计划类型划分的成员分布
- 尚未完成预防性免疫的会员

有关按地区分列的成员分布和尚未完成预防性免疫的成员的示例，请参阅 “其他信息” 部分。

创建仪表板后，您将生成一个数据故事，解释先前分析的见解。数据故事为增加完成预防性免疫接种的成员人数提供了建议。

先决条件和限制

先决条件

- 活跃 AWS 账户的. 该解决方案是在亚马逊弹性计算云（亚马逊 EC2）上的 Amazon Linux 2 上构建和测试的。
- 虚拟私有云 (VPC)，其子网可由您的大型机系统访问。
- 包含业务数据的大型机数据库。有关用于构建和测试此解决方案的示例数据，请参阅附件部分。
- 在 Db2 z/OS 表上启用了更改数据捕获 (CDC)。要在 Db2 z/OS 上启用 CDC，请参阅 IBM 文档。
- 安装在托管源数据库的 z/OS 系统上的 precist Connect connect for z/OS。适用于 z/OS 的 Precission Connect CDC 镜像在 IBM z/OS Amazon 机器映像 (AMI) 的数据复制中作为 zip 文件提供。AWS Mainframe Modernization 要在大型机上安装适用于 z/OS 的 Precission Connect CDC，请参阅 Precist 安装文档。

限制

- 您的大型机 Db2 数据应采用 Precision Connect CDC 支持的数据类型。有关支持的数据类型的列表，请参阅 Precission Connect CDC 文档。
- 您在亚马逊 MSK 上的数据应采用亚马逊 Redshift 支持的数据类型。有关支持的数据类型的列表，请参阅亚马逊 Redshift 文档。
- Amazon Redshift 对不同的数据类型有不同的行为和大小限制。有关更多信息，请参阅亚马逊 Redshift 文档。
- 中的近乎实时的数据 QuickSight 取决于为 Amazon Redshift 数据库设置的刷新间隔。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。Amazon Q in QuickSight 目前并非在所有支持的地区都可用 QuickSight。有关特定终端节点，请参阅服务终端节点和配额页面，然后选择服务的链接。

产品版本

- AWS Mainframe Modernization 使用 4.1.44 版本精确复制数据
- Python 版本 3.6 或更高版本
- Apache Kafka 版本 3.5.1

架构

目标架构

下图显示了一种架构，该架构通过使用 Preclist 和 Amazon Q 中的AWS Mainframe Modernization QuickSight数据复制从大型机数据生成业务见解。

图表显示了以下工作流：

1. Precission Log Reader Agent 从 Db2 日志中读取数据，并将数据写入大型机 OMVS 文件系统上的临时存储中。
2. 发布者代理从临时存储中读取原始 Db2 日志。
3. 本地控制器守护程序对操作进行身份验证、授权、监控和管理操作。
4. 应用代理是使用预配置 EC2 的 AMI 在亚马逊上部署的。它使用 TCP/IP 通过控制器守护程序与发布者代理连接。应用代理使用多个工作程序将数据推送到 Amazon MSK 以实现高吞吐量。
5. 工作人员以 JSON 格式将数据写入 Amazon MSK 主题。作为复制消息的中间目标，Amazon MSK 提供了高度可用的自动故障转移功能。
6. 亚马逊 Redshift 流媒体提取提供从亚马逊 MSK 到亚马逊 Redshift 无服务器数据库的低延迟、高速数据摄取。Amazon Redshift 中的存储过程将大型机变更数据 (insert/update/deletes) 协调到亚马逊 Redshift 表中。这些 Amazon Redshift 表用作的数据分析来源。 QuickSight
7. 用户访问中的数据 QuickSight 以进行分析和见解。您可以使用 Amazon Q 通过自然语言提示与数据进行交互。 QuickSight

工具

AWS 服务

- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速向外扩展或向内扩展。
- AWS Key Management Service (AWS KMS) 帮助您创建和控制加密密钥以帮助保护您的数据。
- Amazon Managed Streaming for Apache Kafka (Amazon MSK) 是一项完全托管式服务，可帮助您构建并运行使用 Apache Kafka 来处理流数据的应用程序。
- Amazon QuickSight 是一项云规模的商业智能 (BI) 服务，可帮助您在单个控制面板中可视化、分析和报告数据。这种模式使用了 Amazon Q 中的生成式商业智能功能 QuickSight。

- Amazon Redshift Serverless 是 Amazon Redshift 的无服务器选项，无需设置和管理数据仓库基础设施，即可在几秒钟内更高效地运行和扩展分析。
- AWS Secrets Manager 有助于您通过对 Secrets Manager 的 API 调用来替换代码中的硬编码凭证（包括密码），以编程方式检索密钥。

其他工具

- P@@ recission Connect CDC 收集来自传统系统的数据并将其集成到云和数据平台中。

代码存储库

此模式的代码可在 GitHub Mainframe_ DataInsights _change_data_ regniforment 存储库中找到。该代码是亚马逊 Redshift 中的一个存储过程。此存储过程将大型机数据更改（插入、更新和删除）从 Amazon MSK 调整到 Amazon Redshift 表中。这些 Amazon Redshift 表用作的数据分析来源。QuickSight

最佳实践

- 在设置 Amazon MSK 集群时，请遵循最佳实践。
- 遵循 Amazon Redshift 数据解析最佳实践以提高性能。
- 为精确设置创建 AWS Identity and Access Management (IAM) 角色时，请遵循最低权限原则并授予执行任务所需的最低权限。有关更多信息，请参阅 IAM 文档中的授予最低权限和安全最佳实践。

操作说明

在 Amazon 上使用 Precist 设置 AWS Mainframe Modernization 数据复制 EC2

Task	描述	所需技能
设置安全组。	要连接到控制器守护程序和 Amazon MSK 集群，请为该 EC2 实例创建一个安全组。添加以下入站和出站规则： • 入站规则 1： • 对于 “类型”，选择 “自定义 TCP”。	DevOps 工程师，AWS DevOps

Task	描述	所需技能
	• 对于协议，选择 TCP。 • 对于端口范围，选择 2626（精确控制器守护程序的默认端口）或在大型机上运行的控制器守护程序的端口号。 • 对于源，选择 CIDR 块。 • 入站规则 2： • 对于类型，选择自定义 TCP。 • 对于协议，请选择 SSH。 • 对于端口范围，请选择 22。 • 对于来源，选择 IP 地址或前缀列表。 • 入站规则 3： • 对于类型，选择自定义 TCP。 • 对于协议，选择 TCP。 • 对于端口范围，请选择 9092-9098。 • 对于源，选择 CIDR 块。 • 出站规则 1： • 对于类型，选择自定义 TCP。 • 对于协议，选择 TCP。 • 对于端口范围，请选择 9092-9098。 • 对于源，选择 CIDR 块。 • 出站规则 2：	

Task	描述	所需技能
	• 对于类型，选择自定义 TCP。 • 对于协议，选择 TCP。 • 对于端口范围，选择 2626（精确控制器守护程序的默认端口）或在大型机上运行的控制器守护程序的端口号。 • 对于源，选择 CIDR 块。 记下安全组的名称。在启动 EC2 实例和配置 Amazon MSK 集群时，您需要引用该名称。	
创建 IAM 策略和一个 IAM 角色。	1. 要创建 IAM 策略和 IAM 角色，请按照 AWS 文档中的说明进行操作。 IAM 策略授予在 Amazon MSK 集群上创建主题和向这些主题发送数据的权限。 2. 创建 IAM 角色后，将该策略与其关联。 记下 IAM 角色名称。当您启动实例时，该角色将用作 IAM EC2 实例配置文件。	DevOps 工程师，AWS 系统管理员

Task	描述	所需技能
配置实 EC2 例。	要将 EC2 实例配置为运行 Precist CDC 并连接到 Amazon MSK，请执行以下操作： 1. 登录 AWS Marketplace，订阅 AWS 大型机现代化——适用于 IBM z/OS 的数据复制。 2. 从托管订阅中选择 AMI，然后选择启动新实例。 3. 提供其他配置详细信息，例如实例名称、实例类型、key pair、VPC 和子网。有关更多信息，请参阅 Amazon EC2 文档。 4. 在下拉列表中，选择您之前创建的安全组。 5. 在 “高级详细信息”、“IAM 实例配置文件” 下，您必须选择之前创建的角色。 6. 选择启动实例。	AWS 管理员、 DevOps 工程师

设置 Amazon MSK

Task	描述	所需技能
创建 Amazon MSK 集群。	要创建 Amazon MSK 集群，请执行以下操作： 1. 登录并打开 Amazon MSK 控制台，网址为https://console.aws.amazon.com/	AWS DevOps，云管理员

Task	描述	所需技能
	msk/。 AWS Management Console 2. 选择创建集群。 3. 对于集群创建方法，选择自定义创建，对于集群类型，选择已配置。 4. 为群集提供一个名称。 5. 根据需要更新群集设置，并保留其他设置的默认值。 6. 注意 < Kafka 版本 >。 在 Kafka 客户端设置过程中，您将需要它。 7. 选择下一步。 8. 选择您用于 Precist EC2 实例的 VPC 和子网，然后选择之前创建的安全组。 9. 在 “安全设置” 部分中，同时启用 S ASL/SCRAM 和 IAM 基于角色的身份验证。Precissift Connect CDC 使用 SASL/SCRAM（简单身份验证和安全层/加盐质询响应机制），连接亚马逊 Redshift 需要 IAM。 10选择下一步。 11要进行查看，请选择监控和代理日志传送方法。 12选择 “下一步”，然后选择 “创建集群”。 典型的预配置集群最多需要 15 分钟才能创建。创建集群	

Task	描述	所需技能
	后，其状态从 “创建中” 变为 “活动”。	
设置 SASL/SCRAM 身份验证。	要为 Amazon MSK 集群设置 SASL/SCRAM 身份验证，请执行以下操作： 1. 要在 Secrets Manager 中设置密钥，请按照AWS 文档中的说明进行操作。 2. 打开 Amazon MSK 控制台，然后选择您之前创建的 Amazon MSK 集群。 3. 选择属性选项卡。 4. 选择 “关联密钥”，选择密钥，选择您创建的密钥，然后选择 “关联密钥”。 您将看到一条类似于以下内容的成功消息： `Successfully associated 1 secret for cluster <chosen cluster name>` 5. 选择集群名称。 6. 在群集摘要中，选择查看客户机信息。 7. 记下身份验证类型 SASL/SCRAM 的私有端点连接字符串。	云架构师

Task	描述	所需技能

创建 Amazon MSK 主题。	要创建 Amazon MSK 主题，请执行以下操作：
	1. 通过运行以下命令连接到您之前创建的 EC2 实例，并安装最新更新：
	`sudo yum update -y`
	2. 通过运行以下命令安装 Java 和 Kafka 库：
	`sudo yum install -y java-11 librdkafka librdkafka-devel`
	3. 要创建名为kafka中的文件夹/home/ec2-user ，请导航到该文件夹，然后运行以下命令：
	`mkdir kafka;cd kafka`
	4. 将kafka客户端库下载到kafka文件夹，<YOUR MSK VERSION>替换为您在创建 Amazon MSK 集群时记下的 Kafka 版本：
	`wget https://archive.apache.org/dist/kafka//kafka_2.13-<YOUR MSK VERSION>.tgz`
	5. 要解压缩下载的文件，请运行以下命令，替换YOUR MSK VERSION>：
		云管理员

Task	描述	所需技能

	`tar -xzf kafka_2.13-<YOUR MSK VERSION>.tgz`
	6. 要导航到该kafka libs目录并下载 Java IAM 身份验证 Java 存档 (JAR) 文件，请运行以下命令，替换<YOUR MSK VERSION>：
	`cd kafka_2.13-<YOUR MSK VERSION>/libs`
	`wget https://github.com/aws/aws-msk-iam-auth/releases/download/v1.1.1/aws-msk-iam-auth-1.1.1-all.jarkafka`
	7. 要导航到 Kafka bin 目录并创建client.properties 文件，请运行以下命令：
	`cd /home/ec2-user/kafka/kafka_2.13-<YOUR MSK VERSION>/bin`
	`cat >client.properties`
	8. 使用以下内容更新client.properties 文件：
	`security.protocol=SASL_SSL`	

Task	描述	所需技能
	sasl.mechanism=AWS_MSK_IAM sasl.jaas.config=software.amazon.msk.auth.iam.IAMLoginModule required; sasl.client.callback.handler.class=software.amazon.msk.auth.iam.IAMClientCallbackHandler 9. 要创建 Kafka 主题，请导航到 Kafka bin 并运行以下命令，替换<kafka broker>为您在创建 Amazon MSK 集群时记下的 IAM 引导服务器私有终端节点： ./kafka-topics.sh --bootstrap-server <kafka broker> --command-config client.properties --create --replication-factor 3 –partitions 6 --topic <topic name> 10.当消息Created topic <topic name>出现时，请记下主题名称。	

在 Amazon 上配置精确应用引擎 EC2

Task	描述	所需技能

设置精确脚本以复制数据更改。	要设置 Precist Connect CDC 脚本以将更改后的数据从大型机复制到 Amazon MSK 主题，请执行以下操作：
	1. 要精确地创建文件夹名称并更改为该文件夹，请运行以下命令：
	`mkdir /home/ec2-user/precisely;cd /home/ec2-user/precisely`
	2. 要在其中创建两个精确名为 and 的文件夹`ddls`，`scripts`然后切换到该`scripts`文件夹，请运行以下命令：
	`mkdir scripts;mkdir ddls;cd scripts`
	3. 要在`scripts`文件夹`sqdata_kafka_producer.conf` 中创建名为的文件，请运行以下命令：
	`cat >sqdata_kafka_producer.conf`
	4. 使用以下内容更新`sqdata_kafka_producer.conf` 文件：
		应用程序开发人员、云架构师

Task	描述	所需技能
	builtin.features=SASL_SCRAM security.protocol=SASL_SSL sasl.mechanism=SCRAM-SHA-512 sasl.username=<User Name> sasl.password=<Password> metadata.broker.list=<SASL/SCRAM Bootstrap servers>	
	Note <SASL/SCRAM Bootstrap servers>使用您之前配置的 Amazon MSK SASL/SCRAM 代理列表进行更新。<Password> 使用你之前在 Secrets Manager 中设置的用户名和密码进行更新<User Name>。	
	5. 在scripts文件夹中创建script.sqd 文件。 cat >script.sqd 应用引擎script.sqd 用于处理源数据并将源数据复制到目标。有关 Apply	

Task	描述	所需技能

	Engine 脚本的示例，请参阅“其他信息”部分。
	6. 要切换到该ddls文件夹并为每个 Db2 表创建一个.ddl 文件，请运行以下命令：
	`cd /home/ec2-user/precisely/ddls`
	`cat >mem_details.ddl`
	`cat >mem_plans.ddl`
	例如.ddl 文件，请参阅“其他信息”部分。	

Task	描述	所需技能

生成网络 ACL 密钥。	要生成网络访问控制列表（网络 ACL）密钥，请执行以下操作：
	
	1. 要导出sqdata安装路径，请运行以下命令：
	
	`export PATH=$PATH:/usr/sbin:/opt/precisely/di/sqdata/bin`
	
	2. 要切换到/home/ec2-user目录并生成网络 ACL 密钥，请运行以下命令：
	
	`cd /home/ec2-user`
	`sqdutil keygen --force`
	
	生成公钥和私钥后，将显示以下消息：
	
	`SQDUT04I Generating a private key in file /home/ec2-user/.nacl/id_nacl`
	`SQDC017I sqdutil(pid=27344) terminated successfully`
	
	3. 记下存储在.nacl文件夹中的生成的公钥。
		AWS 云架构师 DevOps

准备大型机源环境

Task	描述	所需技能
在 ISPF 屏幕中配置默认值。	要在交互式系统生产率工具 (ISPF) 中配置默认设置，请按照 Precist 文档中的说明进行操作。	大型机系统管理员
配置控制器守护程序。	要配置控制器守护程序，请执行以下操作： 1. 在 SQData z/OS 主菜单屏幕上，选择选项 2。 2. 在 “将守护程序添加到列表” 屏幕上的 “守护程序名称” 字段中，输入守护程序的名称，然后按 Enter。	大型机系统管理员
配置发布者。	要配置发布器，请执行以下操作： 1. 在 SQData z/OS 主菜单屏幕上，选择选项 3。这会将您带到 “捕获/发布者摘要” 屏幕。 2. 选择添加 CAB 文件的选项。这将带您进入将 CAB 文件添加到列表屏幕。 3. 在 “名称” 字段中，输入 CAB 文件的名称。对于 Db2，输入 “类型为”。D 4. 按 Enter 键。这将带您进入新建 Db2 Capture CAB 文件屏幕。 5. 在 zfS 目录字段中，指定存储装载点。	大型机系统管理员

Task	描述	所需技能
	6. 按 Enter 键保存并继续。	

更新守护程序配置文件。	要更新控制器守护程序配置文件中的发布者详细信息，请执行以下操作：
	1. 在 SQData z/OS 主菜单屏幕上，选择选项 2。
	2. 在您创建的守护程序S附近输入以查看守护程序的详细信息。
	3. 输入1，然后按 Enter 键编辑代理文件。
	4. 添加您的 CAB 文件详细信息。以下示例显示名为的 CAB 文件的详细信息DB2ZTOMSK 。请改用您的大型机用户 ID。<userid>
	`ÝDB2ZTOMSK¨ type=capture cab=/u/<userid>/sqdata/DB2ZTOMSK.cab`
	5. 按 F3。
	6. 输入2可编辑 ACL 文件。将您的userid添加到acl配置文件中，如以下示例所示：
	`Ýacls¨ prod=admin,<userid>`
	7. 按 F3 保存并退出。
		大型机系统管理员

Task	描述	所需技能
创建启动控制器守护程序的作业。	要创建作业，请执行以下操作： 1. 在选项中，输入G。 2. 输入任务卡、作业和过程库以及Db2 load库详细信息。 3. 输入网络 ACL 文件详细信息，然后输入选项 2，在指定的作业库中生成作业控制语言 (JCL) 文件。	大型机系统管理员
生成捕获发布者 JCL 文件。	要生成捕获发布者 JCL 文件，请执行以下操作： 1. 在 SQData z/OS 主菜单屏幕上，选择选项 3。这会将您带到 “捕获/发布者摘要” 屏幕。 2. 在 CAB 文件S旁边输入以将其选中。这会将您带到 “Db2 捕获/发布者详细信息” 屏幕。 3. 在选项G中，输入生成capture/publisher 作业的选项。 4. 输入 JOB 卡、作业和过程库以及 Db2 加载库的详细信息。 5. 要创建作业，请选择选项 4。该作业是在作业库中指定的作业库中创建的。	大型机系统管理员

Task	描述	所需技能

查看并更新 CDC。	1. 通过运行以下查询来检查 Db2 表的DATACAPTURE 标志，更改<table name>为 Db2 表的名称：
	`SELECT DATACAPTU RE FROM SYSIBM.SY STABLES WHERE NAME='<table name>';`
	确认结果显示DATACAPTU RE 为Y。
	2. 如果不DATACAPTU RE 是Y，请运行以下查询以在 Db2 表上启用 CDC，<table name>并更改为您的 Db2 表名：
	`ALTER TABLE <table name> DATA CAPTURE CHANGES;`
		大型机系统管理员

提交 JCL 文件。	提交您在前面步骤中配置的以下 JCL 文件： • 启动控制器守护程序的 JCL 文件 • 开始捕获和发布的 JCL 文件 提交 JCL 文件后，您可以精确地在 EC2 实例上启动 Apply Engine。	大型机系统管理员

运行并验证 CDC

Task	描述	所需技能

启动应用引擎并验证 CDC。	要在 EC2 实例上启动应用引擎并验证 CDC，请执行以下操作：
	1. 要连接到 EC2 实例，请按照AWS 文档中的说明进行操作。
	2. 切换到包含该`script.sqd` 文件的目录：
	`cd /home/ec2-user/precisely/scripts`
	3. 要启动应用引擎，请运行以下`sqdeng`启动命令：
	`sqdeng -s script.sqd --identity=/home/ec2-user/.nacl/id_nacl`
	Apply Engine 将开始等待来自大型机来源的更新。
	4. 要测试 CDC，请在 Db2 表中插入或更新一些记录。
	5. 验证 Apply Engine 日志是否显示捕获并写入目标的记录数。
		云架构师、应用程序开发人员

验证 Amazon MSK 主题上的记录。	要阅读来自 Kafka 主题的消息，请执行以下操作： 1. 要更改到 EC2 实例上 Kafka 客户端安装路径	应用程序开发人员、云架构师

Task	描述	所需技能

	的bin目录，请运行以下命令，<Kafka version>替换为您的版本：
	
	`cd /home/ec2-user/kafka/kafka_2.13-<Kafka version>/bin`
	
	2. 要验证在 Kafka 主题中以消息形式写入的 Db2 CDC，请运行以下命令，将<kafka broker>和<Topic Name>替换为之前创建的主题：
	
	`./kafka-console-consumer.sh --bootstrap-server <kafka broker>:9098 --topic <Topic Name> --from-beginning --consumer.config client.properties`
	
	3. 验证消息是否与 Db2 表中更新的记录数相匹配。	

将大型机变更数据存储在 Amazon Redshift 无服务器数据仓库中

Task	描述	所需技能
设置 Amazon Redshift Serverless。	要创建 Amazon Redshift 无服务器数据仓库，请按照文档中的说明进行操作。AWS	数据工程师

Task	描述	所需技能
	在 Amazon Redshift Serverless 控制面板上，验证命名空间和工作组是否已创建并可用。对于此示例模式，该过程可能需要 2–5 分钟。	

Task	描述	所需技能
设置直播摄取所需的 IAM 角色和信任策略。	要设置来自亚马逊 MSK 的 Amazon Redshift Serverless 直播提取，请执行以下操作： 1. 为亚马逊 Redshift 创建访问亚马逊 MSK 的 IAM 策略。 `[region]`使用 AWS 区域 适用于 Amazon MSK 的、`[account-id]` 您的 AWS 账户 ID 和 `[msk-cluster-name]` Amazon MSK 集群名称替换，运行以下代码： `{"Version": "2012-10-17","Statement": [{"Sid": "MSKIAMpolicy","Effect": "Allow","Action": ["kafka-cluster:ReadData","kafka-cluster:DescribeTopic","kafka-cluster:Connect"],"Resource": ["arn:aws:kafka:[region]:[account-id]:cluster/[msk-cluster-name]/*","arn:aws:kafka:[region]:[account-id]:topic/[msk-cluster-name]/*"]},{"Effect":`	数据工程师

Task	描述	所需技能
	`"Allow","Action":["kafka-cluster:AlterGroup","kafka-cluster:DescribeGroup"],"Resource":["arn:aws:kafka:[region]:[account-id]:group/[msk-cluster-name]/*"]}]}` 您可以在 Amazon MSK 控制台上找到集群名称和亚马逊资源名称 (ARN)。在控制台上，选择集群摘要，然后选择 ARN。 2. 要创建 IAM 角色并附加策略，请按照AWS 文档中的说明进行操作。 3. 要将 IAM 角色附加到 Amazon Redshift 无服务器命名空间，请执行以下操作： a. 登录控制台，然后在上打开亚马逊 Redshift 控制台。https://console.aws.amazon.com/redshiftv2/ b. 选择 Serverless 控制面板。 c. 选择 “命名空间”。 d. 选择 “安全和加密” 选项卡。 e. 选择权限，然后附加您创建的 IAM 角色。	

Task	描述	所需技能
	4. 在您的 Amazon Redshift 无服务器安全组中，创建一条包含以下详细信息的入站规则： • 对于类型，选择自定义 TCP。 • 对于协议，选择 TCP。 • 对于端口范围，请选择 9098、919 8。 • 对于来源，选择 Amazon MSK 安全组。 5. 在您的 Amazon MSK 安全组中，创建一条包含以下详细信息的入站规则： • 对于类型，选择自定义 TCP。 • 对于协议，选择 TCP。 • 对于端口范围，请选择 9098、919 8。 • 对于来源，选择 Amazon Redshift 安全组。 此模式使用端口进行 Amazon Redshift 和 Amazon MSK 配置的 IAM 身份验证。有关更多信息，请参阅AWS 文档（步骤 2）。 6. 为 Amazon Redshift 无服务器工作组开启增强型 VPC 路由。有关更多信息，请参阅 AWS 文档。	

Task	描述	所需技能
将 Amazon Redshift Serverless 连接到亚马逊 MSK。	要连接亚马逊 MSK 主题，请在 Amazon Redshift Serverless 中创建外部架构。在 Amazon Redshift 查询编辑器 v2 中，运行以下 SQL 命令，`'iam_role_arn'` 替换为你之前创建的角色，将 `'MSK_cluster_arn '`替换为集群的 ARN。 `CREATE EXTERNAL SCHEMA member_schema FROM MSK IAM_ROLE 'iam_role _arn' AUTHENTICATION iam URI 'MSK_cluster_arn';`	迁移工程师

Task	描述	所需技能
创建物化视图。	要在 Amazon Redshift Serverless 中使用来自亚马逊 MSK 主题的数据，请创建一个物化视图。在 Amazon Redshift 查询编辑器 v2 中，运行以下 SQL 命令，<MSK_Topic_name> 替换为亚马逊 MSK 主题的名称。 `CREATE MATERIALIZED VIEW member_view AUTO REFRESH YES AS SELECT kafka_partition, kafka_offset, refresh_time, json_parse(kafka_value) AS Data FROM member_schema.<MSK_Topic_name> WHERE CAN_JSON_PARSE(kafka_value);`	迁移工程师

Task	描述	所需技能

在 Amazon Redshift 中创建目标表。	亚马逊 Redshift 表为提供了输入。 QuickSight此模式使用表member_dtls 和member_plans ，它们与大型机上的源 Db2 表相匹配。
	
	要在 Amazon Redshift 中创建这两个表，请在亚马逊 Redshift 查询编辑器 v2 中运行以下 SQL 命令：
	
	`-- Table 1: members_dtls`
	`CREATE TABLE members_dtls (`
	`memberid INT ENCODE AZ64,`
	`member_name VARCHAR(100) ENCODE ZSTD,`
	`member_type VARCHAR(50) ENCODE ZSTD,`
	`age INT ENCODE AZ64,`
	`gender CHAR(1) ENCODE BYTEDICT,`
	`email VARCHAR(100) ENCODE ZSTD,`
	`region VARCHAR(50) ENCODE ZSTD`
	`) DISTSTYLE AUTO;`
	
	`-- Table 2: member_plans`
	`CREATE TABLE member_plans (`
	`memberid INT ENCODE AZ64,`
	`medical_plan CHAR(1) ENCODE BYTEDICT,`
		迁移工程师

Task	描述	所需技能
	dental_plan CHAR(1) ENCODE BYTEDICT, vision_plan CHAR(1) ENCODE BYTEDICT, preventive_immuniz ation VARCHAR(50) ENCODE ZSTD) DISTSTYLE AUTO;	
在 Amazon Redshift 中创建存储过程。	这种模式使用存储过程将源大型机的更改数据 (INSERT、UPDATE、DELETE) 同步到目标 Amazon Redshift 数据仓库表，以便在中进行分析。 QuickSight 要在 Amazon Redshift 中创建存储过程，请使用查询编辑器 v2 运行存储库中的存储过程代码。 GitHub	迁移工程师

Task	描述	所需技能
从流式实例化视图中读取并加载到目标表。	存储过程从流式实例化视图读取数据更改，并将数据更改加载到目标表。要运行存储过程，请使用以下命令： `call SP_Members_Load();` 您可以使用 Amazon EventBridge 在 Amazon Redshift 数据仓库中安排任务，以便根据您的数据延迟要求调用此存储过程。 EventBridge 以固定的间隔运行作业。要监视先前对该过程的调用是否完成，可能需要使用诸如AWS Step Functions状态机之类的机制。有关更多信息，请参阅以下资源： • 创建按计划运行的 Amazon EventBridge 规则 • 使用 AWS Step Functions 和 Amazon Redshift 数据 API 加快 ELT 流程的编排。 另一种选择是使用 Amazon Redshift 查询编辑器 v2 来安排刷新。有关更多信息，请参阅使用查询编辑器 v2 安排查询。	迁移工程师

连接 QuickSight 亚马逊 Redshift 中的数据

Task	描述	所需技能
设置 QuickSight。	要进行设置 QuickSight，请按照AWS 文档中的说明进行操作。	迁移工程师
在 QuickSight 和亚马逊 Redshift 之间建立安全连接。	要在 QuickSight 和 Amazon Redshift 之间建立安全连接，请执行以下操作 1. 要授权来自亚马逊 Redshift 的连接， QuickSight 请打开亚马逊 Redshift 控制台，然后在亚马逊 Redshift 安全组中添加入站规则。该规则应允许从您设置的 CIDR 范围向端口 5439（默认 Redshift 端口）发送流量。 QuickSight有关 AWS 区域 及其的 IP 地址的列表，AWS 区域 请参阅支持 QuickSight。 2. 在 Amazon Redshift 控制台上，选择 “工作组”、“数据访问”、“网络和安全”，然后启用 “可公开访问”。	迁移工程师
为创建数据集 QuickSight。	要 QuickSight 从 Amazon Redshift 创建数据集，请执行以下操作： 1. 在 QuickSight 控制台的导航窗格中，选择数据集。 2. 在数据集页面上，选择新数据集。	迁移工程师

Task	描述	所需技能
	3. 选择 Redshift 手动连接。 4. 在 “新建 Redshift 数据源” 窗口中，输入连接信息： • 在数据源名称中，输入 Amazon Redshift 数据源的名称。 • 对于数据库服务器，请输入 Amazon Redshift 集群的终端节点。您可以从 Amazon Redshift Serverless 控制面板上集群工作组一般信息部分的终端节点字段中获取终端节点值。服务器地址是冒号之前端点的第一部分，如以下示例所示： `mfdata-insights.NNNNNNNN.us-east-1.redshift-serverless.amazonaws.com:5439/dev` • 对于端口，输入5439（Amazon Redshift 的默认端口）。 • 输入数据库的名称（在端点的斜杠之后）。在本例中，数据库名称为dev。 • 在用户名和密码中，输入 Amazon Redshift 数据库的用户名和密码。 5. 选择 “验证连接”。如果成功，您应该会看到一个绿色复选标记，表示已验证。如	

Task	描述	所需技能
	果验证失败，请参阅 “故障排除” 部分。 6. 选择创建数据来源。	
加入数据集。	要在中创建分析 QuickSight，请按照AWS 文档中的说明将这两个表格连接起来。 在 “联接配置” 窗格中，为 “联接类型” 选择 “左”。在 “联接子句” 下，使用`memberid from member_plans = memberid from members_details` 。	迁移工程师

使用 Amazon Q 从大型机数据中获取业务见解 QuickSight

Task	描述	所需技能
在中设置 Amazon Q QuickSight。	要在 QuickSight 生成式 BI 中设置 Amazon Q 功能，请按照AWS 文档中的说明进行操作。	迁移工程师
分析大型机数据并构建可视化仪表板。	要在中分析和可视化您的数据 QuickSight，请执行以下操作： 1. 要创建大型机数据分析，请按照AWS 文档中的说明进行操作。对于数据集，请选择您创建的数据集。 2. 在分析页面上，选择生成视觉对象。	迁移工程师

Task	描述	所需技能
	3. 在创建用于分析的主题窗口中，选择更新现有主题。 4. 在选择主题下拉列表中，选择您之前创建的主题。 5. 选择主题链接。 6. 链接主题后，选择 “构建视觉对象” 以打开 Amazon Q “构建可视化” 窗口。 7. 在提示栏中。写下你的分析问题。用于此模式的示例问题如下： • 按地区显示成员分布 • 显示按年龄划分的成员分布 • 按性别显示成员分布 • 按计划类型显示成员分布 • 显示未完成预防性免疫的会员 输入问题后，选择 “构建”。Amazon Q in QuickSight 创作了视觉效果。 8. 要将视觉对象添加到可视化仪表板，请选择添加至分析。 完成后，您可以发布仪表板以与组织中的其他人共享。有关示例，请参阅 “其他信息” 部分中的大型机可视化仪表板。	

使用 Amazon Q QuickSight 从大型机数据中创建数据故事

Task	描述	所需技能
创建数据故事。	创建数据故事以解释先前分析的见解，并提出建议，以增加成员的预防性免疫接种： 1. 要创建数据故事，请按照AWS 文档中的说明进行操作。 2. 对于数据故事提示，请使用以下内容： `Build a data story about Region with most numbers of members. Also show the member distribution by medical plan, vision plan, dental plan. Recommend how to motivate members to complete immunization. Include 4 points of supporting data for this pattern.` 您还可以创建自己的提示，为其他业务见解生成数据故事。 3. 选择 “添加视觉对象”，然后添加与数据故事相关的视觉对象。对于这种模式，请使用您之前创建的视觉效果。	迁移工程师

Task	描述	所需技能
	4. 选择构建。 5. 有关数据故事输出的示例，请参阅 “其他信息” 部分中的数据故事输出。	
查看生成的数据故事。	要查看生成的数据故事，请在数据故事页面上选择该故事。	迁移工程师
编辑生成的数据故事。	要更改数据故事中的格式、布局或视觉效果，请按照AWS 文档中的说明进行操作。	迁移工程师
分享数据故事。	要共享数据故事，请按照AWS 文档中的说明进行操作。	迁移工程师

故障排除

事务	解决方案
QuickSight 对于 Amazon Redshift 数据集的创建，`Validate Connection` 已经失败了。	1. 确认连接到 Amazon Redshift Serverless 实例的安全组允许来自与您设置的区域关联的 IP 地址范围的入站流量。 QuickSight 2. 确认部署 Amazon Redshift Serverless 的 VPC 已公开可用。 3. 确认您使用的 Amazon Redshift 用户名和密码正确。您可以在亚马逊 Redshift 控制台上重置用户名和密码。
尝试在 EC2 实例上启动 Apply 引擎时会返回以下错误： `-bash: sqdeng: command not found`	通过运行以下命令导出`sqdata`安装路径： `export PATH=$PATH:/usr/sbin:/opt/precisely/di/sqdata/bin`

事务	解决方案
尝试启动 Apply Engine 时会返回以下连接错误之一： • `SQDD018E Cannot connect to transfer socket(rc==0x18468). Agent:<Agent Name > Socket:/u /./sqdata/.DB2ZTOMSK.cab.data` • `SQDUR06E Error opening url cdc:// <VPC end point name>:2626/ DB2ZTOMSK/DB2ZTOMSK : errno:112 8 (Unknown error 1128)`	检查大型机线轴以确保控制器守护程序作业正在运行。

相关资源

- 通过使用 AWS Mainframe Modernization 和 Amazon Q QuickSight（模式）生成见解
- 使用 AWS Mainframe Modernization 和 Amazon Q 生成数据见解 QuickSight（演示）
- AWS Mainframe Modernization -适用于 IBM z/OS 的数据复制
- Amazon Redshift 将数据提取流式传输到物化视图

其他信息

.ddl 文件示例

成员_details.ddl

```
CREATE TABLE MEMBER_DTLS (
memberid INTEGER NOT NULL,
member_name VARCHAR(50),
member_type VARCHAR(20),
age INTEGER,
gender CHAR(1),
email VARCHAR(100),
region VARCHAR(20)
);
```

member_plans.ddl

```
CREATE TABLE MEMBER_PLANS (
memberid INTEGER NOT NULL,
medical_plan CHAR(1),
dental_plan CHAR(1),
vision_plan CHAR(1),
preventive_immunization VARCHAR(20)
);
```

.sqd 文件示例

<kafka topic name>替换为您的 Amazon MSK 主题名称。

script.sqd

```
-- Name: DB2ZTOMSK: DB2z To MSK JOBNAME DB2ZTOMSK;REPORT EVERY 1;OPTIONS
 CDCOP('I','U','D');-- Source Descriptions
JOBNAME DB2ZTOMSK;
REPORT EVERY 1;
OPTIONS CDCOP('I','U','D');

-- Source Descriptions
BEGIN GROUP DB2_SOURCE;
DESCRIPTION DB2SQL /var/precisely/di/sqdata/apply/DB2ZTOMSK/ddl/mem_details.ddl AS
 MEMBER_DTLS;
DESCRIPTION DB2SQL /var/precisely/di/sqdata/apply/DB2ZTOMSK/ddl/mem_plans.ddl AS
 MEMBER_PLANS;
END GROUP;
-- Source Datastore
DATASTORE cdc://<zos_host_name>/DB2ZTOMSK/DB2ZTOMSK
OF UTSCDC
AS CDCIN
DESCRIBED BY GROUP DB2_SOURCE ;
-- Target Datastore(s)
DATASTORE 'kafka:///<kafka topic name>/key'
OF JSON
AS TARGET
DESCRIBED BY GROUP DB2_SOURCE;
PROCESS INTO TARGET
SELECT
{
REPLICATE(TARGET)
```

```
}
FROM CDCIN;
```

大型机可视化仪表板

以下数据视觉效果由 Amazon Q QuickSight 为分析问题创建`show member distribution by region`。

以下数据视觉效果由 Amazon Q QuickSight 为该问题创建`show member distribution by Region who have not completed preventive immunization, in pie chart`。

数据故事输出

以下屏幕截图显示了 Amazon Q QuickSight 为提示创建的数据故事的各个部分`Build a data story about Region with most numbers of members. Also show the member distribution by age, member distribution by gender. Recommend how to motivate members to complete immunization. Include 4 points of supporting data for this pattern`。

在导言中，数据故事建议选择成员最多的区域，以便从免疫工作中获得最大的影响。

数据故事提供了对这四个区域的成员人数的分析。东北、西南和东南地区的成员最多。

数据故事按年龄对成员进行了分析。

数据报道的重点是中西部的免疫工作。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS 大型机现代化和 Amazon Q 生成数据见解 QuickSight

由 Shubham Roy (AWS)、Roshna Razack (AWS) 和 Santosh Kumar Singh (AWS) 创作

摘要

如果您的组织在大型机环境中托管关键业务数据，那么从这些数据中获取见解对于推动增长和创新至关重要。通过解锁大型机数据，您可以构建更快、更安全、更可扩展的商业智能，从而加快 Amazon Web Services ()AWS云中数据驱动的决策、增长和创新。

这种模式提供了一种解决方案，用于通过使用 BMC 和 Amazon Q 中的 “AWS Mainframe Modernization 文件传输” 来生成业务见解并根据大型机数据创建可共享的叙述。 QuickSight通过使用 BMC 的 AWS 大型机现代化文件传输，将大型机数据集传输到亚马逊简单存储服务 (Amazon S 3) Simple Storage Service。 AWS Lambda 函数格式化并准备大型机数据文件以加载到 Amazon QuickSight。

在中提供数据后 QuickSight，您可以使用带有 Amazon Q 的自然语言提示 QuickSight 来创建数据摘要、提问和生成数据故事。您不必编写 SQL 查询或学习商业智能 (BI) 工具。

业务背景

这种模式为大型机数据分析和数据洞察用例提供了解决方案。使用该模式，您可以为公司的数据构建可视化仪表板。为了演示解决方案，这种模式使用了一家医疗保健公司，该公司为其在美国的成员提供医疗、牙科和视力计划。在此示例中，成员人口统计和计划信息存储在大型机数据集中。可视化仪表板显示以下内容：

- 按地区划分的成员分布
- 按性别分列的成员分布
- 按年龄划分的成员分布
- 按计划类型划分的成员分布
- 尚未完成预防性免疫的会员

创建仪表板后，您将生成一个数据故事，解释先前分析的见解。数据故事为增加完成预防性免疫接种的成员人数提供了建议。

先决条件和限制

先决条件

- 活跃的 AWS 账户

- 包含业务数据的大型机数据集
- 有权在大型机上安装文件传输代理

限制

- 您的大型机数据文件应采用支持的 QuickSight文件格式之一。有关支持的文件格式列表，请参阅 Amazon QuickSight 文档。

  此模式使用 Lambda 函数将大型机文件转换为支持的格式。 QuickSight

架构

下图显示了通过使用 BMC 和 Amazon Q 中的 AWS Mainframe Modernization 文件传输从大型机数据生成业务见解的架构。 QuickSight

图表显示了以下工作流：

1. 使用带有 BMC AWS Mainframe Modernization 的文件传输功能将包含业务数据的大型机数据集传输到 Amazon S3。
2. Lambda 函数将文件传输目标 S3 存储桶中的文件转换为逗号分隔值 (CSV) 格式。
3. Lambda 函数将转换后的文件发送到源数据集 S3 存储桶。
4. 文件中的数据由摄取。 QuickSight
5. 用户访问中的数据 QuickSight。您可以使用 Amazon Q 通过自然语言提示与数据进行交互。 QuickSight

工具

Amazon Web Services

- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Mainframe Modernization 使用 BMC 进行文件传输可将大型机数据集转换并传输到 Amazon S3，用于大型机现代化、迁移和增强用例。
- Amazon QuickSight 是一项云规模的 BI 服务，可帮助您在单个控制面板中可视化、分析和报告数据。这种模式使用了 Amazon Q 中的生成式商业智能功能 QuickSight。

- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

最佳实践

- 当您使用 BMC 和 Lambda 函数为 AWS Mainframe Modernization 文件传输创建 AWS Identity and Access Management (IAM) 角色时，请遵循最低权限原则。
- 确保您的源数据集支持的数据类型 QuickSight。如果您的源数据集包含不支持的数据类型，请将其转换为支持的数据类型。有关不支持的大型机数据类型以及如何将其转换为 Amazon Q 支持的数据类型的信息 QuickSight，请参阅相关资源部分。

操作说明

使用 B AWS Mainframe Modernization MC 设置文件传输

Task	描述	所需技能
安装文件传输代理。	要在大型机上安装 AWS Mainframe Modernization 文件传输代理，请按照AWS 文档中的说明进行操作。	大型机系统管理员
为大型机文件传输创建 S3 存储桶。	创建 S3 存储桶以存储 BMC AWS Mainframe Modernization 文件传输的输出文件。在架构图中，这是文件传输目标存储桶。	迁移工程师
创建数据传输端点。	1. 创建 S3 存储桶以暂存输入的大型机文件，以便使用 BM AWS Mainframe Modernization C 进行文件传输。 2. 要创建大型机数据传输端点，请按照AWS 文档中的说明进行操作。	AWS 大型机现代化专家

转换大型机文件扩展名以进行集成 QuickSight

Task	描述	所需技能
创建 S3 存储桶。	为 Lambda 函数创建 S3 存储桶，将转换后的大型机文件从源存储桶复制到最终目标存储桶。	迁移工程师
创建一个 Lambda 函数。	要创建用于更改文件扩展名并将大型机文件复制到目标存储桶的 Lambda 函数，请执行以下操作： 1. 登录并导航到 AWS Lambda 控制台。 AWS Managemen t Console 2. 选择 “创建函数”，然后选择 “从头开始创作”。 3. 在函数名称中，输入函数的名称。 4. 在 “运行时间” 下拉列表中，选择 Python.3.X。 5. 展开更改默认执行角色，然后选择使用基本 Lambda 权限创建新角色。 6. 选择创建函数。 7. 选择 “代码” 选项卡，然后粘贴 “其他信息” 部分中提供的 `S3CopyLambda.py` Python 代码。Python 代码是在微软 Visual Studio 集成开发环境 (IDE) 中使用 Amazon Q Developer 生成的。	迁移工程师

Task	描述	所需技能
	8. 编辑为您之前创建的 S3 存储桶的名称和`change destination_file_key` 大型机文件名。`destination_bucket_name` 9. 部署 Lambda 函数。	
创建 Amazon S3 触发器来调用 Lambda 函数。	要配置调用 Lambda 函数的触发器，请执行以下操作： 1. 在 Lambda 控制台上，打开函数页面。 2. 选择 Lambda 函数。 3. 在函数概述中，选择添加触发器。 4. 在“触发器配置”下拉列表中，选择 S3。 5. 在 Bucket 字段中，输入您的源存储桶的名称。 6. 在“事件类型”下拉列表中，选择“所有对象创建事件”。 7. 选中“我确认不建议对输入和输出使用相同的 S3 存储桶”复选框，然后选择“添加”。 有关更多信息，请参见教程：使用 Amazon S3 触发器调用 Lambda函数。	迁移主管

Task	描述	所需技能
为 Lambda 函数提供 IAM 权限。	Lambda 函数需要有 IAM 权限才能访问文件传输目标和源数据集 S3 存储桶。通过允许文件传输目标 S3 存储桶`s3:GetObject` 和`s3:PutObject` 访问源数据集 S3 存储桶的`s3:DeleteObject` 权限来更新与 Lambda 函数执行角色相关的策略。 有关更多信息，请参阅教程：使用 Amazon S3 触发器调用 Lambda 函数中的创建权限策略部分。	迁移主管

定义大型机数据传输任务

Task	描述	所需技能
创建传输任务以将大型机文件复制到 S3 存储桶。	要创建大型机文件传输任务，请按照AWS Mainframe Modernization 文档中的说明进行操作。 **Note** 将源代码页编码指定为 IBM1047，将目标代码页编码指定为 UTF-8。	迁移工程师
验证转移任务。	要验证数据传输是否成功，请按照AWS Mainframe Modernization 文档中的说明进	迁移主管

Task	描述	所需技能
	行操作。确认大型机文件位于文件传输目标 S3 存储桶中。	
验证 Lambda 复制函数。	验证 Lambda 函数是否已启动，并且文件已使用.csv 扩展名复制到源数据集 S3 存储桶。 由 Lambda 函数创建的.csv 文件是的输入数据文件。QuickSight有关示例数据，请参阅 “附件” 部分中的`Sample-data-member-healthcare-APG` 文件。	迁移主管

Connect QuickSight 连接到大型机数据

Task	描述	所需技能
设置 QuickSight。	要进行设置 QuickSight，请按照AWS 文档中的说明进行操作。	迁移主管
为创建数据集 QuickSight。	要为创建数据集 QuickSight，请按照AWS 文档中的说明进行操作。输入数据文件是在定义大型机数据传输任务时创建的转换后的大型机文件。	迁移主管

使用 Amazon Q 从大型机数据中获取业务见解 QuickSight

Task	描述	所需技能
在中设置 Amazon Q QuickSight。	此功能需要企业版。要在中设置 Amazon Q QuickSight，请执行以下操作： 1. 要获取 Amazon Q 附加组件，请按照AWS 文档中的步骤 1：获取 Q 附加组件的说明进行操作。 2. 要使用 Amazon Q 中的生成式 BI 功能，请升级您的用户账户。按照AWS 文档中的说明进行操作。 3. 使用您之前创建的数据集创建 Amazon Q 主题。按照AWS 文档中的说明进行操作。 4. 要配置主题元数据使其适合自然语言，请按照文档中的说明进行操作。AWS	迁移主管
分析大型机数据并构建可视化仪表板。	要在中分析和可视化您的数据 QuickSight，请执行以下操作： 1. 要创建大型机数据分析，请按照AWS 文档中的说明进行操作。对于数据集，请选择在上一步中创建的数据集。 2. 在分析页面上，选择生成视觉对象。 3. 在创建用于分析的主题窗口中，选择更新现有主题。	迁移工程师

Task	描述	所需技能
	4. 在选择主题下拉列表中，选择您之前创建的主题。 5. 选择主题链接。 6. 链接主题后，选择 “构建视觉对象” 以打开 Amazon Q “构建可视化” 窗口。 7. 在提示栏中。写下你的分析问题。用于此模式的示例问题如下： • 按地区显示成员分布 • 显示按年龄划分的成员分布 • 按性别显示成员分布 • 按计划类型显示成员分布 • 显示成员未完成预防性免疫接种 输入问题后，选择 “构建”。Amazon Q in QuickSight 创建了视觉效果。 8. 要将视觉对象添加到可视化仪表板，请选择添加至分析。 完成后，您可以发布仪表板以与组织中的其他人共享。有关示例，请参阅 “其他信息” 部分中的大型机可视化仪表板。	

使用 Amazon Q QuickSight 从大型机数据中创建数据故事

Task	描述	所需技能
创建数据故事。	创建数据故事以解释先前分析的见解，并提出建议，以增加成员的预防性免疫接种： 1. 要创建数据故事，请按照AWS 文档中的说明进行操作。 2. 对于数据故事提示，请使用以下内容： `Build a data story about Region with most numbers of members. Also show the member distribution by medical plan, vision plan, dental plan. Recommend how to motivate members to complete immunization. Include 4 points of supporting data for this pattern.` 您还可以创建自己的提示，为其他业务见解生成数据故事。 3. 选择 “添加视觉对象”，然后添加与数据故事相关的视觉对象。对于这种模式，请使用您之前创建的视觉效果。	迁移工程师

Task	描述	所需技能
	4. 选择构建。 5. 有关数据故事输出的示例，请参阅 “其他信息” 部分中的数据故事输出。	
查看生成的数据故事。	要查看生成的数据故事，请按照AWS 文档中的说明进行操作。	迁移主管
编辑生成的数据故事。	要更改数据故事中的格式、布局或视觉效果，请按照AWS 文档中的说明进行操作。	迁移主管
分享数据故事。	要共享数据故事，请按照AWS 文档中的说明进行操作。	迁移工程师

故障排除

事务	解决方案
找不到在 “使用 BMC 进行文件传输” 中的 “创建传输任务” 的数据集搜索条件中输入的大型机 AWS Mainframe Modernization 文件或数据集。	1. 首先，在 “使用 BMC 传输” 控制台上选择 “数据 AWS Mainframe Modernization 传输端点”，检查连接。如果上次心跳时间超过两分钟，则文件传输连接尚未建立。如果在大型机上运行的代理的最后一次心跳时间少于 2 分钟，则代理连接成功。继续执行步骤 2。 2. 检查 AWS Secrets Manager 设置。必须在 Secrets Manager 中将密钥配置为密钥`userId`（大写字母 I），其值为大型机的用户 ID，密钥`password`的值为大型机密码的值。`userId`和`password`密钥区分大小写，必须按原样输入。

相关资源

要将 PACKED-DECIMAL (COMP-3) 或二进制（COMP 或 COMP-4）等大型机数据类型转换为支持的数据类型 QuickSight，请参阅以下模式：

- 使用 Python 将 EBCDIC 数据转换并解压缩为 ASCII AWS
- 在 Amazon S3 中使用以下命令将大型机文件从 EBCDIC 格式转换为字符分隔的 ASCII 格式 AWS Lambda

其他信息

S3 CopyLambda .py

以下 Python 代码是通过在 IDE 中使用 Amazon Q Developer 的提示生成的：

```
#Create a lambda function triggered by S3. display the S3 bucket name and key
import boto3
s3 = boto3.client('s3')
def lambda_handler(event, context):
print(event)
bucket = event['Records'][0]['s3']['bucket']['name']
key = event['Records'][0]['s3']['object']['key']
print(bucket, key)
#If key starts with object_created, skip copy, print "copy skipped". Return lambda with
 key value.
if key.startswith('object_created'):
print("copy skipped")
return {
'statusCode': 200,
'body': key
}
# Copy the file from the source bucket to the destination bucket.
 Destination_bucket_name = 'm2-filetransfer-final-opt-bkt'. Destination_file_key =
 'healthdata.csv'
copy_source = {'Bucket': bucket, 'Key': key}
s3.copy_object(Bucket='m2-filetransfer-final-opt-bkt', Key='healthdata.csv',
 CopySource=copy_source)
print("file copied")
#Delete the file from the source bucket.
s3.delete_object(Bucket=bucket, Key=key)
return {
'statusCode': 200,
```

```
'body': 'Copy Successful'
}
```

大型机可视化仪表板

以下数据视觉对象是由 Amazon Q QuickSight 为分析问题创建的`show member distribution by region`。

以下数据视觉效果由 Amazon Q QuickSight 为该问题创建`show member distribution by Region who have not completed preventive immunization, in pie chart`。

数据故事输出

以下屏幕截图显示了 Amazon Q QuickSight 为提示创建的数据故事的各个部分 `Build a data story about Region with most numbers of members. Also show the member distribution by medical plan, vision plan, dental plan. Recommend how to motivate members to complete immunization. Include 4 points of supporting data.`

在导言中，数据故事建议选择成员最多的区域，以便从免疫工作中获得最大的影响。

该数据报道分析了排名前三个地区的成员人数，并将西南地区列为专注于免疫工作的领先地区。

> **Note**
>
> 西南和东北地区各有八个成员。但是，西南地区有更多成员没有完全接种疫苗，因此它更有可能从提高免疫率的举措中受益。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

将 Stonebranch Universal Controller 与 AWS Mainframe Modernization 集成

由 Vaidy Sankaran (AWS)、Robert Lemieux (Stonebranch)、Huseyin Gomleksizoglu (Stonebranch) 和 Pablo Alonso Prieto (AWS) 创建

摘要

此模式解释了如何将Stonebranch Universal Automation Center (UAC) 工作负载编排与Amazon Web Services (AWS) Mainframe Modernization 服务集成。AWS Mainframe Modernization 将主机应用程序迁移至 Amazon Web Services Cloud 并对其进行现代化改造。它提供了两种模式：采用 Micro Focus Enterprise 技术的 AWS 大型机现代化改造平台和采用 AW S Blu Age 的 AWS 大型机现代化自动重构。

Stonebranch UAC 是实时 IT 自动化和编排平台。UAC 旨在跨混合 IT 系统（从本地到 AWS） 自动执行和协调作业、活动和工作流。使用主机系统的企业客户正在过渡至以云为中心的现代化基础架构和应用程序。Stonebranch 的工具和专业服务有助于将现有调度程序和自动化功能迁移至 AWS Cloud。

当您使用 AWS Mainframe Modernization 服务将主机程序迁移至 Amazon Web Services Cloud 或对其进行现代化改造时，您可以使用此集成来自动执行批量调度、提高灵活性、改善维护并降低成本。

此模式提供了将 Stonebranch 调度程序与迁移到 AWS 大型机现代化服务 M icro Focus Enterprise 运行时的大型机应用程序集成的说明。此模式适用于解决方案架构师、开发人员、顾问、迁移专家和其他从事迁移、现代化、运营或。 DevOps

目标成果

这种模式重点提供以下目标结果：

- 能够安排、自动化和运行Stonebranch Universal Controller中的 AWS Mainframe Modernization 服务（Microfocus 运行时系统）的主机批处理作业。
- 通过 Stonebranch Universal Controller 监控应用程序的批处理过程。
- Start/Restart/Rerun/Stop通过 Stonebranch 通用控制器自动或手动进行批处理。
- 检索 AWS Mainframe Modernization 批处理流程的结果。
- 在 Stonebranch 通用控制器中捕获批处理任务的 AWS CloudWatch 日志。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 一款包含任务控制语言 (JCL) 文件的 Micro Focus Bankdem o 应用程序，以及部署在 AWS 大型机现代化服务（Micro Focus 运行时）环境中的批处理流程
- 有关如何构建和部署在 Micro Focus Enterprise Server上运行的主机应用程序的基础知识
- Stonebranch Universal Controller基础知识
- Stonebranch 试用许可证（请联系 Stonebranch）
- 具有至少四个内核、8 GB 内存和 2 GB 磁盘空间的 Windows 或 Linux 亚马逊弹性计算云 (Amazon EC2) 实例（例如 xlarge）实例
- Apache Tomcat 8.5.x 或 9.0.x 版
- Oracle Java 运行时环境（JRE）或 OpenJDK 第 8 版或第 11 版
- Amazon Aurora MySQL 兼容版
- Amazon Simple Storage Service (Amazon S3) 存储桶，可导出存储库
- 适用于代理 Stonebranch Universal Message Service (OMS) 连接实现高可用性的Amazon Elastic File System (Amaon EFS)。
- Stonebranch Universal Controller 7.2 Universal Agent 7.2 安装文件
- AWS Mainframe Modernization 任务计划模板（最新版 .zip 文件）

限制

- 该产品和解决方案仅在 OpenJDK 8 和 11 版本上进行了测试和兼容性验证。
- aws-mainframe-modernization-stonebranch集成任务计划模板仅适用于 AWS 大型机现代化服务。
- 此任务计划模板只能在 Unix、Linux 或 Windows 版本 Stonebranch 代理上运行。
- 某些 AWS 服务并非在所有 AWS 区域都可用。有关区域可用性，请参阅按地区划分的 AWS 服务。有关特定终端节点，请参阅服务终端节点和配额页面，然后选择服务的链接。

架构

目标状态架构

下图显示了此试点所需示例 AWS 环境。

1. Stonebranch Universal Automation Center (UAC) 包括两个主要组件：Universal Controller和 Universal Agent。Stonebranch OMS 用作控制器和各个代理间的消息总线。

2. Universal Controller 使用 Stonebranch UAC 数据库。该数据库可以兼容 MySQL Server、Microsoft SQL Server、Oracle 或者 Aurora MySQL。
3. AWS 大型机现代化服务 — 部署了BankDemo 应用程序的 Micro Focus 运行时环境。 BankDemo 应用程序文件将存储在 S3 存储桶中。此存储桶还包含主机 JCL 文件。
4. Stonebranch UAC 可运行以下函数进行批量运行：
   a. 使用链接至 AWS Mainframe Modernization服务的 S3 存储桶中的 JCL 文件名启动批处理作业。
   b. 获取批处理作业运行状态。
   c. 等待批处理作业运行完成。
   d. 获取批处理作业运行日志。
   e. 重新运行失败批处理作业。
   f. 批量作业运行时取消批量作业。
5. Stonebranch UAC 可为应用程序运行以下函数：
   a. 启动应用程序
   b. 获取应用程序状态
   c. 等待应用程序启动或停止
   d. 停止应用程序
   e. 获取应用程序操作日志

Stonebranch 作业转换

下图显示了 Stonebranch 在现代化流程中的工作转换过程。它描述了如何将任务计划和任务定义转换为可运行 AWS Mainframe Modernization 批处理任务的兼容格式。

1. 在转换过程中，任务定义从现有的主机系统中导出。
2. 可以将 JCL 文件上传至 Mainframe Modernization 应用程序的 S3 存储桶，这样 AWS Mainframe Modernization 服务就可以部署这些 JCL 文件。
3. 转换工具将导出作业定义转换为 UAC 任务。
4. 创建完所有任务定义和作业计划后，这些对象将导入 Universal Controller。然后，转换后的任务将在 AWS Mainframe Modernization 服务中运行这些流程，而不是在主机上运行这些流程。

Stonebranch UAC 架构

以下架构图代表了高可用性 (HA) 通用控制器的 active-active-passive模型。Stonebranch UAC 部署至多个可用区，以提供高可用性并支持灾难恢复 (DR)。

Universal Controller

两台 Linux 服务器被配置为 Universal Controller。两者都连接至同一个数据库端点。每台服务器都装有一个 Universal Controller 应用程序和 OMS。在配置 Universal Controller 时使用最新版本。

Universal Controller 作为文档根目录部署在 Tomcat Web 应用程序中，并在端口 80 上提供服务。这种部署简化了前端负载均衡器配置。

使用 Stonebranch 通配符证书（例如`https://customer.stonebranch.cloud`）启用基于 TLS 的 HTTP 或 HTTPS。这样可以保护浏览器和应用程序间的通信。

OMS

Universal Agent和 OMS（Opswise 消息服务）驻留在每台 Universal Controller 服务器上。所有从客户端部署的 Universal Agent 都设置为连接至两个 OMS 服务。OMS 充当 Universal Agent 和 Universal Controller 之间的常见消息服务。

Amazon EFS 在每台服务器上都挂载假脱机目录。OMS 使用此共享的缓冲池目录，保存来自控制器和代理的连接和任务信息。OMS 可在高可用性模式下运行。如果主动 OMS 出现故障，则被动 OMS 可访问所有数据，并且它会自动恢复主动操作。Universal Agent 会检测到此更改，并自动连接至新的活动 OMS。

数据库

Amazon Relational Database Service (Amazon RDS) 托管 UAC 数据库，引擎与 Amazon Aurora MySQL 兼容。Amazon RDS 有助于定期管理与提供定时备份。两个 Universal Controller 实例都连接至同一个数据库端点。

负载均衡器

为每个实例设置了应用程序负载均衡器。负载均衡器可以在任何给定时刻将流量引导至活动控制器。您的实例域名指向相应负载均衡器端点。

URLs

您的每个实例都有 URL，如以下示例所示。

环境	实例
生产	`customer.stonebranch.cloud`
开发（非生产）	`customerdev.stonebranch.cloud`
测试（非生产）	`customertest.stonebranch.cloud`

> **Note**
>
> 可以根据需要设置非生产实例名称。

高可用性

高可用性 (HA) 是指系统能够在指定时间段内连续运行且不会出现故障。此类故障包括但不限于存储、CPU 或内存问题导致的服务器通信响应延迟和网络连接。

要满足高可用性要求：

- 所有 EC2 实例、数据库和其他配置都镜像到同一 AWS 区域内的两个独立可用区。
- 控制器通过亚马逊机器映像（AMI）在两个可用区的两台 Linux 服务器上进行配置。例如，如果您在欧洲 eu-west-1 区域进行配置，则您在可用区 eu-west-1a 和可用区 eu-west-1c 中有一个 Universal Controller。
- 不允许任何作业直接在应用程序服务器上运行，也不允许在此服务器上存储任何数据。
- 应用程序负载均衡器对每个 Universal Controller 运行运行状况检查，以识别活动控制器并将流量引导至该控制器。如果一台服务器出现问题，负载均衡器会自动将被动 Universal Controller 提升到主动状态。然后，负载均衡器从运行状况检查中识别出新的活动 Universal Controller 实例，并开始引导流量。故障转移将在四分钟内完成，不会丢失任何工作，且前端 URL 保持不变。
- 兼容 Aurora MySQL 的数据库服务存储 Universal Controller 数据。对于生产环境，数据库集群由位于单个 AWS 区域内的两个不同可用区内的两个数据库实例构建。两个 Universal Controller 都使用指向单个数据库集群端点的 Java 数据库连接 (JDBC) 接口。如果数据库实例出现问题，则数据库集群端点会动态指向运行状况良好的实例。无需手动干预。

备份和清除

Stonebranch Universal Controller 设置为按照表中所示时间表备份和清除旧数据。

类型	计划
活动	7 days
审核	90 天
历史记录	60 天

早于显示日期的备份数据将导出为 .xml 格式并存储在文件系统中。备份过程完成后，将从数据库中清除较旧数据，并在 S3 存储桶中存档长达一年的生产实例。

您可以在 Universal Controller 界面中调整此时间表。但是，增加这些时间范围可能会导致延长维护期间的停机时间。

工具

Amazon Web Services

- AWS Mainframe Modernization 是一个 AWS 云原生平台，可帮助您将主机应用程序现代化为 AWS 托管运行时系统环境。它提供了工具和资源来帮助您规划和实施迁移与现代化。
- Amazon Elastic Block Store (Amazon EBS) 提供用于亚马逊实例的块级存储卷。 EC2
- Amazon Elastic File System (Amazon EFS) 可帮助您在 Amazon Web Services Cloud 中创建和配置共享文件系统。
- Amazon Relational Database Service (Amazon RDS) 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展关系数据库。Amazon Aurora MySQL 兼容版。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- 弹性负载均衡（ELB）将传入的应用程序或网络流量分配到多个目标。例如，您可以在一个或多个可用区的 Amazon EC2 实例、容器和 IP 地址之间分配流量。此模式使用应用程序负载均衡器。

Stonebranch

- Universal Automation Center (UAC) 是一个由企业工作负载自动化产品组成的系统。此模式采用以下 UAC 组件：
  - Universal Contro ller 是一款在 Tomcat Web 容器中运行的 Java Web 应用程序，是通用自动化中心的企业作业调度程序和工作负载自动化代理解决方案。控制器提供了一个用户界面，用于创建、

监视和配置控制器信息；处理调度逻辑；处理进出通用代理的所有消息；并同步通用自动化中心的大部分高可用性操作。

  - Universal Agent 是独立于供应商的调度代理，可与所有主要计算平台（包括传统和分布式）上的现有作业调度器协作。支持在 z/Series、i/Series、Unix、Linux 或 Windows 上运行的全部调度程序。
- Universal Agent 是独立于供应商的调度代理，可与所有主要计算平台（包括传统和分布式）上的现有作业调度器协作。支持在 z/Series、i/Series、Unix、Linux 或 Windows 上运行的全部调度程序。
- Stonebranch aws-mainframe-modernization-stonebranch-integration AWS 大型机现代化通用扩展是在 AWS 大型机现代化平台中运行、监控和重新运行批处理作业的集成模板。

代码

此模式的代码可在 aws-mainframe-modernization-stonebranch- integration GitHub 存储库中找到。

操作说明

在 Amazon 上安装通用控制器和通用代理 EC2

Task	描述	所需技能
下载安装文件。	从 Stonebranch 服务器下载安装程序。若要获取安装文件，请联系 Stonebranch。	云架构师
启动实 EC2 例。	安装 Universal Controller 和 Universal Agent 需要大约 3 GB 的额外空间。因此，请为此实例提供至少 30 GB 的磁盘空间。 向安全组添加端口 8080，使其可访问。	云架构师
检查先决条件。	安装前请执行以下操作： 1. 按照下载 Java 运行时环境所述安装 Java。	云管理员、Linux 管理员

Task	描述	所需技能

	``
	$ sudo yum -y update
	$ sudo yum install
	 java-11-amazon-cor
	retto
	``
	请务必使用所支持的 JAVA 版本之一。之前的命令应安装 java-11。请检查 Java 版本并确保您使用的是第 11 版，然后再继续。
	2. 如安装 Apache Tomcat 中所述，运行以下命令。
	``
	$ sudo yum install
	 tomcat tomcat-admin-
	webapps
	$ sudo systemctl
	 enable tomcat
	$ sudo systemctl start
	 tomcat
	``
	3. 按照创建并连接 Aurora MySQL 数据库集群中所述创建 Amazon Aurora 数据库。使用 Amazon Aurora MySQL 兼容版。
	选择主用户名与主密码。将其他设置保留为默认值。	

Task	描述	所需技能

安装Universal Controller。	1. 将universal-controller-7.2.0.0.tar 安装文件上传到实 EC2 例。
	2. 将安装文件取消存档至某个 temp 文件夹。
	`$ tar -xvf universal-controller-7.2.0.0.tar`
	3. 为安装脚本授予运行权限。
	`$ chmod a+x install-controller.sh`
	4. 安装控制器。此示例使用以下命令在 /usr/share/tomca 下安装 Universal Controller。请使用您在先前步骤中创建的 Amazon Aurora 数据库。
	`$ sudo ./install-controller.sh --tomcat-dir /usr/share/tomcat/ --controller-file universal-controller-7.2.0.0-build.145.war --dbuser admin --dbpass “****” --dbname uc --rdbms mysql --dburl jdbc:mysql://database-2-instance-1.cih63miincgy.us-east-1.rds.amazonaws.com:3306/`
		云架构师、Linux 管理员

Task	描述	所需技能
	脚本输出的最后一行应为“安装完成”。 5. 在 EC2 实例中导航到以下 URL。 `http://<public_ip>:8080/uc` 6. 在登录屏幕上，在用户名部分输入 ops.admin，并将密码 字段留空。 7. 为 `ops.admin` 用户设置新密码。	

Task	描述	所需技能

安装 Universal Agent。	1. 将`sb-7.2.0.1-linux-3.10-x86_64.tar.Z` 安装文件上传到实 EC2 例。
	2. 登录实 EC2 例。
	3. 取消存档 Universal Agent 安装包。
	`$ zcat sb-7.2.0.1-linux-3.10-x86_64.tar.Z | tar xvf -`
	4. 运行以下命令。
	`$ sudo ./unvinst --oms_servers 7878@localhost --oms_autostart yes --python yes`
	5. 创建 PAM 文件。
	`$ cp /etc/pam.d/login /etc/pam.d/ucmd`
	6. 为 Universal Agent 自动启动。
	`$ /sbin/restorecon -v /etc/rc.d/init.d/ubrokerd`
		云管理员、Linux 管理员

Task	描述	所需技能
将 OMS 添加到 Universal Controller。	1. 使用 `ops.admin` 用户登录 Universal Controller。 2. 选择屏幕左上角的服务菜单，然后在系统 中选择OMS 服务器 菜单 3. 在 “OMS 服务器地址” 字段中`localhost` ，键入，然后保存。 4. 您将看到 OMS 服务器的状态为已连接，会话状态为运行。	Universal Controller 管理员

导入 AWS Mainframe Modernization 通用扩展并创建任务

Task	描述	所需技能
导入集成模板。	若要完成此步骤，您需要AWS Mainframe Modernization Universal Extension。确保已下载最新发布的 .zip 文件版本。 1. 使用 `ops.admin` 用户登录 Universal Controller。 2. 导航到服务，导入集成模板。 3. 选择集成模板 .zip 文件 (`aws_mainframe_modernization_stonebranch_extension.zip`)，然后选择导入。	Universal Controller 管理员

Task	描述	所需技能
	导入集成模板后，您将在可用服务下看到 AWS Mainframe Modernization 任务。	

Task	描述	所需技能
启用可解析的凭证。	1. 导航到服务、AWS Mainframe Modernization 任务。 2. 在右侧面板上填写必填字段： • 名称：新的 Mainframe Modernization 任务 • 代理：选择唯一代理 (AGNT0001)。 在 AWS Mainframe Modernization 详情下： • 操作：列出环境 • AWS 证书：如果您在 EC2 实例中添加了 AWS 身份和访问管理 (IAM) 角色，则可以将此字段留空。如果要使用 `AWSAccessKeyID` 和 `AWSSecretKey` ，请选择该字段旁边的图标 ()。 在打开的凭证详细信息窗口中，输入以下信息，然后保存。 • 名称：AWS Mainframe Modernization 凭证 • 运行时用户：在此字段中写入 AWS 访问密钥 ID。 • 运行时密码：在此字段中写入 AWS 密钥密码。 • 终点：确保端点具有正确 AWS 区域。默认	Universal Controller 管理员

Task	描述	所需技能
	为 `https://m2.us-east-1.amazonaws.com` 。 • 区域：输入 AWS Mainframe Modernization 服务区域。默认为 `us-east-1` 。 3. 在其余字段保留默认值，然后保存任务。	

Task	描述	所需技能
启动任务。	1. 在右侧面板顶部选择启动任务。 2. 在确认窗口中，选择启动。之后，Universal Controller 控制台将显示类似如下的消息。 2022 年 8 月 24 日上午 10:11:49 成功启动了通用任务 “新的大型机现代化任务”，任务实例 sys_id 166 NC8 1291493634146313 E38 OZJY。DB8 3. 导航到实例，如果您没有看到实例选项卡，请选择向右箭头向右滚动。 4. 打开列表中任务实例的上下文（右键单击）菜单，选择检索输出，选择检索输出，然后选择检索输出中的提交 5. 在检索输出窗口中，您将看到 STDOUT 中的环境列表。	Universal Controller 管理员

测试启动批量作业

Task	描述	所需技能
为批处理作业创建任务。	1. 导航到服务、AWS Mainframe Modernization 任务。	Universal Controller 管理员

Task	描述	所需技能
	2. 在右侧面板上填写必填字段： • 名称：新的 Mainframe Modernization 任务 • 代理：选择唯一代理 (AGNT0001)。 在 AWS Mainframe Modernization 详情下： • 操作：启动批处理（或者在 AWS 中启动批处理并等待运行批处理作业并等到任务完成） • AWS 证书：如果您已向 EC2 实例添加了 IAM 角色，则可以将此字段留空。如果要使用 `AWSAccessKeyID` 和 `AWSSecretKey` ，请选择该字段旁边的图标 ()。 • 终点：确保端点具有正确 AWS 区域。默认为 https://m2.us-east-1.amazonaws.com。 • 区域：输入 AWS Mainframe Modernization 服务区域。默认为 `us-east-1` 。 • 应用程序：选择字段旁边的图标 ()，然后在刷新应用程序选项中选择提交。这将连接到 AWS Mainframe Modernization 服务并返回应用程序列	

Task	描述	所需技能
	表。现在，您可从下拉列表中选择应用程序。选择要运行的批量作业。 • JCL 文件名：`RUNHELLO.jcl` • 等待成功或失败：如果选择此项，则任务将等到批处理作业的状态为成功或失败。 • 轮询间隔：这是每次轮询间的时间间隔。 • 获取执行日志：如选中此选项，则将在批处理作业完成后自动提取日志。 • 日志格式：这是要打印的日志格式。它可能是文本或 JSON 格式。 3. 在其余字段保留默认值，然后保存任务。	

Task	描述	所需技能
启动任务。	1. 在右侧面板顶部选择启动任务。 2. 在确认窗口中，选择启动。之后，Universal Controller 控制台将显示类似如下的消息。 2022 年 8 月 24 日上午 11:11:59 使用任务实例 sys_id 成功启动 Universal task "Mainframe Modernization Start Batch。 <sys id> 3. 导航到实例，如果您没有看到实例选项卡，请选择向右箭头向右滚动。 4. 打开列表中任务实例的上下文（右键单击）菜单，选择检索输出，选择检索输出，然后选择检索输出中的提交 5. 在检索输出窗口中，您将看到 STDOUT 中的环境列表。	Universal Controller 管理员

为多项任务创建一个工作流

Task	描述	所需技能
复制任务。	1. 打开要为其创建副本的任务的上下文（右键单击）菜单，然后选择 复制。 2. 在 “复制 AWS 大型机现代化任务” 窗口中，为新任务	Universal Controller 管理员

Task	描述	所需技能
	输入以下新名称：大型机现代化 Start Batch-。RUNAWS2 3. 使用以下名称再次复制任务：大型机现代化 Start Batch-RUNAWS3。 4. 使用以下名称再次复制任务：大型机现代化 Start Batch-RUNAWS4。 5. 使用以下名称最后一次复制任务：Mainframe Modernization Start Batch - FOOBAR。	

Task	描述	所需技能
更新任务。	1. 打开（双击）“大型机现代化 Start Batch-” RUNAWS2 任务，将 “JCL 文件名” 字段更改为`RUNAWS2.jcl` ，然后保存。 2. 打开（双击）“大型机现代化 Start Batch-” RUNAWS3 任务，将 “JCL 文件名” 字段更改为 `RUNAWS3.jcl` ，然后保存。 3. 打开（双击）“大型机现代化 Start Batch-” RUNAWS4 任务，将 “JCL 文件名” 字段更改为 `RUNAWS4.jcl` ，然后保存。 4. 打开（双击）Mainframe Modernization Start Batch - FOOBAR 任务，将 JCL 文件名字段更改为 `MISSING.jcl` ，然后保存。此任务将会失败，因为 JCL 文件名值不正确。	Universal Controller 管理员

Task	描述	所需技能
创建工作流。	1. 导航到服务、工作流。 2. 在右侧面板的名称字段中输入Mainframe Modernization Workflow，然后保存。 3. 在右窗格中，选择编辑工作流。 4. 在工作流编辑器选项卡，单击添加任务按钮 (+)。 5. 在任务查找窗口，选择搜索以查看 Universal Controller 中的所有任务。 6. 单击 Mainframe Modernization Start Batch Task 旁边的图标，然后将该图标拖到工作流编辑器中的空白处。 7. 对其他 Mainframe Modernization 任务重复相同的操作，然后按照其他信息部分所示放置它们。 8. 选择连接按钮 ()，，然后将任务连接在一起。要将任务与另一个任务关联，请在任务中间单击，然后将其拖动到目标任务。 9. 按照其他信息部分所示连接任务，然后保存工作流。 10.右键单击“工作流编辑器”中的空白处，选择启动工作流，然后选择确定。	Universal Controller 管理员

Task	描述	所需技能
查看工作流的状态。	1. 在左侧菜单上，选择活动 2. 在窗口中间，选择开始。 您将要在列表中看到任务实例列表。 3. 在列表中打开（双击）Mainframe Modernization 工作流，或者打开上下文（右键单击）菜单并选择工作流任务命令、查看工作流。 您将看到“其他信息” 部分所示的任务。由于您缺失了 JCL 文件，第二个任务预计会失败。	Universal Controller 管理员

对失败的批处理作业进行故障排除，并重新运行

Task	描述	所需技能
修复失败的任务并重新运行。	1. 打开（双击）失败的任务以查看任务的错误。 2. 修复失败的任务时，您有两种选择。 • 修复 JCL 文件名，并将其设置为 `FOOBAR.jcl` 。 • 在 JCL 文件名（临时）中添加正确的 JCL 文件名。此字段将覆盖JCL 文件名字段。 在此试点中，选择第二选项，然后保存任务实例。	Universal Controller 管理员

Task	描述	所需技能
	3. 在工作流监视器中，打开失败任务的上下文（右键单击）菜单，然后选择命令、重新运行。 4. 之后，所有任务将成功完成。	

创建启动应用程序和停止应用程序任务

Task	描述	所需技能
创建启动应用程序操作。	1. 导航到服务、AWS Mainframe Modernization 任务。 2. 在右窗格上，填写必填字段。 • 名称：Mainframe Modernization Start 应用程序 • 代理：选择唯一的代理 (AGNT0001) 在 AWS Mainframe Modernization 详情下： • 操作：启动应用程序 • AWS 证书：如果您已向 EC2 实例添加了 IAM 角色，则可以将此字段留空。如果要使用 `AWSAccessKeyID` 和 `AWSSecretKey` ，请选择之前创建的凭证。	Universal Controller 管理员

Task	描述	所需技能
	• 端点：确保端点具有正确的区域。默认为 `https://m2.us-east-1.amazonaws.com` 。 • 区域：输入 AWS Mainframe Modernization 服务区域。默认为 `us-east-1` 。 • 应用程序：选择字段旁边的图标 ()，然后在刷新应用程序选项中选择提交。这将连接到 AWS Mainframe Modernization 服务并返回应用程序列表。现在，您可从下拉列表中选择应用程序。选择要运行的批量作业。 • 等待成功或失败：如果选择此项，则任务将等到批处理作业的状态为成功或失败。 • 轮询间隔：这是每次轮询间的时间间隔。 • 获取执行日志：如选中此选项，则将在批处理作业完成后自动提取日志。 • 日志格式：这是要打印的日志格式。它可能是文本或 JSON 格式。 3. 在其余字段保留默认值，然后保存任务。	

Task	描述	所需技能
	4. 现在复制此任务并创建“停止应用程序”任务。将名称更改为 Mainframe Moderinization Stop 应用程序，然后将操作更改为停止应用程序。	

创建取消批量执行” 任务

Task	描述	所需技能
创建 Cancel Batch 操作。	1. 导航到服务、AWS Mainframe Modernization 任务。 2. 在右窗格上，填写必填字段。 • 名称：Mainframe Modernization Cancel Batch Execution • 代理：选择唯一的代理 (AGNT0001) 在 AWS Mainframe Modernization 详情下： • 操作：取消批量执行 • AWS 证书：如果您已向 EC2 实例添加了 IAM 角色，则可以将此字段留空。如果要使用 `AWSAccessKeyID` 和 `AWSSecretKey` ，请选择之前创建的凭证。	

Task	描述	所需技能
	• 端点：确保端点具有正确的区域。默认为 `https://m2.us-east-1.amazonaws.com` 。 • 区域：输入 AWS Mainframe Modernization 服务区域。默认为 `us-east-1` 。 • 应用程序：选择字段旁边的图标 ()，然后在刷新应用程序选项中选择提交。这将连接到 AWS Mainframe Modernization 服务并返回应用程序列表。现在，您可从下拉列表中选择应用程序。选择要运行的批量作业。 • 等待成功或失败：如果选择此项，则任务将等到批处理作业的状态为成功或失败。 • 轮询间隔：这是每次轮询间的时间间隔。 • 获取执行日志：如选中此选项，则将在批处理作业完成后自动提取日志。 • 日志格式：这是要打印的日志格式。它可能是文本或 JSON 格式。 3. 在其余字段保留默认值，然后保存任务。	

相关资源

- Universal Controller
- Universal Agent
- LDAP 设置
- SAML 单点登录
- Xpress 转换工具

其他信息

工作流编辑器内的图标

所有已连接任务

工作流状态

使用 Connect from Precisely 将 VSAM 文件迁移和复制到 Amazon RDS 或 Amazon MSK

创建者：Prachi Khanna (AWS) 和 Boopathy GOPALSAMY (AWS)

摘要

此模式向您展示了如何使用 Connect from Precisely 将虚拟存储访问方法 (VSAM) 文件从大型机迁移和复制到 Amazon Web Services Cloud 中的目标环境。此模式涵盖的目标环境包括 Amazon Relational Database Service (Amazon RDS) 和 Amazon Managed Streaming for Apache Kafka (Amazon MSK)。Connect 使用更改数据捕获 (CDC) 持续监控源 VSAM 文件的更新，然后将这些更新传输到您的一个或多个 AWS 目标环境。您可使用这种模式来实现应用程序现代化或数据分析目标。例如，您可使用 Connect 将您的 VSAM 应用程序文件以低延迟迁移到 Amazon Web Services Cloud，或者将 VSAM 数据迁移到 AWS 数据仓库或数据湖进行分析，以容限高于应用程序现代化所需同步延迟。

先决条件和限制

先决条件

- IBM z/OS V2R1 或更高版本
- CICS Transaction Server for z/OS (CICS TS) V5.1 或更高版本（CICS/VSAM 数据捕获）
- IBM MQ 8.0 或更高版本
- 符合 z/OS 安全要求（例如，加载库的 APF 授权）SQData
- VSAM 恢复日志已开启
- （可选）CICS VSAM Recovery Version (CICS VR) 用于自动捕获 CDC 日志
- 一个有效的 Amazon Web Services account
- 具有可由您的旧平台访问的子网的 Amazon 虚拟私有云（VPC）
- 来自 Precisely 的 VSAM Connect 许可证

限制

- Connect 不支持基于源 VSAM 架构或副本自动创建目标表。您必须首次定义目标表的结构。
- 对于非流媒体目标（例如 Amazon RDS），您必须在 Apply Engine 配置脚本中指定目标映射的转换源。
- 记录、监控和警报功能通过外部组件（例如 Amazon）实现， APIs 并且需要外部组件（例如 Amazon CloudWatch）才能完全运行。

产品版本

- SQData 40134 适用于 z/OS
- SQData 4.0.43 适用于亚马逊弹性计算云（亚马逊）上的亚马逊 Linux 亚马逊系统映像 (AMI) EC2

架构

源技术堆栈

- 作业控制语言 (JCL)
- z/OS Unix Shell 和 Interactive System Productivity Facility (ISPF)
- VSAM 实用程序 (IDCAMS)

目标技术堆栈

- Amazon EC2
- Amazon MSK
- Amazon RDS
- Amazon VPC

目标架构

将 VSAM 文件迁移到 Amazon RDS

下图显示了如何通过在源环境（本地大型机）中使用 CDC 代理/Publisher，在目标环境 (AWS Cloud) 中使用 Apply Engine，实时或近乎实时地将 VSAM 文件迁移到关系数据库（例如 Amazon RDS）。

图表显示了以下批处理工作流：

1. Connect 通过比较备份文件中的 VSAM 文件来捕获文件更改以识别更改，然后将更改发送到日志流。
2. Publisher 使用系统日志流中的数据。
3. Publisher 通过 TCP/IP 将捕获的数据更改传达至目标引擎。控制器进程守护程序对源环境和目标环境之间的通信进行身份验证。
4. 目标环境中的 Apply Engine 从 Publisher 代理接收更改并将它们应用到关系或非关系数据库。

图表显示了以下在线工作流：

1. Connect 使用日志副本捕获联机文件更改，然后将捕获的更改流式传输到日志流。
2. Publisher 使用系统日志流中的数据。
3. Publisher 通过 TCP/IP 将捕获的数据更改传达至目标引擎。控制器进程守护程序对源环境和目标环境之间的通信进行身份验证。
4. 目标环境中的 Apply Engine 从 Publisher 代理接收更改，然后将它们应用到关系或非关系数据库。

将 VSAM 文件迁移到 Amazon MSK

下图显示了如何在高性能模式下将 VSAM 数据结构从大型机流式传输到 Amazon MSK，并自动生成与 Amazon MSK 集成的 JSON 或 AVRO 架构转换。

图表显示了以下批处理工作流：

1. Connect 通过使用 CICS VR 或通过比较备份文件中的 VSAM 文件，以捕获和识别文件更改。捕获的更改将发送至日志流。
2. Publisher 使用系统日志流中的数据。
3. Publisher 通过 TCP/IP 将捕获的数据更改传达至目标引擎。控制器进程守护程序对源环境和目标环境之间的通信进行身份验证。
4. 在并行处理模式下运行的 Replicator Engine 将数据拆分为工作缓存单元。
5. Worker 线程从缓存中捕获数据。
6. 数据从 Worker 线程发布到 Amazon MSK 主题。
7. 用户使用连接器将来自亚马逊 MSK 的更改应用到目标，例如亚马逊 DynamoDB、亚马逊简单存储服务 (Amazon S3) S OpenSearch ervice 或亚马逊服务。

图表显示了以下在线工作流：

1. 在线文件中的更改是通过使用日志复制来捕获的。捕获的更改将传输到日志流。
2. Publisher 使用系统日志流中的数据。
3. Publisher 通过 TCP/IP 将捕获的数据更改传达至目标引擎。控制器进程守护程序对源环境和目标环境之间的通信进行身份验证。
4. 在并行处理模式下运行的 Replicator Engine 将数据拆分为工作缓存单元。
5. Worker 线程从缓存中捕获数据。

6. 数据从 Worker 线程发布到 Amazon MSK 主题。
7. 用户使用连接器将 Amazon MSK 中的更改应用于 DynamoDB、Ama OpenSearch zon S3 或服务等目标。

工具

- Amazon Managed Streaming for Apache Kafka (Amazon MSK) 是一项完全托管式服务，可帮助您构建并运行使用 Apache Kafka 来处理流数据的应用程序。
- Amazon Relational Database Service (Amazon RDS) 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展关系数据库。

操作说明

准备源环境（主机）

Task	描述	所需技能
安装 Connect CDC 4.1。	1. 请联系 Precission Support 团队获取许可证和安装包。 2. 使用示例 JCLs 安装 Connect CDC 4.1。有关说明，请参阅 Precist 文档中的使用 JCL 安装 Connect CDC (SQData)。 3. 运行 SETPROG APF 命令授权 Connect 加载库 sqdata.v4nnn.loadLib。	IBM 大型机开发人员/管理员
设置 zFS 目录。	要设置 zfS 目录，请按照 Preclist 文档中 zfS 变量目录中的说明进行操作。 Note 控制器守护程序和 Capture/Publisher	IBM 大型机开发人员/管理员

Task	描述	所需技能
	agent configurations are stored in the z/OS UNIX 系统服务文件系统（称为 ZFS）。控制器进程守护程序、Capture、Storage 和 Publisher 代理需要预定义 zFS 目录结构以存储少量文件。	
配置 TCP/IP 端口。	要配置 TCP/IP 端口，请按照 Precisely 文档中的 TCP/IP 端口中的说明进行操作。 **Note** 控制器守护程序需要源系统上的 TCP/IP 端口。这些端口由目标系统上的引擎引用（在那里处理捕获的更改数据）。	IBM 大型机开发人员/管理员

Task	描述	所需技能
创建 z/OS 日志流。	要创建 z/OS 日志流，请按照 Precisely 文档中的创建 z/OS 系统日志流中的说明进行操作。 **Note** 迁移期间，Connect 使用日志流在源环境和目标环境之间捕获和流式传输数据。 有关创建 z/O LogStream S 的 JCL 示例，请参阅 Precist 文档中的创建 z/OS 系统 Log Streams。	IBM 大型机开发人员
IDs 为 ZFS 用户和已启动的任务进行识别和授权。	使用 RACF 授予对 OMVS zFS 文件系统的访问权限。有关 JCL 的示例，请参阅 Precist 文档 IDs中的识别和授权 zfS 用户并启动任务。	IBM 大型机开发人员/管理员
生成 z/OS 公有密钥/私有密钥以及授权密钥文件。	运行 JCL，以生成密钥对。有关示例，请参阅此模式的其他信息部分中的密钥对示例。 有关说明，请参阅 Precisely 文档中的生成 z/OS 公有密钥和私有密钥以及授权密钥文件。	IBM 大型机开发人员/管理员

Task	描述	所需技能

激活 CICS VSAM 日志复制并将其附加至日志流。	运行以下 JCL 脚本：
	
	``
	 //STEP1 EXEC PGM=IDCAMS
	 //SYSPRINT DD SYSOUT=*
	 //SYSIN DD *
	   ALTER SQDATA.CICS.FILEA -
	   LOGSTREAMID(SQDATA.VSAMCDC.LOG1) -
	   LOGREPLICATE
	``
		IBM 大型机开发人员/管理员

通过 FCT 激活 VSAM File Recovery 日志。	修改 File Control Table (FCT) 以反映以下参数更改：
	
	``
	 Configure FCT Parms
	   CEDA ALT FILE(name) GROUP(groupname)
	   DSNAME(data set name)
	   RECOVERY(NONE|BACKOUTONLY|ALL)
	   FWDRECOVLOG(NO|1-99)
	   BACKUPTYPE(STATIC|DYNAMIC)
	   RECOVERY PARAMETERS
	   RECOVery : None | Backoutonly | All
	   Fwdrecovlog : No | 1-99
	   BAckuptype : Static | Dynamic
	``
		IBM 大型机开发人员/管理员

Task	描述	所需技能
为发布商代理设置 CDCz日志。	1. 创建 CDCz日志发布器 CAB 文件。 2. 加密已发布数据。 3. 准备 CDCz日志发布器运行时 JCL。	IBM 大型机开发人员/管理员
激活控制器进程守护程序。	1. 打开 Tool ISPF 面板，然后运行以下命令，打开 Precisely 菜单：`EXEC 'SQDATA.V4nnnnn.ISPFLIB(SQDC$STA)' 'SQDATA.V4nnnnn'` 2. 要设置控制器进程守护程序，请从菜单中选择选项 2。	IBM 大型机开发人员/管理员
激活 Publisher。	1. 打开 Tool ISPF 面板，然后运行以下命令，打开 Precisely 菜单：`EXEC 'SQDATA.V4nnnnn.ISPFLIB(SQDC$STA)' 'SQDATA.V4nnnnn'` 2. 要设置 Publisher，请从菜单中选择选项 3，然后选择 I 插入。	IBM 大型机开发人员/管理员

Task	描述	所需技能
激活日志流。	1. 打开 Tool ISPF 面板，然后运行以下命令，打开 Precisely 菜单：`EXEC 'SQDATA.V4nnnnn.ISPFLIB(SQDC$STA)' 'SQDATA.V4nnnnn'` 2. 要设置日志流，请从菜单中选择选项 4，选择 I 插入。然后，输入前述步骤中创建的日志流名称。	IBM 大型机开发人员/管理员

准备目标环境 (AWS)

Task	描述	所需技能
精确地安装在 EC2 实例上。	要在适用于亚马逊的亚马逊 Linux AMI 上安装 Precission 的 Connect EC2，请按照 Precist 文档中在 UNIX 上安装 Connect CDC (SQData) 中的说明进行操作。	常规 AWS
打开 TCP/IP 端口。	要修改安全组以包含用于入站和出站访问的控制器进程守护程序端口，请按照 Precisely 文档中的 TCP/IP 中的说明进行操作。	常规 AWS
创建文件目录。	要创建文件目录，请按照 Precisely 文档中的准备目标应用环境中的说明进行操作。	常规 AWS
创建 Aply Engine 配置文件。	在 Apply Engine 的工作目录中创建 Apply Engine 配置文件。	常规 AWS

Task	描述	所需技能

	以下示例配置文件显示 Apache Kafka 作为目标：
	
	`builtin.features=SASL_SCRAM`
	`security.protocol=SASL_SSL`
	`sasl.mechanism=SCRAM-SHA-512`
	`sasl.username=`
	`sasl.password=`
	`metadata.broker.list=`
	
	Note
	有关更多信息，请参阅 Apache Kafka 文档中的安全性。	

为 Apply Engine 处理创建脚本。	为 Apply Engine 创建脚本以处理源数据并将源数据复制到目标。有关更多信息，请参阅 Precisely 文档中的创建应用引擎脚本。	常规 AWS
运行脚本。	使用 SQDPARSE 和 SQDENG 命令运行脚本。有关更多信息，请参阅 Precisely 文档中的解析 zOS 脚本。	常规 AWS

验证环境

Task	描述	所需技能
验证用于 CDC 处理的 VSAM 文件与目标列表。	1. 验证 VSAM 文件，包括复制日志、恢复日志、FCT 参数以及日志流。 2. 验证目标数据库表，包括是否根据所需架构定义、表访问和其他条件创建表。	常规 AWS、大型机
验证 Connect CDC SQData 产品是否已链接。	运行测试作业并验证此作业的返回码是否为 0（成功）。 **Note** Connect CDC SQData 应用引擎状态消息应显示活动连接消息。	常规 AWS、大型机

运行并验证测试用例（批处理）

Task	描述	所需技能
在大型机中运行批处理作业。	使用修改后的 JCL 运行批处理应用程序作业。在修改后的 JCL 中包括以下操作步骤： 1. 备份数据文件。 2. 将备份文件与修改后的数据文件进行比较，生成增量文件，然后记下消息中的增量记录计数。 3. 将增量文件推送至 z/OS 日志流。	常规 AWS、大型机

Task	描述	所需技能
	4. 运行 JCL。有关示例 JCL，请参阅 Precisely 文档中的准备文件比较捕获 JCL。	
检查日志流。	检查日志流，确认您可看到已完成的大型机批处理作业的更改数据。	常规 AWS、大型机
验证源增量更改和目标表的计数。	要确认记录已计数，请执行以下操作： 1. 从批处理 JCL 消息收集源增量计数。 2. 监视 Apply Engine，以了解在 VSAM 文件中插入、更新或删除的记录数量的记录级别计数。 3. 在目标表查询记录数。 4. 比较并统计所有不同记录数。	常规 AWS、大型机

运行和验证测试用例（在线）

Task	描述	所需技能
在 CICS 区域运行在线事务。	1. 运行在线事务，以验证测试用例。 2. 验证事务执行代码（RC=0 - 成功）。	IBM 大型机开发人员
检查日志流。	确认日志流中填充了特定记录级别更改。	AWS 大型机开发人员
验证目标数据库计数。	监视 Apply Engine 以获取记录级别计数。	Precisely、Linux

Task	描述	所需技能
验证目标数据库中的记录计数与数据记录。	查询目标数据库以验证记录计数与数据记录。	常规 AWS

相关资源

- VSAM z/OS（Precisely 文档）
- Apply Engine（Precisely 文档）
- 复制器引擎（Precisely 文档）
- 日志流（IBM 文档）

其他信息

配置文件示例

以下日志流的示例配置文件，其中源环境为大型机，目标环境是 Amazon MSK：

```
 -- JOBNAME -- PASS THE SUBSCRIBER NAME
 -- REPORT  progress report will be produced after "n" (number) of Source records
processed.

 JOBNAME VSMTOKFK;
 --REPORT EVERY 100;
 -- Change Op has been ‘I’ for insert, ‘D’ for delete , and ‘R’ for Replace. For RDS
it is 'U' for update
 -- Character Encoding on z/OS is Code Page 1047, on Linux and UNIX it is Code Page
819 and on Windows, Code Page 1252
 OPTIONS
 CDCOP('I', 'U', 'D'),
 PSEUDO NULL = NO,
 USE AVRO COMPATIBLE NAMES,
 APPLICATION ENCODING SCHEME = 1208;

 --       SOURCE DESCRIPTIONS

 BEGIN GROUP VSAM_SRC;
 DESCRIPTION COBOL ../copybk/ACCOUNT AS account_file;
```

```
 END GROUP;

 --       TARGET DESCRIPTIONS

 BEGIN GROUP VSAM_TGT;
 DESCRIPTION COBOL ../copybk/ACCOUNT AS account_file;
 END GROUP;

 --       SOURCE DATASTORE (IP & Publisher name)

 DATASTORE cdc://10.81.148.4:2626/vsmcdct/VSMTOKFK
 OF VSAMCDC
 AS CDCIN
 DESCRIBED BY GROUP VSAM_SRC ACCEPT ALL;

 --       TARGET DATASTORE(s) - Kafka and topic name

 DATASTORE 'kafka:///MSKTutorialTopic/key'
 OF JSON
 AS CDCOUT
 DESCRIBED BY GROUP VSAM_TGT FOR INSERT;

 --       MAIN SECTION

 PROCESS INTO
 CDCOUT
 SELECT
 {
 SETURL(CDCOUT, 'kafka:///MSKTutorialTopic/key')
 REMAP(CDCIN, account_file, GET_RAW_RECORD(CDCIN, AFTER), GET_RAW_RECORD(CDCIN,
BEFORE))
 REPLICATE(CDCOUT, account_file)
 }
 FROM CDCIN;
```

密钥对示例

以下是关于如何运行 JCL 生成密钥对的示例：

```
//SQDUTIL EXEC PGM=SQDUTIL //SQDPUBL DD DSN=&USER..NACL.PUBLIC, //
 DCB=(RECFM=FB,LRECL=80,BLKSIZE=21200), // DISP=(,CATLG,DELETE),UNIT=SYSDA, //
 SPACE=(TRK,(1,1)) //SQDPKEY DD DSN=&USER..NACL.PRIVATE, //
 DCB=(RECFM=FB,LRECL=80,BLKSIZE=21200), // DISP=(,CATLG,DELETE),UNIT=SYSDA, //
```

```
 SPACE=(TRK,(1,1)) //SQDPARMS DD  keygen //SYSPRINT DD SYSOUT= //SYSOUT DD SYSOUT=* //
SQDLOG DD SYSOUT=* //*SQDLOG8 DD DUMMY
```

使用 Rocket Enterprise Server 和 L PageCenter RS X 实现大型机输出管理的现代化 AWS

由 Shubham Roy (AWS)、Abraham Rondon (Micro Focus) 和 Guy Tucker (Levi、Ray & Shoup Inc) 创建

摘要

通过实现大型机输出管理的现代化，您可以通过 Amazon Web Services (AWS) 云原生技术节省成本，减轻维护传统系统的技术负担， DevOps 并提高弹性和灵活性。此模式将展示如何在 Amazon Web Services Cloud 上实现关键业务大型机输出管理工作负载的现代化。该模式使用 R ocket Enterprise Server 作为现代化大型机应用程序的运行时，Levi、Ray & Shoup, Inc. (LRS) VPSX/MFI（Micro Focus Interface）作为打印服务器，LRS X作为存档服务器。 PageCenterLRS PageCenter X 提供输出管理解决方案，用于查看、索引、搜索、存档和保护对业务输出的访问。

该模式基于更换平台大型机现代化方法。大型机应用程序由 AWS 大型机现代化迁移到亚马逊弹性计算云 (Ama EC2 zon) 上。大型机输出管理工作负载迁移到亚马逊 EC2，大型机数据库（例如适用于 z/OS 的 IBM Db2）迁移到亚马逊关系数据库服务（Amazon RDS）。LRS 目录集成服务器 (LRS/DIS) 与适用于 Microsoft Active Directory 的 AWS Directory Service 协同工作，用于输出管理工作流身份验证和授权。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 大型机输出管理工作负载。
- 有关如何重建和交付在 Rocket Enterprise Server 上运行的大型机应用程序的基础知识。有关更多信息，请参阅 Rocket Software 文档中的 Rocket Enterprise 服务器数据表。
- LRS 云打印解决方案和概念的基础知识。有关详细信息，请参阅 LRS 文档中的输出现代化。
- 火箭企业服务器软件和许可证。欲了解更多信息，请联系火箭软件。
- LRS VPSX/MFI, LRS PageCenterX, LRS/Queue, and LRS/DIS 软件和许可证。有关更多信息，请联系 LRS。您必须提供将安装 LRS 产品的 EC2 实例的主机名。

> **Note**
>
> 有关大型机输出管理工作负载的配置注意事项的更多信息，请参阅此模式的 “其他信息” 部分中的注意事项。

产品版本

- 火箭企业服务器 10.0
- LRS VPSX/MFI
- LRS PageCenter X V1R3 或更高版本

架构

源技术堆栈

- 操作系统 - IBM z/OS
- 编程语言 — 面向业务的通用语言 (COBOL)、作业控制语言 (JCL) 和客户信息控制系统 (CICS)
- 数据库 - IBM Db2 for z/OS、IBM Information Management System (IMS) 数据库和虚拟存储访问方法 (VSAM)
- 安全 — 资源访问控制设施 (RACF)、适用于 z/OS 的 CA 绝密和访问控制设施 2 () ACF2
- 打印和存档解决方案 - IBM 大型机 z/OS 输出和打印产品（IBM Infoprint Server for z/OS、LRS 和 CA Deliver）和存档解决方案（CA Deliver、ASG Mobius 或 CA Bundle）

源架构

下图显示了大型机输出管理工作负载的典型当前状态架构。

图表显示了以下工作流：

1. 用户在基于 COBOL 编写的 IBM CICS 应用程序构建的互动系统 (SoE) 上执行业务事务。
2. SoE 调用大型机服务，该服务将业务交易数据记录在 system-of-records (SoR) 数据库中，例如适用于 z/OS 的 IBM Db2。

3. SoR 保留来自 SoE 的业务数据。
4. 批处理作业调度程序启动批处理作业以生成打印输出。
5. 批处理作业从数据库中提取数据。并根据业务需求对数据进行格式化，然后生成业务输出（例如账单、身份证或贷款对账单）。最后，批处理作业将路由输出至输出管理，以便根据业务需求对输出进行格式化、发布和存储。
6. 输出管理接收批处理作业的输出。输出管理索引、排列输出并将其发布到输出管理系统中的指定目的地，例如 LRS PageCenter X 解决方案（如本模式所示）或 CA View。
7. 用户可以查看、搜索和检索输出。

目标技术堆栈

- 操作系统 — 在亚马逊上运行的 Windows 服务器 EC2
- 计算 — 亚马逊 EC2
- 存储 — 亚马逊 Elastic Block Store (亚马逊 EBS) 和 FSx 适用于 Windows 文件服务器的亚马逊 Elastic Block Store
- 编程语言 - COBOL、JCL 和 CICS
- 数据库 - Amazon RDS
- 安全性 - AWS 托管的 Microsoft AD
- 打印和存档 — AWS 上的 LRS 打印 (VPSX) 和存档 (PageCenterX) 解决方案
- 大型机运行时环境 — Rocket 企业服务器

目标架构

下方图表显示了部署在 Amazon Web Services Cloud 中的大型机输出管理工作负载的架构。

图表显示了以下工作流：

1. 批处理作业调度程序启动批处理作业来创建输出，例如账单、身份证或贷款报表。
2. 大型机批处理作业（平台改编为 Amazon EC2）使用 Rocket Enterprise Server 运行时从应用程序数据库中提取数据，对数据应用业务逻辑并格式化数据。然后，它使用 Rocket Software 打印机退出模块将数据发送到输出目的地（OpenText Micro Focus 文档）。
3. 应用程序数据库（在 Amazon RDS 上运行的 SoR）保留打印输出的数据。

4. LRS VPSX/MFI printing solution is deployed on Amazon EC2, and its operational data is stored in Amazon EBS. LRS VPSX/MFI uses the TCP/IP-based LRS/Queue 传输代理用于通过 Rocket Software JES Print Exit API 收集输出数据。

   LRS VPSX/MFI 进行数据预处理，例如 EBCDIC 到 ASCII 的转换。它还可以执行更复杂的任务，包括将大型机专用的数据流（例如 IBM 高级功能演示 (AFP) 和 Xerox 行条件数据流 (LCDS)）转换为更常见的查看和打印数据流（例如打印机命令语言 (PCL) 和 PDF）。

   在 LRS PageCenter X 的维护窗口内，LR VPSX/MFI persists the output queue and serves as backup for the output queue. LRS VPSX/MFI connects and sends output to LRS PageCenterX by using the LRS/Queue protocol. LRS/Queue S 会交换任务的准备情况和完成情况，以帮助确保数据传输的进行。

   备注：

   有关从 Rocket Software Print Exit 传递到LRS/Queue and LRS VPSX/MFI支持的大型机批处理机制的打印数据的更多信息，请参阅 “其他信息” 部分中的打印数据捕获。

   LRS VPSX/MFI 可以在打印机机群级别执行运行状况检查。有关详细信息，请参阅此模式的其他信息部分中的打印机队列运行状况检查。

5. LRS PageCenter X 输出管理解决方案部署在亚马逊上 EC2，其操作数据存储在亚马 FSx 逊 Windows 文件服务器中。LRS PageCenter X 提供了一个中央报告管理系统，其中包含导入 LRS PageCenter X 的所有文件以及所有能够访问这些文件的用户。用户可以查看特定文件内容或跨多个文件执行搜索以查找匹配条件。

   LRS/NetX component is a multi-threaded web application server that provides a common runtime environment for the LRS PageCenterX application and other LRS applications. The LRS/Web Connect component is installed on your web server and provides a connector from the web server to the LRS/NetXWeb 应用程序服务器。

6. LRS PageCenter X 为文件系统对象提供存储。LRS PageCenter X 的操作数据存储在 Amazon for Window FSx s 文件服务器中。

7. 输出管理身份验证和授权由 AWS 托管 Microsoft AD 和 LRS/DIS 执行。

> **Note**
>
> 目标解决方案通常不需要更改应用程序以适应大型机格式化语言，例如 IBM AFP 或 Xerox LCDS。

AWS 基础设施架构

下图显示了适用于大型机输出管理工作负载的高可用性且安全的 AWS 基础设施架构。

图表显示了以下工作流：

1. 批处理调度器启动批处理流程，并部署在 Amazon EC2 上的多个可用区以实现高可用性 (HA)。

> **Note**
>
> 这种模式不包括批处理调度器的实现。有关实施的详细信息，请参阅调度程序的软件供应商文档。

2. 大型机批处理作业（用 JCL 或 COBOL 等编程语言编写）使用核心业务逻辑来处理和生成打印输出，例如账单、身份证和贷款对账单。批处理任务部署在 Amazon 上， EC2 跨两个可用区进行高可用性。它使用 Rocket Software Print Exit API 将打印输出路由到 LRS VPSX/MFI 进行数据预处理。
3. LRS VPSX/MFI 打印服务器部署在亚马逊上， EC2 跨两个可用区，用于 HA（主用-备用冗余对）。它使用 Amazon EBS 作为操作数据存储。Network Load Balancer 对 LRS VPS EC2 X/MFI 实例执行运行状况检查。如果活动实例处于运行状况不佳状态，负载均衡器会将流量路由到其他可用区中的热备用服务器实例。在每个实例中，打印请求都保存在本地的 LRS Job Queue 中。 EC2 如果发生故障，必须重新启动故障实例，然后 LRS 服务才能继续处理打印请求。

> **Note**
>
> LRS VPSX/MFI 还可以在打印机队列级别执行运行状况检查。有关更多信息，请参阅此模式的其他信息部分中的打印机队列运行状况检查。

4. LRS PageCenter X 输出管理部署在 Amazon 上， EC2 跨两个可用区，用于 HA（主用-备用冗余对）。它使用 FSx 适用于 Windows 的 Amazon 文件服务器作为操作数据存储。如果活动实例处于不健康状态，则负载均衡器会对 LRS PageCenter X EC2 实例执行运行状况检查，并将流量路由到其他可用区的备用实例。
5. N etwork Load Balancer 提供 DNS 名称来将 LRS VPSX/MFI 服务器与 LRS X 集成 PageCenter

> **Note**
>
> LRS PageCenter X 支持第 4 层负载均衡器。

6. LRS PageCenter X 使用 Amazon FSx for Windows File Server 作为操作数据存储，部署在两个可用区中用于 HA。LRS PageCenter X 只能识别文件共享中的文件，而不能理解外部数据库中的文件。
7. AWS Managed Microsoft AD 与 LRS/DIS 协同使用，以执行输出管理工作流身份验证和授权。有关更多信息，请参阅其他信息部分中的打印输出身份验证和授权。

工具

Amazon Web Services

- 适用于 Microsoft Active Directory 的 AWS Directory Service 允许目录感知工作负载和 AWS 资源使用 Amazon Web Services Cloud 中的 Microsoft Active Directory。
- 亚马逊弹性区块存储 (Amazon EBS) E lastic Block Store 提供块级存储卷，用于亚马逊弹性计算云 (Amazon) 实例。 EC2
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- 弹性负载均衡(ELB) 将传入的应用程序或网络流量分配到多个目标。例如，您可以在一个或多个可用区的 Amazon EC2 实例、容器和 IP 地址之间分配流量。此模式使用网络负载均衡器。
- Amazon FSx 提供的文件系统支持行业标准的连接协议，并可在 AWS 区域之间实现高可用性和复制。此模式使用 FSx 适用于 Windows 文件服务器的 Amazon。
- Amazon Relational Database Service (Amazon RDS) 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展关系数据库。

其他工具

- LRS PageCenter X 软件提供可扩展的文档和报告内容管理解决方案，通过自动索引、加密和高级搜索功能，帮助用户从信息中获得最大价值。
- 由 LRS 和 Rocket Software 共同开发的 LRS VPSX/MFI（Micro Focus Interface）可捕获火箭软件 JES 线轴的输出，并将其可靠地传送到指定的打印目的地。
- LRS/Queue is a transmission agent that’s TCP/IP based. LRS VPSX/MFI uses LRS/Queue通过 Rocket Software JES 打印出口编程接口收集或捕获打印数据。
- LRS 目录集成服务器 (LRS/DIS) 用于打印工作流期间的身份验证和授权。
- Rocket Enterprise Server 是大型机应用程序的应用程序部署环境。它为使用任何版本的 Rocket Enterprise Developer 迁移或创建的大型机应用程序提供了运行时环境。

操作说明

设置 Rocket 运行时并部署大型机批处理应用程序

Task	描述	所需技能
设置运行时系统并部署演示应用程序。	要在亚马逊上设置 Rocket Enterprise Server EC2 并部署 Rocket 软件 BankDemo 演示应用程序，请按照 AWS 大型机现代化用户指南中的说明进行操作。 该 BankDemo 应用程序是一个大型机批处理应用程序，用于创建然后启动打印输出。	云架构师

在亚马逊上设置 LRS 打印服务器 EC2

Task	描述	所需技能
创建亚马逊 EC2 Windows 实例。	要启动亚马逊 EC2 Windows 实例，请按照亚马逊 EC2 文档中启动亚马逊 EC2 实例中的说明进行操作。使用与 LRS 产品许可证相同的主机名。 您的实例必须满足 LRS VPSX/MFI 的以下硬件和软件要求： • CPU - 双核 • 内存 - 16 GB • 驱动器 - 500 GB • 最小 EC2 实例 — `m5.xlarge` • OS - Windows	云架构师

Task	描述	所需技能
	• 软件 - Internet 信息服务 (IIS) 或 Apache **Note** 上述硬件和软件要求适用于小型打印机群（大约 500-1000）。要获取完整的要求，请咨询您的 LRS 和 AWS 联系人。 1. 创建 Windows 实例时，请确认 EC2 主机名与用于 LRS 产品许可证的主机名相同。 2. 按照 Amazon EC2 文档中步骤 2：连接到您的实例中的说明连接到您的实例。 EC2 3. 在 Windows 开始菜单上，找到并打开服务器管理器。 4. 在服务器管理器中，选择控制面板、快速启动、添加角色和功能，然后选择服务器角色。 5. 在“服务器角色”中，选择 WebServer (IIS)，然后选择“应用程序开发”。 6. 在“应用程序开发”中，选中CGI复选框。 7. 要安装 CGI，请按照 Windows 服务器管理器添加	

Task	描述	所需技能
	角色和功能向导中的说明进行操作。 8. 在 EC2 实例的 Windows 防火墙中打开端口 5500，进行 LRS/队列通信。	
在实例上安装 LRS VPSX/MFI。 EC2	1. Connect 连接到您的 EC2 实例。 2. 从您应该已收到的 LRS 电子邮件中打开产品下载页面的链接。 **Note** LRS 产品通过电子文件传输 (EFT) 分发。 3. 下载 LRS VPSX/MFI，并解压缩文件（默认文件夹：c:\LRS）。 4. 要安装 LRS VPSX/MFI，请从解压缩的文件夹中启动 LRS 产品安装程序。 5. 在选择功能菜单上，选择 VPSX® 服务器，然后选择下一步以启动安装过程。安装完成后，您将收到一条成功消息。	云架构师

Task	描述	所需技能
安装 LRS/队列。	1. 连接到你的 Rocket 企业服务器 EC2 实例。 2. 从您应该已收到的 LRS 电子邮件中打开指向 LRS 产品下载页面的链接，下载 LRS/Queue，然后解压缩文件。 3. 导航到文件的下载位置，然后启动 LRS 产品安装程序以安装 LRS/队列。 4. 按照 LRS 产品安装程序中的说明完成安装过程。	云架构师
安装 LRS/DIS。	LRS/DIS product often is included in LRS VPSX installation. However, if LRS/DIS未与 LRS VPSX 一起安装，请使用以下步骤进行安装： 1. 连接到你的 LRS VPS EC2 X/MFI 实例。 2. 从您应该收到的 LRS 电子邮件中打开指向 LRS 产品下载页面的链接，下载 LRS/DIS，然后解压缩文件。 3. 导航到文件下载位置，然后启动 LRS 产品安装程序。 4. 在 LRS 产品安装程序中，展开 LRS 杂项工具，选择 LRS DIS，然后选择下一步。 5. 按照 LRS 产品安装程序中的其余说明完成安装过程。	云架构师

Task	描述	所需技能
创建目标组。	遵循为您的网络负载均衡器创建目标组中的说明创建目标组。创建目标组时，将 LRS VPSX/MFI EC2 实例注册为目标： 1. 在指定组详细信息页面上，对于选择目标类型，选择实例。 2. 对于协议，选择 TCP。 3. 对于端口，选择 5500。 4. 在注册目标页面的可用实例部分中，选择 LRS VPS EC2 X/MFI 实例。	云架构师
创建网络负载均衡器。	要创建网络负载均衡器，请遵循弹性负载均衡器文档中的说明进行操作。您的 Network Load Balancer 将流量从 Rocket Enterprise 服务器路由到 LRS VPS EC2 X/MFI 实例。 创建网络负载均衡器时，在侦听器和路由页面上选择以下值： 1. 对于协议，选择 TCP。 2. 对于端口，选择 5500。 3. 对于默认操作，选择转发至您之前创建的目标组。	云架构师

将 Rocket 企业服务器与 LRS/Queue and LRS VPSX/MFI

Task	描述	所需技能
为 LRS/队列集成配置 Rocket 企业服务器。	1. 按照亚马逊 EC2 文档中的说明连接到您的 Rocket Enterprise Server EC2 实例。 2. 在 Windows 的 “开始” 菜单上，打开 Rocket Enterprise 服务器管理用户界面。 3. 在菜单栏中，选择 NATIVE。 4. 在导航窗格中，选择目录服务器，然后选择 BANKDEMO 作为您的企业服务器区域。 5. 从左侧导航窗格中的常规向下滚动到其他部分，将环境变量（LRSQ_ADDRESS 、LRSQ_PORT 、LRSQ_COMMAND ）配置为指向 LRSQ。 • 对于 LRSQ_ADDRESS，输入您之前创建的网络负载均衡器的 IP 地址或 DNS 名称。 • 对于 LRSQ_PORT，输入 VPSX LRSQ 侦听器端口 (5500)。 • 对于 LRSQ_COMMAND，请输入 LRSQ 可执行文件的路径位置。	云架构师

Task	描述	所需技能
	Note LRS 目前支持 DNS 名称的最大字符限制为 50。如果您的 DNS 名称超过 50 个字符，您可以使用网络负载均衡器的 IP 地址作为替代。	

Task	描述	所需技能
为 LRS VPSX/MFI 集成配置 Rocket Enterprise 服务器。	1. 将该VPSX_MFI_R2 文件夹从 LRS VPSX/MFI 安装程序复制到 Rocket Enterprise Server 的位置，网址为。C\BANKDEMO\print 2. 按照亚马逊 EC2 文档中的说明连接到您的 Rocket Enterprise Server EC2 实例。 3. 在 Windows 的 “开始” 菜单上，打开 Rocket Enterprise 服务器管理用户界面。 4. 在菜单栏上，选择 NATIVE。 5. 在左侧导航窗格中，选择目录服务器，然后选择 BANKDEMO。 6. 在 BANKDEMO 下，选择 JES。 7. 在 JES 程序路径下，添加 C\BANKDEMO\print 中的 DLL(VPSX_MFI_R2) 路径。	云架构师

设置打印队列和打印用户

Task	描述	所需技能
将 Rocket 软件打印退出模块与 Rocket Enterprise Server 批处理打印机服务器执行过程相关联	1. 按照亚马逊 EC2 文档中的说明连接到您的 Rocket Enterprise Server EC2 实例。	云架构师

Task	描述	所需技能
	2. 在 Windows 的 “开始” 菜单上，打开 Rocket Focus 企业服务器管理用户界面。 3. 在菜单栏上，选择 NATIVE。 4. 在左侧导航窗格中，选择目录服务器，然后选择 BANKDEMO。 5. 在 BANKDEMO 下，选择 JES，然后向下滚动到打印机。 6. 在打印机中，将 Rocket Software 打印退出模块（LRSPRTE6 用于批处理）与 Rocket Enterprise Server 批处理打印机服务器执行流程 (SEP) 关联 这可以将打印输出路由至 LRS VPSX/MFI。 有关配置的更多信息，请参阅 OpenText Micro Focus 文档中的使用退出。	

Task	描述	所需技能
在 LRS VPSX/MFI 中创建打印输出队列并将其与 LRS X 集成 PageCenter	1. 连接到你的 LRS VPS EC2 X/MFI 实例。 2. 在 Windows 开始菜单上，打开 VPSX Web Interface。 3. 在导航窗格中，选择打印机。 4. 选择添加，再选择添加用户。 5. 在打印机配置页上，对于打印机名称，输入 Local。 6. 要获取 VPSX ID，请输入。VPS1 7. 对于 CommType，请选择 TCPIP/ LRSQ。 8. 在 “主机/IP 地址” 中，输入 LRS X 实例前面的 Network L PageCenter oad Balancer 的 IP 地址。 EC2 9. 对于远程端口，输入 5800。 10对于远程队列，输入将存储输出的 LRS PageCenter X 文档文件夹的名称。 11选择 添加。	云架构师

Task	描述	所需技能
在 LRS VPSX/MFI 中创建打印用户。	1. 连接到你的 LRS VPS EC2 X/MFI 实例。 2. 在 Windows 开始菜单上，打开 VPSX Web Interface。 3. 在导航窗格中，选择安全，然后选择用户。 4. 在用户名列中，选择 admin，然后选择复制。 5. 在“用户配置文件维护”窗口中，在“用户名”中输入用户名（例如 PrintUser）。 6. 对于说明，输入简短说明（例如，测试打印的用户）。 7. 选择更新。这将创建一个打印用户（例如，PrintUser）。 8. 在导航窗格中的用户下， 选择您创建的新用户。 9. 在命令菜单上，选择安全。 10. 在安全规则页面上，选择所有适用的打印机安全和作业安全选项，然后选择保存。 11. 要将新的打印用户添加到管理员组，请在导航窗格中选择安全性，然后选择配置。 12. 在安全配置窗口中，将新的打印用户添加到管理员列中。	云架构师

在亚马逊上设置 LRS PageCenter X 服务器 EC2

Task	描述	所需技能
创建亚马逊 EC2 Windows 实例。	按照亚马逊 EC2 文档中的步骤 1：启动实例中的说明启动亚马逊 EC2 Windows 实例。使用与 LRS 产品许可证相同的主机名。 您的实例必须满足 LRS PageCenter X 的以下硬件和软件要求： • CPU - 双核 • 内存 - 16 GB • 驱动器 - 500 GB • 最小 EC2 实例 — `m5.xlarge` • OS - Windows • 软件 - IIS 或 Apache **Note** 上述硬件和软件要求适用于小型打印机群（大约 500—1000）。要获取完整的要求，请咨询您的 LRS 和 AWS 联系人。 1. 创建 Windows 实例时，请确认 EC2 主机名与用于 LRS 产品许可证的主机名相同。	云架构师

Task	描述	所需技能
	2. 按照 Amazon EC2 文档中的说明连接到您的 EC2 实例。 3. 在 Windows 开始菜单上，打开服务器管理器。 4. 在服务器管理器中，选择控制面板、快速启动、添加角色和功能，然后选择服务器角色。 5. 在 “服务器角色” 中，选择 WebServer (IIS)，然后选择 “应用程序开发”。 6. 在“应用程序开发”中，选中CGI复选框。 7. 要安装 CGI，请按照 Windows 服务器管理器添加角色和功能向导中的说明进行操作。 8. 在 5800 端口上打开入站TCP/IP traffic in the Windows firewall of the EC2 instance. LRS VPSX uses the TCPIP/LRSQ协议端口 5800 以与 LRS X 通信。 PageCenter	

Task	描述	所需技能
在 EC2 实例上安装 LRS PageCenter X。	1. Connect 连接到您的 EC2 实例。 2. 从您应该已收到的 LRS 电子邮件中打开产品下载页面的链接。 **Note** LRS 产品通过电子文件传输 (EFT) 分发。 3. 下载 LRS PageCenter X，然后解压缩文件（默认文件夹：`c:\LRS`）。 4. 要安装 LRS PageCenter X，请从解压缩的文件夹中启动 LRS 产品安装程序。 5. 在“选择功能”菜单上，选择 PageCenterX，然后选择“下一步”开始安装过程。安装完成后，您将收到一条成功消息。	云架构师

Task	描述	所需技能
安装 LRS/DIS。	LRS/DIS product often is included in LRS VPSX installation. However, if LRS/DIS未与 LRS VPSX 一起安装，请使用以下步骤进行安装： 1. 连接到您的 LRS PageCenter X EC2 实例。 2. 从您应该收到的 LRS 电子邮件中打开 LRS 产品下载页面的链接，下载 LRS/DIS，然后解压缩文件。 3. 导航到文件下载位置，然后启动 LRS 产品安装程序。 4. 在 LRS 产品安装程序中，展开 LRS 杂项工具，选择 LRS DIS，然后选择下一步。 5. 按照 LRS 产品安装程序中的其余说明完成安装过程。	云架构师

Task	描述	所需技能
创建目标组。	遵循为您的网络负载均衡器创建目标组中的说明创建目标组。创建目标组时，将 LRS PageCenter X EC2 实例注册为目标： 1. 在指定组详细信息页面上，对于选择目标类型，选择实例。 2. 对于协议，选择 TCP。 3. 对于端口，选择 5800。 4. 在注册目标页面的可用实例部分中，选择 LRS PageCenter X EC2 实例。	云架构师
创建网络负载均衡器。	要创建网络负载均衡器，请遵循弹性负载均衡器文档中的说明进行操作。您的 Network Load Balancer 将流量从 LRS VPSX/MFI 路由到 LRS X 实例。 PageCenter EC2 创建网络负载均衡器时，在侦听器和路由页面上选择以下值： 1. 对于协议，选择 TCP。 2. 对于端口，选择 5800。 3. 对于默认操作，选择转发给您之前创建的目标组。	云架构师

在 LRS X 中设置输出管理功能 PageCenter

Task	描述	所需技能
在 LRS PageCenter X 中启用导入功能。	您可以使用 LRS PageCenter X Import 功能，通过作业名称或表单 ID 等标准来识别 LRS PageCenter X 上的输出。然后，您可以将输出路由到 LRS PageCenter X 中的特定文件夹。 要启用“导入”函数，请执行以下操作： 1. 按照亚马逊 EC2 文档中的说明连接到您的 LRS PageCenter X EC2 实例。 2. 在 Windows 开始菜单上，打开 PCX Web Interface。 3. 在文件夹资源管理器中，选择管理员。 4. 在配置页面上，选择高级、导入参数。 5. 在导入参数部分中，选中高级导入复选框。 6. 要提交更改，请选择更新。	云架构师
配置文档保留策略。	LRS PageCenter X 使用文档保留政策来决定在 L PageCenter RS X 中保留文档多长时间。 要配置文档保留策略，请执行以下操作：	云架构师

Task	描述	所需技能
	1. 连接到您的 LRS PageCenter X EC2 实例。 2. 在 Windows 开始菜单上，打开 PCX Web Interface。 3. 在文件夹资源管理器中，选择管理员。 4. 在管理页面上，选择存档组列表/常规管理员，然后选择保留策略。 5. 在保留策略部分中，选择添加以创建保留策略。 6. 在保留策略信息页面上，输入保留策略名称、说明和文档保留期限。 7. 选择确定以保存您的更改并创建策略。	

Task	描述	所需技能
创建一条规则，将输出文档路由到 LRS PageCenter X 中的特定文件夹。	在 LRS PageCenter X 中，目标决定当报告定义调用此目标时，输出将发送到的文件夹路径。在此示例中，基于报告定义中的表单 ID 文件夹创建一个文件夹，并将输出保存到该文件夹。 1. 连接到您的 LRS PageCenter X EC2 实例。 2. 在 Windows 开始菜单上，打开 PCX Web Interface。 3. 在文件夹资源管理器中，依次选择管理员、高级导入和目标。 4. 在“目标”部分中，选择添加以打开目标维护窗体。 5. 在“目标维护”表单中，输入以下值： • 目标名称 - 表单 • 说明 - 目标的说明，例如基于表单的文件夹结构 • 目标类型 - 文件夹 • 文件夹参数 -导入文件夹路径（文档到达时将在 PageCenter X 中创建的文件夹路径；例如，该路径**/Test/&FORM/&IMPORTDATE/&IMPORTTIME** 将创建基本文件**Test**夹、基于名为 Form-ID 的子文件夹**STD**、基于导入日期的	云架构师

Task	描述	所需技能
	子文件夹，然后根据导入时间创建子文件夹） • 文档名称 - 当文档存储在文件夹中时分配给文档的动态名称。 6. 在下拉列表中，选择保留策略。例如，选择 Year1 可将文档保留 1 年。 7. 选择确定以保存您的更改。	

Task	描述	所需技能
创建作业定义	1. 连接到您的 LRS PageCenter X EC2 实例。 2. 在 Windows 开始菜单上，打开 PCX Web Interface。 3. 在文件夹资源管理器中，选择管理员、高级导入、报告定义，然后 选择添加。 4. 在报告定义维护页面的常规选项卡上，输入报告定义名称。 5. 在常规选项卡上的字段下，您可以指定选择条件，例如作业名称、表单、类别和作者。例如，您可以输入 MFIDEMO 的作业名称。作业名称值是将生成打印输出的批处理作业的名称。 6. 在目标选项卡中的可用目标下，选择之前创建的目标 (表单)。 7. 选择添加将表单目标添加为分配的目标。 **Note** 此示例包括一个报告定义，其中由 MFIDEMO 生成并路由到 LRS PageCenter X 的输出保存在目标定义	云架构师

Task	描述	所需技能
	中定义的文件夹结构中。	

设置输出管理的身份验证和授权

Task	描述	所需技能
使用用户和组创建 AWS Managed Microsoft AD 域。	1. 若要在 AWS Managed Microsoft AD 上创建目录，请按照创建 AWS Managed Microsoft AD 目录中的说明进行操作。 2. 要部署 EC2 实例（活动目录管理器）并安装活动目录工具来管理您的 AWS 托管 Microsoft AD，请按照步骤 3：部署 EC2 实例来管理您的 AWS 托管 Microsoft AD 中的说明进行操作。 3. 要连接到您的 EC2 实例，请按照 Amazon EC2 文档中的说明进行操作。 **Note** 当您连接到 EC2 实例时，在 Windows 安全窗口中，输入您在步骤 1 中创建的目录的管理员凭据。 4. 登录之后，在开始菜单中的 Windows 管理员工具下，选择活动目录用户及计算机。	云架构师

Task	描述	所需技能
	5. 要在活动目录域中创建打印用户，请按照创建用户中的说明进行操作。	
将 EC2 实例加入到 AWS 托管的 Microsoft AD 域中。	自动将 LRS VPSX/MFI 和 LRS X PageCenter 实例加入 EC2 您的 AWS 托管微软 AD 域（AWS 知识中心文档）或手动加入（AWS Di rectory Service 文档）。	云架构师
为 LRS X 实例配置 LRS/DIS 并将其与 AWS 托管的 Microsoft AD 集成。PageCenter EC2	1. 连接到您的 LRS PageCenter X EC2 实例。 2. 在 Windows 开始菜单上，打开 PCX Web Interface。 3. 在文件夹资源管理器中，选择管理员。 4. 在配置页面的安全参数部分中，对于安全类型，选择 LRS/DIS。 5. 在安全参数部分中输入您对其余选项的首选项。 6. 在 Windows 的 “开始” 菜单上，打开 PageCenterX 文件夹，选择 “服务器启动”，然后选择 “服务器停止”。 7. 使用您的 Active Directory 用户名和密码登录 LRS PageCenter X。	云架构师

Task	描述	所需技能
配置导入组以将输出从 LRS VPSX 导入到 LRS X。PageCenter	1. 连接到您的 LRS PageCenter X EC2 实例。 2. 在 Windows 开始菜单上，打开 PCX Web Interface。 3. 在文件夹资源管理器中，选择管理员、安全管理员、组。 4. 在组部分的中，选择添加以打开组首选项表单。 5. 在组首选项表单中，输入组名称和说明的值。 6. 展开常规选项，然后选中导入复选框。 7. 选择确定以保存您的更改。	云架构师
向安全组添加规则。	1. 打开导入组的上下文（右键单击）菜单。 2. 选择高级，然后选择安全性。 3. 在“安全性”部分中，选择导入，然后选中子文件夹复选框。 4. 选择应用以保存更改。	云架构师

Task	描述	所需技能
在 LRS PageCenter X 中创建一个用户来执行从 LRS VPSX/MFI 导入输出。	在 LRS PageCenter X 中创建用户以执行输出导入时，用户名应与 LRS VPSX/MFI 中打印输出队列的 VPSX ID 相同。在本示例中，VPSX ID 为。VPS1 1. 连接到您的 LRS PageCenter X EC2 实例。 2. 在 Windows 开始菜单上，打开 PCX Web Interface。 3. 在文件夹资源管理器中，选择管理员、安全管理员、用户。 4. 选择添加以打开用户配置文件维护表单。 5. 在 “用户配置文件维护” 中，在 “用户名” 栏中输入VPS1。	云架构师

Task	描述	所需技能
将 LRS PageCenter X 导入用户添加到 “仅限导入” 组中。	要为从 LRS VPSX 导入到 LRS X 的文档提供必要的权限，请执行 PageCenter以下操作： 1. 连接到您的 LRS PageCenter X EC2 实例。 2. 在 Windows 开始菜单上，打开 PCX Web Interface。 3. 在文件夹资源管理器中，选择管理员、安全管理员、组。 4. 在“组”部分中，打开仅导入组的上下文（右键单击）菜单，然后选择高级、安全性。 5. 在 “文件夹安全记录” (ImportOnly) 页面上，选择 “用户” 选项卡。 6. 在 “用户” 选项卡的 “名称” 下，VPS1从下拉列表中选择 “用户”，然后选择 “应用”。	云架构师

Task	描述	所需技能
配置LRS/DIS with AWS Managed Microsoft AD for the LRS VPSX/MFI EC2 实例。	1. 连接到你的 LRS VPS EC2 X/MFI 实例。 2. 在 Windows 开始菜单上，打开 VPSX Web Interface。 3. 在导航窗格中，选择存储，然后选择配置。 4. 在安全配置页面的安全参数部分中，对于安全类型，选择 LRS/DIS（外部）。 5. 在“安全参数”部分中输入您对其余选项的首选项。 6. 在 Windows 开始菜单上，打开 LRS 输出管理文件夹，选择服务器启动，然后选择服务器停止。 7. 使用您的活动目录用户名和密码登录 LRS VPSX/MFI。	云架构师

将 FSx 适用于 Windows 文件服务器的 Amazon 配置为 LRS PageCenter X 的操作数据存储

Task	描述	所需技能
为 LRS PageCenter X 创建文件系统。	要在多可用区环境中使用 Amazon FSx for Windows 文件服务器作为 LRS PageCenter X 的操作数据存储，请按照步骤 1：创建文件系统中的说明进行操作。	云架构师
将文件共享映射到 LRS PageCenter X EC2 实例。	要将上一步中创建的文件共享映射到 LRS PageCenter X EC2 实例，请按照步骤 2：将您的文件共享映射到运行	云架构师

Task	描述	所需技能
	Windows Server 的 EC2 实例中的说明进行操作。	
将 LRS PageCenter X 控制目录和主文件夹目录映射到 Amazon FSx 网络共享驱动器。	1. 按照亚马逊 EC2 文档中的说明连接到您的 LRS PageCenter X EC2 实例。 2. 在 Windows 开始菜单上，打开 PCX Web Interface。 3. 在文件夹资源管理器中，依次选择管理员和配置。 4. 在配置页面上，选择目录，然后选择控制目录。 5. 在控制目录中，输入 `\\FSx file share DNS name\share\cntl` 。 6. 在主文件夹目录中，输入 `\\FSx file share DNS name\share\mstr` 。	云架构师

测试输出管理工作流

Task	描述	所需技能
从 Rocket Softwar BankDemo e 应用程序启动批量打印请求。	1. 在 Rocket Enterprise Server 实例中打开 32700 终端模拟器。 EC2 2. 通过运行命令连接到 BankDemo 应用程序`connect 127.0.0.1:9278` 。 3. 在 BankDemo 命令行界面上，对于用户 ID，输入	测试工程师

Task	描述	所需技能
	B0001。对于密码，输入非空白密钥。 4. 对于请求打印报表选项，在空行中输入 X。 5. 在发送语句依据部分中，对于邮件，输入 Y，然后按 F10。	

Task	描述	所需技能
在 LRS PageCenter X 中检查打印输出。	1. 按照亚马逊 EC2 文档中的说明连接到您的 LRS PageCenter X EC2 实例。 2. 在 Windows 开始菜单上，打开 PCX Web Interface。 3. 在导航窗格中，打开测试文件夹，打开 STD 文件夹，然后打开包含作业运行日期的文件夹，例如 `08-03-2023` （年-月-日）。 **Note** 这与故事中定义的文件夹结构相同。创建一条规则，将输出文档路由到 LRS PageCenter X 中的特定文件夹。 4. 打开 `formtest-STD.txt` 文件。 现在，您可以看到账户对账单的打印输出，其中包含账户号码、描述、日期、金额和余额列。有关示例，请参阅此模式的 `batch_print_output` 附件。	测试工程师

相关资源

- LRS

- Advanced Function Presentation 数据流（IBM 文档）
- 线路条件数据流 (LCDS)（比较文档）
- 使用 Micro Focus 在 AWS 上为企业大型机工作负载提供支持（博客文章）
- 在 AWS 上实现大型机在线打印工作负载的现代化（AWS Prescriptive Guidance）
- 在 AWS 上实现大型机批量打印工作负载的现代化 (AWS Prescriptive Guidance)

其他信息

注意事项

在现代化改造过程中，您可能会考虑对大型机批处理和在线流程及其生成的输出进行各种配置。每个使用大型机平台的客户和供应商都根据直接影响打印的特殊要求对平台进行了定制。例如，您当前的平台可能会将 IBM AFP 数据流或 Xerox LCDS 合并到当前工作流中。此外，大型机回车控制字符和通道命令字可能会影响打印页面的外观，可能需要特殊处理。作为现代化规划过程的一部分，我们建议您评测并了解特定打印环境中的配置。

打印数据采集

Rocket Software Print Exit 传递必要的信息，让 LRS VPSX/MFI 能够有效地处理线轴文件。该信息由在相关控制块中传递的字段组成，如下所示：

- JOBNAME
- OWNER (USERID)
- DESTINATION
- FORM
- FILENAME
- WRITER

LRS VPSX/MFI 支持以下用于从 Rocket Enterprise Server 捕获数据的大型机批处理机制：

- BATCH COBOL print/spool processing using standard z/OS JCL SYSOUT DD/OUTPUT 语句。
- BATCH COBOL print/spool processing using standard z/OS JCL CA-SPOOL SUBSYS DD 语句。
- IMS/COBOL print/spool使用 CBLTDLI 接口进行处理。有关支持的方法和编程示例的完整列表，请参阅产品许可证附带的 LRS 文档。

打印机队列运行状况检查

LRS VPSX/MFI（LRS LoadX）可以执行深入的运行状况检查，包括设备管理和操作优化。设备管理可以检测打印机设备中的故障，并将打印请求路由到正常运行的打印机。有关打印机队列的深入运行状况检查的详细信息，请参阅产品许可证附带的 LRS 文档。

打印身份验证和授权

LRS/DIS enables LRS applications to authenticate user IDs and passwords by using Microsoft Active Directory or a Lightweight Directory Access Protocol (LDAP) server. In addition to basic print authorization, LRS/DIS还可以在以下用例中应用粒度级别的打印安全控制：

- 管理谁可以浏览打印机作业。
- 管理其他用户作业的浏览级别。
- 管理操作任务，例如，命令级安全性，例如保留或释放、清除、修改、复制和重新路由。安全性可以通过用户 ID 或组进行设置，类似于活动目录安全组或 LDAP 组。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 Amazon Q 开发人员对 CardDemo 大型机应用程序进行现代化改造

由 Santosh Kumar Singh (AWS) 和 Cheryl du Preez (AWS) 创作

摘要

适用于大型机的 Amazon Q Developer 转换是一款由 AI 驱动的代理，旨在加速大型机应用程序的现代化。它使用生成式 AI 来简化大型机现代化流程。它可以自动执行复杂的任务，例如：遗留代码分析、大型机文档、将整体应用程序分解到业务领域以及代码重构。它通过自动执行复杂任务（例如应用程序分析和迁移顺序规划）来加速现代化项目。在分解单体应用程序时，Amazon Q Developer 会智能地对大型机应用程序转换进行排序，从而帮助您并行转换业务功能。Amazon Q Developer 可以加快决策制定速度，提高运营灵活性和迁移效率。

此模式提供了一些 step-by-step说明，可帮助您使用示例开源大型机应用程序来测试 Amazon Q De CardDemoveloper 的大型机转换功能。

> **Note**
>
> Amazon Q Developer 的转换功能处于预览版，可能会发生变化。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- AWS IAM Identity Center，已启用
- 允许管理员使用 Amazon Q 开发者控制台的@@ 权限
- 允许管理员接受 Amazon Q 开发者转型网络体验的连接请求的@@ 权限

限制

- Amazon Q Developer 仅在某些版本中可用 AWS 区域。有关更多信息，请参阅 Amazon Q 开发者支持的区域。
- 适用于大型机的 Amazon Q Developer 转换仅支持用于代码分析、文档生成和分解的 IBM z/OS 大型机文件。有关支持的文件类型的列表，请参阅支持的大型机应用程序转换文件类型。
- Amazon Q Developer 中有大型机转型功能的服务配额。有关更多信息，请参阅大型机转型功能的服务配额。

- 要在共享工作空间上进行协作，所有用户都必须是与您的 Amazon Q Developer 转换网络体验实例关联的 IAM 身份中心同一个实例的注册用户。
- 亚马逊简单存储服务 (Amazon S3) Service 存储桶和 Amazon Q Developer 订阅必须相同。 AWS 账户

架构

下图显示了您在此模式中设置的架构。

图表显示了以下工作流：

1. Amazon Q Developer 使用连接器访问存储在 Amazon S3 存储桶中的 CardDemo 大型机应用程序。
2. Amazon Q 开发 AWS IAM Identity Center 者用于管理用户访问和身份验证。该系统对身份验证、授权、加密和访问管理实施了多层安全控制，以帮助在处理过程中保护代码和工件。用户通过聊天界面与 Amazon Q 开发者代理互动。他们可以用英语输入目的和目标。
3. 代理解释用户的指令，创建工作计划，将任务划分为可执行的任务，然后自动对其进行操作。用户可以查看和批准转换。转换任务包括以下内容：
   - 代码分析 — Amazon Q Developer 会分析每个文件中的代码，以了解文件名、文件类型、代码行及其路径等详细信息。代理分析源代码、运行分类、创建依赖关系映射并识别任何缺失的工件。
   - 生成文档 — Amazon Q Developer 为大型机应用程序生成文档。通过分析代码，它可以自动创建应用程序的详细文档，包括对旧系统中存在的业务逻辑、流程、集成和依赖关系的描述。
   - 分解 — Amazon Q Developer 将代码分解为多个域，以考虑程序和组件之间的依赖关系。这有助于确保相关的文件和程序适当地分组到同一个域中。它还有助于在分解过程中保持应用程序逻辑的完整性。
   - 迁移浪潮规划 — 根据您在分解阶段创建的域，Amazon Q Developer 会根据建议的现代化顺序生成迁移浪潮计划。
   - 代码重构 — Amazon Q Developer 将所有或选定域文件中的代码重构为 Java 代码。此步骤的目标是保留应用程序的关键业务逻辑，同时将其重构为经过云优化的现代化 Java 应用程序。
4. Amazon Q Developer 将重构后的代码以及其他相关计划和文档存储在 Amazon S3 存储桶中。

工具

AWS 服务

- AWS IAM Identity Center帮助您集中管理对所有应用程序 AWS 账户 和云应用程序的单点登录 (SSO) 访问权限。
- Amazon Q Developer 是一款基于人工智能的生成式对话助手，可以帮助您理解、构建、扩展和操作 AWS 应用程序。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

代码存储库

由创建的开源CardDemo大型机应用程序 AWS 可以帮助您开始大型机现代化。

最佳实践

- 从小处着手 — 从不太复杂的小代码（15,000—20,000 行代码）开始，了解 Amazon Q Developer 是如何分析和转换大型机应用程序的。
- 与人类专业知识相结合 — 使用 Amazon Q Developer 作为加速器，同时应用人类专业知识以获得最佳结果。
- 彻底审查和测试 — 务必仔细检查转换后的代码，并运行全面的测试以验证转换后的功能等效性。
- 提供反馈-要提供反馈和改进建议，请使用中的发送反馈按钮 AWS Management Console 或使用创建案例AWS 支持。有关更多信息，请参阅创建支持案例。您的意见对于服务增强和 future 发展非常有价值。

操作说明

准备大型机应用程序

Task	描述	所需技能
创建存储桶。	使用订阅 Amazon Q Developer AWS 账户 的相同方式创建 Amazon S3 存储桶。您使用此存储桶存储大型机应用程序，Amazon Q Developer 使用此存储段存储重构后的代码和其他与转换相关的文件。	常规 AWS

Task	描述	所需技能
	有关说明，请参阅 Amazon S3 文档中的创建存储桶。	
准备大型机应用程序示例。	1. 输入以下命令将CardDemo存储库克隆到本地工作站： `git clone https://github.com/aws-samples/aws-mainframe-modernization-carddemo.git` 2. 创建一个名为的新文件夹`carddemo`。 3. 将包含大型机源代码的`app`文件夹从克隆的存储库复制到该`carddemo`文件夹。 4. 将该`carddemo`文件夹压缩为 ZIP 文件。 5. 将 ZIP 文件上传到您创建的 Amazon S3 存储桶。有关说明，请参阅 Amazon S3 文档中的上传对象。	应用程序开发者、 DevOps 工程师

配置 Amazon Q 开发者

Task	描述	所需技能
将用户添加到 IAM 身份中心。	将您的潜在用户添加到 IAM 身份中心。有关更多信息，请参阅 IAM 身份中心文档中的 Connect 员工用户。	AWS 管理员

Task	描述	所需技能
为用户订阅 Amazon Q 开发者专业版。	在目标账户中设置 Amazon Q Developer Pro 并订阅用户。有关说明，请参阅为用户订阅 Amazon Q Developer Pro，然后选择与您的类型相对 AWS 账户 应的选项。	AWS 管理员
在 Amazon Q 开发者中启用转换功能。	1. 登录您用来管理 Amazon Q 开发者的。 AWS Management Console AWS 账户 2. 打开 Amazon Q 开发者控制台。 3. 选择设置。 4. 在 Amazon Q 开发者：转换设置部分，选择编辑。 5. 启用 Amazon Q 开发者转换，然后选择保存。 6. 复制应用程序 URL。 7. 在新的浏览器选项卡中，粘贴 URL。这将打开 Amazon Q 开发者转型网页体验。	AWS 管理员

Task	描述	所需技能
配置用户对转换 Web 体验的访问权限。	每位用户必须执行以下步骤才能访问 Amazon Q 开发者转型网页体验： 1. 接受通过电子邮件发送的邀请。 2. 创建密码。 3. 通过执行以下操作来设置多因素身份验证： • 登录 Amazon Q 开发者转型网页体验。 • 出现提示时，注册多重身份验证 (MFA) 设备。按照屏幕上的说明进行操作。	应用程序开发人员、应用程序所有者
登录转换 Web 体验。	1. 登录到目标 AWS Management Console 中 AWS 账户。 2. 打开 Amazon Q 开发者控制台。 3. 选择 “QDev转换”。这将打开 Amazon Q 开发者转型网页体验。	应用程序开发人员、应用程序所有者
设置工作区。	设置一个工作空间，让用户可以在 Amazon Q Developer Pro 网络体验中进行协作。按照 Amazon Q 开发者文档中设置工作空间中的说明进行操作。	AWS 管理员

改造大型机应用程序

Task	描述	所需技能
创建转换作业。	创建转型任务以实现 CardDemo 大型机应用程序的现代化。有关说明，请参阅 Amazon Q 开发者文档中的创建和启动任务。当要求你设置目标时，选择分析代码、生成文档、分解代码、规划迁移顺序以及将代码转换为 Java。	应用程序开发人员、应用程序所有者
设置连接器。	与包含 CardDemo 大型机应用程序的 Amazon S3 存储桶建立连接器。此连接器允许 Amazon Q Developer 访问存储桶中的资源并执行连续的转换功能。有关说明，请参阅 Amazon Q 开发者文档中的设置连接器。	AWS 管理员
执行代码分析。	1. 在指定资产位置页面上，输入您上传的 `carddemo` ZIP 文件的 Amazon S3 存储桶路径。 2. 选择 “批准” 并发送至 Q。Amazon Q 开发者开始分析代码。 3. 在 “工作日志” 选项卡上监控状态。 4. 分析完成后，在左侧导航窗格的 “分析代码” 下，选择 “查看代码分析结果”。	应用程序开发人员、应用程序所有者

Task	描述	所需技能
	5. （可选）选择 “下载” 以下载完整的资产列表、缺失的源代码和依赖文件。 有关更多信息，请参阅 Amazon Q 开发者文档中的代码分析。	

Task	描述	所需技能
生成文档。	1. 在左侧导航窗格的 “生成文档” 下，选择 “选择文件并配置设置”。 2. 展开 COBO L 或 JCL，然后选择一个或多个文件。 3. 选择文档详细信息级别： • 摘要-提供范围内每个文件的高级概述。此外，还会给出每个文件的单行摘要。 • 详细的功能规格-提供大型机应用程序转换范围内每个文件的全面详细信息。一些细节包括逻辑和流程、确定的业务规则、数据流、依赖关系、输入和输出处理以及各种交易详细信息。 4. 选择 “发送到 Q”。 5. 在 “工作日志” 选项卡中监控进度。 **Note** 文档生成时间从几分钟到几小时不等，具体取决于文件数量和代码行数。 6. 完成后，选择查看文档结果以查看 Amazon S3 存储桶中的输出。	应用程序开发人员、应用程序所有者

Task	描述	所需技能
	7. 从 Amazon S3 存储桶中下载 zip 文件并查看生成的文档。 有关更多信息，请参阅 Amazon Q 开发者文档中的生成文档。	

Task	描述	所需技能
分解代码。	1. 在左侧导航窗格中，展开“分解代码”，然后选择“分解为域”。 2. 在操作列表中，选择创建域。 3. 输入新域的名称，例如`Bill payment`。 4. （可选）提供描述。 5. 在“查找文件”搜索栏中`CB00`，搜索并选择该文件。 6. 选择“标记为种子”。 7. 验证种子标志是否从“否”更改为“是”。 8. 选择创建。 9. 在“操作”列表中，选择“配置分解”。 10.调整域名大小，然后单击“保存”。 11.在“操作”列表中，选择“分解”。 12通过选择域名来查看分解输出。 13分解完成后，选择批准并发送至 Q。 有关分解和种子的更多信息，请参阅 Amazon Q Developer 文档中的分解。	应用程序开发人员、应用程序所有者

Task	描述	所需技能
规划迁移浪潮。	规划 CardDemo 应用程序的迁移浪潮。按照 Amazon Q Developer 文档中迁移浪潮规划中的说明查看和编辑迁移计划。	应用程序开发人员、应用程序所有者
重构代码。	将所有或选定域文件中的 CardDemo 大型机应用程序代码重构为 Java 代码。按照 Amazon Q 开发者文档中重构代码中的说明进行操作。	应用程序开发人员、应用程序所有者

故障排除

事务	解决方案
您会收到以下消息： `You do not have sufficient permission on your user to administer CodeWhisperer. Ask your account administrator to provide you with the required codewhisperer:ListProfiles permission to proceed.`	您必须拥有管理员权限才能订阅 Amazon Q Developer，并允许组织中的用户访问 Amazon Q 开发者。有关更多信息，请参阅允许管理员使用 Amazon Q 订阅控制台设置访问权限和所需策略。
订阅选项不适用于 Amazon Q Developer Pro，您会收到以下消息： `Unable to connect to organization instance of IAM Identity Center. Your application must be configured in the same AWS ## as your organization instance of`	在中 AWS Management Console，更改为启用 IAM 身份中心 AWS 区域 的位置。

事务	解决方案
`IAM Identity Center before you can assign users and groups.`	

相关资源

AWS 文档

- 大型机应用程序的转换（Amazon Q 开发者文档）
- Amazon Q 开发者：大型机转型（Amazon Q 开发者文档）

其他 AWS 资源

- 使用 Amazon Q Developer 生成式 AI 代理简化大型机现代化（AWS 博客文章）
- Amazon Q 开发者版 FAQs
- AWS IAM Identity Center FAQs

视频和教程

- Amazon Q 开发者简介：转型（AWS 技能生成器）
- AWS re: Invent 2024-使用 Amazon Q Developer 更快地实现大型机应用程序的现代化 () YouTube
- AWS re: Invent 2024-自动化迁移和现代化以加快转型 () YouTube
- AWS re: Invent 2024-丰田利用一代人工智能推动创新并提高运营效率 () YouTube

使用 Rocket Enterprise Server 和 LRS V AWS PSX/MFI 实现大型机批量打印工作负载的现代化

由 Shubham Roy (AWS)、Abraham Rondon (Micro Focus)、Guy Tucker (Levi, Ray and Shoup Inc) 和 Kevin Yung (AWS) 编写

摘要

此模式向您展示了如何使用 Rocket Enterprise Server 作为现代化大型机应用程序的运行时，将 LRS VPSX/MFI（Micro Focus Interface Interface）用作打印服务器，在亚马逊网络服务 (AWS) 云上实现业务关键型大型机批量打印工作负载的现代化。该模式基于更换平台大型机现代化方法。通过这种方法，您可以将大型机批处理任务迁移到亚马逊弹性计算云 (Amazon EC2)，并将大型机数据库（例如 DB2 适用于 z/OS 的 IBM）迁移到亚马逊关系数据库服务（Amazon RDS）。现代化打印工作流程的身份验证和授权由 Microsoft Active Directory 的 AWS Directory Service（也称为 AWS 托管 Microsoft AD）执行。LRS 目录信息服务器 (LRS/DIS) 与 AWS Managed Microsoft AD 集成。通过实现批量打印工作负载的现代化，您可以降低 IT 基础设施成本，减轻维护传统系统的技术债务，消除数据孤岛，通过 DevOps 模型提高灵活性和效率，并利用 AWS 云中的按需资源和自动化。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 大型机打印或输出管理工作负载
- 有关如何重建和交付在 Rocket Enterprise Server 上运行的大型机应用程序的基础知识（有关更多信息，请参阅 Rocket 文档中的 Rocket Enterprise Server 数据表。）
- LRS 云打印解决方案和概念的基础知识
- Rocket Enterprise Server 软件和许可证（有关更多信息，请联系 Rocket 销售人员
- LRS VPSX/MFI, LRS/Queue, and LRS/DIS 软件和许可证（欲了解更多信息，请联系 LRS 销售人员。）

> **Note**
>
> 有关大型机批量打印工作负载的配置注意事项的更多信息，请参阅此模式的 “其他信息” 部分中的注意事项。

产品版本

- 火箭企业服务器 6.0（产品更新 7）
- LRS VPSX/MFI 或更高版本

架构

源技术堆栈

- 操作系统 - IBM z/OS
- 编程语言 — 面向业务的通用语言 (COBOL)、作业控制语言 (JCL) 和客户信息控制系统 (CICS)
- 数据库 — DB2 适用于 z/OS 的 IBM 和虚拟存储访问方法 (VSAM)
- 安全 — 资源访问控制设施 (RACF)、适用于 z/OS 的 CA 绝密和访问控制设施 2 () ACF2
- 打印和输出管理 - IBM 大型机 z/OS 打印产品（IBM Tivoli Output Manager for z/OS、LRS 和 CA View）

目标技术堆栈

- 操作系统 — 在亚马逊上运行的微软 Windows 服务器 EC2
- 计算 — 亚马逊 EC2
- 编程语言 - COBOL、JCL 和 CICS
- 数据库 - Amazon RDS
- 安全性 - AWS 托管的 Microsoft AD
- 打印和输出管理 - AWS 上的 LRS 打印解决方案
- 大型机运行时环境 — Rocket 企业服务器

源架构

下图显示了大型机批量打印工作负载的典型当前状态架构：

图表显示了以下工作流：

1. 用户在基于 COBOL 编写的 IBM CICS 应用程序构建的互动系统 (SoE) 上执行业务事务。

2. SoE 调用大型机服务，该服务将业务交易数据记录在 system-of-records (SoR) 数据库中，例如 IBM DB2 for z/OS。
3. SoR 保留来自 SoE 的业务数据。
4. 批处理作业调度程序启动批处理作业以生成打印输出。
5. 批处理作业从数据库中提取数据，根据业务需求格式化数据，然后生成业务输出，例如账单、身份证或贷款报表。最后，批处理作业根据业务需求将输出路由到打印输出管理进行处理和输出交付。
6. 打印输出管理接收批处理作业的打印输出，然后将该输出传递到指定的目标，例如电子邮件、使用安全 FTP 的文件共享、使用 LRS 打印解决方案的物理打印机（如本模式所示）或 IBM Tivoli。

目标架构

下图显示了部署在 Amazon Web Services Cloud 中的大型机批量打印工作负载的架构：

图表显示了以下工作流：

1. 批处理作业调度程序启动批处理作业来创建打印输出，例如账单、身份证或贷款报表。
2. 大型机批处理作业（改编为 Amazon EC2）使用 Rocket Enterprise Server 运行时从应用程序数据库中提取数据，对数据应用业务逻辑，格式化数据，然后使用 Rocket Soft ware Print Exit（Micro Focus 文档）将数据发送到打印目的地。
3. 应用程序数据库（在 Amazon RDS 上运行的 SoR）保留打印输出的数据。
4. LRS VPSX/MFI printing solution is deployed on Amazon EC2 and its operational data is stored in Amazon Elastic Block Store (Amazon EBS). LRS VPSX/MFI uses the TCP/IP-based LRS/Queue 传输代理用于通过 Rocket Software JES Print Exit API 收集打印数据并将数据传送到指定的打印机目的地。

> **Note**
>
> 目标解决方案通常不需要更改应用程序以适应大型机格式化语言，例如 IBM 高级功能演示文稿 (AFP) 或施乐线路条件数据流 (LCDS)。有关使用 Rocket 软件在 AWS 上进行大型机应用程序迁移和现代化的更多信息，请参阅使用 Micro Focus 在 AWS 上增强企业大型机工作负载能力博客文章。

AWS 基础设施架构

下图显示了适用于大型机批量打印工作负载的高可用性且安全的 AWS 基础设施架构：

图表显示了以下工作流：

1. 批处理调度器启动批处理流程，并部署在 Amazon EC2 上的多个可用区以实现高可用性 (HA)。

> **Note**
>
> 这种模式不包括批处理调度器的实现。有关实现的详细信息，请参阅调度程序的软件供应商文档。

2. 大型机批处理作业（用 JCL 或 COBOL 等编程语言编写）使用核心业务逻辑来处理和生成打印输出，例如账单、身份证和贷款对账单。该任务在 Amazon 上 EC2 跨两个可用区 HA 部署，并使用 Rocket Software Print Exit 将打印输出路由到 LRS VPSX/MFI 以供最终用户打印。
3. LRS VPSX/MFI uses a TCP/IP-based LRS/Queue transmission agent to collect or capture print data from the Rocket Software JES Print Exit programming interface. Print Exit passes the necessary information to enable LRS VPSX/MFI to effectively process the spool file and dynamically build LRS/Queue 命令。然后使用 Rocket Software 的标准内置函数运行这些命令。

> **Note**
>
> 有关从 Rocket Software Print Exit 传递到LRS/Queue and LRS VPSX/MFI支持的大型机批处理机制的打印数据的更多信息，请参阅此模式的 “其他信息” 部分中的打印数据捕获。

4.

> **Note**
>
> N etwork Load Balanc er 提供 DNS 名称，用于将 Rocket Enterprise Server 与 LRS 集成，VPSX/MFI. : LRS VPSX/MFI supports a Layer 4 load balancer. The Network Load Balancer also does a basic health check on LRS VPSX/MFI并将流量路由到运行状况良好的注册目标。

5.

> **Note**
>
> LRS VPSX/MFI 打印服务器部署在亚马逊上， EC2 跨越两个可用区，用于 HA，并使用 A mazon EBS 作为运营数据存储。LRS 还VPSX/MFI supports both the active-active and active-passive service modes. This architecture uses multiple AZs in an active-passive pair as an active and hot standby. The Network Load Balancer performs a health check on LRS

VPSX/MFI EC2 instances and routes traffic to hot standby instances in the other AZ if an active instance is in an unhealthy state. The print requests are persisted in the LRS Job Queue locally in each of the EC2 instances. In the event of recovery, a failed instance has to be restarted for the LRS services to resume processing the print request. : LRS VPSX/MFI可以在打印机群级别执行运行状况检查。有关更多信息，请参阅此模式的其他信息部分中的打印机队列运行状况检查。

6. AWS Managed Microsoft AD 与 LRS/DIS 集成以执行打印工作流程身份验证和授权。有关更多信息，请参阅此模式的其他信息部分中的打印身份验证和授权。
7. LRS VPSX/MFI 使用 Amazon EBS 进行块存储。您可以将活动 EC2 实例中的 Amazon EBS 数据作为 point-in-time快照备份到 Amazon S3，然后将其恢复到热备用 EBS 卷。要自动创建、保留和删除 Amazon EBS 卷快照，您可以使用 Amazon Data Lifecycle Manager 设置自动快照的频率并根据 RTO/RPO 要求恢复快照。

工具

Amazon Web Services

- Amazon Elastic Block Store (Amazon EBS) 提供用于实例的块级存储卷。 EC2 EBS 卷的行为类似于原始、未格式化的块储存设备。您可以将这些卷作为设备挂载在实例上。
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要使用 Amazon EC2 启动任意数量或数量的虚拟服务器，也可以进行横向扩展或扩展。
- Amazon Relational Database Service (Amazon RDS) 是一项网络服务，可以更轻松地在 AWS 云中设置、操作和扩展关系数据库。它为行业标准的关系数据库提供了经济高效、可调整大小的容量，并管理常见的数据库管理任务。
- 适用于微软的 AWS Directory Service AD（也称为 AWS 托管的 Microsoft AD）使您的目录感知工作负载和 AWS 资源能够在 AWS 云中使用 Microsoft Active Directory。

其他工具

- 由 LRS 和 Rocket Software 共同开发的 LRS VPSX/MFI（Micro Focus Interface）可捕获火箭企业服务器 JES 线轴的输出，并将其可靠地传送到指定的打印目的地。
- LRS 目录信息服务器 (LRS/DIS) 用于打印工作流期间的身份验证和授权。
- TCP/IP-based LRS/Queue transmission agent is used by LRS VPSX/MFI通过 Rocket Software JES 打印出口编程接口收集或捕获打印数据。

- Rocket Enterprise Server 是大型机应用程序的应用程序部署环境。它为使用任何版本的 Rocket Software Enterprise Developer 迁移或创建的大型机应用程序提供了执行环境。

操作说明

在亚马逊上设置 Rocket Enterprise Server EC2 并部署大型机批处理应用程序

Task	描述	所需技能
设置 Rocket 企业服务器并部署演示应用程序。	在亚马逊上设置 Rocket Enterprise Server EC2，然后在亚马逊上部署 Rocket Software BankDemo 演示应用程序 EC2。 该 BankDemo 应用程序是一个大型机批处理应用程序，用于创建然后启动打印输出。	云架构师

在亚马逊上设置 LRS 打印服务器 EC2

Task	描述	所需技能
获取用于打印的 LRS 产品许可证。	要获得 LRS 的 LRS 产品许可证VPSX/MFI, LRS/Queue, and LRS/DIS，请联系 LRS 输出管理团队。您必须提供将安装 LRS 产品的 EC2 实例的主机名。	构建 lead
创建亚马逊 EC2 Windows 实例来安装 LRS VPSX/MFI。	按照亚马逊 EC2 文档中启动亚马逊 EC2 实例中的说明启动亚马逊 EC2 Windows 实例。您的实例必须满足 LRS VPSX/MFI 的以下硬件和软件要求： • CPU - 双核	云架构师

Task	描述	所需技能
	• 内存 - 16 GB • 驱动器 - 500 GB • 最小 EC2 实例 — m5.xlarge • 操作系统 - Windows/Linux • 软件 - 互联网信息服务(IIS)或 Apache **Note** 上述硬件和软件要求适用于小型打印机群（大约 500—1000）。要获取完整的要求，请咨询您的 LRS 和 AWS 联系人。 创建 Windows 实例时，请执行以下操作： 1. 确认 EC2 主机名与 LRS 产品许可证使用的主机名相同。 2. 完成以下操作即可在 Amazon EC2 中启用 CGI： a. 按照中的说明连接到您的 EC2 实例	

Task	描述	所需技能
在实例上安装 LRS VPSX/MFI。 EC2	1. 按照 Amazon EC2 文档中步骤 2：连接到您的实例中的说明连接到您的实例。 EC2 2. **Note** 打开发送至您的邮箱的 LRS 电子邮件中的产品下载页面链接。: LRS 产品通过电子文件传输 (EFT) 分发。 3. 下载 LRS VPSX/MFI 并解压缩文件（默认文件夹：c:\LRS）。 4. 从解压缩的文件夹中启动 LRS 产品安装程序以安装 LRS VPSX/MFI。 5. 在选择功能菜单中，选择 VPSX® Server (V1R3.022)，然后选择下一步开始安装过程。安装完成后，您将收到一条成功消息。	云架构师

Task	描述	所需技能
安装 LRS/队列。	1. 按照亚马逊 EC2 文档中步骤 2：连接到您的 EC2 实例中的说明连接到您的 Rocket Enterprise Server 实例。 2. 从发送至您邮箱的 LRS 电子邮件中打开 LRS 产品下载页面的链接，下载 LRS/Queue，然后解压缩文件。 3. 转到下载文件的位置，然后启动 LRS 产品安装程序以安装 LRS/Queue。	云架构师
安装 LRS/DIS。	1. 按照亚马逊文档中步骤 2：连接到您的 EC2 实例中的说明，连接到您的 LRS VPSX/MFI 实例。 EC2 2. 从发送至您邮箱的 LRS 电子邮件中打开 LRS 产品下载页面的链接，下载 LRS/DIS，然后解压缩文件。 3. 转到下载文件的位置，然后启动 LRS 产品安装程序。 4. 在 LRS 产品安装程序中，展开 LRS 其他工具，选择 LRS DIS，然后选择下一步。 5. 按照 LRS 产品安装程序中的其余说明完成安装过程。	云架构师

Task	描述	所需技能
创建目标组并将 LRS VPSX/MF EC2 I 注册为目标。	按照弹性负载均衡器文档中为网络负载均衡器创建目标组的说明创建目标组。 创建目标组时，请执行以下操作: 1. 在指定组详细信息页面上，对于选择目标类型，选择实例。 2. 对于协议，选择 TCP。 3. 对于端口，选择 5500。 4. 在注册目标页面的可用实例部分，选择 LRS VPS EC2 X/MFI 实例。	云架构师
创建网络负载均衡器。	按照弹性负载均衡器文档中创建网络负载均衡器的说明进行操作。您的 Network Load Balancer 将流量从 Rocket Enterprise 服务器路由到 LRS V EC2 PSX/MFI。 创建网络负载均衡器时，在侦听器和路由页面上执行以下操作： 1. 对于协议，选择 TCP。 2. 对于端口，选择 5500。 3. 对于默认操作，选择转发至您之前创建的目标组。	云架构师

将 Rocket 企业服务器与 LRS 集成 VPSX/MFI and LRS/Queue

Task	描述	所需技能
为 LRS/队列集成配置 Rocket 企业服务器。	1. 按照亚马逊 EC2 文档中步骤 2：连接到您的 EC2 实例中的说明连接到您的 Rocket Enterprise Server 实例。 2. 在 Windows 的 “开始” 菜单中，打开 Rocket 企业服务器管理用户界面。 3. 在菜单栏中，选择 NATIVE。 4. 在导航窗格中，选择 Directory Server，然后选择 BANKDEMO 或您的 Enterprise 服务器区域。 5. 从左侧导航窗格中的常规中，向下滚动到其他部分，将环境变量 (LRSQ_ADDRESS、LRSQ_PORT、LRSQ_COMMAND) 配置为指向 LRSQ。 6. 对于 LRSQ_ADDRESS，输入您之前创建的网络负载均衡器的 IP 地址或 DNS 名称。 7. 对于 LRSQ_PORT，输入 VPSX LRSQ 侦听器端口 (5500)。 8. 对于 LRSQ_COMMAND，请输入 LRSQ 可执行文件的路径位置。	云架构师

Task	描述	所需技能
	Note LRS 目前支持 DNS 名称的最大字符限制为 50，但将来可能会发生变化。如果您的 DNS 名称字符数大于 50，则可以使用网络负载均衡器的 IP 地址作为替代。	

Task	描述	所需技能
为 LRS VPSX/MFI 集成配置 Rocket Enterprise 服务器。	1. 将VPSX_MFI_R2 文件夹从 LRS VPSX/MFI 安装程序复制到 Rocket Enterprise Server 的位置，网址为。C\BANKDEMO\print 2. 按照亚马逊 EC2 文档中步骤 2：连接到您的 EC2 实例中的说明连接到您的 Rocket Enterprise Server 实例。 3. 在 Windows 的 “开始” 菜单中，打开 Rocket 企业服务器管理用户界面。 4. 在菜单栏中，选择 NATIVE。 5. 在导航窗格中，选择 Directory Server，然后选择 BANKDEMO。 6. 在 BANKDEMO 下方选择 JES。 7. 在 JES 程序路径下，添加来自 C\BANKDEMO\print 位置的 DLL(VPSX_MFI_R2) 路径。	云架构师

在 Rocket Enterprise Server 和 LRS VPSX/MFI 中设置打印机和打印用户

Task	描述	所需技能
将 Rocket 软件打印退出模块与 Rocket Enterprise Server 批处理打印机服务器执行过程相关联	1. 按照亚马逊 EC2 文档中步骤 2：连接到您的 EC2 实例中的说明连接到您的 Rocket Enterprise Server 实例。	云架构师

Task	描述	所需技能
	2. 在 Windows 的 “开始” 菜单中，打开 Rocket 企业服务器管理用户界面。 3. 在菜单栏中，选择 NATIVE。 4. 在导航窗格中，选择 Directory Server，然后选择 BANKDEMO。 5. 在 BANKDEMO 下方选择 JES，然后向下滚动到打印机。 6. 在打印机中，将 Rocket Software 打印退出模块（LRSPRTE6 用于批处理）与 Rocket Enterprise Server 批处理打印机服务器执行流程 (SEP) 关联 这一操作将允许将打印输出路由至 LRS VPSX/MFI。 7. 登录到 Enterprise Server 管理 UI。 有关配置的更多信息，请参阅 Micro Focus 文档中的使用退出。	

Task	描述	所需技能
在 LRS VPSX/MFI 中添加打印机。	1. 按照亚马逊文档中步骤 2：连接到您的 EC2 实例中的说明，连接到您的 LRS VPSX/MFI 实例。 EC2 2. 从 Windows 开始菜单打开 VPSX Web 界面。 3. 在导航窗格中，选择打印机。 4. 选择添加，再选择添加用户。 5. 在打印机配置页上，对于打印机名称，输入 Local。 6. 对于 VPSX ID，请输入。VPS1 7. 对于 CommType，请选择 TCPIP/ LRSQ。 8. 对于主机/IP 地址，输入要添加的物理打印机的 IP 地址。 9. 对于设备，输入您的设备名称。 10选择 Windows 驱动程序或 Linux/Mac 驱动程序。 11选择 添加。	云架构师

Task	描述	所需技能
在 LRS VPSX/MFI 中创建打印用户。	1. 按照亚马逊文档中步骤 2：连接到您的 EC2 实例中的说明，连接到您的 LRS VPSX/MFI 实例。 EC2 2. 从 Windows 开始菜单打开 VPSX Web 界面。 3. 在导航窗格中，选择安全，然后选择用户。 4. 在用户名列中，选择 admin，然后选择复制。 5. 在“用户配置文件维护”窗口中，在“用户名”中输入用户名（例如 PrintUser）。 6. 对于说明，输入简短说明（例如，测试打印的用户）。 7. 选择更新。这将创建一个打印用户（例如，PrintUser）。 8. 在导航窗格中的用户下方，选择您创建的新用户。 9. 从命令菜单中选择安全。 10. 在安全规则页面上，选择所有适用的打印机安全和作业安全选项，然后选择保存。 11. 要将新的打印用户添加到管理员组，请在导航窗格中选择安全，然后选择配置。 12. 在安全配置窗口中，将新的打印用户添加到管理员列中。	云架构师

设置身份验证和授权

Task	描述	所需技能
使用用户和组创建 AWS Managed Microsoft AD 域。	1. 按照 AWS Directory Service 文档中创建 AWS Managed Microsoft AD 目录的说明，在 AWS Managed Microsoft AD 上创建 Active Directory。 2. 按照 AWS Directory Service 文档中第 3 步：部署 EC2 实例来管理 AWS 托管的 Microsoft AD 中的说明，部署 EC2 实例（活动目录管理器）并安装活动目录工具来管理您的 AWS 托管 Microsoft AD。 3. **Note** 按照 Amazon EC2 文档中步骤 2：连接到您的实例中的说明连接到您的实例。EC2 ：当您连接到 EC2 实例时，请在 Windows 安全窗口中输入您的管理员凭据（用于您在步骤一中创建的目录）。 4. 在 Windows 开始菜单的 Windows 管理员工具下方，选择 Active Directory 用户及计算机。	云架构师

Task	描述	所需技能
	5. 按照 AWS Directory Service 文档中创建用户的步骤在 Active Directory 域中创建打印用户。	
加入 LRS VPSX/MFI，进入 AWS 托管 EC2 的微软 AD 域。	自动加入 LRS VPSX/MFI EC2 到您的 AWS 托管微软 AD 域（AWS 知识中心文档）或手动加入（AWS Directory Service 文档）。	云架构师
配置 LRS/DIS 并将其与 AWS Managed Microsoft AD 集成。	1. 按照亚马逊文档中步骤 2：连接到您的 EC2 实例中的说明，连接到您的 LRS VPSX/MFI 实例。 EC2 2. 在 Windows 开始菜单中，打开 VPSX Web 界面。 3. 在导航窗格中，选择安全，然后选择配置。 4. 在安全配置页面的安全参数部分中，对于安全类型，选择内部。 5. 在安全参数部分中输入您对其余选项的首选项。 6. 从 Microsoft Windows 开始菜单中打开 LRS 输出管理文件夹，选择服务器启动，然后选择服务器停止。 7. 使用您的 Active Directory 用户名和密码登录 LRS VPSX/MFI。	云架构师

测试打印工作流

Task	描述	所需技能
从 Rocket 软件 BankDemo 应用程序启动批量打印请求。	1. 在你的 Rocket 企业服务器 EC2 实例中打开 3270 终端模拟器。 2. 通过运行以下命令连接到 BankDemo 应用程序：`connect 127.0.0.1:9278` 3. 在 BankDemo 命令行界面上，对于用户 ID，输入 B0001。对于密码，输入非空白密钥。 4. 对于请求打印报表选项，在空行中输入 X。 5. 在发送语句依据部分中，对于邮件，输入 Y，然后按 F10。	测试工程师
检查 LRS VPSX/MFI 中的打印输出。	1. 按照亚马逊文档中步骤 2：连接到您的 EC2 实例中的说明，连接到您的 LRS VPSX/MFI 实例。 EC2 2. 在 Windows 开始菜单中，打开 VPSX Web 界面。 3. 在导航窗格中，选择打印机，然后选择输出队列。 4. 在假脱机 ID列中，选择打印机队列中请求的假脱机 ID。 5. 在操作选项卡的命令列中，选择浏览。	测试工程师

Task	描述	所需技能
	现在，您可以看到账户对账单的打印输出，其中包含账户号码、描述、日期、金额和余额列。有关示例，请参阅此模式的 batch_print_output 附件。	

相关资源

- LRS 输出现代化（LRS 文档）
- ANSI 和机器托架控制（IBM 文档）
- 通道命令字（IBM 文档）
- 使用 Micro Focus 在 AWS 上为企业大型机工作负载提供支持（Amazon Web Services Partner Network 博客）
- 使用 Amazon A EC2 uto Scaling 和 Systems Manager 构建 Micro Focus 企业服务器 PAC（AWS Prescriptive Guidence 文档）
- Advanced Function Presentation (AFP)数据流（IBM 文档）
- 线路条件数据流 (LCDS)（比较文档）

其他信息

注意事项

在现代化改造过程中，您可以考虑大型机批处理及其生成的输出的各种配置。每个使用大型机平台的客户和供应商都根据直接影响打印的特殊要求对平台进行了定制。例如，您当前的平台可能会将 IBM 高级功能演示 (AFP) 或 Xerox 线路条件数据流 (LCDS) 合并到当前工作流程中。此外，大型机回车控制字符和通道命令字可能会影响打印页面的外观，可能需要特殊处理。作为现代化规划过程的一部分，我们建议您评测并了解特定打印环境中的配置。

打印数据采集

Rocket Software Print Exit 传递必要的信息，使 LRS VPSX/MFI 能够有效地处理线轴文件。该信息由在相关控制块中传递的字段组成，例如：

- JOBNAME

- OWNER (USERID)
- DESTINATION
- FORM
- FILENAME
- WRITER

LRS VPSX/MFI 支持以下大型机批处理机制，用于从 Rocket Enterprise Server 捕获数据。

- 批处理 COBOL 语句 print/spool processing using standard z/OS JCL SYSOUT DD/OUTPUT
- 批处理 COBOL print/spool processing using standard z/OS JCL CA-SPOOL SUBSYS DD 语句
- IMS/COBOL print/spool使用 CBLTDLI 接口进行处理（有关支持的方法和编程示例的完整列表，请参阅产品许可证附带的 LRS 文档。）

打印机队列运行状况检查

LRS VPSX/MFI (LRS LoadX) 可执行深入的运行状况检查，包括设备管理和操作优化。设备管理可以检测打印机设备中的故障，并将打印请求路由到正常运行的打印机。有关打印机队列的深入运行状况检查的详细信息，请参阅产品许可证附带的 LRS 文档。

打印身份验证和授权

LRS/DIS enables LRS applications to authenticate user IDs and passwords by using Microsoft Active Directory or an LDAP server. In addition to basic print authorization, LRS/DIS还可以在以下用例中应用粒度级别的打印安全控制：

- 管理谁可以浏览打印机作业。
- 管理其他用户作业的浏览级别。
- 管理操作任务。例如，命令级安全性，例如保留/释放、清除、修改、复制和重新路由。安全性可以通过用户 ID 或组（类似于 AD 组或 LDAP 组）来设置。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

大型机现代化： AWS 使用 Rock DevOps et 软件企业套件开启

由 Kevin Yung (AWS) 编写

摘要

注意： AWS CodeCommit 已不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

客户面临的挑战

在大型机硬件上运行核心应用程序的组织，在硬件需要纵向扩展以满足数字创新需求时，通常会遇到一些挑战。这些挑战包括以下制约因素。

- 由于大型机硬件组件不灵活，更换成本高昂，因此大型机开发和测试环境无法扩展。
- 由于新开发人员对传统大型机开发工具不熟悉、不感兴趣，大型机开发正面临技能短缺的问题。诸如容器、连续integration/continuous delivery (CI/CD（连续）管道和现代测试框架之类的现代技术在大型机开发中不可用。

模式结果

为了应对这些挑战，Amazon Web Services (AWS) 和 AWS Partner Network (APN) 合作伙伴Rocket Software Micro Focus合作创建了这种模式。该解决方案旨在帮助您实现以下成果。

- 提高开发人员的工作效率。开发人员可以在几分钟内获得新的大型机开发实例。
- 使用创建 AWS Cloud 容量几乎无限的新大型机测试环境。
- 快速配置新的大型机 CI/CD 基础架构。使用 AWS CloudFormation 和 AWS 可以在一小时内完成置备 AWS Systems Manager。
- 原生使用大型机开发 AWS DevOps 工具，包括 AWS CodeBuild、、、和 Amazon Elastic Contain AWS CodePipeline er Re AWS CodeDeploy gistry (Amazon ECR) Registry。 AWS CodeCommit
- 将大型机项目中的传统瀑布式开发转变为灵活开发。

技术摘要

在此模式中，目标堆栈包含以下组件。

逻辑组件	实施方案	描述

源代码存储库	火箭软件 AccuRev 服务器、CodeCommit、亚马逊 ECR	源代码管理-该解决方案使用两种类型的源代码： • 大型机源代码，例如 COBOL 和 JCL。 • AWS 基础架构模板和自动化脚本 两种类型的源代码都需要版本控制，但它们的管理方式不同 SCMs。部署到大型机或火箭软件企业服务器中的源代码在 Rocket Software Micro Focus AccuRev 服务器中管理。 AWS 模板和自动化脚本在中进行管理 CodeCommit。Amazon ECR 用于 Docker 映像存储库。
企业开发人员实例	亚马逊 Elastic Compute Cloud（亚马逊 EC2），Eclipse 的火箭软件企业开发者	大型机开发人员可以使用适用于 Eclipse EC2 的 Rocket Software Enterprise 开发者在亚马逊中开发代码。这样就无需依赖大型机硬件来编写和测试代码。
火箭软件企业套件许可证管理	火箭软件企业套件许可证管理器	对于集中的火箭软件企业套件许可证管理和治理，该解决方案使用Rocket Software Enterprise Suite许可证管理器来托管所需的许可证。

CI/CD 管道	CodePipeline、、 CodeBuild CodeDeploy、容器里的 Rocket Software Enterprise Developer、容器中的 Rocket Software 企业测试服务器、	大型机开发团队需要 CI/CD 管道来执行代码编译、集成测试和回归测试。在 AWS，CodePipeline 并且 CodeBuild 可以在容器中使用 Rocket Software Enterprise 开发人员和企业测试服务器。

先决条件和限制

先决条件

名称	描述
py3270	py3270 是 x3270（IBM 3270 终端模拟器）的 Python 接口。它为 x3270 或 s3270 子进程提供 API。
x3270	x3270 是用于 X Window System 和 Windows 的 IBM 3270 终端模拟器。 开发人员可以使用它来进行本地单元测试。
机器人框架-大型机-3270-库	Mainframe3270 是基于 py3270 项目的 Robot Framework 库。
火箭软件 Verastream	Rocket Software Verastream 是一个集成平台，它支持以测试移动应用程序、Web 应用程序和 SOA Web 服务的方式测试大型机资产。
Rocket Software 统一功能测试 (UFT) 安装程序和许可证	Rocket Software 统一功能测试是为软件应用程序和环境提供功能和回归测试自动化的软件。
火箭软件企业服务器安装程序和许可证	Enterprise Server 为大型机应用程序提供运行时系统环境。
火箭软件企业测试服务器安装程序和许可证	Rocket Software 企业测试服务器是一个 IBM 大型机应用程序测试环境。

Rocket Software 服务器 AccuRev 安装程序和许可证，以及 Rocket Software Micro Focus AccuRev 安装程序和 Windows	AccuRev 提供源代码管理 (SCM)。该 AccuRev 系统专为一组正在开发一组文件的人员使用而设计。
Eclipse 安装程序、补丁和许可证的 Rocket 软件企业开发人员	Enterprise Developer 为大型机开发人员提供了一个平台，用于开发和维护核心大型机在线和批处理应用程序。

限制

- 中不支持构建 Windows Docker 镜像。 CodeBuild此报告问题需要 Windows 内核/HCS 和 Docker 团队的支持。解决方法是使用 Systems Manager 创建 Docker 映像构建运行手册。这种模式使用变通方法为 Eclipse 和 Rocket Software Micro Focus 企业测试服务器容器镜像构建 Rocket Software Enterp
- Windows 尚不支持来自 CodeBuild 的虚拟私有云 (VPC) 连接，因此该模式不使用 Rocket 软件许可证管理器来管理 Rocket Software Enterprise Developer 和 OpenText Rocket Software 企业测试服务器容器中的许可证。

产品版本

- 火箭软件企业开发者 5.5 或更高版本
- 火箭软件企业测试服务器 5.5 或更高版本
- 火箭软件企业服务器 5.5 或更高版本
- 火箭软件 AccuRev 7.x 或更高版本
- 适用于 Rocket Software 企业开发人员和企业测试服务器的 Windows Docker 基础镜像：微软/dotnet-framework-4.7.2-runtime
- 适用于 AccuRev 客户的 Linux Docker 基础镜像：amazonlinux: 2

架构

大型机环境

在传统的大型机开发中，开发人员需要使用大型机硬件来开发和测试程序。它们面临容量限制，例如，开发/测试环境的每秒百万条指令数(MIPS)受到限制，并且它们必须依赖大型计算机上可用的工具。

在许多组织中，大型机开发遵循瀑布式开发方法，团队依靠较长的周期来发布变更。这些产品的发布周期通常长于数字产品开发。

下图显示了多个大型机项目共享大型机硬件进行开发的情况。在大型机硬件中，为更多项目横向扩展开发和测试环境的成本很高。

AWS 架构

这种模式将大型机开发扩展到. AWS Cloud首先，它使用 AccuRev SCM 托管大型机源代码。 AWS然后，它使企业开发人员和企业测试服务器可用于构建和测试大型机代码。 AWS

下文将介绍该模式的三个主要组成部分。

1。SCM

在中 AWS，该模式用于 AccuRev 为大型机源代码创建一组 SCM 工作空间和版本控制。其基于流的架构可实现多个团队的并行大型机开发。要合并更改，请 AccuRev 使用升级概念。要将该更改添加到其他工作空间，请 AccuRev 使用更新概念。

在项目级别，每个团队可以创建一个或多个直播 AccuRev 来跟踪项目级别的变化。这些被称为项目流。这些项目流继承自同一父流。父流用于合并来自不同项目流的更改。

每个项目流都可以将代码提升到 AccuRev，并且设置了提升后触发器来启动 C AWS I/CD 管道。项目流更改的成功生成可以提升到其父流，以进行更多回归测试。

通常，父流被称为系统集成流。当从项目流提升到系统集成流时，提升后触发器会启动另一个 CI/CD 管道来运行回归测试。

除大型机代码外，此模式还包括 AWS CloudFormation 模板、Systems Manager Automation 文档和脚本。遵循 infrastructure-as-code最佳实践，它们在中受版本控制。 CodeCommit

如果您需要将大型机代码同步回大型机环境进行部署，Rocket Software 提供了企业同步解决方案，该解决方案可以将代码从 SCM 同步回大型机 AccuRev SCM。

2。开发人员和测试环境

在大型组织中，要扩展成百甚至上千名大型机开发人员是一项挑战。为了解决这一限制，该模式使用亚马逊 EC2 Windows 实例进行开发。在这些实例上，安装了 Eclipse 的企业开发人员工具。开发人员可以在实例上本地执行所有大型机代码测试和调试。

AWS Systems Manager 状态管理器和自动化文档用于自动配置开发人员实例。创建开发者实例的平均时间在 15 分钟以内。已准备好以下软件和配置：

- AccuRev Windows 客户端，用于检出源代码并将其提交到 AccuRev
- Eclipse 的企业开发人员工具，用于在本地编写、测试和调试大型机代码
- 开源测试框架 Python 行为驱动开发 (BDD) 测试框架 Behave、py3270 和 x3270 模拟器，用于创建脚本来测试应用程序
- 一个 Docker 开发人员工具，用于生成企业测试服务器 Docker 映像并在企业测试服务器 Docker 容器中测试应用程序

在开发周期中，开发人员使用该 EC2 实例在本地开发和测试大型机代码。成功测试本地更改后，开发人员会将更改推广到 AccuRev 服务器。

3。CI/CD 管道

在该模式中，CI/CD 管道用于在部署到生产环境之前进行集成测试和回归测试。

正如 SCM 部分所述， AccuRev 使用两种类型的流：项目流和集成流。每个流都与 CI/CD 管道挂钩。为了在 AccuRev 服务器和之间执行集成 AWS CodePipeline，该模式使用升级 AccuRev 后脚本创建一个事件来启动 CI/CD。

例如，当开发者在中推广对项目流的更改时 AccuRev，它会启动一个升级后脚本以在 AccuRev Server 中运行。然后，该脚本将更改的元数据上传到 Amazon Simple Storage Service (Amazon S3)桶以创建 Amazon S3 事件。此事件将启动 CodePipeline 已配置的管道运行。

集成流及其关联的管道使用相同的事件启动机制。

在 CI/CD 管道中， CodeBuild 与 AccuRev Linux 客户端容器一起 CodePipeline 使用来查看流中的最新代码。 AccuRev 然后，管道开始 CodeBuild 使用企业开发人员 Windows 容器来编译源代码，并使用中的企业测试服务器 Windows 容器 CodeBuild 来测试大型机应用程序。

CI/CD pipelines are built using CloudFormation templates, and the blueprint will be used for new projects. By using the templates, it takes less than an hour for a project to create a new CI/CD管道进来 AWS了。

为了扩展你的大型机测试能力 AWS，该模式构建了 Rocket Software DevOps 测试套件、Verastream 和 UFT 服务器。通过使用现代 DevOps 工具，您可以根据 AWS 需要运行任意数量的测试。

下图显示了开启 Rocket Softwar AWS e 的大型机开发环境示例。

目标技术堆栈

本节将详细介绍该模式中每个组件的架构。

1。源代码存储库 — AccuRev SCM

AccuRev SCM 的设置是为了管理大型机源代码版本。为了获得高可用性， AccuRev 支持主模式和副本模式。操作员在主节点上执行维护时可以故障转移到副本。

为了加快 CI/CD 管道的响应速度，该模式使用 Amazon E CloudWatch vents 来检测源代码更改并启动管道的启动。

1. 该管道已设置为使用 Amazon S3 来源。
2. 设置 CloudWatch 事件规则是为了从源 S3 存储桶中捕获 S3 事件。
3. CloudWatch 事件规则为管道设置了目标。
4. AccuRev SCM 配置为在升级完成后在本地运行升级后脚本。
5. AccuRev SCM 生成一个包含促销元数据的 XML 文件，脚本会将该 XML 文件上传到源 S3 存储桶。
6. 上传后，源 S3 存储桶发送事件以匹配 CloudWatch 事件规则， CloudWatch 事件规则启动管道以运行。

当管道运行时，它会启动一个 CodeBuild 项目，使用 AccuRev Linux 客户端容器从关联 AccuRev 的流中查看最新的大型机代码。

下图显示了 AccuRev 服务器设置。

2。企业开发人员模板

该模式使用 Amazon EC2 模板来简化开发者实例的创建。通过使用状态管理器，它可以一致地将软件和许可证设置应用于 EC2 实例。

Amazon EC2 模板在其 VPC 上下文设置和默认实例设置中构建，并且遵循企业标签要求。通过使用模板，团队可以创建自己的新开发实例。

当开发人员实例启动时，通过与标签关联，Systems Manager 使用 State Manager 来应用自动化。自动化包括以下常规步骤。

1. 安装企业开发人员软件并安装补丁。
2. 安装适用于 Windows 的 AccuRev 客户端。
3. 安装预先配置的脚本以供开发者加入 AccuRev 直播。初始化 Eclipse 工作区。
4. 安装开发工具，包括 x3270、py3270 和 Docker。
5. 将许可证设置配置为指向 License Manager 负载平衡器。

下图显示了由 Amazon EC2 模板创建的企业开发者实例，状态管理器将软件和配置应用于该实例。企业开发者实例连接到 AWS License Manager 以激活其许可证。

3。 CI/CD 管道

正如 AWS 架构部分所解释的那样，在模式中，有项目级CI/CD pipelines and system integration pipelines. Each mainframe project team creates a pipeline or multiple CI/CD pipelines for building the programs that they are developing in a project. These project CI/CD管道从关联 AccuRev 的流中检出源代码。

在项目团队中，开发人员在关联的 AccuRev 流中推广他们的代码。然后，促销会启动项目管道以生成代码并运行集成测试。

每个项目 CI/CD 管道都使用带有企业开发者工具 Amazon ECR 映像和企业测试服务器工具 Amazon ECR 映像的 CodeBuild 项目。

CodePipeline 并 CodeBuild 用于创建 CI/CD 管道。因为 CodeBuild 并且 CodePipeline 没有预付费用或承诺，因此您只需为实际使用量付费。与大型机硬件相比，该 AWS 解决方案大大缩短了硬件配置提前期，降低了测试环境的成本。

在现代开发中，使用了多种测试方法。例如，测试驱动开发 (TDD)、BDD 和机器人框架。通过此模式，开发人员可以使用这些现代工具进行大型机测试。例如，通过使用 x3270、py3270 和行为 python

测试工具，可以定义联机应用程序的行为。您还可以在这些 CI/CD 管道中使用构建大型机 3270 机器人框架。

下图显示了团队流 CI/CD 管道。

下图显示了 Mainframe3270 Robot Framework CodePipeline 中生成的项目 CI/CD 测试报告。

下图显示了 Py3270 和 Behave BDD CodePipeline 中生成的项目 CI/CD 测试报告。

成功通过项目级测试后，将测试的代码手动提升到 AccuRev SCM 中的集成流。在团队对其项目管道的测试覆盖率有信心后，可以自动执行此步骤。

升级代码时，系统集成 CI/CD 管道会签出合并的代码并执行回归测试。合并的代码将从所有并行项目流中提升。

根据对测试环境的要求程度，客户可以在不同的环境中拥有更多的系统集成 CI/CD 管道，例如 UAT、Pre-Production。

在该模式中，系统集成管道中使用的工具是企业测试服务器、UFT Server 和 Verastream。所有这些工具都可以部署到 Docker 容器中并与之一起 CodeBuild使用。

成功测试大型机程序后，构件将通过版本控制存储在 S3 存储桶中。

下图显示了系统集成 CI/CD 管道。

在系统集成 CI/CD 管道中成功测试构件后，可以将其提升为生产部署。

如果您需要将源代码部署回大型机，Rocket Software提供了企业同步解决方案，可以将源代码从 Mainframe Endeavor同步 AccuRev 回Mainframe Endeavor。

下图显示了将构件部署到企业服务器的生产 CI/CD 管道。在此示例中， CodeDeploy 协调了将经过测试的大型机工件部署到企业服务器中的过程。

除了 CI/CD 管道的架构演练外，有关在和中测试大型机应用程序的更多信息，请参阅 AWS DevOps 博客文章 “使用 Micro Focus Enterprise Suite 自动执行数千个大型机测试”。 AWS CodeBuild CodePipeline（Micro Focus 现在是 Rocket 软件。 ）有关在上进行大型机测试的最佳实践和详细信息，请参阅博客文章。 AWS

工具

AWS 自动化工具

- AWS CloudFormation
- 亚马逊 CloudWatch 活动
- AWS CodeBuild
- AWS CodeDeploy
- AWS CodePipeline
- Amazon ECR
- Amazon S3
- AWS Secrets Manager
- AWS Systems Manager

火箭软件工具

- Eclipse 的火箭企业开发人员
- 火箭企业测试服务器
- 火箭企业服务器（生产部署）
- 火箭软件 AccuRev
- 火箭软件企业套件许可证管理器
- 火箭软件 Verastream 主机集成商
- 火箭软件 UFT One

其他工具

- x3270
- py3270

- 机器人框架-大型机-3270-库

操作说明

创建 AccuRev SCM 基础架构

Task	描述	所需技能
使用 CloudFormation部署主 AccuRev SCM 服务器。		AWS CloudFormation
创建 AccuRev 管理员用户。	登录 AccuRev SCM 服务器，然后运行 CLI 命令创建管理员用户。	AccuRev SCM 服务器管理员
创建 AccuRev 直播。	按顺序创建 AccuRev 从上层流继承的直播：制作、系统集成、团队直播。	AccuRev SCM 管理员
创建开发者 AccuRev 登录账户。	使用 AccuRev SCM CLI 命令为大型机开发人员创建 AccuRev 用户登录帐户。	AccuRev SCM 管理员

创建企业开发者 Amazon EC2 启动模板

Task	描述	所需技能
使用部署 Amazon EC2 启动模板 CloudFormation。	用于 CloudFormation 为企业开发者实例部署 Amazon EC2 启动模板。该模板包括 Rocket Enterprise 开发者实例的 Systems Manager 自动化文档。	AWS CloudFormation
使用 Amazon EC2 模板创建企业开发者实例。		AWS 管理控制台登录和大型机开发人员技能

创建企业开发者工具 Docker 镜像

Task	描述	所需技能
创建企业开发者工具 Docker 镜像。	使用 Docker 命令和企业开发者工具 Dockerfile 创建 Docker 镜像。	Docker
在 Amazon ECR 中创建 Docker 存储库。	在 Amazon ECR 控制台上，为企业开发者 Docker 镜像创建存储库。	Amazon ECR
将企业开发者工具 Docker 镜像推送到亚马逊 ECR。	运行 Docker push 命令以推送企业开发人员工具 Docker 映像，以将其保存在 Amazon ECR 的 Docker 存储库中。	Docker

创建企业测试服务器 Docker 镜像

Task	描述	所需技能
创建企业测试服务器 Docker 镜像。	使用 Docker 命令和企业测试服务器 Dockerfile 创建 Docker 镜像。	Docker
在 Amazon ECR 中创建 Docker 存储库。	在 Amazon ECR 控制台上，为企业测试服务器 Docker 镜像创建 Amazon ECR 存储库。	Amazon ECR
将企业测试服务器 Docker 镜像推送到 Amazon ECR。	运行 Docker push 命令以在 Amazon ECR 中推送和保存企业测试服务器 Docker 映像。	Docker

创建团队流 CI/CD 管道

Task	描述	所需技能
创建 CodeCommit 存储库。	在 CodeCommit 控制台上，为基础架构和 CloudFormation 代码创建基于 Git 的存储库。	AWS CodeCommit
将 CloudFormation 模板和自动化代码上传到 CodeCommit 存储库。	运行 Git push 命令将 CloudFormation 模板和自动化代码上传到存储库。	Git
使用部署团队流 CI/CD 管道。CloudFormation	使用准备好的 CloudFormation 模板部署团队流 CI/CD 管道。	AWS CloudFormation

创建系统集成 CI/CD 管道

Task	描述	所需技能
创建 UFT Docker 镜像。	使用 Docker 命令和 UFT Dockerfile 创建 Docker 镜像。	Docker
在 Amazon ECR 中为 UFT 镜像创建 Docker 存储库。	在 Amazon ECR 控制台上，为 UFT 镜像创建 Docker 存储库。	Amazon ECR
将 UFT Docker 镜像推送到亚马逊 ECR。	运行 Docker push 命令以在 Amazon ECR 中推送和保存企业测试服务器 Docker 映像。	Docker
创建 Verastream Docker 镜像。	使用 Docker 命令和 Verastream Dockerfile 创建 Docker 镜像。	Docker
在 Amazon ECR 中为 Verastream 镜像创建 Docker 存储库。	在 Amazon ECR 控制台上，为 Verastream 镜像创建 Docker 存储库。	Amazon ECR

Task	描述	所需技能
使用部署系统集成 CI/CD 管道。 CloudFormation	使用准备好的 CloudFormation 模板部署系统集成 CI/CD 管道。	AWS CloudFormation

创建生产部署 CI/CD 管道

Task	描述	所需技能
使用 AWS 快速入门部署企业服务器。	要使用部署企业服务器 CloudFormation，请在 “ AWS 快速入门” 中启动企业服务器。	AWS CloudFormation
部署生产部署 CI/CD 管道。	在 CloudFormation 控制台上，使用 CloudFormation 模板部署生产部署 CI/CD 管道。	AWS CloudFormation

相关资源

参考

- AWS DevOps 博客-使用 Micro Focus 企业套件自动执行数千次大型机测试（ Micro Focus 现在是 Rocket Software。 ） AWS
- py3270/py3270存储库 GitHub
- Altran-pt-GDC/robot-Framework-Mainframe-3270-Librar GitHub
- 欢迎浏览 behave ！
- APN 合作伙伴博客——标签：Micro Focus（ Mic ro Focus 现在是 Rocket 软件。 ）
- 从启动模板启动实例

AWS Marketplace

- 火箭软件 UFT One

AWS Quick Start（快速入门）

- Rocket 企业服务器开启 AWS

使用 Micro Focus 企业服务器和 LRS VPSX/MFI 在 AWS 上实现大型机在线打印工作负载的现代化

由 Shubham Roy (AWS)、Abraham Rondon (Micro Focus)、Guy Tucker (Levi, Ray and Shoup Inc) 和 Kevin Yung (AWS) 编写

摘要

此模式向您展示如何使用 Micro Focus Enterprise Server 作为现代化大型机应用程序的运行时系统，并使用 LRS VPSX/MFI（Micro Focus 接口）作为打印服务器，在 Amazon Web Services (AWS) Cloud 上实现业务关键型大型机在线打印工作负载的现代化。该模式基于更换平台大型机现代化方法。通过这种方法，您可以将大型机在线应用程序迁移到亚马逊弹性计算云 (Amazon EC2)，并将大型机数据库（例如适用 DB2 于 z/OS 的 IBM）迁移到亚马逊关系数据库服务（Amazon RDS）。现代化打印工作流程的身份验证和授权由 Microsoft Active Directory 的 AWS Directory Service（也称为 AWS 托管 Microsoft AD）执行。LRS 目录信息服务器 (LRS/DIS) 与 AWS 托管 Microsoft AD 集成，用于打印工作流身份验证和授权。通过实现在线打印工作负载的现代化，您可以降低 IT 基础设施成本，减轻维护传统系统的技术债务，消除数据孤岛，通过 DevOps 模型提高灵活性和效率，并利用 AWS 云中的按需资源和自动化。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 大型机联机打印或输出管理工作负载
- 有关如何重建和交付在 Micro Focus 企业服务器上运行的大型机应用程序的基本知识（有关详细信息，请参阅 Micro Focus 文档中的企业服务器数据表。）
- LRS 云打印解决方案和概念的基本知识（有关更多信息，请参阅 LRS 文档中的输出现代化。）
- Micro Focus Enterprise Server 软件和许可证（有关更多信息，请联系 Micro Focus 销售。）
- LRS VPSX/MFI, LRS/Queue, and LRS/DIS 软件和许可证（欲了解更多信息，请联系 LRS 销售人员。）

> **Note**
>
> 有关大型机在线打印工作负载的配置注意事项的更多信息，请参阅此模式的 “其他信息” 部分中的注意事项。

产品版本

- Micro Focus 企业服务器 8.0 或更高版本
- LRS VPSX/MFI V1R3 或更高版本

架构

源技术堆栈

- 操作系统 - IBM z/OS
- 编程语言 - 面向业务的通用语言 (COBOL) 和客户信息控制系统 (CICS)
- 数据库 — DB2 适用于 z/OS 的 IBM IBM 信息管理系统 (IMS) 和虚拟存储访问方法 (VSAM)
- 安全 — 资源访问控制设施 (RACF)、适用于 z/OS 的 CA 绝密和访问控制设施 2 () ACF2
- 打印和输出管理 - IBM 大型机 z/OS 打印产品（ IBM Infoprint Server for z/OS、LRS 和 CA View ）

目标技术堆栈

- 操作系统 — 在亚马逊上运行的微软 Windows 服务器 EC2
- 计算 — 亚马逊 EC2
- 编程语言 - COBOL 和 CICS
- 数据库 - Amazon RDS
- 安全性 - AWS 托管的 Microsoft AD
- 打印和输出管理 - AWS 上的 LRS 打印解决方案
- 大型机运行时环境 - Micro Focus 企业服务器

源架构

下图显示了大型机在线打印工作负载的典型当前状态架构。

图表显示了以下工作流：

1. 用户在基于 COBOL 编写的 IBM CICS 应用程序构建的互动系统 (SoE) 上执行业务事务。
2. SoE 调用大型机服务，该服务将业务交易数据记录在 system-of-records (SoR) 数据库中，例如 IBM DB2 for z/OS。

3. SoR 保留来自 SoE 的业务数据。
4. 用户发起请求以从 CICS SoE 生成打印输出，CICS SoE 启动打印事务应用程序来处理打印请求。
5. 打印事务应用程序（例如 CICS 和 COBOL 程序）从数据库中提取数据，根据业务需求格式化数据，并生成业务输出（打印数据），例如账单、身份证或贷款对账单。然后，应用程序使用虚拟电信访问方法 (VTAM) 发送打印请求。z/OS 打印服务器（例如 IBM Infoprint Server）使用 NetSpool 或类似的 VTAM 组件来拦截打印请求，然后使用 JES 输出参数在 JES 缓冲池上创建打印输出数据集。JES 输出参数指定打印服务器用于将输出传输到特定网络打印机的路由信息。术语 VTAM 指的是 z/OS 通信服务器和 z/OS 的系统网络架构 (SNA) 服务元素。
6. 打印输出传输组件将输出打印数据集从 JES 假脱机传输到远程打印机或打印服务器，例如 LRS（如此模式中所示）、IBM Infoprint Server 或电子邮件目标。

目标架构

下图显示了部署在 Amazon Web Services Cloud 中的大型机在线打印工作负载的架构：

图表显示了以下工作流：

1. 用户从在线 (CICS) 用户界面发起打印请求以创建打印输出，例如账单、身份证或贷款对账单。
2. 大型机在线应用程序（已重定向 Amazon EC2）使用 Micro Focus Enterprise Server 运行时从应用程序数据库中提取数据，对数据应用业务逻辑，格式化数据，然后使用 Micro Focus CICS 打印出口 (DFHUPRNT) 将数据发送到打印目的地。
3. 应用程序数据库（在 Amazon RDS 上运行的 SoR）保留打印输出的数据。
4. LRS VPSX/MFI printing solution is deployed on Amazon EC2, and its operational data is stored in Amazon Elastic Block Store (Amazon EBS). LRS VPSX/MFI uses a TCP/IP-based LRS/Queue transmission agent to collect print data through the Micro Focus CICS Print Exit API (DFHUPRNT) and deliver the data to a specified printer destination. The original TERMID (TERM) that's used in the modernized CICS application is used as the VPSX/MFI 队列名称。

> **Note**
>
> 目标解决方案通常不需要更改应用程序以适应大型机格式化语言，例如 IBM 高级功能演示文稿 (AFP) 或施乐线路条件数据流 (LCDS)。有关在 AWS 上使用 Micro Focus 进行大型机应用程序迁移和现代化的更多信息，请参阅 AWS 文档中的使用 Micro Focus 在 AWS 上为企业大型机工作负载提供支持。

AWS 基础设施架构

下图显示了适用于大型机在线打印工作负载的高可用性且安全的 AWS 基础设施架构：

图表显示了以下工作流：

1. 大型机在线应用程序（用 CICS 或 COBOL 等编程语言编写）使用核心业务逻辑来处理和生成打印输出，例如账单、身份证和贷款对账单。该在线应用程序 EC2 跨两个可用区 (AZ) 部署在亚马逊上以实现高可用性 (HA)，并使用 Micro Focus CICS Print Exit 将打印输出路由到 LRS VPSX/MFI 以供最终用户打印。
2. LRS VPSX/MFI uses a TCP/IP-based LRS/Queue transmission agent to collect or capture print data from the Micro Focus online Print Exit programming interface. Online Print Exit passes the necessary information to enable LRS VPSX/MFI to effectively process the print file and dynamically build LRS/Queue 命令。

   > **Note**
   >
   > 有关用于打印的各种 CICS 应用程序编程方法以及 Micro Focus Enterprise 服务器和 LRS VPSX/MFI 如何支持这些方法的更多信息，请参阅此模式其他信息部分中的打印数据捕获。

3.

   > **Note**
   >
   > N etwork Load Balanc er 提供 DNS 名称，用于将 Micro Focus Enterprise Server 与 LRS 集成，VPSX/MFI. : LRS VPSX/MFI supports a Layer 4 load balancer. The Network Load Balancer also does a basic health check on LRS VPSX/MFI并将流量路由到运行状况良好的注册目标。

4. LRS VPSX/MFI 打印服务器部署在亚马逊上， EC2 跨越两个可用区，用于 HA，并使用 A mazon EBS 作为运营数据存储。LRS VPSX/MFI supports both the active-active and active-passive service modes. This architecture uses multiple Availability Zones in an active-passive pair as an active and hot standby. The Network Load Balancer performs a health check on LRS VPSX/MFI EC2 实例并在活动实例处于不健康状态时将流量路由到另一个可用区的热备用实例。在每个实例中，打印请求都保存在本地的 LRS Job Queue 中。 EC2 在恢复的情况下，必须重新启动失败的实例，LRS 服务才能继续处理打印请求。

> **Note**
>
> LRS VPSX/MFI 还可以在打印机机群级别执行运行状况检查。有关更多信息，请参阅此模式的其他信息部分中的打印机队列运行状况检查。

5. AWS Managed Microsoft AD 与 LRS/DIS 集成以执行打印工作流程身份验证和授权。有关更多信息，请参阅此模式的其他信息部分中的打印身份验证和授权。
6. LRS VPSX/MFI 使用 Amazon EBS 进行块存储。您可以将活动 EC2 实例中的 Amazon EBS 数据作为 point-in-time快照备份到 Amazon S3，然后将其恢复到热备用 EBS 卷。要自动创建、保留和删除 Amazon EBS 卷快照，您可以使用 Amazon Data Lifecycle Manager 设置自动快照的频率并根据 RTO/RPO 要求恢复快照。

工具

Amazon Web Services

- Amazon Elastic Block Store (Amazon EBS) 提供用于亚马逊实例的块级存储卷。 EC2 EBS 卷的行为类似于原始、未格式化的块储存设备。您可以将这些卷作为设备挂载在实例上。
- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- Amazon Relational Database Service (Amazon RDS) 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展关系数据库。
- 适用于 Microsoft Active Directory (AD) 的 AWS Directory Service，也称为 AWS Managed Microsoft Active Directory，可使您的目录感知工作负载和 AWS 资源在 AWS 中使用托管 Active Directory。

其他工具

- LRS VPSX/MFI（Micro Focus 接口）由 LRS 和 Micro Focus 共同开发，可捕获 Micro Focus Enterprise Server JES 假脱机的输出，并将其可靠地传送到指定的打印目的地。
- LRS 目录信息服务器 (LRS/DIS) 用于打印工作流期间的身份验证和授权。
- LRS/Queue is a TCP/IP-based LRS/Queue transmission agent, used by LRS VPSX/MFI，通过 Micro Focus 在线 Print Exit 编程接口收集或捕获打印数据。
- Micro Focus Enterprise Server 是大型机应用程序的应用程序部署环境。它为使用任何版本的 Micro Focus Enterprise Developer 迁移或创建的大型机应用程序提供执行环境。

操作说明

在 Amazon 上设置 Micro Focus 企业服务器 EC2 并部署大型机在线应用程序

Task	描述	所需技能
设置 Micro Focus Enterprise Server 并部署演示在线应用程序。	在亚马逊上设置 Micro Focus Enterprise Server EC2，然后 EC2 按照 Micro Focus 文档中的教程：CICS Support 中的说明在亚马逊上部署 Micro Focus 账户演示应用程序（ACC T 演示）。 ACCT 演示应用程序为大型机在线 (CICS) 应用程序，用于创建并启动打印输出。	云架构师

在亚马逊上设置 LRS 打印服务器 EC2

Task	描述	所需技能
获取用于打印的 LRS 产品许可证。	要获得 LRS 的 LRS 产品许可证VPSX/MFI, LRS/Queue, and LRS/DIS，请联系 LRS 输出管理团队。您必须提供将安装 LRS 产品的 EC2 实例的主机名。	构建 lead
创建亚马逊 EC2 Windows 实例来安装 LRS VPSX/MFI。	按照亚马逊 EC2 文档中的步骤 1：启动实例中的说明启动亚马逊 EC2 Windows 实例。您的实例必须满足 LRS VPSX/MFI 的以下硬件和软件要求： • CPU - 双核 • 内存 - 16 GB	云架构师

Task	描述	所需技能
	• 驱动器 - 500 GB • 最小 EC2 实例 — m5.xlarge • 操作系统 - Windows/Linux • 软件 - 互联网信息服务(IIS)或 Apache **Note** 上述硬件和软件要求适用于小型打印机群（大约 500—1000）。要获取完整的要求，请咨询您的 LRS 和 AWS 联系人。 创建 Windows 实例时，请执行以下操作： 1. 确认 EC2 主机名与 LRS 产品许可证使用的主机名相同。 2. 完成以下操作即可在 Amazon EC2 中启用 CGI： a. 按照 Amazon EC2 文档中步骤 2：连接到您的实例中的说明连接到您的实例。 EC2 b. 在 Windows 开始菜单中，找到并打开服务器管理器。 c. 在服务器管理器中，依次选择控制面板、快速启	

Task	描述	所需技能
	动、添加角色和功能。然后，选择服务器角色。 d. 在 “服务器角色” 中，选择 WebServer (IIS)，然后选择 “应用程序开发”。 e. 在应用程序开发中，选中 CGI 复选框。 f. 按照 Windows Server Manger 添加角色和功能向导中的说明安装 CGI。 g. 在 EC2 实例的 Windows 防火墙中打开端口 5500，进行 LRS/队列通信。	

Task	描述	所需技能
在实例上安装 LRS VPSX/MFI。 EC2	1. 按照 Amazon EC2 文档中步骤 2：连接到您的实例中的说明连接到您的实例。 EC2 2. **Note** 打开发送至您的邮箱的 LRS 电子邮件中的产品下载页面链接。: LRS 产品通过电子文件传输 (EFT) 分发。 3. 下载 LRS VPSX/MFI 并解压缩文件（默认文件夹：c:\LRS）。 4. 从解压缩的文件夹中启动 LRS 产品安装程序以安装 LRS VPSX/MFI。 5. 在选择功能菜单中，选择 VPSX® Server (V1R3.022)，然后选择下一步开始安装过程。安装完成后，您将收到一条成功消息。	云架构师

Task	描述	所需技能
安装 LRS/队列。	1. 按照亚马逊 EC2 文档中步骤 2：连接到您的 EC2 实例中的说明连接到您的 Micro Focus Enterprise Server 实例。 2. 从发送至您邮箱的 LRS 电子邮件中打开 LRS 产品下载页面的链接，下载 LRS/Queue，然后解压缩文件。 3. 转到下载文件的位置，然后启动 LRS 产品安装程序以安装 LRS/Queue。	云架构师
安装 LRS/DIS。	1. 按照亚马逊文档中步骤 2：连接到您的 EC2 实例中的说明，连接到您的 LRS VPSX/MFI 实例。 EC2 2. 从发送至您邮箱的 LRS 电子邮件中打开 LRS 产品下载页面的链接，下载 LRS/DIS，然后解压缩文件。 3. 转到下载文件的位置，然后启动 LRS 产品安装程序。 4. 在 LRS 产品安装程序中，展开 LRS 其他工具，选择 LRS DIS，然后选择下一步。 5. 按照 LRS 产品安装程序中的其余说明完成安装过程。	云架构师

Task	描述	所需技能
创建目标组并将 LRS VPSX/MF EC2 I 注册为目标。	按照弹性负载均衡器文档中为网络负载均衡器创建目标组的说明创建目标组。 创建目标组时，请执行以下操作: 1. 在指定组详细信息页面上，对于选择目标类型，选择实例。 2. 对于协议，选择 TCP。 3. 对于端口，选择 5500。 4. 在注册目标页面的可用实例部分中，选择 LRS VPS EC2 X/MFI 实例。	云架构师
创建网络负载均衡器。	按照弹性负载均衡器文档中创建网络负载均衡器的说明进行操作。您的 Network Load Balancer 将流量从 Micro Focus 企业服务器路由到 LRS V EC2 PSX/MFI。 创建网络负载均衡器时，在侦听器和路由页面上执行以下操作： 1. 对于协议，选择 TCP。 2. 对于端口，选择 5500。 3. 对于默认操作，选择转发至您之前创建的目标组。	云架构师

将 Micro Focus 企业服务器与 LRS 集成 VPSX/MFI and LRS/Queue

Task	描述	所需技能
配置 Micro Focus Enterprise Server 以进行 LRS/队列集成。	1. 按照亚马逊 EC2 文档中步骤 2：连接到您的 EC2 实例中的说明连接到您的 Micro Focus Enterprise Server 实例。 2. 在 Windows 开始菜单上，打开 Micro Focus Enterprise Server 管理 UI。 3. 在菜单栏中，选择 NATIVE。 4. 在导航窗格中，选择 Directory Server，然后选择 BANKDEMO 或您的 Enterprise 服务器区域。 5. 从左侧导航窗格中的常规中，向下滚动到其他部分，将环境变量 (LRSQ_ADDRESS、LRSQ_PORT、LRSQ_COMMAND) 配置为指向 LRSQ。 6. 对于 LRSQ_ADDRESS，输入您之前创建的网络负载均衡器的 IP 地址或 DNS 名称。 7. 对于 LRSQ_PORT，输入 VPSX LRSQ 侦听器端口 (5500)。 8. 对于 LRSQ_COMMAND，请输入 LRSQ 可执行文件的路径位置。	云架构师

Task	描述	所需技能
	9. **Note** LRS 目前支持 DNS 名称的最大字符限制为 50，但将来可能会发生变化。如果您的 DNS 名称字符数大于 50，则可以使用网络负载均衡器的 IP 地址作为替代。	

Task	描述	所需技能
使 CICS 打印出口 (DFHUPRNT) 可用于 Micro Focus Enterprise Server 初始化。	1. 按照亚马逊 EC2 文档中步骤 2：连接到您的 EC2 实例中的说明连接到您的 Micro Focus Enterprise Server 实例。 2. **Note** 将 CICS Print Exit (DFHUPRNT) 从 LRS VPSX/MFI 可执行文件夹（已命名VPSX_MFI_R2 ）复制到 Micro Focus Enterprise Server 实例位置。 EC2 对于 32 位系统，该位置是 C:\Program Files (x86)\Micro Focus\Enterprise Server\bin 。对于 64 位系统，该位置是 C:\Program Files (x86) \Micro Focus\Enterprise Server\bin64 。：复制DFHUPRNT.dll 时必须将DFHUPRNT_	云架构师

Task	描述	所需技能
	64.dll 文件重命名为。 验证 Micro Focus Enterprise Server 是否已检测到 CICS Print Exit (DFHUPRNT) 1. 停止并启动 Micro Focus Enterprise Server。 2. 在 Micro Focus Enterprise Server 的管理面板中，依次打开监控、日志、控制台日志。 3. 检查控制台日志中是否有以下消息：“3270 打印机用户退出 DFHUPRNT 安装成功。”	

Task	描述	所需技能
将 CICS 打印机的终端 ID (TERMIDs) 定义为 Micro Focus 企业服务器。	在 Micro Focus Enterprise Server 中启用 3270 打印 1. 在 Micro Focus Enterprise Server 的管理面板中，依次打开 CICS、资源、按组。 2. 在左侧导航面板中，选择 SIT（系统初始化表），然后选择 BNKCICV。 3. 在常规部分中，向下滚动到 3270，然后选中 3270 打印复选框。 在 Micro Focus Enterprise Server 中定义 CICS 打印机终端 1. 在 Micro Focus Enterprise Server 管理面板中，依次打开 CICS、资源、按类型。 2. 在左侧导航窗格中，选择术语，然后选择新术语。即将打开创建终端资源表单。 3. 对于名称，输入 LRS 打印队列的名称。（注意：此模式使用“P275”作为 CICS 打印机的终端 ID 和 LRS VPSX 打印队列。） 4. 对于组，请输入 BANKTERM。 5. 对于“自动安装 - 型号”，输入“否”。	云架构师

Task	描述	所需技能
	6. 在 “终端标识符-终端类型” 中，输入DFHPRT32。 7. 在 “网络名称” 中，输入VTAMP275。 8. 对于终端使用，请选择服务中复选框。 9. 滚动至页面顶部并选择保存。 10选择安装。弹出显示安装成功的消息。	

在 Micro Focus Enterprise Server 和 LRS VPSX/MFI 中设置打印机和打印用户

Task	描述	所需技能
在 LRS VPSX 中创建打印队列。	1. 按照亚马逊文档中步骤 2：连接到您的 EC2 实例中的说明，连接到您的 LRS VPSX/MFI 实例。 EC2 2. 从 Windows 开始菜单打开 VPSX Web 界面。 3. 在导航窗格中，选择打印机。 4. 选择添加，然后选择添加打印机。 5. 在打印机配置页上，对于打印机名称，输入 P275。 6. 要获取 VPSX ID，请输入。VPS1 7. 对于 CommType，请选择 TCPIP/ LRSQ。	云架构师

Task	描述	所需技能
	8. 对于主机/IP 地址，输入要添加的物理打印机的 IP 地址。 9. 对于设备，输入您的设备名称。 10选择 Windows 驱动程序或 Linux/Mac 驱动程序。 11选择 添加。 **Note** 打印队列必须等同于 Micro Focus Enterprise Server 中 TERMIDs 创建的打印队列。	

Task	描述	所需技能
在 LRS VPSX/MFI 中创建打印用户。	1. 按照亚马逊文档中步骤 2：连接到您的 EC2 实例中的说明，连接到您的 LRS VPSX/MFI 实例。 EC2 2. 从 Windows 开始菜单打开 VPSX Web 界面。 3. 在导航窗格中，选择安全，然后选择用户。 4. 在用户名列中，选择 admin，然后选择复制。 5. 在“用户配置文件维护”窗口中，在“用户名”中输入用户名（例如 PrintUser）。 6. 对于说明，输入简短说明（例如，测试打印的用户）。 7. 选择更新。这将创建一个打印用户（例如，PrintUser）。 8. 在导航窗格中的用户下方，选择您创建的新用户。 9. 从命令菜单中选择安全。 10. 在安全规则页面上，选择所有适用的打印机安全和作业安全选项，然后选择保存。 11. 要将新的打印用户添加到管理员组，请在导航窗格中选择安全，然后选择配置。 12. 在安全配置窗口中，将新的打印用户添加到管理员列中。	云架构师

设置身份验证和授权

Task	描述	所需技能
使用用户和组创建 AWS Managed Microsoft AD 域。	1. 按照 AWS Directory Service 文档中创建 AWS Managed Microsoft AD 目录的说明，在 AWS Managed Microsoft AD 上创建 Active Directory。 2. 按照 AWS Directory Service 文档中步骤 3：部署 EC2 实例来管理 AWS 托管的 Microsoft AD 中的说明，部署 EC2 实例（活动目录管理器）并安装活动目录工具来管理你的 AWS 托管的 Microsoft AD。 3. **Note** 按照 Amazon EC2 文档中步骤 2：连接到您的实例中的说明连接到您的实例。EC2 ：当您连接到 EC2 实例时，请在 Windows 安全窗口中输入您的管理员凭据（用于您在步骤一中创建的目录）。 4. 在 Windows 开始菜单的 Windows 管理员工具下方，选择 Active Directory 用户及计算机。	云架构师

Task	描述	所需技能
	5. 按照 AWS Directory Service 文档中创建用户的步骤在 Active Directory 域中创建打印用户。	
加入 LRS VPSX/MFI，进入 AWS 托管 EC2 的微软 AD 域。	自动加入你的 AWS 托管微软 AD 域（AWS 知识中心文档）或手动加入 LRS VPSX/MFI EC2（AWS Directory Service 文档）。	云架构师
配置 LRS/DIS 并将其与 AWS Managed Microsoft AD 集成。	1. 按照亚马逊文档中步骤 2：连接到您的 EC2 实例中的说明，连接到您的 LRS VPSX/MFI 实例。 EC2 2. 在 Windows 开始菜单中，打开 VPSX Web 界面。 3. 在导航窗格中，选择安全，然后选择配置。 4. 在安全配置页面的安全参数部分中，对于安全类型，选择内部。 5. 在安全参数部分中输入您对其余选项的首选项。 6. 从 Microsoft Windows 开始菜单中打开 LRS 输出管理文件夹，选择服务器启动，然后选择服务器停止。 7. 使用您的 Active Directory 用户名和密码登录 LRS VPSX/MFI。	云架构师

测试在线打印工作流

Task	描述	所需技能
从 Micro Focus ACCT 演示应用程序发起在线打印请求。	1. **Note** 在 Micro Focus 企业服务器 EC2 实例中打开 TN327 0 终端仿真器。(: 此模式使用 3270 终端仿真器。) 2. 连接到 TN327 0 终端仿真器 (Rumba)。对于主机名地址，请使用 127.0.0.1。对于 Telnet 端口，请使用 9270。 3. 连接到 3270 屏幕后，按 CTRL+SHIFT+Z 清屏。 4. 要启动 ACCT 演示应用程序，请在清晰的屏幕中输入 ACCT。此操作将打开 ACCT 在线演示 (CICS) 应用程序主屏幕。注意：主屏幕包括菜单选项，例如账户文件、按名称搜索、输入、请求类型、账户和打印机。 5. 若要从 ACCT 在线演示 (CICS) 应用程序提交打印请求，请在请求类型字段中输入 P，在账户字段中输入 11111，在打印机字段中输入 P275。确保将打印机字	云架构师

Task	描述	所需技能
	段中的值设置为 CICS 打印机终端 ID 值。 6. 按 Enter 键。 “打印请求已安排”消息出现在屏幕底部。该消息表明 ACCT 演示版应用程序已生成在线打印请求，并已将其发送到 LRS VPS/MFI 进行打印处理。	

Task	描述	所需技能
检查 LRS VPSX/MFI 中的打印输出。	1. 按照亚马逊文档中步骤 2：连接到您的 EC2 实例中的说明，连接到您的 LRS VPSX/MFI 实例。 EC2 2. 在 Windows 开始菜单中，打开 VPSX Web 界面。 3. 在导航窗格中，选择打印机，然后选择输出队列。找到您之前为在线打印创建的 P275 打印队列。 4. 对于打印队列 (P275)，在假脱机 ID 列中，选择打印机队列中请求的假脱机 ID。 5. 在操作选项卡的命令列中，选择浏览。 现在，您可以看到账户对账单的打印输出，其中包含账号、姓氏、名字、地址、电话、发卡数量、发行日期、金额和余额等列。 有关示例，请参阅此模式的 online_print_output 附件。	测试工程师

相关资源

- LRS 输出现代化（LRS 文档）
- VTAM 网络概念（IBM 文档）
- 逻辑单元 (LU) 类型摘要（IBM 文档）
- ANSI 和机器托架控制（IBM 文档）

- 使用 Micro Focus 在 AWS 上为企业大型机工作负载提供支持（Amazon Web Services Partner Network 博客）
- 使用 Amazon A EC2 uto Scaling 和 Systems Manager 构建 Micro Focus 企业服务器 PAC（AWS Prescriptive Guidence 文档）
- Advanced Function Presentation (AFP)数据流（IBM 文档）
- 线路条件数据流 (LCDS)（比较文档）

其他信息

注意事项

在现代化改造过程中，您可以考虑大型机联机流程的各种配置及其生成的输出。每个使用大型机平台的客户和供应商都根据直接影响打印的特殊要求对平台进行了定制。例如，您当前的平台可能会将 IBM 高级功能演示 (AFP) 或 Xerox 线路条件数据流 (LCDS) 合并到当前工作流程中。此外，大型机回车控制字符和通道命令字可能会影响打印页面的外观，可能需要特殊处理。作为现代化规划过程的一部分，我们建议您评测并了解特定打印环境中的配置。

打印数据采集

本节总结了可在 IBM 大型机环境中用于打印的 CICS 应用程序编程方法。LRS VPSX/MFI components provide techniques to allow the same application programs to create data in the same way. The following table describes how each application programming method is supported in a modernized CICS application running in AWS and Micro Focus Enterprise Server with an LRS VPSX/MFI 打印服务器。

方法	描述	在现代化环境中支持该方法
执行 CICS 发送文本或执行 CICS 发送地图。	这些 CICS 和 VTAM 方法负责创建 3270/SCS 打印数据流并将其传送到 LUTYPE0 和打印设备。 LUTYPE1 LUTYPE3	Micro Focus 在线打印出口 (DFHUPRNT) 应用程序接口 (API) 允许使用这两种方法中的任何一种创建的VPSX/MFI when 3270/SCS打印数据流来处理打印数据。
执行 CICS SEND TEXT 或执行 CICS SEND MAP（使用第三方 IBM 大型机软件）。	CICS 和 VTAM 方法负责创建 3270/SCS 打印数据流并将其传送到 LUTYPE0 和打印设备。 LUTYPE1 LUTYPE3 第三	Micro Focus 在线打印退出 (DFHUPRNT) API 允许使用这两种方法中的任何一种创建的

	方软件产品拦截打印数据，使用 ASA/MCH 控制字符将数据转换为标准打印格式数据，并将数据放置在 JES 假脱机上，以便由使用 JES 的基于大型机的打印系统进行处理。	VPSX/MFI when 3270/SCS打印数据流来处理打印数据。
执行 CICS SPOOLOPEN	CICS 应用程序使用此方法将数据直接写入 JES 假脱机。然后，这些数据就可以由使用 JES 的基于大型机的打印系统进行处理。	Micro Focus Enterprise Server 将数据假脱机到企业服务器缓冲池，在那里可以通过 VPSX/MFI Batch Print Exit LRSPRTE6 () 进行处理，该退出将数据后台处理到 VPSX。
DRS/API	LRS 提供的编程接口用于将打印数据写入 JES。	VPSX/MFI 提供了一个替换接口，可将打印数据直接后台打印到 VPSX。

打印机队列运行状况检查

LRS VPSX/MFI (LRS LoadX) 可执行深入的运行状况检查，包括设备管理和操作优化。设备管理可以检测打印机设备中的故障，并将打印请求路由到正常运行的打印机。有关打印机队列的深入运行状况检查的详细信息，请参阅产品许可证附带的 LRS 文档。

打印身份验证和授权

LRS/DIS enables LRS applications to authenticate user IDs and passwords by using Microsoft Active Directory or an LDAP server. In addition to basic print authorization, LRS/DIS还可以在以下用例中应用粒度级别的打印安全控制：

- 管理谁可以浏览打印机作业。
- 管理其他用户作业的浏览级别。
- 管理操作任务。例如，命令级安全，例如保留/释放、清除、修改、复制和重新路由。安全性可以通过用户 ID 或组（类似于 AD 组或 LDAP 组）来设置。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 Transfer Family 将大型机文件直接移动到 Amazon S3

由 Luis Gustavo Dantas (AWS) 编写

摘要

作为现代化之旅的一部分，您可能会面临在本地服务器和 Amazon Web Services (AWS) 云之间传输文件的挑战。从大型机传输数据可能是一项重大挑战，因为大型机通常无法访问 Amazon Simple Storage Service (Amazon S3)、Amazon Elastic Block Store (Amazon EBS) 或 Amazon Elastic File System (Amazon EFS) 等现代数据存储。

许多客户使用中间暂存资源（如本地 Linux、Unix 或 Windows 服务器）将文件传输到 AWS Cloud。您可以通过使用 AWS Transfer Family 和 Secure Shell (SSH) 文件传输协议 (SFTP) 将大型机文件直接上传到 Amazon S3 来避免这种间接方法。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 具有可由您的旧平台访问的子网的虚拟私有云（VPC）
- 您 VPC 的 Transfer Family 端点
- 大型机虚拟存储访问方法 (VSAM) 文件转换为连续的固定长度文件（IBM 文档）

限制

- SFTP 默认以二进制模式传输文件，这意味着文件上传到 Amazon S3 时会保留 EBCDIC 编码。如果您的文件不包含二进制或打包数据，那么您可以使用 sftp ascii 子命令（IBM 文档）在传输过程中将文件转换为文本。
- 您必须解压包含打包和二进制内容的大型机文件 (AWS Prescriptive Guidance)，才能在目标环境中使用这些文件。
- Amazon S3 对象的大小范围从最小 0 字节到最大 5 TB。有关 Amazon S3 功能的更多信息，请参阅亚马逊 S3 FAQs。

架构

源技术堆栈

- 作业控制语言 (JCL)
- z/OS Unix shell 和 ISPF
- SFTP
- VSAM 和平面文件

目标技术堆栈

- Transfer Family
- Amazon S3
- Amazon Virtual Private Cloud(Amazon VPC)

目标架构

下图显示了将 Transfer Family 与 SFTP 结合使用以将大型机文件直接上传到 S3 存储桶的参考架构。

图表显示了以下工作流：

1. 您可以使用 JCL 作业通过 Direct Connect 将大型机文件从旧版大型机传输到 Amazon Web Services Cloud。
2. Direct Connect 使您的网络流量能够保留在 AWS 全球网络上，并绕过公共互联网。Direct Connect 还提高了网络速度，从 50 Mbps 开始，扩展到 100 Gbps。
3. VPC 端点无需使用公共互联网即可在您的 VPC 资源和支持的服务之间建立连接。对 Transfer Family 和 Amazon S3 的访问通过位于两个私有子网和可用区中的弹性网络接口实现高可用性。
4. Transfer Family 对用户进行身份验证，并使用 SFTP 从旧环境接收文件并将其移动到 S3 存储桶。

自动化和扩缩

Transfer Family 服务到位后，您可以使用 JCL 作业作为 SFTP 客户端，将无限数量的文件从大型机传输到 Amazon S3。当您准备好传输大型机文件时，还可以使用大型机批处理作业计划程序运行 SFTP 作业，从而自动执行文件传输。

工具

- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。
- AWS Transfer Fam ily 使您能够使用 SFTP、FTPS 和 FTP 协议安全地将定期 business-to-business 文件传输扩展到亚马逊 S3 和亚马逊 EFS。

操作说明

创建 S3 存储桶和访问策略

Task	描述	所需技能
创建 S3 存储桶。	创建一个 S3 存储桶来托管您从旧环境传输的文件。	常规 AWS
创建 IAM 角色和策略。	Transfer Family 使用您的 AWS Identity and Access Management (IAM) 角色授予对您之前创建的 S3 存储桶的访问权限。 创建 IAM 角色，包含以下 IAM policy： `{ "Version": "2012-10-17", "Statement": [ { "Sid": "UserFolderListing", "Action": [ "s3:ListBucket", "s3:GetBucketLocation" ], "Effect": "Allow",`	常规 AWS

Task	描述	所需技能
	"Resource": ["arn:aws:s3:::<your-bucket-name>"] }, { "Sid": "HomeDirObjectAccess", "Effect": "Allow", "Action": ["s3:PutObject", "s3:GetObjectAcl", "s3:GetObject", "s3:DeleteObjectVersion", "s3:DeleteObject", "s3:PutObjectAcl", "s3:GetObjectVersion"], "Resource": "arn:aws:s3:::<your-bucket-name>/*" }] }	

Task	描述	所需技能
	Note 创建 IAM 角色时，必须选择转移用例。	

定义传输服务

Task	描述	所需技能
创建 SFTP 服务器。	1. 登录 AWS 管理控制台，打开 Transfer Family 控制台，然后选择创建服务器。 2. 仅选择 SFTP（SSH 文件传输协议）- 通过 Secure Shell 协议传输文件，然后选择下一步。 3. 对于身份提供商，选择服务托管，然后选择下一步。 4. 对于端点类型，选择 VPC 托管。 5. 对于访问，选择内部。 6. 对于 VPC，选择您的 VPC。 7. 在可用区部分，请选择您的可用区和子网。 8. 在安全组部分，选择您的安全组，然后选择下一步。 9. 对于域，选择 Amazon S3，然后选择下一步。 10. 保留配置其他详细信息页面上的默认选项，然后选择下一步。	常规 AWS

Task	描述	所需技能
	11选择 Create server (创建服务器)。 **Note** 有关如何设置 SFTP 服务器的更多信息，请参阅创建支持 SFTP 的服务器（AWS Transfer Family 用户指南）。	
获取服务器地址。	1. 打开 Transfer Family 控制台，然后在 Server ID 列中选择您的服务器 ID。 2. 在端点详细信息部分中，对于端点类型，选择端点 ID。您将进入 Amazon VPC 控制台。 3. 在 Amazon VPC 控制台的详细信息选项卡上，找到 DNS 名称旁边的 DNS 名称。	常规 AWS
创建 SFTP 客户端密钥对。	为微软 Windows 或创建 SSH 密钥对macOS/Linux/UNIX。	常规 AWS、SSH

Task	描述	所需技能
创建 SFTP 用户。	1. 打开 Transfer Family 控制台，从导航窗格中选择服务器，然后选择您的服务器。 2. 在服务器 ID 列中，选择您的服务器的服务器 ID，然后选择添加用户。 3. 对于用户名，输入与您的 SSH 密钥对用户名匹配的用户名。 4. 对于角色，选择您之前创建的 IAM 角色。 5. 对于主目录，选择您之前创建的 S3 存储桶。 6. 对于 SSH 公有密钥， 输入您之前 创建的密钥对。 7. 选择 添加。	常规 AWS

传输大型机文件

Task	描述	所需技能

将 SSH 私钥发送到大型机。	使用 SFTP 或 SCP 将 SSH 私钥发送到旧版环境。
	
	SFTP 示例：
	
	`sftp [USERNAME@mainframeIP]`
	`[password]`
	`cd [/u/USERNAME]`
	`put [your-key-pair-file]`
	
	SCP 示例：
		大型机、z/OS Unix shell、FTP、SCP

Task	描述	所需技能
	`scp [your-key-pair-file] [USERNAME@MainframeIP]:/[u/USERNAME]` 接下来，将 SSH 密钥存储在 z/OS Unix 文件系统中稍后将运行文件传输批处理作业的用户名下（例如，`/u/CONTROLM `）。 **Note** 有关 z/OS Unix 外壳的更多信息，请参阅 z/OS 外壳简介（IBM 文档）。	

Task	描述	所需技能

创建 JCL SFTP 客户端。	由于大型机没有本机 SFTP 客户端，因此您必须使用 BPXBATCH 实用程序从 z/OS Unix shell 运行 SFTP 客户端。
	
	在 ISPF 编辑器中，创建 JCL SFTP 客户端。例如：
	
	`//JOBNAM JOB ...`
	`//************`
	`******************`
	`******************`
	`******************`
	`****`
	`//SFTP EXEC PGM=BPXBA`
	`TCH,REGION=0M`
	`//STDPARM DD *`
	`SH cp "//'MAINF`
	`RAME.FILE.NAME'"`
	` filename.txt;`
	`echo 'put filename.txt'`
	` > uplcmd;`
	`sftp -b uplcmd -i`
	` ssh_private_key_fi`
	`le ssh_username@<tran`
	`sfer service ip or`
	` DNS>;`
	`//SYSPRINT DD SYSOUT=*`
	`//STDOUT DD SYSOUT=*`
	`//STDENV DD *`
	`//STDERR DD SYSOUT=*`
	
	Note
	有关如何在 z/OS Unix 外壳中运行命令的更多信息，请参阅 BPXBATCH 实用程序
		JCL、大型机、z/OS Unix shell

Task	描述	所需技能
	（IBM 文档）。有关如何在 z/OS 中创建或编辑 JCL 作业的更多信息，请参阅什么是 ISPF？和 ISPF 编辑器（IBM 文档）。	
运行 JCL SFTP 客户端。	1. 在 ISPF 编辑器中，输入 SUB，然后在创建 JCL 作业后按 ENTER 键。 2. 在 SDSF 中监视大型机的文件传输批处理作业活动。 **Note** 有关如何检查批处理作业活动的更多信息，请参阅 z/OS SDSF 用户指南（IBM 文档）。	大型机、JCL、ISPF
验证文件传输。	1. 登录 AWS 管理控制台，打开 Amazon S3 控制台，然后从导航窗格中选择存储桶。 2. 选择与您的 Transfer Family 关联的存储桶。 3. 在对象选项卡的对象部分中，找到您从大型机传输的文件。	常规 AWS

Task	描述	所需技能
自动执行 JCL SFTP 客户端。	使用作业调度程序自动触发 JCL SFTP 客户端。 **Note** 您可以使用大型机作业调度程序，例如 BMC Control-M 或 CA Workload Automation，根据时间和其他批处理作业依赖性自动执行文件传输的批处理作业。	作业调度程序

相关资源

- AWS Transfer Family 的工作原理
- 利用 AWS 实现大型机现代化

使用可信上下文在 AWS 上的 Db2 联合身份验证数据库中保护和简化用户访问

由 Sai Parthasaradhi (AWS) 创建

摘要

许多公司正在将其传统的大型机工作负载迁移至 Amazon Web Services (AWS)。此次迁移包括在亚马逊弹性计算云（亚马逊）上将适用于 z/OS 数据库的 IBM Db2 迁移到适用于 Linux、Unix 和 Windows 的 Db2 (LUW)。 EC2在从内部部署到 AWS 的分阶段迁移过程中，用户可能需要访问 IBM Db2 z/OS 和 Amazon 上的 Db2 LUW 中的数据， EC2 直到所有应用程序和数据库都完全迁移到 Db2 LUW。鉴于不同平台使用不同身份验证机制，故在该远程数据访问场景中进行用户身份验证可能具有挑战性。

此模式涵盖了如何通过将 db2 for z/OS 作为远程数据库在 Db2 for LUW 上设置联合身份验证服务器。此模式使用可信上下文将用户身份从 Db2 LUW 传播至 Db2 z/OS，而无需在远程数据库上重新进行身份验证。有关可信上下文的更多信息，请参阅其他信息部分。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 在 Amazon EC2 实例上运行的 Db2 实例
- 在本地运行的适用于z/OS 数据库的远程 Db2
- 通过 AWS Site-to-SiteVPN 或 AWS Di rect Connect 连接到 AWS 的本地网络

架构

目标架构

工具

Amazon Web Services

- 亚马逊弹性计算云 (Amazon EC2) 在 AWS 云中提供可扩展的计算容量。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- AWS Site-to-Site VPN 可帮助您在您在 AWS 上启动的实例和您自己的远程网络之间传递流量。

其他工具

- db2cli 是 Db2 交互式命令行界面（CLI）命令。

操作说明

在 AWS 上运行的 Db2 LUW 数据库上启用联合身份验证

Task	描述	所需技能
在 DB2 LUW 数据库上启用联合。	要在 DB2 LUW 上启用联合，请运行以下命令。 `update dbm cfg using federated YES`	数据库管理员
重新启动数据库。	若要重新启动数据库，请运行以下命令。 `db2stop force; db2start;`	数据库管理员

对远程数据库编目

Task	描述	所需技能
对远程 Db2 z/OS 子系统编目。	若要对在 AWS 上运行的 Db2 LUW 的远程 Db2 z/OS 数据库进行编目，请使用以下示例命令。 `catalog TCPIP NODE tcpnode REMOTE mainframehost SERVER mainframeport`	数据库管理员

Task	描述	所需技能
对远程数据库编目。	要对远程数据库进行编目，请使用以下示例命令。 `catalog db dbnam1 as ndbnam1 at node tcpnode`	数据库管理员

创建远程服务器定义

Task	描述	所需技能
收集远程 Db2 z/OS 数据库用户凭证。	在继续执行后续步骤前，请收集以下信息： • Db2 z/OS 子系统名称 – 上一步骤在 LUW 上编入目录的 Db2 z/OS 名称（例如`ndbnam1`） • Db2 z/OS 版本 – Db2 z/OS 子系统版本（例如 12） • Db2 z/OS 用户 ID – 具有 BIND 权限（仅用于创建服务器定义）用户（例如`dbuser1`） • Db2 z/OS 密码 – `dbuser1` 的密码（例如`dbpasswd`） • Db2 z/OS 代理用户 – 代理用户的 ID，用于建立可信连接（例如`zproxy`） • Db2 z/OS 代理密码 – `zproxy` 用户的密码（例如`zproxy`）	数据库管理员

Task	描述	所需技能
创建 DRDA 包装器。	若要创建 DRDA 角色，请运行以下命令。 `CREATE WRAPPER DRDA;`	数据库管理员
创建服务器定义。	若要创建服务器定义，请运行以下示例命令。 `CREATE SERVER ndbserver TYPE DB2/ZOS VERSION 12 WRAPPER DRDA AUTHORIZATION "dbuser1" PASSWORD "dbpasswd " OPTIONS ( DBNAME 'ndbnam1',FED_PROXY_USER 'ZPROXY' );` 在此定义中，FED_PROXY_USER 指定将用于建立与 Db2 z/OS 数据库的可信连接的代理用户。只有在 Db2 LUW 数据库中创建远程服务器对象时，才需要授权用户 ID 和密码。将来不用于运行时。	数据库管理员

创建用户映射

Task	描述	所需技能
为代理用户创建用户映射。	若要创建代理用户的用户映射，请运行以下命令。 `CREATE USER MAPPING FOR ZPROXY SERVER ndbserver OPTIONS (REMOTE_AUTHID`	数据库管理员

Task	描述	所需技能
	'ZPROXY', REMOTE_PA SSWORD 'zproxy');	
在 Db2 LUW 上为每个用户创建用户映射。	为 AWS 上 Db2 LUW 数据库中所有需要通过代理用户访问远程数据的用户创建用户映射。若要创建用户组映射，请运行以下命令。 CREATE USER MAPPING FOR PERSON1 SERVER ndbserver OPTIONS (REMOTE_AUTHID 'USERZID', USE_TRUST ED_CONTEXT 'Y'); 该语句指定 Db2 LUW (PERSON1) 上的用户可以与远程 Db2 z/OS 数据库 (USE_TRUSTED_CONTEXT 'Y') 建立可信连接。通过代理用户建立连接后，用户可以使用 Db2 z/OS 用户 ID (REMOTE_AUTHID 'USERZID') 访问数据。	数据库管理员

创建可信上下文对象

Task	描述	所需技能
创建可信上下文对象	若要在远程 Db2 z/OS 数据库上创建可信上下文对象，请使用以下示例命令。 CREATE TRUSTED CONTEXT CTX_LUW_ZOS	数据库管理员

Task	描述	所需技能
	BASED UPON CONNECTIO N USING SYSTEM AUTHID ZPROXY ATTRIBUTES (ADDRESS '10.10.10.10') NO DEFAULT ROLE ENABLE WITH USE FOR PUBLIC WITHOUT AUTHENTIC ATION; 在此定义中，CTX_LUW_ZOS 是可信上下文对象的任意名称。该对象包含代理用户 ID 以及可信连接的必需服务器的 IP 地址。在此示例中，服务器为 AWS 上的 Db2 LUW 数据库。您可以使用域名，而不是 IP 地址。该子句 WITH USE FOR PUBLIC WITHOUT AUTHENTICATION 表示允许在可信连接上切换每个用户 ID。不需要提供密码。	

相关资源

- IBM 资源访问控制设施 (RACF)
- IBM Db2 LUW 联合身份验证
- 可信上下文

其他信息

Db2 可信上下文

可信上下文是定义了联合服务器和远程数据库服务器的信任关系的 Db2 数据库对象。若要定义信任关系，则利用可信上下文指定信任属性。有三类可信属性：

- 发出初始数据库连接请求的系统授权 ID
- 建立连接的 IP 地址或域名
- 数据库服务器和数据库客户端之间的数据通信加密设置

当连接请求的所有属性都与服务器上定义的任何可信上下文对象中指定的属性匹配时，就会建立可信连接。有两类可信连接：隐式连接与显式连接。建立隐式可信连接后，用户将继承在该可信连接定义范围之外无法使用的角色。建立显式可信连接后，无论是否进行身份验证，都可在同一个物理连接上切换用户。此外，可向 Db2 用户授予用于指定仅限可信连接使用的权限的角色。此模式使用了显式可信连接。

此模式下的可信上下文

模式完成后， PERSON1 在 Db2 上，LUW 使用联合可信上下文访问来自 Db2 z/OS 的远程数据。如果连接来自 PERSON1 可信上下文定义中指定的 IP 地址或域名，则通过代理用户建立连接。建立连接后，无需重新进行身份验证即可切换相应 PERSON1的 Db2 z/OS 用户 ID，并且用户可以根据为该用户设置的 Db2 权限访问数据或对象。

联合可信上下文的优势

- 此方法坚持了最低权限原则，避免了通用用户 ID 或应用程序 ID 获取所有用户所需权限的超集。
- 在联合身份验证数据库和远程数据库执行事务的用户的真实身份始终为已知并可予以审计。
- 性能得到提高，因为用户之间可以重复使用物理连接，无需联合服务器重新进行身份验证。

以 CSV 文件形式将大规模 Db2 z/OS 数据传输到 Amazon S3

创建者：Bruno Sahinoglu (AWS)、Ivan Schuster (AWS) 和 Abhijit Kshirsagar (AWS)

摘要

在许多企业中，大型机仍然是一个记录系统，它包含大量数据，包括含有当前和历史业务交易记录的主数据实体。它通常是孤立的，不容易从同一企业中的分布式系统访问。随着云技术的出现和大数据的民主化，企业有兴趣利用隐藏在大型机数据中的见解来开发新的业务能力。

为了实现这一目标，企业希望将其大型机 Db2 数据开放到他们的 Amazon Web Services（AWS）Cloud 环境。业务原因有几个，传输方法因案例而异。您可能更喜欢将应用程序直接连接到大型机，或者您可能更喜欢近乎实时地复制数据。如果用例是为数据仓库或数据湖提供数据，那么拥有 up-to-date副本就不再是一个问题，此模式中描述的过程可能就足够了，尤其是在您想避免任何第三方产品许可成本的情况下。另一个用例可能是迁移项目的大型机数据传输。在迁移场景中，执行功能等效性测试需要数据。本文中描述的方法是将 Db2 数据传输到 Amazon Web Services Cloud 环境的一种经济高效的方法。

由于亚马逊简单存储服务 (Amazon S3) Simple Service 是集成度最高的 AWS 服务之一，因此您可以使用其他 AWS 服务（例如亚马逊 Athena、AWS Lambda 函数或亚马逊）直接从那里访问数据并收集见解。 QuickSight 您也可以使用 AWS Glue 或 AWS Database Migration Service（AWS DMS）将数据加载到 Amazon Aurora 或 Amazon DynamoDB。考虑到这一目标，本文描述了如何在大型机上以 ASCII 格式卸载 CSV 文件中的 Db2 数据，然后将文件传输到 Amazon S3。

为此，开发了大型机脚本来帮助生成作业控制语言 (JCLs)，以便根据需要卸载和传输任意数量的 Db2 表。

先决条件和限制

先决条件

- 有权运行重构扩展执行程序 (REXX) 和 JCL 脚本的 IBM z/OS 操作系统用户。
- 访问 z/OS Unix 系统服务 (USS) 以生成 SSH (Secure Shell) 私有密钥和公有密钥。
- 一个可写的 S3 存储桶。有关更多信息，请参阅 Amazon S3 文档中的创建第一个 S3 存储桶。
- 一台启用 AWS Transfer Family SSH 文件传输协议 (SFTP) 的服务器，使用托管服务作为身份提供者，使用 Amazon S3 作为 AWS 存储服务。有关更多信息，请参阅 AWS Transfer Family 文档中的创建启用 SFTP 的服务器。

限制

- 这种方法不适用于近实时或实时的数据同步。
- 只能将数据从 Db2 z/OS 移动到 Amazon S3，反之则不然。

架构

源技术堆栈

- 在 z/OS 上运行 Db2 的大型机

目标技术堆栈

- AWS Transfer Family
- Amazon S3
- Amazon Athena
- Amazon QuickSight
- AWS Glue
- Amazon Relational Database Service (Amazon RDS)
- Amazon Aurora
- Amazon Redshift

源架构和目标架构

下图显示了生成、提取采用 ASCII CSV 格式的 Db2 z/OS 数据并将其传输到 S3 存储桶的过程。

1. 从 Db2 目录中选择用于数据迁移的表列表。
2. 该列表用于驱动具有外部格式的数字列和数据列的卸载作业的生成。
3. 然后，使用 AWS Transfer Family 将数据传输到 Amazon S3。
4. AWS Glue 提取、转换、加载（ETL）作业可以转换数据并将其加载到指定格式的已处理存储桶，或者 AWS Glue 可以将数据直接馈送到数据库中。
5. Amazon Athena 和 A QuickSight mazon 可用于查询和呈现数据以推动分析。

下图是整个过程的逻辑流程。

1. 第一个 JCL 名为 TABNAME，它将使用 Db2 实用程序 DSNTIAUL 来提取和生成您计划从 Db2 卸载的表的列表。要选择表，必须手动调整 SQL 输入以选择并添加筛选条件以包含一个或多个 Db2 架构。
2. 第二个 JCL 名为 REXXEXEC，它将使用提供的 JCL 骨架和 REXX 程序来处理由 JCL TABNAME 创建的表列表并为每个表名生成一个 JCL。每个 JCL 都将包含一个用于卸载表的步骤和另一个使用 SFTP 协议将文件发送到 S3 存储桶的步骤。
3. 最后一步包括运行 JCL 以卸载表，然后将文件传输到 AWS。整个过程都可以在本地或 AWS 上使用计划程序自动完成。

工具

Amazon Web Services

- Amazon Athena 是一种交互式查询服务，使您可使用标准 SQL 直接分析 Amazon Simple Storage Service（Amazon S3）中的数据。
- Amazon Aurora 是与 MySQL 和 PostgreSQL 兼容的完全托管式的云端关系数据库引擎。
- AWS Glue 是一项完全托管的提取、转换、加载（ETL）服务。它可以帮助您在数据存储和数据流之间对数据进行可靠地分类、清理、扩充和移动。
- Amazon QuickSight 是一项云规模的商业智能 (BI) 服务，可帮助您在单个控制面板中可视化、分析和报告数据。
- Amazon Redshift 是在 Amazon Web Services Cloud 上托管的 PB 级数据仓库服务。
- Amazon Relational Database Service (Amazon RDS) 可帮助您在 Amazon Web Services Cloud 中设置、操作和扩展关系数据库。
- Amazon Simple Storage Service(Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Transfer Family 是一种安全的传输服务，使您能够将文件传入和传出 AWS 存储服务。

大型机工具

- SSH 文件传输协议 (SFTP)是一种安全的文件传输协议，允许远程登录服务器并在服务器之间传输文件。SSH 通过加密所有流量来提供安全性。
- DSNTIAUL 是 IBM 提供的用于卸载数据的示例程序。
- DSNUTILB 是 IBM 提供的实用程序批处理程序，用于从 DSNTIAUL 中卸载具有不同选项的数据。

- z/OS OpenSSH 是在 IBM 操作系统 z/OS 下的 Unix 系统服务上运行的开源软件 SSH 端口。SSH 是在 TCP/IP 网络上运行的两台计算机之间的安全、加密的连接程序。它提供了多种实用程序，包括 ssh-keygen。
- REXX（重构扩展执行程序）脚本用于通过 Db2 卸载和 SFTP 步骤自动生成 JCL。

代码

此模式的代码可在 GitHub unloadd b2 存储库中找到。

最佳实践

对于第一次卸载，生成的 JCLs 应该卸载整个表数据。

第一次完全卸载后，执行增量卸载以提高性能和节省成本。更新模板 JCL deck 中的 SQL 查询以适应对卸载过程的任何更改。

您可以手动转换架构，也可以在 Lambda 上使用脚本将 Db2 SYSPUNCH 作为输入。对于工业流程，AWS Schema Conversion Tool（SCT）是首选。

最后，使用基于大型机的计划程序或在 AWS 上的计划程序（在大型机上装有代理）来帮助管理和自动化整个流程。

操作说明

设置 S3 存储桶

Task	描述	所需技能
创建 S3 存储桶。	有关说明，请参阅创建第一个 S3 存储桶。	常规 AWS

设置 Transfer Family 服务器

Task	描述	所需技能
创建启用 SFTP 的服务器。	要在 AWS Transfer Family 控制台上打开并创建 SFTP 服务器，请执行以下操作：	常规 AWS

Task	描述	所需技能
	1. 在“选择协议”页面上，选中 SFTP（SSH 文件传输协议）- 通过 Secure Shell 传输文件复选框。 2. 对于身份提供者，请选择服务托管。 3. 对于端点，选择可公开访问。 4. 对于域，选择 Amazon S3。 5. 在配置其他详细信息页面上，请保留默认设置。 6. 创建服务器。	
为 Transfer Family 创建 IAM 角色。	要创建 AWS Identity and Access Management（IAM）角色以供 Transfer Family 访问 Amazon S3，请按照创建 IAM 角色和策略中的说明操作。	AWS 管理员
添加 Amazon S3 服务托管用户。	要添加 Amazon S3 服务托管用户，请按照 AWS 文档中的说明进行操作，并使用大型机用户 ID。	常规 AWS

保护通信协议

Task	描述	所需技能
创建 SSH 密钥。	在大型机 USS 环境下，运行以下命令。 `ssh-keygen -t rsa`	大型机开发人员

Task	描述	所需技能
	Note 当提示输入密码时，请将其留空。	

为 SSH 文件夹和密钥文件提供正确的授权级别。	默认情况下，公有密钥和私有密钥将存储在用户目录 `/u/home/username/.ssh` 中。
	
	您必须为密钥文件授权 644，对文件夹授权 700。
	
	`chmod 644 .ssh/id_rsa`
	`chmod 700 .ssh`
		大型机开发人员

将公有密钥内容复制到 Amazon S3 服务托管用户。	要复制 USS 生成的公有密钥内容，请打开 AWS Transfer Family 控制台。 1. 在导航窗格中，选择服务器。 2. 选择服务器 ID 列中的标识符以查看服务器详细信息 3. 在用户下方，选择一个用户名以查看用户详细信息 4. 在 SSH 公有密钥下方，选择添加 SSH 公有密钥以向用户添加公有密钥。对于 SSH 公有密钥，请输入您的公有密钥。在添加新用户之前，您的密钥已通过服务验证。 5. 选择添加密钥。	大型机开发人员

生成 JCLs

Task	描述	所需技能
生成范围内 Db2 表列表。	提供输入 SQL 以创建限定数据迁移范围的表的列表。此步骤要求您使用 SQL where 子句指定查询 Db2 目录表 SYSIBM.SYSTABLES 的选择标准。可以对过滤器进行自定义，使其包含以特定前缀开头或基于增量卸载的时间戳开头的特定架构或表名。输出是在大型机上的物理序列 (PS) 数据集中捕获的。该数据集将作为 JCL 生成下一阶段的输入。 在使用 JCL TABNAME（如有必要，可以对其进行重命名）之前，请进行以下更改： 1. <Jobcard> 替换为作业类和有权运行 Db2 实用程序的用户。 2. 替换 < HLQ1 > 或自定义输出数据集名称以符合您的网站标准。 3. 根据您的站点标准更新 STEPLIB 堆栈 PDSEs（分区数据集已扩展）。此模式中的示例使用 IBM 默认值。 4. 将 PLAN 名称和 LIB 替换为特定于安装的值。 5. 将 <Schema> 和 <Prefix> 替换为 Db2 目录的选择标准。	大型机开发人员

Task	描述	所需技能

	6. 将生成的 JCL 保存到 PDS（分区数据集）库中。
	7. 提交 JCL。
	
	Db2 表列表提取作业
	
	`<Jobcard>`
	`//*`
	`//* UNLOAD ALL THE TABLE NAMES FOR A PARTICULAR SCHEMA`
	`//*`
	`//STEP01 EXEC PGM=IEFBR14`
	`//*`
	`//DD1 DD DISP=(MOD,DELETE,DELETE),`
	`// UNIT=SYSDA,`
	`// SPACE=(1000,(1,1)),`
	`// DSN=<HLQ1>.DSN81210.TABLIST`
	`//*`
	`//DD2 DD DISP=(MOD,DELETE,DELETE),`
	`// UNIT=SYSDA,`
	`// SPACE=(1000,(1,1)),`
	`// DSN=<HLQ1>.DSN81210.SYSPUNCH`
	`//*`
	`//UNLOAD EXEC PGM=IKJEFT01,DYNAMNBR=20`
	`//SYSTSPRT DD SYSOUT=*`
	`//STEPLIB DD DISP=SHR,DSN=DSNC10.DBCG.SDSNEXIT`
	`// DD DISP=SHR,DSN=DSNC10.SDSNLOAD`	

Task	描述	所需技能

	`//  DD  DISP=SHR,`
	`DSN=CEE.SCEERUN`
	`//  DD  DISP=SHR,`
	`DSN=DSNC10.DBCG.RU`
	`NLIB.LOAD`
	`//SYSTSIN  DD  *`
	`  DSN SYSTEM(DBCG)`
	`  RUN  PROGRAM(D`
	`SNTIAUL) PLAN(DSNT`
	`IB12) PARMS('SQL') -`
	`  LIB('DSNC`
	`10.DBCG.RUNLIB.LOAD')`
	`  END`
	`//SYSPRINT DD SYSOUT=*`
	`//*`
	`//SYSUDUMP DD SYSOUT=*`
	`//*`
	`//SYSREC00 DD DISP=(NEW`
	`,CATLG,DELETE),`
	`//  UNIT=SYSD`
	`A,SPACE=(32760,(10`
	`00,500)),`
	`//  DSN=<HLQ1`
	`>.DSN81210.TABLIST`
	`//*`
	`//SYSPUNCH DD DISP=(NEW`
	`,CATLG,DELETE),`
	`//  UNIT=SYSD`
	`A,SPACE=(32760,(10`
	`00,500)),`
	`//  VOL=SER=S`
	`CR03,RECFM=FB,LREC`
	`L=120,BLKSIZE=12`
	`//  DSN=<HLQ1`
	`>.DSN81210.SYSPUNCH`
	`//*`
	`//SYSIN  DD *`
	`  SELECT CHAR(CREA`
	`TOR), CHAR(NAME)`
	`  FROM SYSIBM.SY`
	`STABLES`	

Task	描述	所需技能
	WHERE OWNER = '<Schema>' AND NAME LIKE '<Prefix>%' AND TYPE = 'T'; /*	

Task	描述	所需技能
修改 JCL 模板。	此模式提供的 JCL 模板包含通用作业卡和库名称。但是，对于数据集名称、库名称和作业卡，大多数大型机站点都有自己的命名标准。例如，可能需要特定的作业类才能运行 Db2 作业。Job Entry Subsystem 实现 JES2 并 JES3 可以施加其他更改。标准负载库的第一个限定符可能与 IBM 默认值 SYS1 不同。因此，在运行模板之前，请根据贵站点特定标准对其进行自定义。 在骨架 JCL UNLDSKEL 中进行以下更改： 1. 将作业卡修改为作业类和有权运行 Db2 实用程序的用户。 2. 自定义输出数据集名称以符合贵站点标准。 3. PDSEs 根据您的网站标准更新 STEPLIB 堆栈。此模式中的示例使用 IBM 默认值。 4. 将 <DSN> 替换为 Db2 子系统名称和关联 ID。 5. 将生成的 JCL 保存在作为 ISPSLIB 堆栈一部分的 PDS 库中，该堆栈是 ISPF 的标准骨架模板库。 卸载和 SFTP JCL 骨架	大型机开发人员

Task	描述	所需技能
	`//&USRPFX.U JOB (DB2UNLOAD),'JOB', CLASS=A,MSGCLASS=A, // TIME=1440 ,NOTIFY=&USRPFX //* DELETE DATASETS //STEP01 EXEC PGM=IEFBR14 //DD01 DD DISP=(MOD ,DELETE,DELETE), // UNIT=SYSD A, // SPACE=(TR K,(1,1)), // DSN=&USRPFX..DB2.P UNCH.&JOBNAME //DD02 DD DISP=(MOD ,DELETE,DELETE), // UNIT=SYSD A, // SPACE=(TR K,(1,1)), // DSN=&USRPFX..DB2.U NLOAD.&JOBNAME //* //* RUNNING DB2 EXTRACTION BATCH JOB FOR AWS DEMO //* //UNLD01 EXEC PGM=DSNUTILB,REGIO N=0M, // PARM='<DSN>,UNLOAD' //STEPLIB DD DISP=SHR,DSN=DSNC1 0.DBCG.SDSNEXIT // DD DISP=SHR, DSN=DSNC10.SDSNLOAD //SYSPRINT DD SYSOUT=* //UTPRINT DD SYSOUT=* //SYSOUT DD SYSOUT=*`	

Task	描述	所需技能
	`//SYSPUN01 DD DISP=(NEW,CATLG,DELETE), // SPACE=(CYL,(1,1),RLSE), // DSN=&USRPFX..DB2.PUNCH.&JOBNAME //SYSREC01 DD DISP=(NEW,CATLG,DELETE), // SPACE=(CYL,(10,50),RLSE), // DSN=&USRPFX..DB2.UNLOAD.&JOBNAME //SYSPRINT DD SYSOUT=* //SYSIN DD * UNLOAD DELIMITED COLDEL ',' FROM TABLE &TABNAME UNLDDN SYSREC01 PUNCHDDN SYSPUN01 SHRLEVEL CHANGE ISOLATION UR; /* //* //* FTP TO AMAZON S3 BACKED FTP SERVER IF UNLOAD WAS SUCCESSFUL //* //SFTP EXEC PGM=BPXBATCH,COND=(4,LE),REGION=0M //STDPARM DD * SH cp "//'&USRPFX..DB2.UNLOAD.&JOBNAME'" &TABNAME..csv; echo "ascii " >> uplcmd; echo "PUT &TABNAME..csv " >>>> uplcmd;`	

Task	描述	所需技能

	sftp -b uplcmd -i .ssh/id_rsa &FTPUSER.@&FTPSITE;
	rm &TABNAME..csv;
	//SYSPRINT DD SYSOUT=*
	//STDOUT DD SYSOUT=*
	//STDENV DD *
	//STDERR DD SYSOUT=*	

Task	描述	所需技能
生成“批量卸载 JCL”。	此步骤涉及使用 JCL 在 ISPF 环境下运行 REXX 脚本。提供在第一步中创建的范围内表的列表，根据 TABLIST DD 名称进行批量生成 JCL 的输入。JCL 将在根据 ISPFILE DD 名称指定的用户指定的分区数据集中为每个表名生成一个新的 JCL。事先分配这个库。每个新 JCL 都将分为两个步骤：一个步骤将 Db2 表卸载到文件中，另一个步骤将文件发送到 S3 存储桶。 在 JCL REXXEXEC 中进行以下更改（您可以更改名称）： 1. 替换 Job card user ID 为在表格上具有卸载权限的大型机用户 ID。替换 SYSPROC、ISPPLIB、ISPSL 和 ISPTLIB <HLQ1> 值或自定义 DSN，以满足贵站点标准。要找出特定于安装的值，请使用命令 TSO ISRDDN。 2. 替换 <MFUSER> 为在安装中具有作业运行权限的用户 ID。 3. 替换 <FTPUSER> 为在安装中具有 USS 和 FTP 权限的用户 ID。假设此用户 ID 及其 SSH 安全密钥位于大型机上相应的 Unix 系统服务目录中。	大型机开发人员

Task	描述	所需技能

	4. 替换 <AWS TransferFamily IP> 为 AWS Transfer Family IP 地址或域名。此地址将用于 SFTP 步骤。
	5. 在应用站点标准调适并按下述方式更新 REXX 程序后提交 JCL。
	
	批量 JCL 生成作业
	
	`//RUNREXX JOB (CREATEJCL),'RUNS ISPF TABLIST', CLASS=A,MSGCLASS=A,`
	`//  TIME=1440 ,NOTIFY=&SYSUID`
	`//* Most of the values required can be updated to your site specific`
	`//* values using the command 'TSO ISRDDN' in your ISPF session.`
	`//* Update all the lines tagged with //update marker to desired`
	`//* site specific values.`
	`//ISPF EXEC PGM=IKJEFT01,REGION=2048K,DYNAMNBR=25`
	`//SYSPROC DD DISP=SHR,DSN=USER.Z23D.CLIST`
	`//SYSEXEC DD DISP=SHR,DSN=<HLQ1>.TEST.REXXLIB`	

Task	描述	所需技能

	`//ISPPLIB DD DISP=SHR,DSN=ISP.SISPPENU`
	`//ISPSLIB DD DISP=SHR,DSN=ISP.SISPSENU`
	`// DD DISP=SHR,DSN=<HLQ1>.TEST.ISPSLIB`
	`//ISPMLIB DD DSN=ISP.SISPMENU,DISP=SHR`
	`//ISPTLIB DD DDNAME=ISPTABL`
	`// DD DSN=ISP.SISPTENU,DISP=SHR`
	`//ISPTABL DD LIKE=ISP.SISPTENU,UNIT=VIO`
	`//ISPPROF DD LIKE=ISP.SISPTENU,UNIT=VIO`
	`//ISPLOG DD SYSOUT=*,RECFM=VA,LRECL=125`
	`//SYSPRINT DD SYSOUT=*`
	`//SYSTSPRT DD SYSOUT=*`
	`//SYSUDUMP DD SYSOUT=*`
	`//SYSDBOUT DD SYSOUT=*`
	`//SYSTSPRT DD SYSOUT=*`
	`//SYSUDUMP DD SYSOUT=*`
	`//SYSDBOUT DD SYSOUT=*`
	`//SYSHELP DD DSN=SYS1.HELP,DISP=SHR`
	`//SYSOUT DD SYSOUT=*`	

Task	描述	所需技能

	`//* Input list of tablenames`
	`//TABLIST DD DISP=SHR,DSN=<HLQ1>.DSN81210.TABLIST`
	`//* Output pds`
	`//ISPFILE DD DISP=SHR,DSN=<HLQ1>.TEST.JOBGEN`
	`//SYSTSIN DD *`
	`ISPSTART CMD(ZSTEPS <MFUSER> <FTPUSER> <AWS TransferFamily IP>)`
	`/*`
	
	使用 REXX 脚本之前，请执行以下更改：
	
	1. 将 REXX 脚本保存在上一步编辑的 JCL REXXEXEC 中的 SYSEXEC 堆栈下定义的 PDS 库中，成员名为 ZSTEPS。如果要对其进行重命名，则应更新 JCL 以满足需求。
	2. 此脚本使用跟踪选项打印其他信息，以防出现错误。相反，您可以在 EXECIO、ISPEXEC 和 TSO 语句之后添加错误处理代码，然后删除跟踪行。
	3. 此脚本使用 LODnnnnn 命名约定生成成员名称，命名约定最多可支持 10 万个成员。如果您有超过 10 万张表，请使用较短的前缀，然	

Task	描述	所需技能
	后调整 tempjob 语句中的数字。 ZSTEPS REXX 脚本 `/*REXX - - -

Task	描述	所需技能

	<pre>/* member will be
	saved in ISPDSN library
	allocated in JCL */
	ADDRESS ISPEXEC
	"FTCLOSE NAME("tem
	pjob")"
	END
	
	ADDRESS TSO "FREE
	F(TABLIST) "
	ADDRESS TSO "FREE
	F(ISPFILE) "
	
	exit 0</pre>	

运行 JCLs

Task	描述	所需技能
执行 Db2 卸载步骤。	生成 JCL 后，将有 JCLs 尽可能多的表需要卸载。 本情节使用 JCL 生成的示例来解释结构和最重要的步骤。 您无需执行任何操作。以下信息仅供参考。如果您打算提交在上 JCLs 一步中生成的内容，请跳至提交 LODnnnnn JCLs任务。 使用带有 IBM 提供的 DSNUTILB Db2 实用程序的 JCL 卸载 Db2 数据时，必须确保卸载的数据不包含压缩的数字数据。为此，请使用	大型机开发人员、系统工程师

Task	描述	所需技能
	DSNUTILB DELIMITED 参数。 该 DELIMITED 参数支持卸载 CSV 格式的数据，方法是为文本字段添加一个字符作为分隔符和双引号，删除 VARCHAR 列中的填充，并将所有数值字段（包括日期字段）转换为外部格式。 以下示例使用逗号字符作为分隔符，显示生成的 JCL 中的卸载步骤是什么样子。 `UNLOAD DELIMITED COLDEL ',' FROM TABLE SCHEMA_NAME.TBNAME UNLDDN SYSREC01 PUNCHDDN SYSPUN01 SHRLEVEL CHANGE ISOLATION UR;`	

Task	描述	所需技能
执行 SFTP 步骤。	要使用 JCL 中的 SFTP 协议，请使用 BPXBATCH 实用程序。 SFTP 实用程序无法直接访问 MVS 数据集。您可以使用复制命令（cp）将顺序文件 &USRPFX..DB2.UNLOAD.&JOBNAME 复制到 USS 目录中，它将在那里变成 &TABNAME..csv 。 使用私有密钥（id_rsa）并使用 RACF 用户 ID 作为用户名运行 sftp 命令，以连接到 AWS Transfer Family IP 地址。 `SH cp "//'&USRPFX..DB2.UNLOAD.&JOBNAME'" &TABNAME..csv; echo "ascii " >> uplcmd; echo "PUT &TABNAME..csv " >>>> uplcmd; sftp -b uplcmd -i .ssh/id_rsa &FTPUSER.@&FTP_TF_SITE; rm &TABNAME..csv;`	大型机开发人员、系统工程师

Task	描述	所需技能
提交 LODnnnnn JCLs.	之前的 JCL 已经生成了所有需要卸载、转换为 CSV 并传输到 S3 存储桶的 LODnnnnn JCL 表。 对所有已生成的内容运行 JCLs 该`submit`命令。	大型机开发人员、系统工程师

相关资源

有关本文档中使用的不同工具和解决方案的更多信息，请参阅以下内容：

- z/OS OpenSSH 用户指南
- Db2 z/OS – 卸载控制语句示例
- Db2 z/OS – 卸载带分隔符的文件
- Transfer Family – 创建一台启用 SFTP 的服务器
- Transfer Family – 与服务托管用户合作

其他信息

在 Amazon S3 上获得 Db2 数据后，您可以通过多种方式获得新的见解。由于 Amazon S3 与 AWS 数据分析服务集成，因此您可以在分布式端自由使用或公开这些数据。例如，您可以执行以下操作：

- 在 Amazon S3 上构建数据湖，无需移动数据 query-in-place，即可使用分析和机器学习工具提取宝贵的见解。
- 通过设置与 AWS Transfer Family 集成的上传后处理工作流程来启动 Lambda 函数。
- 使用 AWS Glue 开发新的微服务，以用于访问 Amazon S3 或完全托管数据库中的数据。AWS Glue 是一项无服务器数据集成服务，可轻松发现、准备和组合数据，以用于分析、机器学习和应用程序开发。

在迁移用例中，由于您可以将任何数据从大型机传输到 S3，因此您可以执行以下操作：

- 停用物理基础设施，使用 Amazon S3 Glacier 和 S3 Glacier Deep Archive 创建经济实惠的数据存档策略。

- 使用 Amazon S3 和其他 Amazon Web Services（例如 S3 Glacier 和 Amazon Elastic File System（Amazon EFS））构建可扩展、耐用、安全的备份和恢复解决方案，以增强或取代现有的本地功能。

更多模式

- 使用 Terraform 为 AWS WAF 解决方案部署安全自动化
- 使用 Precission Connect 将大型机数据库复制到 AWS

管理

主题

- 成本管理
- 高性能计算
- 混合云
- 管理与治理
- 消息和通信

成本管理

主题

- 使用 AWS Cost Explorer 成本管理服务为 AWS Glue 作业创建详细的成本和使用情况报告
- 使用 AWS Cost Explorer 成本管理服务为 Amazon EMR 集群创建详细的成本和使用情况报告
- 更多模式

使用 AWS Cost Explorer 成本管理服务为 AWS Glue 作业创建详细的成本和使用情况报告

由 Parijat Bhide (AWS) 和 Aromal Raj Jayarajan (AWS) 创建

摘要

此模式显示如何通过配置用户定义的成本分配标签来跟踪 AWS Glue 数据集成任务的使用成本。您可以使用这些标签在 AWS Cost Explorer 成本管理服务中为跨多个维度的任务创建详细的成本和使用情况报告。例如，您可以在团队、项目或成本中心级别跟踪使用成本。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 一个或多个激活了用户定义标签的 AWS Glue 任务

架构

目标技术堆栈

- AWS Glue
- AWS Cost Explorer 成本管理服务

下图显示了如何应用标签来跟踪 AWS Glue 作业的使用成本。

图表显示了以下工作流：

1. 数据工程师或 AWS 管理员为 AWS Glue 作业创建用户定义的成本分配标签。
2. AWS 管理员激活标签。
3. 标签将元数据报告给 AWS Cost Explorer 成本管理服务。

工具

- AWS Glue 是一项完全托管的提取、转换、加载（ETL）服务。它可以帮助您在数据存储和数据流之间对数据进行可靠地分类、清理、扩充和移动。
- AWS Cost Explorer 成本管理服务可帮助您查看和分析 AWS 成本和使用情况。

操作说明

为您的 AWS Glue 作业创建并激活标签

Task	描述	所需技能
为您的 AWS Glue 作业创建用户定义的成本分配标签。	将标签添加到现有 AWS Glue 作业 1. 登录到 AWS 管理控制台，然后打开 AWS Glue 控制台。 2. 在左侧导航窗格中，选择 ETL 下的作业。 3. 在您的作业部分中，选择要标记的作业的名称。 4. 选择 Job details（任务详细信息）选项卡。然后，展开高级属性部分。 5. 在标签下，选择添加新标签。 6. 对于键，输入标签名称。 7. （可选）对于值，输入要与键关联的值。 8. （可选）对要为作业创建的每个标签重复步骤 5-7。 9. 选择保存。	数据工程师

Task	描述	所需技能
	向新的 AWS Glue 作业添加标签 1. 根据您的使用案例要求创建新的 AWS Glue 作业。有关说明，请参阅 AWS Glue 开发人员指南中的在 AWS Glue 控制台上处理作业。 2. 在配置作业详细信息设置时，请执行此任务的向现有 AWS Glue 作业添加标签部分的步骤 4-9。 **Note** 有关更多信息，请参阅 AWS Glue 开发人员指南中的 AWS Glue 中的 AWS 标签。	
激活用户定义的成本分配标签。	按照 Amazon Web Services Billing 用户指南中的激活用户定义的成本分配标签中的说明操作。	AWS 管理员

为 AWS Glue 作业创建成本和使用报告

Task	描述	所需技能
使用 AWS Cost Explorer 成本管理服务中的标签筛选条件为您的 AWS Glue 作业创建成本和使用情况报告。	1. 登录 AWS 管理控制台 并打开 AWS 成本管理控制台。 2. 在左侧导航窗格中，选择报告。	常规 AWS、AWS 管理员

Task	描述	所需技能
	3. 选择创建新报告。 4. 对于选择报告类型，选择成本和使用情况（推荐）。然后，选择创建报告。 5. 对于筛选条件，选择服务。此时将显示服务下拉列表。 6. 选择 Glue 旁边的复选框。然后，选择应用筛选条件。 7. 对于筛选条件，请选择标签。此时将显示标签下拉列表。 8. 选择团队。然后，选中已为其分配标记的团队旁边的复选框。排除任何尚未为其分配标记的团队。然后，选择应用筛选条件。 9. 在图表顶部，选择标签。然后，选择要为其创建报告的 AWS Glue 作业的标签。 10.在图表顶部，选择过去 3 个月下拉列表，然后选择您希望报告涵盖的时间范围。然后，选择每月下拉列表，然后选择您希望如何根据时间范围聚合报告中的订单项。 11选择另存为。然后，输入报告的标题。 12选择保存报告。 有关更多信息，请参阅 AWS Cost Management 用户指南中	

Task	描述	所需技能
	的使用 Cost Explorer 浏览数据。	

使用 AWS Cost Explorer 成本管理服务为 Amazon EMR 集群创建详细的成本和使用情况报告

由 Parijat Bhide (AWS) 和 Aromal Raj Jayarajan (AWS) 创建

摘要

此模式显示如何通过配置用户定义的成本分配标签来跟踪 Amazon EMR 集群的使用成本。您可以使用这些标签在 AWS Cost Explorer 成本管理服务中为跨多个维度的集群创建详细的成本和使用情况报告。例如，您可以在团队、项目或成本中心级别跟踪使用成本。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 一个或多个已激活用户自定义标签的 EMR 集群

架构

目标技术堆栈

- Amazon EMR
- AWS Cost Explorer 成本管理服务

目标架构

下图显示了如何应用标签来跟踪特定 Amazon EMR 集群的使用成本。

图表显示了以下工作流：

1. 数据工程师或 AWS 管理员为 Amazon EMR 集群创建用户定义的成本分配标签。
2. AWS 管理员激活标签。
3. 标签将元数据报告给 AWS Cost Explorer 成本管理服务。

工具

工具

- Amazon EMR 是一个托管集群平台，可简化在 AWS 上运行大数据框架以处理和分析海量数据的操作。
- AWS Cost Explorer 成本管理服务可助您您查看和分析成本与使用情况。

操作说明

为您的 Amazon EMR 集群创建和激活标签

Task	描述	所需技能
为您的 Amazon EMR 集群创建用户定义的成本分配标签。	向现有 Amazon EMR 集群添加标签 按照 Amazon EMR 管理指南中的向现有集群添加标签中的说明操作。 向新的 Amazon EMR 集群添加标签 按照 Amazon EMR 管理指南中的向新集群添加标签中的说明操作。 有关如何设置 Amazon EMR 集群的更多信息，请参阅 Amazon EMR 管理指南中的计划和配置集群。	数据工程师
激活用户定义的成本分配标签。	按照 Amazon Web Services Billing 用户指南中的激活用户定义的成本分配标签中的说明操作。	AWS 管理员

为您的 Amazon EMR 集群创建成本和使用情况报告

Task	描述	所需技能
使用 AWS Cost Explorer 成本管理服务中的标签筛选条件为您的 Amazon EMR 集群创建成本和使用情况报告。	1. 登录 AWS 管理控制台并打开 AWS 成本管理控制台。 2. 在左侧导航窗格中，选择报告。 3. 选择创建新报告。 4. 对于选择报告类型，选择成本和使用情况（推荐）。然后，选择创建报告。 5. 对于筛选条件，选择服务。此时将显示服务下拉列表。 6. 选中 EMR（弹性 MapReduce）和 EC2-实例（弹性计算云 — 计算）旁边的复选框。然后，选择应用筛选条件。 7. 对于筛选条件，请选择标签。此时将显示标签下拉列表。 8. 选择团队。然后，选中已为其分配标记的团队旁边的复选框。排除任何尚未为其分配标记的团队。然后，选择应用筛选条件。 9. 在图表顶部，选择标签。然后，选择要为其创建报告的 Amazon EMR 集群的标签。 10. 在图表顶部，选择过去 3 个月下拉列表，然后选择您希望报告涵盖的时间范围。然后，选择每月下拉列表，然	常规 AWS、AWS 管理员

Task	描述	所需技能
	后选择您希望如何根据时间范围聚合报告中的订单项。 11选择另存为。然后，输入报告的标题。 12选择保存报告。 有关更多信息，请参阅 AWS Cost Management 用户指南中的使用 Cost Explorer 浏览数据。	

更多模式

- 使用 Amazon Bedrock 实现 AWS 基础设施运营自动化
- 自动清点多个账户和地区的 AWS 资源
- 使用 AWS 自动创建 AppStream 2.0 资源 CloudFormation
- 使用 DynamoDB TTL 自动将项目归档到 Amazon S3
- 使用 AWS Systems Manager 维护窗口自动停止和启动 Amazon RDS 数据库实例
- 为 Amazon RDS 和 Amazon Aurora 创建详细的成本和使用情况报告
- 使用和删除未使用的 Amazon EBS 卷 AWS ConfigAWS Systems Manager
- 估算 Amazon DynamoDB 表的存储成本
- 估算按需容量的 DynamoDB 表成本
- 使用 Amazon EKS Pod Identity 和 KEDA 在 Amazon EKS 中设置事件驱动的自动缩放
- 使用 AWS Fargate WaitCondition 挂钩构造协调资源依赖关系和任务执行

高性能计算

主题

- 使用 Terraform 和 DRA 部署 Lustre 文件系统以实现高性能数据处理
- 为 AWS 设置一个 Grafana 监控控制面板 ParallelCluster
- 使用 NICE EnginFrame 和 NICE DCV 会话管理器设置自动缩放虚拟桌面基础架构

使用 Terraform 和 DRA 部署 Lustre 文件系统以实现高性能数据处理

由 Arun Bagal (AWS) 和 Ishwar Chauthaiwale (AWS) 创作

摘要

这种模式会自动在上面部署 Lustre 文件系统，并将其与亚马逊弹性计算云 (亚马逊) AWS 和亚马逊简单存储服务 (Amazon S3 EC2) Simple Service 集成。

该解决方案可帮助您快速设置具有集成存储、计算资源和 Amazon S3 数据访问权限的高性能计算 (HPC) 环境。它将 Lustre 的存储功能与 Amazon 提供的灵活计算选项 EC2 以及 Amazon S3 中的可扩展对象存储相结合，因此您可以处理机器学习、HPC 和大数据分析中的数据密集型工作负载。

该模式使用 HashiCorp Terraform 模块和 Ama FSx zon for Lustre 来简化以下流程：

- 配置 Lustre 文件系统
- 在 Lustre 和 S3 存储桶之间 FSx 建立数据存储库关联 (DRA)，将 Lustre 文件系统与 Amazon S3 对象关联起来
- 创建实 EC2 例
- 在实例上安装带有与 Amazon S3 关联的 DRA 的 Lustre 文件系统 EC2

此解决方案的好处包括：

- 模块化设计。您可以轻松维护和更新此解决方案的各个组件。
- 可扩展性。您可以跨区域快速部署一致 AWS 账户 的环境。
- 灵活性。您可以自定义部署以满足您的特定需求。
- 最佳实践。此模式使用遵循 AWS 最佳实践的预配置模块。

有关 Lustre 文件系统的更多信息，请访问 Lustre 网站。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 最低权限 AWS Identity and Access Management (IAM) 策略（参见说明）

限制

FSx for Lustre 将 Lustre 文件系统限制为单个可用区，如果您有高可用性需求，这可能会是一个问题。如果包含文件系统的可用区出现故障，则对文件系统的访问将丢失，直到恢复。为了实现高可用性，您可以使用 DRA 将 Lustre 文件系统与 Amazon S3 链接，并在可用区域之间传输数据。

产品版本

- Terraform 版本 1.9.3 或更高版本
- HashiCorp AWS 提供商版本 4.0.0 或更高版本

架构

下图显示了 Lustre FSx 的架构，以及 AWS 服务 中的互补架构。 AWS Cloud

该架构包括以下内容：

- S3 存储桶可用作耐用、可扩展且经济实惠的数据存储位置。f FSx or Lustre 和 Amazon S3 之间的集成提供了一个与 Amazon S3 无缝关联的高性能文件系统。
- FSx for Lustre 运行和管理 Lustre 文件系统。
- Amazon Log CloudWatch s 从文件系统收集和监控日志数据。这些日志可让您深入了解 Lustre 文件系统的性能、运行状况和活动。
- 亚马逊 EC2 用于通过开源 Lustre 客户端访问 Lustre 文件系统。 EC2 实例可以从同一虚拟私有云 (VPC) 内的其他可用区访问文件系统。网络配置允许在 VPC 内跨子网进行访问。将 Lustre 文件系统挂载到实例上后，您可以像使用本地文件系统一样处理其文件和目录。
- AWS Key Management Service (AWS KMS) 通过为静态数据提供加密来增强文件系统的安全性。

自动化和扩缩

Terraform 可以更轻松地在多个环境中部署、管理和扩展 Lustre 文件系统。在 F FSx or Lustre 中，单个文件系统有大小限制，因此您可能需要通过创建多个文件系统来进行水平扩展。您可以使用 Terraform 根据您的工作负载需求配置多个 Lustre 文件系统。

工具

AWS 服务

- Amazon CloudWatch Lo gs 可帮助您集中所有系统和应用程序的日志， AWS 服务 这样您就可以监控它们并安全地将其存档。
- 亚马逊弹性计算云 (Amazon EC2) 在中提供可扩展的计算容量 AWS Cloud。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- Amazon FSx for Lustre 可以轻松且经济高效地启动、运行和扩展高性能 Lustre 文件系统。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥以帮助保护您的数据。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

代码存储库

此模式的代码可在使用 Terraform 存储库的 Lustre 文件系统 GitHub 配置 FSx 中找到。

最佳实践

- 以下变量定义了 Lustre 文件系统。请务必根据您的环境正确配置这些内容，如长篇故事部分所示。
  - `storage_capacity`— Lustre 文件系统的存储容量，在。 GiBs最低和默认设置为 1200 GiB。
  - `deployment_type`— Lustre 文件系统的部署类型。有关这两个选项`PERSISTENT_1`和`PERSISTENT_2`（默认）的说明，请参阅 for Lustre 文档。FSx
  - `per_unit_storage_throughput`— 读取和写入吞吐量，以 MBs 每秒 TiB 为单位。
  - `subnet_id`— 您要 FSx 为 Lustre 部署到的私有子网的 ID。
  - `vpc_id`— 您要在 Lustre 上部署 AWS 的虚拟私有云 FSx 的 ID。
  - `data_repository_path`— 将链接到 Lustre 文件系统的 S3 存储桶的路径。
  - `iam_instance_profile`— 用于启动实例的 IAM EC2 实例配置文件。
  - `kms_key_id`— 将用于数据加密的 AWS KMS 密钥的 Amazon 资源名称 (ARN)。
- 使用和`vpc_id`变量确保在 VPC 内正确访问`security_group`和放置网络。
- 按照 Epics 部分所述运行`terraform plan`命令以在应用更改之前预览和验证更改。这有助于 catch 发现潜在的问题，并确保你知道将要部署的内容。
- 使用 Epics 部分所述的`terraform validate`命令来检查语法错误并确认您的配置是否正确。

操作说明

设置您的环境

Task	描述	所需技能
安装 Terraform。	要在本地计算机上安装 Terraform，请按照 Ter raform 文档中的说明进行操作。	AWS DevOps， DevOps 工程师
设置 AWS 凭据。	要为账户设置 AWS Command Line Interface (AWS CLI) 配置文件，请按照AWS 文档中的说明进行操作。	AWS DevOps， DevOps 工程师
克隆 GitHub 存储库。	要克隆 GitHub 存储库，请运行以下命令： `git clone https://github.com/aws-samples/provision-fsx-lustre-with-terraform.git`	AWS DevOps， DevOps 工程师

为 Lustre FSx 进行配置和部署

Task	描述	所需技能
更新部署配置。	1. 在本地计算机上的克隆存储库中，导航到以下`fsx_deployment` 目录： `cd fsx_deployment` 2. 打开该`terraform.tfvars` 文件，然后更新以下变量的值：	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	• vpc_id • subnet_id • data_repository_path • iam_instance_profile • kms_key_id 有关这些变量的描述，请参阅 “最佳实践” 部分。 3. 在同一目录中，打开 locals.tf 文件并更新fsx_inress 和fsx_egress 安全组变量的 CIDR 范围。 4. 如果需要，请打开variables.tf 文件并更新以下变量的默认值： • storage_capacity • deployment_type • per_unit_storage_throughput 有关这些变量的描述，请参阅 “最佳实践” 部分。	
初始化 Terraform 环境。	要初始化您的环境以运行 Terraform fsx_deployment 模块，请运行： `terraform init`	AWS DevOps， DevOps 工程师

Task	描述	所需技能
验证 Terraform 语法。	要检查语法错误并确认您的配置是否正确，请运行： `terraform validate`	AWS DevOps， DevOps 工程师
验证 Terraform 配置。	要创建 Terraform 执行计划并预览部署，请运行： `terraform plan -var-file terraform.tfvars`	AWS DevOps， DevOps 工程师
部署 Terraform 模块。	要部署 for FSx Lustre 资源，请运行： `terraform apply -var-file terraform.tfvars`	AWS DevOps， DevOps 工程师

清理 AWS 资源

Task	描述	所需技能
移除 AWS 资源。	使用 for Lustre 环境后，您可以移除 Terraform 部署的 AWS 资源，以避免产生不必要的费用。 FSx 代码存储库中提供的 Terraform 模块可自动执行此清理工作。 1. 在您的本地存储库中，导航到该`fsx_deployment` 目录： `cd fsx_deployment` 2. 运行命令：	AWS DevOps， DevOps 工程师

Task	描述	所需技能
	`terraform destroy -var-file terraform.tfvars`	

故障排除

事务	解决方案
FSx for Lustre 会返回错误。	要获得有关 Lustre 问题的帮助，请参阅 for Lustre 文档中的 Amazon f FSx or Lustre 故障排除。 FSx FSx

相关资源

- 使用 Terraform（Terraform 文档中的AWS 提供商参考） FSx 为 Lustre 构建亚马逊
- 开始使用 Amazon f FSx or Lustre（查看 FSx Lustre 文档）
- AWS 关于 Amazon for Lustre FSx 的博客文章

为 AWS 设置一个 Grafana 监控控制面板 ParallelCluster

由 Dario La Porta (AWS) 和 William Lu (AWS) 编写

摘要

AWS ParallelCluster 可帮助您部署和管理高性能计算 (HPC) 集群。支持 AWS Batch 和 Slurm 开源作业计划程序。尽管 ParallelCluster AWS 与 Amazon CloudWatch 集成了日志和指标，但它没有为工作负载提供监控控制面板。

适用于 AWS 的 Grafana 控制面板 GitHub () 是 ParallelCluster AWS 的监控控制面板。 ParallelCluster 它提供了操作系统级别的作业调度程序见解和详细的监控指标。有关此解决方案中包含的仪表板的更多信息，请参阅 GitHub 存储库中的示例仪表板。这些指标可帮助您更好地了解 HPC 工作负载及性能。但是，控制面板代码不会针对最新版本的 AWS ParallelCluster 或解决方案中使用的开源软件包进行更新。此模式增强解决方案，提供以下优势：

- 支持 AWS ParallelCluster v3
- 使用最新版开源包，包括 Prometheus、Grafana、Prometheus Slurm Exporter 和 NVIDIA DCGM-Exporter
- 增加 Slurm 作业使用 GPUs 的 CPU 内核数量
- 添加任务监控控制面板
- 增强具有 4 或 8 个图形处理单元的节点的 GPU 节点监控仪表板 (GPUs)

此版本的增强型解决方案已在 AWS 客户的 HPC 生产环境中实施和验证。

先决条件和限制

先决条件

- AWS ParallelCluster CLI，已安装并配置。
- AWS 支持的网络配置 ParallelCluster。此模式使用使用 ParallelCluster 使用两个子网的 AWS 配置，这需要公有子网、私有子网、Internet 网关和 NAT 网关。
- 所有 AWS ParallelCluster 集群节点都必须能够访问互联网。这是必要条件，这样安装脚本才能下载开源软件和 Docker 映像。
- 亚马逊弹性计算云（亚马逊 EC2）中的密钥对。具有此密钥对的资源具有对头节点的 Secure Shell (SSH) 访问权限。

限制

- 此示例旨在支持 Ubuntu 20.04 LTS。如果您使用的是其他版本的 Ubuntu，或者您使用的是 Amazon Linux 或 CentOS，则需要修改此解决方案提供的脚本。这些修改不包含在此模式中。

产品版本

- Ubuntu 20.04 LTS
- ParallelCluster 3.X

账单与成本注意事项

- 以这种模式部署的解决方案并不在免费套餐范围内。亚马逊 EC2、亚马逊 Lustre、亚马逊 VPC 中的 NAT 网关和亚马逊 Route 53 均 FSx 需收费。

架构

目标架构

下图显示了用户如何在头节点 ParallelCluster 上访问 AWS 的监控控制面板。头节点运行 NICE DCV、Prometheus、Grafana、Prometheus Slurm Exporter、Prometheus Node Exporter 以及 NGINX Open Source。计算节点运行 Prometheus Node Exporter，如果节点包含，它们还会运行 NVIDIA dcgm-Exporter。 GPUs头节点从计算节点检索信息，并将此数据显示在 Grafana 控制面板中。

在大多数情况下，头节点的负载并不重，因为作业调度程序不需要大量的 CPU 或内存。用户通过端口 443 上的 SSL 访问头节点上的控制面板。

所有授权查看者都可以匿名查看监控控制面板。仅 Grafana 管理员可以修改控制面板。您可在 `aws-parallelcluster-monitoring/docker-compose/docker-compose.head.yml` 文件中为 Grafana 管理员配置密码。

工具

Amazon Web Services

- NICE DCV 是一种高性能远程显示协议，可帮助您在不同的网络条件下将远程桌面和应用程序流从任何云或数据中心传送到任何设备。

- AWS ParallelCluster 可帮助您部署和管理高性能计算 (HPC) 集群。支持 AWS Batch 和 Slurm 开源作业计划程序。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。

其他工具

- Docker 是一组平台即服务（PaaS）产品，它们使用操作系统级别的虚拟化技术在容器中交付软件。
- Grafana 是一款开源软件，可帮助您查询、可视化、提醒和浏览指标、日志和跟踪。
- NGINX Open Source 是一个开源 Web 服务器和反向代理。
- NVIDIA 数据中心 GPU 管理器 (DCGM) 是一套工具，用于在集群环境中管理和监控 NVIDIA 数据中心图形处理单元 (GPUs)。在这种模式中，您使用 dcgm-Exporter，它可以帮助您从 Prometheus 中导出 GPU 指标。
- Prometheus 是开源系统监控工具包，可将其指标收集并存储为时间序列数据，以及相关的键值对（称为标签）。在此模式下，您还可使用 Prometheus Slurm Exporter 收集和导出指标，您可使用 Prometheus Node Exporter 导出来自结算节点的指标。
- Ubuntu 是基于 Linux 的开源操作系统，专为企业服务器、桌面、云环境和物联网而设计。

代码存储库

此模式的代码可在 GitHub pcluster-monitoring-dashboard存储库中找到。

操作说明

创建所需资源

Task	描述	所需技能
创建 S3 存储桶。	创建 Amazon S3 存储桶。您可使用此存储桶存储配置脚本。有关说明，请参阅 Amazon S3 文档中的创建存储桶。	常规 AWS

Task	描述	所需技能
克隆存储库。	通过运行以下命令克隆 GitHub pcluster-monitoring-dashboard存储库。 `git clone https://github.com/aws-samples/parallelcluster-monitoring-dashboard.git`	DevOps 工程师
创建管理员密码。	1. 选择文件夹，选择 `aws-parallelcluster-monitoring` 文件夹，选择 `docker-compose` 文件夹，然后打开 docker-compose.head.yml 文件。 2. 在 `GF_SECURITY_ADMIN_PASSWORD` 变量中，使用所选密码替换 `Grafana4PC!` 。这是您用于管理 Grafana 账户的管理密码。 3. 保存和关闭 docker-compose.head.yml 文件。	Linux Shell 脚本
将所需文件复制至 S3 存储桶。	将 post_install.sh 脚本和aws-parallelcluster-monitoring文件夹复制到您创建的 S3 存储桶中。有关说明，请参阅 Amazon S3 文档中的上传对象。	常规 AWS

Task	描述	所需技能
为头节点配置其他安全组。	1. 为头节点创建安全组。该安全组将允许入站流量到达头节点上的监控控制面板。有关说明，请参阅 Amazon VPC 文档中的 创建安全组。 2. 将入站规则添加到安全组。有关说明，请参阅 Amazon VPC 文档中的 向安全组添加规则。对规则使用以下参数： • 类型 – HTTPS • 协议 – TCP • 端口范围 –443 • 源 - 输入您的 IP 地址 • 描述 - 允许用户访问监控面板	AWS 管理员
为头节点配置 IAM policy。	为头节点创建基于身份的策略。该策略允许节点从 Amazon 检索指标数据 CloudWatch。该 GitHub 存储库包含一个示例策略。有关说明，请参阅 AWS Identity and Access Management (IAM) 文档中的创建 IAM policy。	AWS 管理员

Task	描述	所需技能
为计算机节点配置 IAM policy。	为计算机节点创建基于身份的策略。此策略允许节点创建包含作业 ID 和任务拥有者的标签。该 GitHub 存储库包含一个示例策略。有关说明，请参阅 IAM 文档中的创建 IAM policy。 如您使用提供的示例文件，请替换以下值： • <REGION> — 托管集群的 AWS 区域 • <ACCOUNT_ID> — Amazon Web Services account ID	AWS 管理员

创建集群

Task	描述	所需技能
修改所提供的集群模板文件。	创建 AWS ParallelCluster 集群。使用提供的 cluster.yaml A CloudFormation WS 模板文件作为创建集群的起点。替换所提供模板中的以下值： • <REGION> — 托管集群的 AWS 区域。 • <HEADNODE_SUBNET> — VPC 的公有子网。 • <ADDITIONAL_HEAD_NODE_SG> — 您为头节点创建的安全组的名称。	AWS 管理员

Task	描述	所需技能
	• <KEY_NAME>— 输入现有 Amazon EC2 密钥对的名称。具有此密钥对的资源具有对头节点的 Secure Shell (SSH) 访问权限。 • <ALLOWED_IPS> — 输入允许与头节点建立 SSH 连接的 CIDR 格式的 IP 地址范围。 • <ADDITIONAL_HEAD_NODE_POLICY> — 输入您为头节点创建的 IAM policy 的名称。 • <BUCKET_NAME> — 输入您创建的 S3 存储桶的名称。 • <COMPUTE_SUBNET> — 输入 VPC 中私有子网的名称。 • <ADDITIONAL_COMPUTE_NODE_POLICY> — 输入您为计算节点创建的 IAM policy 的名称。	

Task	描述	所需技能
创建集群。	在 AWS ParallelCluster CLI 中，输入以下命令。这将部署 CloudFormation 模板并创建集群。有关此命令的更多信息，请参阅 AWS 文档中的 pcluster create-cluster。 ParallelCluster `pcluster create-cluster -n <cluster_name> -c cluster.yaml`	AWS 管理员
监控集群创建。	输入以下命令，以监控集群创建。有关此命令的更多信息，请参阅 AWS 文档中的 pcluster describe- cluster。 ParallelCluster `pcluster describe-cluster -n <cluster_name>`	AWS 管理员

使用 Grafana 控制面板

Task	描述	所需技能
访问 Grafana 门户。	1. 输入以下命令以检索头节点的公共 IP 地址。 `pcluster describe-cluster -n <cluster_name> --query headNode.publicIpAddress`	AWS 管理员

Task	描述	所需技能
	2. 在 Web 浏览器中，导航到以下 URL 以访问 Grafana 控制面板。 https://<head_node_public_ip_address> 3. 在 Grafana 首页上，选择左侧菜单上的 4 方形控制面板图标，然后选择常规。这显示了已配置的控制面板列表。Grafana 中提供以下控制面板： • 集群成本 — 包含有关集群成本的信息 • 集群日志 — 包含有关集群日志的信息 • 计算节点详细信息 — 包含有关计算节点使用情况统计信息的信息 • 计算节点列表 — 包含集群的计算节点列表 • GPU 节点 — 包含有关 GPU 节点使用情况统计信息的信息 • 作业详情 — 包含有关作业资源利用率的信息 • 头节点详细信息 — 包含有关头节点使用情况统计信息的信息 • ParallelCluster 摘要-包含有关集群使用情况的信息	

清理解决方案，以停止产生相关成本

Task	描述	所需技能
请删除集群。	输入以下命令以删除集群。有关此命令的更多信息，请参阅 AWS 文档中的 pcluster delete-cluster。 ParallelCluster `pcluster delete-cluster -n <cluster_name>`	AWS 管理员
删除 IAM policy。	删除您为头节点与计算节点创建的策略。有关删除策略的更多信息，请参阅 IAM 文档中的删除 IAM policy。	AWS 管理员
删除安全组和规则。	删除您为头节点创建的安全组。有关更多信息，请参阅 Amazon VPC 文档中的删除安全组规则和删除安全组。	AWS 管理员
删除 S3 存储桶。	删除您创建的用于存储配置脚本的 S3 存储桶。有关更多信息，请参阅 Amazon S3 文档中的删除存储桶。	常规 AWS

故障排除

事务	解决方案
头节点在浏览器中不可访问。	检查安全组并确认入站端口 443 已经打开。
无法打开 Grafana。	在头节点上，查看 `docker logs Grafana` 的容器日志。
部分指标没有数据。	在头节点，检查所有容器的容器日志。

相关资源

AWS 文档

- 适用于亚马逊的 IAM 政策 EC2

其他 AWS 资源

- AWS ParallelCluster
- AWS 监控控制面板 ParallelCluster（AWS 博客文章）

其他资源

- Prometheus 监控系统
- Grafana

使用 NICE EnginFrame 和 NICE DCV 会话管理器设置自动缩放虚拟桌面基础架构

由 Dario La Porta (AWS) 和 Salvatore Maccarone (AWS) 创作

摘要

NICE DCV 是一种高性能远程显示协议，可帮助您在不同的网络条件下将远程桌面和应用程序从任何云或数据中心流式传输到任何设备。借助 NICE DCV 和亚马逊弹性计算云 (Amazon EC2)，您可以在亚马逊 EC2 实例上远程运行图形密集型应用程序，并将其用户界面流式传输到更简单的远程客户端计算机上。这消除了对昂贵的专用工作站的需要，以及在云和客户端计算机之间传输大量数据的需要。

此模式建立了一个功能齐全、自动扩缩的 Linux 和 Windows 虚拟桌面基础架构（VDI），可通过基于 Web 的用户界面进行访问。VDI 解决方案为研发 (R&D) 用户提供易于访问且高性能的用户界面，用于提交图形密集型分析请求并远程审查结果。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- 管理员权限和一组访问密钥。
- AWS Cloud Development Kit (AWS CDK) 工具包，已安装和配置。有关更多信息，请参阅安装 AWS CDK。
- AWS Command Line Interface (AWS CLI)，已为您安装和配置 AWS 账户。有关更多信息，请参阅安装或更新 AWS CLI的最新版本。
- Python，已安装并配置。有关更多信息，请参见源版本（Python 网站）。
- 一个或多个虚拟私有云 (VPCs) 可用。
- 两个或多个 Elastic IP 地址可用。有关默认限制的更多信息，请参阅 Elastic IP 地址限制。
- 对于 Linux Amazon EC2 实例，请设置安全外壳 (SSH) 密钥对。有关更多信息，请参阅密钥对和 Linux 实例。

产品版本

- AWS CDK 版本 2.26.0 或更高版本
- Python 版本 3.8 或更高版本

架构

目标架构

下图显示了该 VDI 解决方案的不同组件。根据适用于 Windows 和 Linux NICE EnginFrame DCV EC2 实例的 Amazon A EC2 uto Scaling 群组，用户与 NICE 交互启动亚马逊实例。

自动化和扩缩

此模式中包含的代码可创建自定义 VPC、公有子网和私有子网、互联网网关、NAT 网关、Application Load Balancer、安全组和 AWS Identity and Access Management (IAM) 策略。 AWS CloudFormation 还用于创建 Linux 和 Windows NICE DCV 服务器群。

工具

AWS 服务

- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，可帮助您在代码中定义和配置 AWS Cloud 基础架构。
- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在资源的整个生命周期中跨地区对其 AWS 账户 进行管理。
- NICE DCV 是一种高性能远程显示协议，可帮助您在不同的网络条件下将远程桌面和应用程序流从任何云或数据中心传送到任何设备。在这种模式下，它提供了一种带宽高效的体验，可远程传输高性能计算（HPC）3D 图形。
- NICE DCV Session Manager 可帮助您在一组 NICE DCV 服务器实例集上创建和管理 NICE DCV 会话的生命周期。
- NICE EnginFrame 是一个高级的前端 Web 界面，用于访问云端的技术和科学应用程序。

代码存储库

此模式的代码可在带有 NICE EnginFrame 和 NICE DCV 会话管理器存储库的 Auto Scaling VDI 解决方案中找到。

操作说明

部署虚拟桌面基础设施

Task	描述	所需技能
克隆存储库。	克隆包含代码的存储库： `git clone https://github.com/aws-samples/elastic-vdi-infrastructure.git`	云架构师

安装所需的 AWS CDK 库。	安装 AWS CDK 库：
	
	`cd elastic-vdi-infrastructure`
	`python3 -m venv .venv`
	`source .venv/bin/activate`
	`pip3 install -r requirements.txt`
		云架构师

更新参数。	1. 在选定的文本编辑器中打开 app.py 文件。 2. 替换以下必需参数的 `CHANGE_ME` 值： • `region`— 目标 AWS 区域。有关完整列表，请参阅AWS 区域。 • `account`— 目标的 ID AWS 账户。有关更多信息，请参阅查找您的 AWS 账户 身份证。 • `key_name`— 用于访问 Linux Amazon EC2 实例的密钥对。	云架构师

Task	描述	所需技能
	3. （可选）修改以下参数的值，为您的环境自定义解决方案： • `ec2_type_enginframe` — EnginFrame 实例类型 • `ec2_type_broker` — Session Manager Broker 实例类型 • `ebs_enginframe_size` — 实例的亚马逊 Elastic Block Store (Amazon EBS) 卷的大小 EnginFrame • `ebs_broker_size` — 会话管理器代理实例的 Amazon EBS 卷的大小 • `TagName and TagValue`— 资源的账单标签 • `efadmin_uid` — EnginFrame 管理员 (efadmin) 用户的唯一标识符 • `linux_shared_storage_size` — OpenZFS 大小（以字节为单位） • `Shared_Storage_Linux` — 共享存储的挂载点 • `Enginframe_installer` — 的下载链接 EnginFrame	

Task	描述	所需技能
	• `Session_Manager_Broker_Installer` — Session Manager Broker 的下载链接 4. 保存并关闭 app.py 文件。	

部署解决方案。	按顺序运行以下命令：
	`cdk bootstrap`
	`cdk deploy Assets-Stack Parameters-Stack`
	`cdk deploy Elastic-Vdi-Infrastructure`
	部署完成后，会返回以下两个输出：
	• `Elastic-Vdi-Infrastructure.EnginFrameURL` — EnginFrame 门户网站的 HTTPS 地址
	• `Elastic-Vdi-InfrastruSecretEFadminPassword` — 包含 efadmin 用户密码的密钥的 Amazon 资源名称（ARN）
	记下这些值。您稍后将在此模式中使用这些值。
		云架构师

Task	描述	所需技能
部署 Linux 服务器实例集。	1. 登录并 AWS Management Console打开CloudFormation 控制台。 2. 选择创建堆栈，然后选择使用新资源。 3. 在 cloudformation_files 文件夹中，选择.yaml 文件。dcv-linux-fleet 4. 在指定堆栈详细信息页面上，定义以下参数： • 堆栈名称 — 堆栈的名称。 • DcvFleet— NICE DCV 车队的名称。请勿将此值留空或使用空格。 • InstanceType— 队列的实例类型。 • RootVolumeSize— Linux Amazon EC2 实例的根卷大小。 • MinSize— 应可用且未运行任何 DCV 会话的最小节点数。例如，如果您输入 2，则解决方案将从 2 个节点开始。当用户创建会话时，可用节点的数量减少到 1，解决方案会创建另一个节点以保持最小值。 • MaxSize— 队列中节点的最大数量。如果已达到最大会话限制，则用户无法开始新会话。	云架构师

Task	描述	所需技能
	• BillingTagName— 用于计费的标签名称。此标签名称必须与用于 Windows 堆栈的名称存在不同。 • BillingTagValue— 用于计费的标签值。 5. 完成堆栈创建向导，然后选择提交开始创建堆栈。	

Task	描述	所需技能
部署 Windows 服务器实例集。	1. 登录并 AWS Management Console打开CloudFormation 控制台。 2. 选择创建堆栈，然后选择使用新资源。 3. 在 cloudformation_files 文件夹中，选择.yaml 文件。dcv-windows-fleet 4. 在指定堆栈详细信息页面上，定义以下参数： • 堆栈名称 — 堆栈的名称。 • DcvFleet— NICE DCV 车队的名称。请勿将此值留空或使用空格。 • InstanceType— 队列的实例类型。 • RootVolumeSize— Windows Amazon EC2 实例的根卷大小。 • MinSize— 应可用且未运行任何 DCV 会话的最小节点数。 • MaxSize— 队列中节点的最大数量。 • BillingTagName— 用于计费的标签名称。此标签名称必须与用于 Linux 堆栈的名称存在不同。 • BillingTagValue— 用于计费的标签值。	云架构师

Task	描述	所需技能
	5. 完成堆栈创建向导，然后选择提交开始创建堆栈。	

访问已部署的环境

Task	描述	所需技能
检索 EnginFrame 管理员密码。	EnginFrame 管理帐户名为 efadmin，密码 AWS Secrets Manager 作为密钥存储在中。密钥的 ARN 是动态生成的，并且在部署的输出中可见。 AWS CDK 1. 在上一篇操作说明中，在部署解决方案说明中，在 `Elastic-Vdi-Infrastructure.SecretEFadminPassword` 输出下方，找到生成的密钥的 ARN。 2. 执行以下操作之一，找回密钥： • 使用 Secrets Manager 控制台。有关更多信息，请参阅找回密钥。 • 输入 get-secret-value 命令。 `aws secretsmanager get-secret-value \ --secret-id <secret_arn> \ --query SecretString \`	云架构师

Task	描述	所需技能
	`--output text`	
访问门 EnginFrame 户。	1. 在上一篇长篇故事中，在“部署解决方案” 故事中，在`Elastic-Vdi-Infrastructure.EnginFrameURL` 输出下方，找到 EnginFrame 门户的 HTTPS 地址。 2. 在 Web 浏览器中，键入门户的 HTTPS 地址。 3. 输入 efadmin 用户的凭证。	云架构师
启动 Windows 会话。	1. 在 EnginFrame 门户网站的菜单中，选择 Windows 桌面。 2. 当系统提示您以 Windows 管理员身份登录时，请输入与 efadmin 用户相同的密码。 3. 确认 Windows 会话成功启动。	云架构师
启动 Linux 会话。	1. 在 EnginFrame 门户网站的菜单中，选择 Linux 桌面。 2. 当系统提示您登录时，输入 efadmin 用户的凭证。 3. 确认 Linux 会话成功启动。	云架构师

清理

Task	描述	所需技能
删除堆栈。	在 CloudFormation 控制台中，删除 Windows 和 Linux 服务器队列的堆栈。有关更多信息，请参阅删除堆栈。	云架构师
删除基础设施。	使用以下 AWS CDK 命令删除已部署的基础架构： `cdk destroy --all`	云架构师

故障排除

事务	解决方案
部署未完成，因为它已经中断。	按照清理说明中的说明进行操作，然后重复此模式以再次部署环境。

相关资源

- NICE DCV
- 不错 EnginFrame

混合云

主题

- 使用混合链接模式将数据中心扩展配置为 VMware Cloud on AWS
- 将 VMware vRealize Automation 配置为在 A VMs W VMware S 上的云端上
- 使用 AWS 上的 VMware C VMware loud 在 AWS 上部署 SDDC
- 在 AWS 上将 VMware vRealize 网络洞察与 VMware 云集成
- 使用 HCX 操作系统辅助迁移在 AWS 上迁移 VMs 到 VMware 云端
- 使用 VMware Aria 日志操作 AWS 将日志从 VMware Cloud 发送到 Splunk
- 使用 AWS CDK 在 Amazon ECS Anywhere 上为混合工作负载设置 CI/CD 管道和 GitLab
- 更多模式

使用混合链接模式将数据中心扩展配置为 VMware Cloud on AWS

由 Deepak Kumar (AWS) 编写

摘要

注意：自 2024 年 4 月 30 日起，VMware Cloud on AWS 不再由其渠道 AWS 合作伙伴转售。该服务将继续通过博通提供。我们鼓励您联系您的 AWS 代表了解详情。

此模式描述了如何使用混合链接模式通过单个 vSpher VMware e Client 界面查看和管理本地数据中心和 VMware Cloud on AWS 软件定义数据中心 (SDDC) 中的库存。

通过配置混合链接模式，您可以将本地虚拟机 (VMs) 和应用程序迁移到云端 SDDC。然后，您的 IT 团队可以使用熟悉的 VMware 工具管理基于云的资源，而无需任何新工具。您还可以使用 VMware Cloud Gateway 设备确保一致的操作和简化的管理。

此模式提供了两个用于配置混合链接模式的选项，但您一次只能使用一个选项。第一个选项安装云网关设备并使用它从本地 vCenter Server 链接到云 SDDC。第二个选项通过云 SDDC 配置混合链接模式。

先决条件和限制

先决条件（两个选项）

- 现有本地数据中心和云 SDDC。
- 本地数据中心和云 SDDC 之间的现有连接，使用 AWS Direct Connect、VPN 或两者兼而有之。
- 本地数据中心和云 SDDC 与网络时间协议 (NTP) 或其他权威时间源同步。
- 本地数据中心和云 SDDC 之间往返的最大时延不超过 100 毫秒。
- 可以访问本地环境的云管理员。
- vCenter Server 的完全限定域名（FQDN）必须解析为私有 IP 地址。

选项 1 的先决条件

- 本地环境应在 vSphere 6.5.0d 或更高版本上运行。
- 云网关设备和 vCenter Server 可以通过 AWS Direct Connect、VPN 或两者进行通信。

- 云网关设备符合硬件要求。
- 防火墙端口已经打开。

选项 2 的先决条件

- 本地 vCenter Server 在 vSphere 6.0 Update 3 或更高版本、或者 vSphere 6.5.0d 或更高版本上运行。
- 本地 vSphere 单点登录（SSO）域可以使用登录凭证。
- 本地环境中的用户对基本专有名称 (Base DN) 具有只读访问权限。
- 本地域名系统 (DNS) 服务器已配置为 VMware 管理网关。
- 使用连接验证器进行网络 VMware 连接测试。
- 防火墙端口已经打开。

限制

- 混合链接模式只能连接一个本地 vCenter Sever 增强型链接模式域。
- 混合链接模式仅支持运行版本 6.7 或更高版本的本地 vCenter Server。

架构

下图显示了配置混合链接模式的两个选项。

使用混合链接模式迁移不同工作负载类型

混合链接模式支持使用冷迁移或通过 VMware vSpher e vMotion 进行实时迁移，在本地数据中心和云 SDDC 之间迁移工作负载。选择迁移方法时，必须考虑的因素包括虚拟交换机类型和版本、与云 SDDC 的连接类型以及虚拟硬件版本。

冷迁移适用于 VMs 这种停机时间。您可以关闭 VMs，迁移它们，然后将其重新打开。迁移时间更快，因为不需要复制活动内存。我们建议对接受停机的应用程序（例如，第 3 层应用程序或开发和测试工作负载）使用冷迁移。如果您 VMs 无法遇到停机时间，则应考虑使用vMotion对任务关键型应用程序进行实时迁移。

下图概述了使用混合链接模式的不同工作负载迁移类型。

工具

- VMware Cloud on AWS 是一款由 AWS 和 AWS 联合开发的集成云产品 VMware。
- VMware Cloud Gatew ay Appliance 支持许多将本地资源连接到云资源的混合云用例。
- VMware vSphere VMware 是虚拟化平台，可将数据中心转变为包括 CPU、存储和网络资源的聚合计算基础架构。

操作说明

选项 1 - 在云网关设备上使用混合链接模式

Task	描述	所需技能
配置云网关设备。	1. 在 AWS 控制台上登录 VMware 云并下载云网关设备。 2. 通过以下两个步骤，在本地环境中安装云网关设备： • 选择开始进行配置，然后部署云网关设备。 • 配置混合链接模式。 有关更多信息和详细步骤，请参阅文档中的使用 vCenter Cloud Gateway 设备配置混合链接模式。 VMware	云管理员

选项 2 - 使用云端 SDDC 中的混合链接模式

Task	描述	所需技能
从云 SDDC 配置混合链接模式。	1. 登录 AWS 上的 VMware Cloud 控制台，使用连接验	云管理员

Task	描述	所需技能
	证器检查所有必需的网络连接。有关这方面的更多信息，请参阅 VMware 文档中的验证混合链接模式的网络连接。 2. 登录云 SDDC 的 vSphere 客户端，选择菜单，选择管理，然后选择域。 3. 在混合云部分中，选择链接域，然后连接至您的本地 vCenter Server。 4. 将身份源添加到云 SDDC 轻量级目录访问协议 (LDAP) 域。有关这方面的更多信息，请参阅 VMware 文档中的向 SDDC LDAP 域添加身份源。	

相关资源

- 配置混合链接模式
- 在 AWS 上为 VMware 云配置混合链接模式

将 VMware vRealize Automation 配置为在 A VMs W VMware S 上的云端上

由 Deepak Kumar (AWS) 编写

摘要

注意：自 2024 年 4 月 30 日起， VMware Cloud on AWS 不再由其渠道 AWS 合作伙伴转售。该服务将继续通过博通提供。我们鼓励您联系您的 AWS 代表了解详情。

VMware vReal ize Automation 是一款自动化软件，可用于请求和管理 IT 资源。通过选择在 AWS 上使用 VMware Cloud 配置 vRealize Automation，您可以跨多个数据中心和云环境自动交付虚拟机 (VMs)、应用程序和 IT 服务。

IT 团队可以创建目录项来配置服务预调配和运营功能，用户可以请求这些功能并将其与现有 vRealize Automation 工具配合使用。您还可以通过将 AWS 上的 Cloud 与 vRealize Automation VMware Cloud Assembl y 集成，来提高 IT

此模式描述了如何将 VMware vRealize Automation 配置为在 AWS VMware 云上自动构建 VMs 或应用功能。

先决条件和限制

先决条件

- 现有的本地数据中心和 VMware 云上的 AWS 软件定义数据中心 (SDDC)。有关云 SDCC 的更多信息，请参阅文档中的关于软件定义的数据中心。 VMware
- 本地数据中心与云 SDDC 之间的现有连接，使用 AWS Direct Connect、VPN（基于路由或基于策略）或两者兼而有之。
- 本地数据中心和云 SDDC 与网络时间协议（NTP）或其他权威时间源同步。
- 本地数据中心和云 SDDC 之间往返的最大时延不超过 100 毫秒。
- vCenter Server 的完全限定域名（FQDN）必须解析为私有 IP 地址。
- 可以访问本地环境的 Cloud SDDC 用户。
- 具有 vRealize Automation Cloud Assembly 服务角色的组织所有者访问权限。
- 在 vRealize Automation Service Broker 中拥有使用服务权限的最终用户。
- 本地数据中心的无类域间路由 (CIDR) 范围必须处于开放状态，才能 VMware 在 AWS 控制台上从云端生成 API 令牌。以下列表提供了生成 API 令牌所需最低角色：

- 组织成员
- 组织所有者
- 服务角色-AWS 上的 VMware 云
- 管理员
- NSX Cloud 管理员
- NSX Cloud 审计员

有关这方面的更多信息，请参阅 AWS 合作伙伴网络博客 SDDCs中的 AWS VMware 云连接选项。

限制

- 在一个 vRealize Automation 中，您只能配置 20 个带有公共终端节点的 VMware 云帐户。有关这方面的更多信息，请参阅文档中的可扩展性和并发最大值。 VMware

产品版本

- vRealize Automation 8.x 或更高版本
- VMware vRealize 身份管理器 3.x 或更高版本
- VMware vRealize Suite 生命周期管理器 8.x 版或更高版本

架构

下图显示了 vRealize Automation 服务，这些服务可以使用本地和 VMware 云端 AWS 环境中的基础设施。

VMware 云端装配组件

VMware Cloud Assembly 是 vRealize Automation 的核心组件，您可以使用它来部署 VMs 、配置和计算资源。下表描述了必须配置的 VMware 云端组件，才能在 AWS VMware 云 VMs 上进行配置。

组成部分	定义
云账户	云账户提供连接详细信息（例如，服务器名称、用户名和密码、访问密钥和 API 令牌）。

	VMware Cloud Assembly 使用云账户收集您的资源清单。
云区域	云区域识别云账户中的资源边界（例如，AWS 区域 和云 SDDC）。云区域将计算资源与 Cloud Assembly 项目进行关联。
项目	项目是一个由用户和资源（例如云区域）组成的逻辑实体。它还包括构建虚拟机时使用的资源限额和虚拟机命名策略。
规格模板映射	Flavor mapping 提供有关云模板中使用的虚拟机容量（例如，内存数量 CPUs 和数量）的信息。
镜像映射	图像映射映射云模板中使用的 VMware vSphere 虚拟机模板和 Amazon Web Services (AWS) 映像。有关这方面的更多信息，请参阅文档中的了解有关 vRealize Automation 中图像映射的 VMware 更多信息。
网络配置文件	网络配置文件控制在虚拟机预调配期间选择网络的放置决策。
存储配置文件	网络配置文件控制在虚拟机预调配期间选择存储的放置决策。
云模板	VMware 云模板是 vRealize Automation 的重要组成部分，因为它们定义了云基础架构的配置和编排。云模板是资源的规范，包括资源类型、资源属性和需要从用户处收集的输入。

工具

- VMware vRealize Aut omation — vRealize Automation 是一个基础设施自动化平台，具有事件驱动的状态管理和合规性。它旨在帮助组织控制和保护自助服务云、具有治理功能的多云自动化以及DevOps基于基础架构的交付。
- VMware AWS 上的 C VMware loud — A WS 上的 Cloud on AWS 是一款由 AWS 和 AWS 联合开发的集成云产品 VMware。

操作说明

生成 API 令牌

Task	描述	所需技能
从您的 VMware Cloud on AWS 账户生成 API 令牌。	1. 登录 VMware 云控制台。 2. 在 VMware 云服务工具栏上，选择我的账户，然后选择 API 令牌。 3. 输入 API 令牌的名称，提供所需使用期限，然后定义令牌的范围。 4. 选中打开 ID 复选框，然后选择生成。 5. 记录 API 令牌的凭证。 有关这方面的更多信息，请参阅 VMware 文档中的如何生成 API 令牌。	云管理员

在本地数据中心安装 vRealize Automation

Task	描述	所需技能
下载所需软件。	从 “我的 VMware 门户” 下载 VMware vRealize Suite ISO 文件。此软件包含 vRealize Suite 生命周期管理器、 VMware 身份管理器和 vRealize Automation	云管理员
安装 软件。	按照文档中使用适用于 vRealize Automationand Id VMware entity Manager 的简易安装程序安装 vRealize Suite	云管理员、云架构师

Task	描述	所需技能
	生命周期管理器中的说明安装软件并连接到您的云 SDCC。VMware **Important** 请确保您的安装可以使用以下内容： • 本地 VMware vCenter 服务器设置和登录凭据 • vRealize Automation IP 和子网的网络详细信息 • vRealize Automation 许可证密钥	

使用 VMware Cloud Assembly 连接 AWS 上的 VMware 云

Task	描述	所需技能
配置云账户。	1. 在 VMware 云控制台上，打开基础架构选项卡，选择管理-云帐户，然后选择添加云帐户。 2. 选择 AWS 上的 VMware Cloud 作为类型。 3. 粘贴之前记录的 API 令牌信息。这会填充您的 Cloud on AWS 组织 SDDCs 中的所有可用 VMware 云。 4. 选择所需云 SDCC，然后为 SDDC 提供 vCenter 用户名和密码。	云架构师、云管理员

Task	描述	所需技能
	5. 成功通过身份验证后，您可以查看状态为 “正常” 的集成 VMware Cloud on AWS 账户。 有关这方面的更多信息，请参阅 VMware 文档中的在 vRealiz VMware e Automation 中创建 AWS 云端账户。	
配置项目。	1. 在 VMware 云控制台上，打开项目选项卡，然后选择新建项目。 2. 输入项目名称。 3. 打开云区域选项卡，然后选择默认 VMware AWS 云账户。	云管理员
配置云区域。	1. 在 VMware 云控制台上，打开云区域，然后为您的 SDDC 数据中心选择云区域。 2. 默认情况下，`cloudadmin@vmc.local`（云 SDDC 的 vCenter 的默认本地用户 ID）只能访问 `Compute-ResourcePool` 中的预调配。 3. 打开 “云区域” 下的 “计算” 选项卡，然后选择 “计算-” ResourcePool。	云管理员

Task	描述	所需技能
配置规格模板映射。	1. 打开规格模板映射选项卡并创建新的规格模板映射。 2. 输入规格名称，选择 VMware Cloud on AWS 账户，然后提供 v 的数量 CPUs 和内存量。	云管理员
配置镜像映射。	1. 打开镜像映射并创建新的镜像映射。 2. 输入镜像名称。 3. 选择 VMware Cloud on AWS 账户并提供所需的云账户模板。	云管理员
配置网络配置文件。	1. 打开网络配置文件并创建新的网络配置文件。 2. 输入网络配置文件名称。 3. 打开网络选项卡，然后选择要用于预调配的现有网络。	云管理员
配置存储配置文件。	1. 打开存储配置文件并选择新建存储配置文件。 2. 输入存储配置文件的名称。 3. 在策略部分中，创建一条新的策略。 4. 选择工作负载数据存储。默认情况下，`cloudadmin@vmc.local` 只能访问工作负载数据存储中的预调配。	云管理员

Task	描述	所需技能
创建云模板。	1. 打开设计选项卡，选择云模板，然后选择新建自和空白画布。 2. 提供云模板的名称和描述。 3. 选择您以前创建的项目。 4. 在云模板资源设计页面中，根据您的要求将组件拖到空白画布中。 5. 选择测试以测试模板并修复所有问题。 6. 选择部署并提供部署名称以进行部署 VMs。 有关这方面的更多信息，请参阅 VMware 文档中的创建基本云模板。	云管理员

相关资源

- 将 vRealize Automation 8.x 连接到您的 SDDC：
- 在 AWS 控制台上从 VMware 云端部署 SDDC
- AWS Direct Connect 与 AWS 上的 VMware 云集成

使用 AWS 上的 VMware C VMware loud 在 AWS 上部署 SDDC

由 Deepak Kumar (AWS) 和 Derek Cox (AWS) 创作

摘要

注意：自 2024 年 4 月 30 日起， VMware Cloud on AWS 不再由其渠道 AWS 合作伙伴转售。该服务将继续通过博通提供。我们鼓励您联系您的 AWS 代表了解详情。

此模式描述了如何创建托管在亚马逊网络服务 (AWS) 云中的 VMware基于软件定义的数据中心 (SDDC)。您可以部署 SDDC 将 VMware 基于 vSphere 的工作负载迁移到 AWS 云，并在使用现有 VMware 工具和技能的同时利用 AWS 服务。您可以使用此 SDDC 在 VMware 基于 vSphere 的私有云、公有云和混合云环境中运行生产应用程序，并优化对 AWS 服务的访问权限。例如您可将 SDDC 用作灾难恢复的辅助站点，也可以将数据中心扩展到不同的地理位置。

VMware Cloud on AWS 是一项 pay-as-you-go（按需）服务，它允许各种规模的企业使用各种 AWS 服务在 VMware 基于 vSphere 的云环境中运行工作负载。在生产环境中，您可从每个 SDDC 集群至少 2 台主机开始，然后扩展到每个集群 16 台主机。有关更多信息，请参阅 AWS 上的 VMware Cloud 网站。要了解更多信息 SDDCs，请参阅文档中的软件定义数据中心简介。 VMware

先决条件和限制

先决条件

- 注册 “我的VMware 账户” 并填写所有字段。
- 注册 Amazon Web Services account。有关说明，请参阅 AWS Knowledge Center。
- 注册 AWS 上的 My VMware Cloud 账户。 激活链接将发送到您在注册时指定的电子邮件地址。

限制

- 请参阅 VMware 网站上的 VMware Cloud on AWS 配置限制页面。

产品版本

- 请参阅 VMware 文档中的 VMware Cloud on AWS 发行说明。

架构

目标技术堆栈

下图显示了在 AWS 裸机专用基础设施上运行的 VMware 软件堆栈，包括 vSphere、vCenter、vSAN 和 NSX-T。您可以通过与其他 AWS 服务（如亚马逊弹性计算云 (亚马逊)、亚马逊简单存储服务 (Amazon S3 EC2)、Amazon Redshift、AWS Direct Connect、亚马逊关系数据库服务 (Amazon RDS) 和亚马逊 DynamoDB 等其他 AWS 服务的无缝集成，管理 VMware基于您的资源和工具。

VMware Cloud on AWS 的基本实体是 SDDC，它包括以下组件：

- 计算：计算组件是 AWS SDDC 上 VMware 云的最低层。 VMware AWS 上的 Cloud 在亚马逊 EC2 裸机实例类型上运行。它们包括 `i3.metal`、 `i3en.metal` 和 `i4i.metal`，以及提供对物理资源（如处理器和内存）的直接访问。

> **Important**
>
> VMware Cloud on AWS 的`i3.metal`实例类型（包括一年期和三年期的按需和订阅选项）将于 2026 年 12 月 31 日终止生命周期并终止支持。此外，新客户目前无法申请 `i3.metal` 实例。有关更多信息，请参阅 VMware Cloud 博客上的公告。

- 存储：SDDC 集群支持 VMware 采用全闪存配置的 vSAN，使用非易失性内存 express NVMe () 闪存进行存储，可提供快速和高性能的存储。 从 SDDC 版本 1.20 开始，C VMware loud on AWS 支持两种类型的外部存储：亚马逊 FSx 版 NetApp ONTAP 和 VMware Cloud Flex 存储。
- 联网：联网功能和策略通过在 SDD VMware C 群集中使用 NSX-T 进行管理。在 SDDC 集群中创建多层虚拟网络，用于将网络资源与物理设备分开。这使得 VMware Cloud on AWS 用户能够创建逻辑的、软件定义的网络。

工具

- VMware Cloud on AWS 是一款由 AWS 和 AWS 联合开发的集成云产品 VMware。

操作说明

在您的 Amazon Web Services account 中创建 VPC 和子网

Task	描述	所需技能
登录您的 Amazon Web Services account。	使用具有管理员权限的凭证登录 Amazon Web Services account。	云管理员
创建新的 VPC。	在本步骤中，您将定义链接至 SDC 的虚拟私有云（VPC）。如果您已有要用于 SDDC 的 VPC，请跳过此步骤。 1. 选择要在 AWS SDDC 上部署您的 VMware 云的 AWS 区域。 2. 打开位于 https://console.aws.amazon.com/vpc/ 的 Amazon VPC 控制台。 3. 在导航窗格中，选择您的 VPCs。 4. 选择创建 VPC。 5. 指定 VPC 设置，例如 VPC 名称标签、IPv4 CIDR 块、租赁（保持默认），然后选择创建 VPC。 6. 在创建 VPC 后，选择Close（关闭）。 有关更多信息，请参阅 AWS 文档中的创建和配置 VPC。	云管理员
创建私有子网。	现在，您将为每个可用区域的弹性网络接口（ENI）创建一个	云管理员

Task	描述	所需技能
	私有子网。我们建议您使用未连接互联网网关的子网。 1. 打开位于 https://console.aws.amazon.com/vpc/ 的 Amazon VPC 控制台。 2. 在导航窗格中，选择 Subnets(子网)。 3. 选择 Create Subnet（创建子网）。 4. 在创建子网页面，选择之前创建的 VPC。 5. 完成子网的设置，包括子网名称、可用区和 IPv4CIDR 块。 6. 选择创建子网。 重复这些步骤，为区域中的每个可用区创建子网。	

在 AWS 上激活 VMware Cloud

Task	描述	所需技能
激活服务。	当您注册 “我的VMware 帐户” 时， VMware 会向您发送一封欢迎电子邮件和激活链接到您指定的电子邮件地址。 1. 在浏览器中打开欢迎电子邮件中的激活服务链接。 2. 使用我的VMware 凭据登录。	云管理员

Task	描述	所需技能
	3. 查看并接受使用服务的条款和条件。 4. 完成账户激活进程。您将被重定向到 AWS 控制台上的 VMware 云端。（注意：VMware Cloud on AWS 账户以组织为基础，该组织代表订阅该账户的团体或业务领域。 该组织与 AWS Organizations 没有任何关系。 ） 5. 在 “选择或创建组织” 页面上，创建与 “我的VMware 账户” 关联的组织。 6. 输入组织名称和地址，以进行逻辑区分。 7. 选择创建组织以完成该过程。 有关此过程的更多信息，请参阅 AWS 文档中的 AWS 上的 SDDC 部署和最佳实践指南。	

Task	描述	所需技能
分配 IAM 角色。	创建组织后，为特定用户分配特权访问权限，以访问云服务和 SDDC 控制台、SDDC 和 NSX 组件。有关说明，请参阅 VMware 文档中的为组织成员分配 VMC 服务角色。 有两种类型的组织角色： • 组织拥有者可以添加、删除和修改用户，并访问所有云资源。 • 组织成员仅能访问云资源。	云管理员

部署 SDDC

Task	描述	所需技能
在您的 C VMware loud on AWS 账户中部署 SDDC。	⚠ Important AWS 账户作为登记卖家与 VMware 组织关联后，无法更新 AWS 账号。每个 VMware 组织只能有一个记录在案的 AWS 卖家。 要部署 SDDC，请执行以下操作： 1. 登录 VMC 控制台，网址为https://vmc.vmware.com。	云管理员、云架构师

Task	描述	所需技能
	2. 从可用服务中选择 VMware Cloud on AWS 服务。 3. 选择 Create SDDC（创建 SDDC）。 4. 输入 SDDC 属性，例如 AWS 区域、部署(单主机、多主机或延伸集群)、主机类型、SDDC 名称、主机数量、主机容量和总容量，然后选择下一步。 5. 连接到您的 Amazon Web Services account，然后选择下一步。 6. 选择您之前配置的 VPC 和子网，然后选择下一步。 7. 输入 SDDC 的管理子网 CIDR 块，然后选择下一步。有关更多信息，请参阅 VMware 云博客上的 “为 SDDC 选择 IP 子网和连接”。 8. 选中两个复选框以确认您对部署 SDDC 的费用负责，然后选择部署 SDDC。 当您选择部署 SDDC 时，您需要付费。您将无法暂停或取消部署过程，这需要一定的时间才能完成。 有关创建 SDDC 的更多信息，请参阅文档中的从 VMC 控制台部署 SDDC。 VMware	

相关资源

- 部署和管理软件定义的数据中心（文档）VMware
- VMware AWS 上的 Cloud 功能（AWS 网站）
- 利@@ 用 AWS 上的云加速 VMware 云迁移和现代化（视频）

在 AWS 上将 VMware vRealize 网络洞察与 VMware 云集成

由 Deepak Kumar (AWS)、Piotr Pitera (AWS) 和 Sachin Trivedi (AWS) 创作

摘要

注意：自 2024 年 4 月 30 日起， VMware Cloud on AWS 不再由其渠道 AWS 合作伙伴转售。该服务将继续通过博通提供。我们鼓励您联系您的 AWS 代表了解详情。

此模式描述了如何将 VMware vRealize Network Insight 与 VMware Cloud 相集成， AWS 以及如何检查来自虚拟机的流量。这种集成还可以帮助您规划应用程序迁移到 VMware Cloud on。 AWS

vRealize Network Insight 可让您深入了解您的网络基础 它提供网络监控和分析功能，以提高安全性、降低迁移风险和优化性能。您可以使用此工具监控来自虚拟机的流量，并根据观察到的流量查看推荐的安全规则。有关 vRealize Network Insight 的更多信息，请参阅VMware 文档。

VMware Cloud on AWS 是一 pay-as-you-go项（按需）服务，它使各种规模的企业能够使用各种各样的，在 VMware 基于 vSphere 的云环境中运行工作负载。 AWS 服务在生产环境中，您可从每个 SDDC 集群至少 2 台主机开始，然后扩展到每个集群 16 台主机。有关更多信息，请 AWS访问网站上的VMware 云端。要了解更多信息 SDDCs，请参阅文档中的软件定义数据中心简介。 VMware

先决条件和限制

先决条件

- VMware AWS 上的 Cloud SDDC，已部署

限制

- 有关已知限制，请参阅VMware 文档。

产品版本

- vRealize 网络洞察力版本 5.0.0
- VMware Cloud on AWS SDDC 版本 1.24

架构

源技术堆栈

- vRealize 网络洞察

目标技术堆栈

- VMware Cloud on AWS

目标架构

下图显示了 VMware Cloud on AWS 和 vRealize Network Insight 内部部署之间的连接。

工具

- VMware Cloud on AWS 是由 AWS 和联合开发的集成云产品 VMware。
- VMware vRealize Network Insight 是一款监控和分析工具，可提供对网络基础架构的可见性，以便进行安全规划和故障排除。

操作说明

为 vRealize Network Insight 设置环境

Task	描述	所需技能
创建 VMware 用户帐户。	创建 VMware 用户帐户或登录您的现有 VMware 帐户。 要开设新账户，请执行以下操作： 1. 填写注册表即可注册 Cust VMware omer Connect 账户。	云管理员

Task	描述	所需技能
	新用户将收到一封激活其帐户的电子邮件。 2. 输入电子邮件中的身份验证码。 3. 登录 Customer Connect。	
下载 vRealize 网络洞察的 OVA 文件。	下载 vRealize Network Insight 的 OVA 文件： 1. 导航到 https://my.vmware.com/group/vmware/home 上的 VMware 产品下载页面。 2. 搜索 vRealize 网络洞察。 3. 下载最新的 vRealize Network Insight 版本 5.0.0 平台和收集器 OVA 文件。	云管理员
部署 vRealize 网络洞察。	有关部署说明，请参阅VMware 文档。	云管理员

添加数据源和收集器

Task	描述	所需技能
添加数据源。	1. 登录 vRealize 网络洞察。 2. 选择 “设置”、“帐户和数据源”、“添加源”。 3. 对于类型，选择本地 vCenter 服务器。 有关更多信息，请参阅 VMware 文档。	云管理员

Task	描述	所需技能
为数据源设置收集器。	有关说明，请参阅VMware 文档。	云管理员

分析应用程序依赖关系

Task	描述	所需技能
创建应用程序。	如果你在 vRealize Network Insight 中没有现有的应用程序，请按照VMware 文档中的步骤创建一个应用程序。	云管理员
发现和分析您的应用程序。	1. 使用 vRealize 网络洞察来发现您的应用程序。有关说明，请参阅VMware 文档。 2. 分析您的应用程序。有关说明，请参阅VMware 文档。	云管理员

相关资源

- 使用 CI VMware oud on 在 AWS 上部署 VMware SDDC AWS（AWS 规范性指南）
- AWS 使用混合链接模式（AWS 规范性指导）将数据中心扩展配置到 VMware Cloud on
- AWS 使用 VMware HCX 将 VMware SDDC 迁移到 VMware 云端（AWS 规范性指南）
- VMware vRealize 网络洞察文档（VMware 网站）

使用 HCX 操作系统辅助迁移在 AWS 上迁移 VMs 到 VMware 云端

由 Deepak Kumar (AWS) 和 Himanshu Gupta (AWS) 编写

摘要

注意：自 2024 年 4 月 30 日起， VMware Cloud on AWS 不再由其渠道 AWS 合作伙伴转售。该服务将继续通过博通提供。我们鼓励您联系您的 AWS 代表了解详情。

此模式描述了如何使用操作系统辅助迁移 (OSAM) 将虚拟机 (VM) 从非 vSphere 环境迁移到亚马逊网络服务 VMware 云 (AWS)。

OSAM 是 VMware 混合云扩展 (HCX) 的一部分，该扩展包含在 AWS 上的 VMware 云中。您可以使用 OSAM 将非 vSphere 环境（例如 VMware KVM 或 Hyper-V）迁移到 AWS 上的云端。 VMware OSAM 使用 Sentinel 软件，您可以将其安装在 Windows 或 Linux 客户虚拟机上，以帮助将虚拟机从本地环境复制到 AWS 云端的软件定义数据中心 (SDDC)。 VMware

此模式说明了如何启用 OSAM、在 Windows 虚拟机上安装 Sentinel 软件、在源站点与 HCX Sentinel Gateway (SGW) 设备连接和注册，以及如何与目标站点的 HCX Sentinel 数据接收器 (SDR) 设备建立转发连接，以启动迁移。

有关 OSAM 的更多信息，请参阅VMware 文档。

先决条件和限制

先决条件

- 在源环境与目标环境中安装 HCX。有关 HCX 先决条件，请参阅 AWS Prescriptive Guidence 文档中的使用 VMware HCX 将 VMware SDDC 迁移到 AWS 上的 VMware 云。
- 有关 OSAM 先决条件，请参阅 VMware 文档中的安装清单。
- 有关 OSAM 端口信息，请参阅端口和协议网站上的 VMware HCX VMware 端口要求。

限制

- VMware HCX 4.2.0 配置限制
- OSAM 部署注意事项

- 受支持的来宾操作系统
- 来宾操作系统注意事项

产品版本

- VMware HCX 4.2.0
- VMware SDDC 1.12

架构

下图显示了 HCX OSAM 如何与 Sentinel 软件配合使用，将非 vSphere VMs 从本地环境复制到 AWS 上的 Cloud。 VMware

OSAM 由三部分组成：

- Sentinel Gateway (SGW) 设备，用于连接和转发基于源代码的环境中的工作负载和应用程序 VMware
- Sentinel 数据接收器 (SDR)，用于目标 Cloud VMware on AWS 环境中，用于接收来自源端的迁移工作负载
- Sentinel 软件，必须安装至要迁移的每台来宾虚拟机上

OSAM 使用安装在 Windows 或 Linux 客户机上的 Sentinel 软件 VMs 来帮助将虚拟机从本地复制到 SDDC。 VMware 您在客户机上安装的 Sentinel 软件从客户机虚拟机 VMs 收集系统配置并协助进行数据复制。此信息还用于创建要迁移的客户机清单，并帮助 VMs 为复制和迁移目的准备副本虚拟机上的磁盘。

工具

- VMware HCX 4.2.0
- VMware AWS 上的云 SDDC

操作说明

配置 HCX

Task	描述	所需技能
部署 HCX Cloud 和 HCX Connector。	按照文档中 HCX 连接器和 HCX 云安装中的 VMware 说明进行操作。	云管理员、系统管理员

配置 OSAM 并进行迁移 VMs

Task	描述	所需技能
安装 HCX Sentinel。	若要在 Linux 上安装 Sentinel： 1. 在 HCX Connector 的 vCenter Server，选择互联、多站点服务网格、哨点管理。 2. 选择下载 Linux 捆绑包。 3. 在 Linux 计算机安装 Sentinel 代理。 有关更多信息，请参阅文档中的下载和安装 HCX Sentinel Agent 软件。 VMware	云管理员
迁移 VMs。	要迁移您的 VMs 群组（称为移动组），请执行以下步骤： 1. 在 vSphere Client 中，从 HCX 插件中选择服务、迁移。 2. 选择 Migrate。	云管理员

Task	描述	所需技能
	3. 选择 非 vSphere 清单、远程连接。这将显示你安装了 HCX Sentinel 的列表 VMs 。 4. 在组名中，输入要为创建的移动组的名称 VMs。 5. 选择 VMs 要迁移的，然后选择添加将其添加到移动组。 6. 对于每台虚拟机： a. 选择目标计算容器。 b. 选择目标存储器。 c. 选择迁移配置文件。 d. 选择目标文件夹。 7. 若要启动迁移过程，请选择开始。 HCX 会在迁移开始前验证您的虚拟机选择。 有关更多信息，请参阅 VMware 文档中的使用移动组迁移虚拟机和使用移动组监控和估算迁移。	

相关资源

VMware 文档：

- VMware HCX 用户指南
- 使用 VMC SDDC 目标环境安装核对清单 B-HCX
- VMware AW VMware S 上云端的 HCX
- HCX 操作系统辅助迁移 AWS VMware 云端

- VMware HCX 4.2.1 版本说明

使用 VMware Aria 日志操作 AWS 将日志从 VMware Cloud 发送到 Splunk

由 Deepak Kumar (AWS) 和 Piotr Pitera (AWS) 创作

摘要

注意：自 2024 年 4 月 30 日起， VMware Cloud on AWS 不再由其渠道 AWS 合作伙伴转售。该服务将继续通过博通提供。我们鼓励您联系您的 AWS 代表了解详情。

此模式描述了如何使用 A VMware ria Operations for Logs 将 Cl VMware oud 上的 AWS 事件或日志转发到系统日志或 HTTP 端点（例如 Splunk）。

VMware Aria Operations for Logs 是一款日志分析工具，可在 VMware Cloud on AWS 环境中增强可见性并加快故障排除速度。您可以将此工具配置为将 VMware Cloud 中的全部或部分日志或事件发送 AWS 到 syslog 或 HTTP 端点。终端节点可以是软件即服务 (SaaS) 端点，也可以是本地端点，例如 Splunk。（此模式提供了 Splunk 的使用说明。）要了解有关 VMware Aria 日志操作的更多信息，请参阅VMware 文档。

VMware Cloud on AWS 是一 pay-as-you-go项（按需）服务，它使各种规模的企业能够使用各种各样的，在 VMware 基于 vSphere 的云环境中运行工作负载。 AWS 服务在生产环境中，您可以从每个软件定义的数据中心 (SDDC) 集群最少 2 台主机开始，然后将每个集群扩展到最多 16 台主机。有关更多信息，请 AWS访问网站上的VMware 云端。要了解更多信息 SDDCs，请参阅文档中的软件定义数据中心简介。 VMware

先决条件和限制

先决条件

- Splunk，在本地配置

限制

你可以注册免费试用 A VMware ria Operations for Logs。此订阅有效期为 30 天，并有以下限制：

- 您可以转发的最大日志大小：每天 50 GB 日志
- 您可以创建的最大日志转发配置数：10
- 您可以激活的最大日志转发配置数：5

要访问所有服务功能，必须升级到高级订阅。

有关试用版和高级版订阅的更多信息，请参阅 VMware 文档中的 A VMware ria 日志操作 (SaaS) 订阅和计费。有关使用限制的更多信息，请参阅 VMware 文档中的功能使用限制。

产品版本

- VMware AWS SDDC 上的 Cloud 版本 1.24
- VMware 日志的 Aria 操作版本 8.10
- 本地部署 Splunk 版本 9.x

架构

源技术堆栈

- VMware Cloud on AWS
- VMware 日志的 Aria 操作

目标技术堆栈

- 本地 Splunk

目标架构

下图显示了企业数据中心与 VMware Aria Operations for Logs in VMware Cloud 之间的连接 AWS。

工具

- VMware Cloud on AWS 是由 AWS 和联合开发的集成云产品 VMware。
- VMware Aria Operations for Logs 是 VMware Cloud on 的日志分析和故障排除工具 AWS。

操作说明

部署 SDDC 并启用 VMware Aria 日志操作

Task	描述	所需技能
在 AWS SDDC 上部署 VMware 云。	按照 AWS 规范性指南中的使用 VMware 云在上 AWS 部署 VMware SDDC AWS 中的说明进行操作。	云架构师、云管理员
注册 A VMware ria 操作日志。	有关说明，请参阅VMware 文档。	云架构师

部署云代理

Task	描述	所需技能
部署云代理。	要将日志转发到 Splunk 的本地实例，您必须为 VMware Aria 日志操作添加云代理。此代理从本地数据中心接收信息，并将其发送到 VMware Aria Operations 进行日志进行分析。 要下载并安装云代理，请执行以下操作： 1. 确保您的本地环境和 C VMware loud on 之间的端口 443、22 和 514 处于打开状态。 AWS对于其他端口，您可以使用 1514/TCP 或 6514/TCP。有关端口的更多信息，请参阅 VMware	云管理员、云架构师

Task	描述	所需技能
	文档中的 VMware Aria 操作日志防火墙建议。 2. 登录日志版 VMware Aria 操作。 3. 在主页上，在构件中选择“添加收集器”。 4. 在云代理虚拟设备屏幕上，复制令牌密钥。您必须在 24 小时内使用此密钥才能完成以下步骤。 5. 选择 OVA 文件的下载链接。 6. 导航到 VMware vSphere Web 客户端，选择您的集群，然后选择部署 OVF 模板。 7. 当系统提示您输入密钥时，请粘贴您在步骤 4 中复制的令牌密钥。 8. 选择“完成”以安装云代理。	

将日志转发到本地 Splunk 终端节点

Task	描述	所需技能
配置日志转发。	要将日志转发到 Splunk 端点，请执行以下操作： 1. 登录日志版 VMware Aria 操作。 2. 导航到“日志管理”。 3. 选择“日志转发”。	

Task	描述	所需技能
	4. 选择“新建配置”，然后完成以下设置： • 提供日志转发配置的名称。 • 对于目的地，选择本地。 • 对于云代理，请选择您之前安装的云代理。 • 对于端点类型，选择 TCP。 • 对于终端节点 URL，请按以下格式提供您的本地 Splunk 网址： `tcp://x.x.x.x (your Splunk IP address): 514` • （可选）对于标签，您可以指定标签名称和值以方便查询。 • 选择“应用于所有日志”或“应用于特定日志”。如果您想将所有 VMware Cloud on AWS 日志发送到 Splunk，请选择应用于所有日志。 5. 选择验证。 6. 选择保存。 有关更多信息，请参阅 VMware 文档中的转发来自 VMware Aria Operations 的日志以获取日志。	

相关资源

- VMware AWS 网站上的云端
- 关于软件定义的数据中心（文档）VMware
- 使用 Cl VMware oud on 在上 AWS 部署 VMware SDDC AWS（AWS 规范性指南）
- 使用 VMware HCX（AWS 规范性指南 AWS ）将工作负载迁移到 VMware 云端
- AWS 使用混合链接模式（AWS 规范性指导）将数据中心扩展配置到 VMware Cloud on

使用 AWS CDK 在 Amazon ECS Anywhere 上为混合工作负载设置 CI/CD 管道和 GitLab

由 Rahul Sharad Gaikwad 博士 (AWS) 编写

摘要

注意： CodeCommit AWS 不再向新客户开放。AWS 的现有客户 CodeCommit 可以继续照常使用该服务。了解更多。

Amazon ECS Anywhere 是 Amazon Elastic Container Service (Amazon ECS) 的扩展。它支持将外部实例（例如本地服务器或虚拟机 (VM)）注册到 Amazon ECS 集群。该功能有助于降低成本并减轻复杂的本地容器编排和操作。您可以使用 ECS Anywhere 在本地和云环境中部署和运行容器应用程序。它使您的团队无需学习多个域和技能组合，也无需自行管理复杂的软件。

此模式描述了 step-by-step一种使用亚马逊网络服务 (AWS) 云开发套件 (AWS CDK) 堆栈为亚马逊 ECS 集群配置亚马逊 ECS Anywhere 实例的方法。然后，您可以使用 AWS CodePipeline 来设置持续集成和持续部署 (CI/CD) 管道。然后，您将 GitLab 代码存储库复制到 AWS， CodeCommit 并将您的容器化应用程序部署在 Amazon ECS 集群上。

此模式旨在帮助那些使用本地基础设施运行容器应用程序并 GitLab 用于管理应用程序代码库的用户。您可使用 Amazon Web Services Cloud 服务管理这些工作负载，而不会干扰您现有的本地基础设施。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 在本地基础设施上运行的容器应用程序。
- 用于管理应用程序代码 GitLab 库的存储库。有关更多信息，请参阅存储库 (GitLab)。
- 已安装和配置 AWS 命令行界面（AWS CLI）。有关更多信息，请参阅安装或更新最新版本的 AWS CLI（AWS CLI 文档）。
- AWS CDK Toolkit，已安装并全局配置。有关更多信息，请参阅安装 AWS CDK（AWS CDK 文档）。
- npm，已为中的 AWS CDK 安装和配置。 TypeScript有关更多信息，请参阅下载和安装 Node.js 和 npm（npm 文档）。

限制

- 有关限制和注意事项，请参阅 Amazon ECS 文档中的外部实例 (Amazon ECS Anywhere)。

产品版本

- AWS CDK Toolkit 版本 2.27.0 或更高版本
- npm 版本 7.20.3 或更高版本
- Node.js 版本 16.6.1 或更高版本

架构

目标技术堆栈

- AWS CDK
- AWS CloudFormation
- AWS CodeBuild
- AWS CodeCommit
- AWS CodePipeline
- Amazon ECS Anywhere
- Amazon Elastic Container Registry (Amazon ECR)
- AWS Identity and Access Management (IAM)
- AWS Systems Manager
- GitLab 存储库

目标架构

此图表表示此模式中描述的两个主要工作流程，即配置 Amazon ECS 集群和设置CI/CD pipeline that sets up and deploys the CI/CD管道，如下所示：

1. 预调配 Amazon ECS 集群
   a. 当您部署第一个 AWS CDK 堆栈时，它会在 AWS 上创建一个 CloudFormation 堆栈。
   b. 此 CloudFormation 堆栈预配置 Amazon ECS 集群和相关的 AWS 资源。
   c. 要将外部实例注册到 Amazon ECS 集群，您必须在虚拟机上安装 AWS Systems Manager Agent (SSM Agent) 并将该虚拟机注册为 AWS Systems Manager 托管实例。

d. 在您的虚拟机上，您还必须安装 Amazon ECS 容器代理和 Docker，以将其注册为 Amazon ECS 集群的外部实例。

e. 当外部实例注册并配置到 Amazon ECS 集群时，它可以在注册为外部实例的虚拟机上运行多个容器。

f. Amazon ECS 集群处于活动状态，可通过容器运行应用程序工作负载。Amazon ECS Anywhere 容器实例在本地环境中运行，但与云 Amazon ECS 集群关联。

2. 设置和部署 CI/CD 管线

a. 当您部署第二个 AWS CDK 堆栈时，它会在 AWS 上创建另一个 CloudFormation 堆栈。

b. 此 CloudFormation 堆栈预置管道 CodePipeline 和相关的 AWS 资源。

c. 您可以将应用程序代码更改推送并合并到本地 GitLab 存储库。

d. GitLab 存储库会自动复制到 CodeCommit 存储库。

e. 对 CodeCommit 存储库的更新会自动开始 CodePipeline。

f. CodePipeline 从中复制代码 CodeCommit 并创建可部署的 CodeBuild内置应用程序。

g. CodePipeline 创建 CodeBuild 构建环境的 Docker 镜像并将其推送到 Amazon ECR 存储库。

h. CodePipeline 启动从 Amazon ECR 存储库中提取容器映像的 CodeDeploy 操作。

i. CodePipeline 在 Amazon ECS 集群上部署容器映像。

自动化和扩缩

此模式使用 AWS CDK 作为基础设施即代码（IaC）工具来配置和部署此架构。AWS CDK 可帮助您编排 AWS 资源，并设置 Amazon ECS Anywhere 和 CI/CD 管线。

工具

Amazon Web Services

- AWS Cloud Development Kit (AWS CDK) 是一个软件开发框架，可帮助您在代码中定义和预调配 Amazon Web Services Cloud 基础设施。
- AWS CodeCommit 是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。
- AWS CodePipeline 可帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件变更所需的步骤。
- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。

- Amazon Elastic Container Registry (Amazon ECR) 是一项安全、可扩展且可靠的托管容器映像注册表服务。
- Amazon Elastic Container Service (Amazon ECS)是一项快速且可扩展的容器管理服务，可帮助运行、停止和管理集群上的容器。这种模式还使用 Amazon ECS Anywhere，它支持向 Amazon ECS 集群注册本地部署服务器或虚拟机。

其他工具

- Node.js 是一个事件驱动的 JavaScript 运行时环境，专为构建可扩展的网络应用程序而设计。
- npm 是在 Node.js 环境中运行的软件注册表，用于共享或借用软件包以及管理私有软件包的部署。
- Vagrant 是一个开源实用程序，用于构建和维护便携式虚拟软件开发环境。出于演示目的，此模式使用 Vagrant 创建本地虚拟机。

代码存储库

此模式的代码可在使用 AWS GitHub CDK 存储库的 Amazon ECS Anywhere 的 CI/CD 管道中找到。

最佳实践

部署此模式时，请考虑以下最佳实践：

- 使用 AWS CDK 开发和部署云基础设施最佳实践
- 使用 AWS CDK 开发云应用程序的最佳实践（AWS Blog 文章）

操作说明

验证 AWS CDK 配置

Task	描述	所需技能
验证 AWS CDK 版本。	通过输入以下命令验证 AWS CDK Toolkit 的版本。 `cdk --version` 此模式需要版本 2.27.0 或更高版本。如果您拥有早期版本，	DevOps 工程师

Task	描述	所需技能
	请按照 AWS CDK 文档中的说明对其进行更新。	
验证 npm 版本。	通过输入以下命令验证 npm 的版本。 `npm --version` 此模式需要版本 7.20.3 或更高版本。如果您拥有早期版本，请按照 npm 文档中的说明对其进行更新。	DevOps 工程师

设置 AWS 凭证。	通过输入 `aws configure` 命令并按照提示进行操作设置凭证。
	`$aws configure`
	`AWS Access Key ID [None]: <your-access-key-ID>`
	`AWS Secret Access Key [None]: <your-secret-access-key>`
	`Default region name [None]: <your-Region-name>`
	`Default output format [None]:`
		DevOps 工程师

引导 AWS CDK 环境

Task	描述	所需技能
克隆 AWS CDK 代码存储库。	1. 输入以下命令，使用此模式的使用 AWS CDK 的	DevOps 工程师

Task	描述	所需技能

	Amazon ECS Anywhere 的 CI/CD 管线存储库。
	`git clone https://github.com/aws-samples/amazon-ecs-anywhere-cicd-pipeline-cdk-sample.git`
	2. 通过输入以下命令导航到克隆的目录。
	`cd amazon-ecs-anywhere-cicd-pipeline-cdk-sample`	

引导环境。	输入以下命令，将 CloudFormation 模板部署到您要使用的账户和 AWS 区域。 `cdk bootstrap <account-number>/<Region>` 有关更多信息，请参阅 AWS CDK 文档中的引导。	DevOps 工程师

为 Amazon ECS Anywhere 构建和部署基础设施

Task	描述	所需技能

安装软件包依赖关系并编译 TypeScript 文件。	通过输入以下命令安装软件包依赖关系并编译 TypeScript 文件。
	`$cd EcsAnywhereCdk`
	`$npm install`
		DevOps 工程师

Task	描述	所需技能
	`$npm fund` 这些命令安装示例存储库内的所有软件包。有关更多信息，请参阅 npm 文档中的 npm ci 和 npm 安装。如果在输入这些命令时出现任何关于丢失程序包的错误，请参阅此模式的故障排除部分。	
构建 项目。	要构建项目代码，请输入以下命令。 `npm run build` 有关构建和部署项目的更多信息，请参阅 AWS CDK 文档中的您的第一个 AWS CDK 应用程序。	DevOps 工程师

Task	描述	所需技能

部署 Amazon ECS Anywhere 基础设施堆栈。	1. 通过输入以下命令列出堆栈。
	`$cdk list`
	2. 确认输出是否返回了 `EcsAnywhereInfraStack` 和 `ECSAnywherePipelineStack` 堆栈。
	3. 通过输入以下命令部署 `EcsAnywhereInfraStack` 堆栈。
	`$cdk deploy EcsAnywhereInfraStack`
		DevOps 工程师

验证堆栈创建和输出。	1. 登录 AWS 管理控制台并在上打开 CloudFormation 控制台https://console.aws.amazon.com/cloudformation/。 2. 在堆栈页面，选择 `EcsAnywhereInfraStack` 堆栈。 3. 确认堆栈状态为 `CREATE_IN_PROGRESS` 或 `CREATE_COMPLETE` 。 设置 Amazon ECS 集群可能需要一定的时间。在堆栈创建完成前，请勿继续。	DevOps 工程师

设置本地虚拟机

Task	描述	所需技能
设置您的虚拟机。	通过从 Vagrantfile 所在的根目录中输入 `vagrant up` 命令来创建 Vagrant 虚拟机。有关更多信息，请参阅 Vagrant 文档。	DevOps 工程师
将您的虚拟机注册为外部实例。	1. 使用 `vagrant ssh` 命令登录 Vagrant 虚拟机。有关更多信息，请参阅 Vagrant 文档。 2. 按照 AWS CLI 安装说明并输入以下命令，在虚拟机上安装 AWS CLI。 `$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" \ > -o "awscliv2.zip" $sudo apt install unzip $unzip awscliv2.zip $sudo ./aws/install $aws configure AWS Access Key ID [None]: <your-access-key-ID> AWS Secret Access Key [None]: <your-secret-access-key> Default region name [None]: <your-Region-name> Default output format [None]:`	DevOps 工程师

Task	描述	所需技能

	1. 创建激活码和 ID，您可以使用该激活码和 ID 在 AWS Systems Manager 注册虚拟机并激活外部实例。此命令输出包括激活 ID 和激活码值。
	
	`aws ssm create-activation \ > --iam-role EcsAnywhereInstanceRole \ > | tee ssm-activation.json`
	
	如果您在运行此命令时收到错误，请参阅故障排除部分。
	2. 导出激活 ID 与代码值。
	
	`export ACTIVATION_ID=<activation-ID> export ACTIVATION_CODE=<activation-code>`
	
	3. 下载安装脚本至虚拟机。
	
	`curl --proto "https" -o "ecs-anywhere-install.sh" \ > "https://amazon-ecs-agent.s3.amazonaws.com/ecs-anywhere-install-latest.sh"`	

Task	描述	所需技能
	4. 在您的虚拟机上运行安装脚本。 `sudo bash ecs-anywhere-install.sh \ --cluster EcsAnywhereCluster \ --activation-id $ACTIVATION_ID \ --activation-code $ACTIVATION_CODE \ --region <region-name>` 这会将您的虚拟机设置为 Amazon ECS Anywhere 外部实例，并将该实例注册至 Amazon ECS 集群。有关更多信息，请参阅 Amazon ECS 文档中的向集群注册外部实例。如果您遇到任何问题，请参阅故障排除部分。	
验证 Amazon ECS Anywhere 和外部虚拟机状态。	要验证您的虚拟机是否已连接到 Amazon ECS 控制面板并正在运行，请使用以下命令。 `$aws ssm describe-instance-information $aws ecs list-container-instances --cluster $CLUSTER_NAME`	DevOps 工程师

部署 CI/CD 管道

Task	描述	所需技能
在 CodeCommit 存储库中创建一个分支。	通过为存储库创建第一个提交，创建一个`main`在 CodeCommit repo 中命名的分支。您可以按照 AWS 文档在中创建提交 CodeCommit。以下命令是一个示例。 `aws codecommit put-file \ --repository-name EcsAnywhereRepo \ --branch-name main \ --file-path README.md \ --file-content "Test" \ --name "Dev Ops" \ --email "devops@example.com" \ --commit-message "Adding README."`	DevOps 工程师
设置存储库镜像。	您可以将 GitLab 存储库镜像到外部源或从外部源镜像。您可选择哪个存储库作为源。分支、标签和提交自动同步。在托管应用程序的 GitLab 存储库和存储 CodeCommit 库之间设置推送镜像。有关说明，请参阅设置从 GitLab 到的推送镜像 CodeCommit（GitLab 文档）。	DevOps 工程师

Task	描述	所需技能
	Note 默认情况下，镜像会自动同步存储库。如果要手动更新存储库，请参阅更新镜像（GitLab 文档）。	
部署 CI/CD 管线堆栈。	通过输入以下命令部署 `EcsAnywherePipelineStack` 堆栈。 `$cdk deploy EcsAnywherePipelineStack`	DevOps 工程师

Task	描述	所需技能
测试 CI/CD 管线。	1. 更改应用程序代码并将其推送到源本地 GitLab 存储库。有关更多信息，请参阅推送选项（GitLab 文档）。例如，编辑 `../application/index.html` 文件以更新应用程序版本值。 2. 当代码复制到 CodeCommit 存储库时，这会启动 CI/CD 管道。请执行以下操作之一： • 如果您使用自动镜像将 GitLab 存储库与存储库同步，请继续下一步。CodeCommit • 如果您使用的是手动镜像，请按照更新镜像（GitLab 文档）中的说明将应用程序代码更改推送到 CodeCommit 存储库。 3. 在您的本地计算机上，在 Web 浏览器中输入 http://localhost:80。打开 NGINX 网页，因为端口 80 被转发到 Vagrantfile 中的本地主机。确认您可查看更新的应用程序版本值。验证管线和映像部署。 4. （可选）如果要在 AWS 管理控制台 中验证部署，请执行以下操作： a. 打开 Amazon ECS 控制台，网址为https://	DevOps 工程师

Task	描述	所需技能
	console.aws.amazon.com/ecs/。 b. 从导航栏中，选择要使用的区域。 c. 在导航窗格中，选择集群。 d. 在集群页面上，选择集EcsAnywhereCluster群。 e. 选择任务定义。 f. 确认容器正在运行。	

清理

Task	描述	所需技能

清理和删除资源。	完成此模式后，应移除您创建的 proof-of-concept资源。要进行清理，请输入以下命令。
	`$cdk destroy EcsAnywherePipelineStack`
	`$cdk destroy EcsAnywhereInfraStack`
		DevOps 工程师

故障排除

事务	解决方案
安装程序包依赖项时，出现缺少程序包的错误。	输入下列命令之一，解决程序包缺失问题。 `$npm ci`

事务	解决方案
	或 `$npm install -g @aws-cdk/<package_name>`
在虚拟机上运行 `aws ssm create-activation` 命令时，您将收到以下错误。 `An error occurred (ValidationException) when calling the CreateActivation operation: Nonexistent role or missing ssm service principal in trust policy: arn:aws:iam::000000000000:role/EcsAnywhereInstanceRole`	`EcsAnywhereInfraStack` 堆栈尚未完全部署，运行此命令所需 IAM 角色也尚未创建。在 CloudFormation 控制台中检查堆栈状态。状态变为 `CREATE_COMPLETE` 后重试该命令。

Amazon ECS 运行状况检查返回 `UNHEALTHY` ，您会在 Amazon ECS 控制台的集群的服务部分看到以下错误。
 	
`service EcsAnywhereService was unable to place a task because no container instance met all of its requirements. Reason: No Container Instances were found in your cluster.`
	输入以下命令，在 Vagrant 虚拟机重新启动 Amazon ECS 代理。
	
	`$vagrant ssh`
	`$sudo systemctl restart ecs`
	`$sudo systemctl status ecs`

相关资源

- Amazon ECS Anywhere 营销页面
- Amazon ECS Anywhere 文档
- Amazon ECS Anywhere 演示（视频）
- 亚马逊 ECS Anywhere 研讨会示例 (GitHub)
- 存储库镜像（GitLab 文档）

更多模式

- 使用 AWS Transit Gateway 自动设置区域间对等互连
- 通过使用 AWS CDK 设置 Amazon ECS Anywhere 来管理本地容器应用程序
- 使用迁移器将 Hadoop 数据迁移到 Amazon S3 WANdisco LiveData
- 使用 Pow VMware VMs erCLI 通过 HCX 自动化进行迁移
- 使用 VMware HCX 将工作负载迁移到 AWS 上的 VMware 云端
- 在 AWS 上从 F5 迁移到应用程序负载均衡器时修改 HTTP 标头
- 在 Amazon Web Services Cloud 中重新托管本地工作负载：迁移核对清单
- 使用 BMC Discovery 查询提取迁移数据以进行迁移规划

管理与治理

主题

- 在 Amazon Data Firehose 资源未使用密钥加密时进行识别并发出警报 AWS KMS
- 通过 AWS Systems Manager 自动添加或更新 Windows 注册表项
- 使用 Python 在 AMS 中自动创建 RFC
- 使用 AWS Systems Manager 维护窗口自动停止和启动 Amazon RDS 数据库实例
- 使用 Terraform 在 AWS Organizations 中集中分发软件包
- 使用在 Amazon Logs 中为.NET 应用程序配置 CloudWatch 日志记录 NLog
- 跨不同 Amazon Web Services account 和 AWS 区域 复制 AWS Service Catalog 产品
- 为云运营模式创建 RACI 或 RASCI 矩阵
- 使用 Amazon CloudWatch 异常检测为自定义指标创建警报
- 创建使用具有默认加密的 Amazon EBS 卷的 AWS Cloud9 IDE
- 自动创建基于标签的 Amazon CloudWatch 控制面板
- 记录您的 AWS 着陆区设计
- 通过在多个 AWS 区域、账户中启用 Amazon DevOps Guru 并 OUs 使用 AWS CDK 来提高运营绩效
- 使用引导管道实现 Account Factory for Terraform (AFT)
- 管理多个 Amazon Web Services account 和 AWS 区域中的 AWS Service Catalog 产品
- 将 AWS 成员账户从 AWS Organizations 迁移至 AWS Control Tower
- 使用 AWS 服务监控 SAP RHEL Pacemaker 集群
- 监控多个共享的 Amazon 系统映像的使用情况 AWS 账户
- 在 AWS Organizations 中设置程序账户关闭警报
- 查看您的 Amazon Web Services account 或组织 EBS 快照详情
- 更多模式

在 Amazon Data Firehose 资源未使用密钥加密时进行识别并发出警报 AWS KMS

创建者：Ram Kandaswamy (AWS)

摘要

为了合规起见，某些组织必须对数据传输资源（例如 Amazon Data Firehose）启用加密。此模式显示了一种在资源不合规时进行监控、检测和通知方法。

为了满足加密要求，可以使用此模式自动监控和检测未使用 AWS Key Management Service (AWS KMS) 密钥加密的 Amazon Data Firehose 交付资源。 AWS 该解决方案会发送警报通知，并且可对其进行扩展以执行自动修复。此解决方案可以应用于个人账户或多账户环境，例如使用 landing zone 的环境 AWS 或。 AWS Control Tower

先决条件和限制

先决条件

- Amazon Data Firehose 传输流
- 足够的权限和熟悉程度 AWS CloudFormation，用于此基础架构自动化

限制

- 该解决方案不是实时的，因为它使用 AWS CloudTrail 事件进行检测，并且在创建未加密的资源和发送通知之间存在延迟。

架构

目标技术堆栈

此解决方案使用无服务器技术和以下服务：

- AWS CloudTrail
- Amazon CloudWatch
- AWS Command Line Interface (AWS CLI)
- AWS Identity and Access Management (IAM)

- Amazon Data Firehose
- AWS Lambda
- Amazon Simple Notification Service (Amazon SNS)

目标架构

该图说明了以下步骤：

1. 用户创建或修改亚马逊数据 Firehose。
2. 检测并匹配 CloudTrail 事件。
3. 调用了 AWS Lambda。
4. 已识别出不合规的资源。
5. 发送邮件通知。

自动化和扩缩

您可以使用通过单个命令 AWS CloudFormation StackSets 将此解决方案应用于多个 AWS 区域 或帐户。

工具

- AWS CloudTrail AWS 服务 可帮助您实现治理、合规以及运营和风险审计 AWS 账户。用户、角色或执行的操作将作为事件记录 AWS 服务 在中 CloudTrail。事件包括在 AWS Management Console、 AWS CLI AWS SDKs、和 API 操作中执行的操作。
- Amazon CloudWatch Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，允许您使用命令行 shell 中的命令进行交互。 AWS 服务
- AWS Identity and Access Management (IAM) 是一项 Web 服务，可帮助您安全地控制对 AWS 资源的访问。可以使用 IAM 来控制谁通过了身份验证（准许登录）并获得授权（具有相应权限）来使用资源。
- Amazon Data Firehos e 是一项完全托管的服务，用于提供实时流数据。使用 Firehose，您无需编写应用程序或管理资源。您可以配置数据生成工具向 Firehose 发送数据，然后 Firehose 会将数据自动传输到您指定的目标。

- AWS Lambda是一项计算服务，支持在不预置或管理服务器的情况下运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。您只需为所消耗的计算时间付费，当您的代码未运行时，不收取任何费用。
- 亚马逊简单通知服务 (Amazon SNS) Simple Not ification Service 是一项托管服务，可提供从发布者到订阅者（也称为制作者和消费者）的消息传送服务。

操作说明

强制加密以实现合规

Task	描述	所需技能
部署 AWS CloudFormation StackSets。	在中 AWS CLI，使用`firehose-encryption-checker.yaml` 模板（附件）通过运行以下命令来创建堆栈集。 为参数提供有效 Amazon SNS 主题 Amazon 资源名称（ARN）。部署应成功创建 CloudWatch 事件规则、Lambda 函数和具有必要权限的 IAM 角色，如模板中所述。 `aws cloudformation create-stack-set --stack-set-name my-stack-set --template-body file://firehose-encryption-checker.yaml`	云架构师、系统管理员
创建堆栈实例。	可以在您选择 AWS 区域 的账户中创建堆栈，也可以在一个或多个账户中创建堆栈。 要创建堆栈实例，请运行以下命令。将堆栈名称、账号和区域	云架构师、系统管理员

Task	描述	所需技能
	替换为您自己的堆栈名称、账号和区域。 `aws cloudformation create-stack-instances --stack-set-name my-stack-set --accounts 123456789012 223456789012 --regions us-east-1 us-east-2 us-west-1 us-west-2 --operation-preferences FailureToleranceCount=1`	

相关资源

- 与 AWS CloudFormation StackSets
- 什么是 Amazon CloudWatch 活动？

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

通过 AWS Systems Manager 自动添加或更新 Windows 注册表项

由 Appasaheb Bagali (AWS) 创建

摘要

AWS Systems Manager 是一款适用于亚马逊弹性计算云 (亚马逊 EC2) 实例的远程管理工具。Systems Manager 可让您查看和控制您在Amazon Web Services 上的基础设施。此多功能工具可以用于修复被安全漏洞扫描报告识别为漏洞的 Windows 注册表更改。

此模式涵盖了通过自动更改注册表来保护运行 Windows 操作系统的 EC2 实例安全的步骤，这些更改是为了您的环境安全而建议的。该模式使用“运行”命令运行 Command 文档。该代码已附上，其中一部分包含在代码部分中。

先决条件和限制

- 一个有效的 Amazon Web Services account
- 访问 EC2 实例和 Systems Manager 的权限

架构

目标技术堆栈

- 具有两个子网和一个网络地址转换 (NAT) 网关的虚拟私有云（VPC）
- 用于添加或者更新注册表名称和值的 Systems Manager Command 文档
- Systems Manager 运行命令在指定 EC2 实例上运行命令文档

目标架构

工具

工具

- IAM policy 和角色 – AWS Identity and Access Management (IAM) 是一项 Web 服务，可帮助您安全地控制对 AWS 资源的访问。可以使用 IAM 来控制谁通过了身份验证（准许登录）并获得授权（具有相应权限）来使用资源。

- Amazon Simple Storage Service – Amazon Simple Storage Service (Amazon S3) 是一项面向互联网的存储服务。该服务旨在降低开发人员进行网络规模级计算的难度。在此模式中，S3 存储桶用于存储 Systems Manager 日志。
- AWS Systems Manager – AWS Systems Manager 是一项 Amazon Web Services，可用于查看和控制 AWS 上的基础设施。Systems Manager 通过扫描托管实例并报告其检测到的任何策略违规行为（或采取纠正措施）来帮助您维护安全性和合规性。
- AWS Systems Manager Command 文档 – 通过 Run Command 使用的 AWS Systems Manager Command 文档。大多数命令文档在所有 Systems Manager 所支持的 Linux 和 Windows Server 操作系统上受支持。
- AWS Systems Manager Run Command – AWS Systems Manager Run Command 您提供了一种远程安全地管理托管实例配置的方法。利用 Run Command，您可以自动完成常用管理任务以及大规模执行一次性配置更改。

代码

你可以使用以下示例代码来添加或更新 Microsoft Windows 注册表名称 `Version`、注册表路径 `HKCU:\Software\ScriptingGuys\Scripts` 和值 `2`。

```
#Windows registry path which needs to add/update
$registryPath ='HKCU:\\Software\\ScriptingGuys\\Scripts'
#Windows registry Name  which needs to add/update
$Name = 'Version'
#Windows registry value  which needs to add/update
$value = 2
# Test-Path cmdlet to see if the registry key exists.
IF(!(Test-Path $registryPath))
        {
           New-Item -Path $registryPath -Force | Out-Null
           New-ItemProperty -Path $registryPath -Name $name -Value     $value ` -
PropertyType DWORD -                 Force | Out-        Null
        } ELSE {
                      New-ItemProperty -Path $registryPath -Name $name -Value $value `
 -PropertyType            DWORD        -Force | Out-Null
            }
echo 'Registry Path:'$registryPath
 echo 'Registry Name:'$registryPath
 echo 'Registry Value:'(Get-ItemProperty -Path $registryPath -Name $Name).version
```

随函附上完整的 Systems Manager 命令文档 JavaScript 对象表示法 (JSON) 代码示例。

操作说明

设置 VPC

Task	描述	所需技能
创建 VPC。	在 AWS 管理控制台，创建具有公有和私有子网以及一个 NAT 网关的 VPC。有关更多信息，请参阅 AWS 文档。	云管理员
创建安全组。	确保每个安全组都允许远程桌面协议 (RDP) 从源 IP 地址访问。	云管理员

创建 IAM policy 和 IAM 角色

Task	描述	所需技能
创建一个 IAM 策略。	创建一个 IAM 策略，提供对 Amazon S3、Amazon 和 System EC2 s Manager 的访问权限。	云管理员
创建一个 IAM 角色。	创建一个 IAM 角色，并附加允许访问 Amazon S3、Amazon 和 Systems Manager 的 IA EC2 M 策略。	云管理员

运行自动化

Task	描述	所需技能
创建 Systems Manager 命令文档。	创建 Systems Manager Command 文档，用于部署	云管理员

Task	描述	所需技能
	要添加或更新的 Microsoft Windows 注册表更改。	
运行 Systems Manager 运行命令。	运行 Systems Manager Run Command，选择 Command 文档和 Systems Manager 目标实例。这会将所选 Command 文档中的 Microsoft Windows 注册表更改推送至目标实例。	云管理员

相关资源

- AWS Systems Manager
- AWS Systems Manager 文档
- AWS Systems Manager 运行命令

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 Python 在 AMS 中自动创建 RFC

由 Gnanasekaran Kailasam (AWS) 创建

摘要

AWS Managed Services (AMS) 通过持续管理您的 Amazon Web Services (AWS) 基础设施，帮助您更高效、更安全地运营基于云的基础设施。若要对托管环境进行更改，您需要创建并提交新的变更请求 (RFC)，其中包含特定操作或操作的更改类型 (CT) ID。

但是，手动创建 RFC 可能需要大约五分钟，您组织中的团队可能需要 RFCs 每天提交多个 RFC。此模式可帮助您自动执行 RFC 创建进程，缩短每个 RFC 的创建时间，并消除手动错误。

此模式描述了如何使用 Python 代码自动创建 `Stop EC2 instance` RFC，从而停止您的 AMS 账户中的亚马逊弹性计算云 (Amazon EC2) 实例。然后，您可以将此模式的方法和 Python 自动应用于其他类型的 RFC 。

先决条件和限制

先决条件

- AMS 高级账户。有关这方面的更多信息，请参阅 AWS Managed Services 文档中的 AMS 运营计划。
- 您的 AMS 账户中至少有一个现有 EC2 实例。
- 了解如何在 AMS RFCs 中创建和提交。
- 熟悉 Python。

限制

- 您只能 RFCs 用于更改您的 AMS 账户。您的 Amazon Web Services Account 使用不同过程执行类似更改。

架构

技术堆栈

- AMS
- AWS 命令行界面（AWS CLI）

- 适用于 Python 的 Amazon SDK (Boto3)
- Python 及其必需软件包 (JSON 和 Boto3)

自动化和扩缩

此模式提供了自动化 `Stop EC2 instance` RFC 的示例代码，但您可以将此模式的示例代码和方法用于其他 RFCs模式。

工具

- AWS Managed Services – AMS 可帮助您更高效、更安全地运营 AWS 基础设施。
- AWS CLI – AWS 命令行界面（AWS CLI）是一款统一工具，可用于管理 Amazon Web Services。在 AMS 中，变更管理 API 提供用于创建和管理的操作 RFCs。
- 适用于 Python 的 Amazon SDK (Boto3) – 适用于 Python 的开发工具包可以轻松集成 Python 应用程序、库或脚本与 Amazon Web Services。

代码

`AMS Stop EC2 Instance.zip` 文件（附件）包含用于创建 `Stop EC2 instance` RFC 的 Python 代码。您也可以将此代码配置为为多个 EC2 实例提交单个 RFC。

操作说明

选项 1 – 设置适用于 macOS 或 Linux 的环境

Task	描述	所需技能
安装并验证 Python。	1. 打开终端窗口并运行 `brew install python3` 命令。 2. 通过运行 `python --version` 命令验证 Python 是否已正确安装。 3. 通过运行 `pip --version` 命令验证 `pip` 是否已正确安装。	AWS 系统管理员

Task	描述	所需技能
安装 AWS CLI。	运行 `pip install awscli --upgrade –user` 命令，以安装 AWS CLI。	AWS 系统管理员
安装 Boto3。	运行 `pip install boto3` 命令，以安装 Boto3。	AWS 系统管理员
安装 JSON。	运行 `pip install json` 命令，以安装 JSON。	AWS 系统管理员
设置 AMS CLI。	登录 AWS 管理控制台，打开 AMS 控制台，然后选择文档。下载包含 AMS CLI 的 .zip 文件，将其解压缩，然后将其安装至本地计算机。 在安装 AMS CLI 后，运行 `aws amscm help` 命令。输出提供了 AMS 变更管理流程相关信息。	AWS 系统管理员

选项 2 - 设置适用于 Windows 的环境

Task	描述	所需技能
安装并验证 Python。	1. 打开适用 Windows 的 Python 版本页面，下载最新版本，然后安装 Python。 2. 通过运行 `python --version` 命令验证 Python 是否已正确安装。 3. 通过运行 `pip --version` 命令验证 pip 是否已正确安装。	AWS 系统管理员

Task	描述	所需技能
安装 AWS CLI。	运行 `pip install awscli --upgrade –user` 命令，以安装 AWS CLI。	AWS 系统管理员
安装 Boto3。	运行 `pip install boto3` 命令，以安装 Boto3。	AWS 系统管理员
安装 JSON。	运行 `pip install json` 命令，以安装 JSON。	AWS 系统管理员
设置 AMS CLI。	登录 AWS 管理控制台，打开 AMS 控制台，然后选择文档。下载包含 AMS CLI 的 .zip 文件，将其解压缩，然后将其安装至本地计算机。 在安装 AMS CLI 后，运行 `aws amscm help` 命令。输出提供了 AMS 变更管理流程相关信息	AWS 系统管理员

提取适用于 RFC 的 CT ID 与执行参数

Task	描述	所需技能
提取适用于 RFC 的 CT ID、版本与执行参数。	每个 RFC 都有不同的 CT ID、版本以及执行参数。您可以使用以下选项之一提取此信息： 1. 按照 AWS Managed Services 文档中的 RFC 使用示例的通过 CLI 查找变更请求 (RFC)部分的说明。 2. 通过 AMS 控制台，打开一个类似类型的现有 RFC 或	AWS 系统管理员

Task	描述	所需技能
	创建新的 RFC 来测试。使用 RFC 的 CT ID 与执行参数。有关这方面的更多信息，请参阅 AWS Managed Services 文档中的使用控制台查找 RFC。 **Note** 要使此模式的 Python 自动化适应其他模式 RFCs，请将文件（附件）中的 `ams_stop_ec2_instance` Python 代码`AMS Stop EC2 Instance.zip` 文件中的 CT 类型和参数值替换为您提取的值。	

运行 Python 自动化

Task	描述	所需技能
运行 Python 自动化。	1. 将 `AMS Stop EC2 Instance.zip` 文件（附件）下载至本地计算机并将其解压缩。 2. `input_instances` 使用您的 EC2 实例信息进行更新。 3. 打开终端并导航至代码提取路径	AWS 系统管理员

Task	描述	所需技能
	4. 运行 `pythonams_stop_ec2_instance.py` 命令。	

相关资源

- 什么是变更类型？
- CLI 教程：高可用性双层堆栈 (Linux/RHEL)

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS Systems Manager 维护窗口自动停止和启动 Amazon RDS 数据库实例

由 Ashita Dsilva (AWS) 创建

摘要

此模式演示如何使用 AWS Systems Manager 维护时段按特定计划自动停止和启动 Amazon Relational Database Service (Amazon RDS) 数据库实例（例如，在工作时间以外关闭数据库实例以降低成本）。

AWS Systems Manager 自动化提供了`AWS-StopRdsInstance`和`AWS-StartRdsInstance`运行手册，用于停止和启动 Amazon RDS 数据库实例。这意味着您无需使用 AWS Lambda 函数编写自定义逻辑或创建 Amazon Ev CloudWatch ents 规则。

Systems Manager 提供两种调度任务的功能：状态管理器和维护窗口。State Manager 一次性或按特定计划设置和维护您的 Amazon Web Services (AWS) 账户中资源所需的状态配置。Maintenance Windows 在特定时间段内对账户资源运行任务。尽管您可以将此模式的方法用于状态管理器或维护窗口，但我们建议您使用维护窗口，因为它可以根据分配的优先级运行一个或多个任务，也可以运行 AWS Lambda 功能和 AWS Step Functions 任务。有关状态管理器和维护窗口的更多信息，请参阅 Systems Manager 文档中的在状态管理器和维护窗口之间进行选择。

此模式为配置两个使用 cron 表达式停止并启动 Amazon RDS 数据库实例的单独维护时段提供了详细步骤。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- 您想要按特定计划停止和启动的现有 Amazon RDS 数据库实例。
- 适用于所需计划的 Cron 表达式。例如，表达式在每个星期一、星期二、星期三、星期四和星期五的 09:00 `cron(0 9 ? * MON-FRI *)` 运行任务。有关更多信息，请参阅 System s Manager 文档中的维护时段的 Cron 和费率表达式。
- 熟悉 Systems Manager。
- 启动和停止 RDS 实例的权限。欲了解更多信息，请参阅 Epics 部分。

限制

- 每次最多可以停用 7 天的 Amazon RDS 数据库实例。七天后，数据库实例将自动重启，以确保其收到所有必要维护更新。
- 您无法停止具有只读副本或作为只读副本的数据库实例。
- 您无法停止多可用区配置中的 Amazon RDS for SQL Server 数据库实例。
- 服务限额适用于 Maintenance Windows 和 Systems Manager Automation。有关服务配额的更多信息，请参阅 AWS 一般参考 文档中的AWS Systems Manager 终端节点和配额。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额页面，然后选择服务的链接。

架构

下图显示了自动停止和启动 Amazon RDS 数据库实例的工作流程。

工作流程由以下步骤组成：

1. 创建维护时段，并使用 cron 表达式定义 Amazon RDS 数据库实例的停止和启动计划。

2. 使用 `AWS-StopRdsInstance` 或 `AWS-StartRdsInstance` 运行手册将 Systems Manager 自动化任务注册至维护时段。

3. 使用基于标签的资源组为您的 Amazon RDS 数据库实例注册带维护时段的目标。

技术堆栈

- AWS CloudFormation
- AWS Identity and Access Management (IAM)
- Amazon RDS
- Systems Manager

自动化和扩缩

您可以同时停止和启动多个 Amazon RDS 数据库实例，方法是标记所需 Amazon RDS 数据库实例，创建包含所有已标记数据库实例的资源组，并将此资源组注册为维护时段目标。

工具

- AWS CloudFormation是一项可帮助您建模和设置 AWS 资源的服务。

- AWS Identity and Access Management (IAM) 是一项 Web 服务，可帮助您安全地控制对 AWS 资源的访问。
- Amazon Relational Database Service (Amazon RDS) 是一项网络服务，可以更轻松地在中设置、操作和扩展关系数据库 AWS Cloud。
- AWS Resource Groups可帮助您将 AWS 资源分组、标记资源以及管理、监控和自动执行分组资源上的任务。
- AWS Systems Manager可以 AWS 服务 用来查看和控制您的基础架构 AWS。此模式使用 Systems Manager 的以下功能：
  - AWS Systems Manager 自动化简化了亚马逊弹性计算云 (Amazon EC2) 实例和其他 AWS 资源的常见维护和部署任务。
  - AWS Systems Manager 维护窗口可帮助您定义何时对实例执行可能造成中断的操作的时间表。

操作说明

为 Systems Manager Automation 创建并配置 IAM 服务角色

Task	描述	所需技能
为 Systems Manager Automation 配置 IAM 服务角色。	登录 AWS Management Console 并创建 Systems Manager 自动化的服务角色。您可以使用以下两种方法之一创建此服务角色： • 用于 AWS CloudFormation 为 Systems Manager 自动化配置服务角色 • 使用 IAM 为 Systems Manager Automation 配置角色 Systems Manager Automation 工作流程通过使用服务角色在 Amazon RDS 数据库实例上执	AWS 管理员

Task	描述	所需技能
	行启停操作，以调用 Amazon RDS。 必须使用以下内联策略配置服务角色，该策略有权启动和停止 Amazon RDS 数据库实例： `{ "Version": "2012-10-17", "Statement": [ { "Sid": "RdsStartStop", "Effect": "Allow", "Action": [ "rds:StopDBInstance", "rds:StartDBInstance" ], "Resource": "<RDS_Instance_ARN>" }, { "Sid": "RdsDescribe", "Effect": "Allow", "Action": "rds:DescribeDBInstances", "Resource": "*" } ] }`	

Task	描述	所需技能
	请务必<RDS_Instance_ARN> 替换为 Amazon RDS 数据库实例的亚马逊资源名称 (ARN)。 如果您不熟悉 IAM 策略和角色的使用，请按照计划 Amazon RDS 停止并开始使用 AWS Systems Manager博客文章的“解决方案概述” 部分中的说明进行操作。 **Important** 请务必记录服务角色的 ARN。	

创建资源组

Task	描述	所需技能
标记 Amazon RDS 数据库实例。	打开 Amazon RDS 控制台并标记要添加至资源组的 Amazon RDS 数据库实例。标签是分配给 AWS 资源的元数据，由键值对组成。我们建议您使用 Acti on StartStop作为标签键和值。 有关这方面的更多信息，请参阅 Amazon RDS 文档中的添加、列出和移除标签。	AWS 管理员
为您标记的 Amazon RDS 数据库实例创建资源组。	打开AWS Resource Groups 控制台，根据您为 Amazon RDS	AWS 管理员

Task	描述	所需技能
	数据库实例创建的标签创建资源组。 在分组条件下，确保DBInstance为资源类型选择 AWS::RDS::，然后提供标签的键值对（例如，“Action-StartStop”）。这可以确保该服务仅检查 Amazon RDS 数据库实例，而不检查带此标签的其他资源。请确保记录资源组的名称。 有关更多信息和详细步骤，请参阅 AWS Resource Groups 文档中的生成基于标签的查询并创建群组。	

配置维护时段，以停止 Amazon RDS 数据库实例

Task	描述	所需技能
创建维护时段。	1. 打开 Systems Manager 控制台，选择维护窗口，然后选择创建维护窗口。为维护时段提供名称（例如 StopRdsInstance “”），输入描述，然后取消选中 “允许未注册的目标”。 2. 选择 Cron/rate 表达式，并提供计划表达式以定义何时应停止 Amazon RDS 数据库实例。在持续时间 输入 1，在停止启动任务中输入 0。默认情况下，时区显示为 UTC。您可根据 cron 表	AWS 管理员

Task	描述	所需技能
	达式中定义的时间戳更改时区，以启动维护时段。 3. 选择 Create maintenance window。系统会返回至维护时段页面，且维护时段状态为已启用。 **Important** 停止数据库实例的任务在启动时几乎立即运行，并且不会跨越整个维护时段。此模式提供了持续时间和停止启动任务的最小值，因其为维护时段的必需参数。 有关更多信息和详细步骤，请参阅 Systems Manager 文档中的创建维护窗口（控制台）。	

Task	描述	所需技能
为维护时段分配目标。	1. 在 Systems Manager 控制台上，选择维护窗口，选择操作，然后选择注册目标。 2. 在目标区域中，指定选择资源组，然后选择账户中现有资源组的名称。 3. 对于资源类型，选择 AWS::RDS::，DBInstance然后选择注册目标。 有关更多信息和详细步骤，请参阅 Systems Manager 文档中的为维护时段（控制台）分配目标。	AWS 管理员

Task	描述	所需技能
为维护时段分配任务。	1. 在 Systems Manager 控制台上，选择维护窗口，然后选择您的维护时段。选择“操作”，然后选择“注册自动化”任务。 2. 对于文档，请选择 AWS-StopRdsInstance。 3. 在“目标”部分，选择“选择注册的目标组”，然后选择在当前维护时段中注册的维护时段目标。 4. 对于速率控制，请 指定 并发和错误阈值为百分之百。 您可以根据任务并发和错误阈值要求更改速率控制值。有关这方面的更多信息，请参阅 Systems Manager 文档中的关于并发和错误阈值。 5. 在 IAM 服务角色部分中，对于服务角色，将此框留空或创建自己的自定义角色。如果将该框留空，Systems Manager 会自动创建服务相关角色 AWSServiceRoleForAmazonSSM，然后将该角色与任务关联起来。要创建自己的自定义角色，请参阅为维护窗口（控制台）创建自定义服务角色，然后将该自定义角色与任务关联。	AWS 管理员

Task	描述	所需技能
	6. 在输入参数部分中，为运行手册指定以下参数： • InstanceId: `{{RESOURCE_ID}}` **Note** 对于 InstanceId，使用虚拟参数从 ARN 中提取 Amazon RDS 数据库资源 ID。要了解有关伪参数的更多信息，请参阅 Systems Manager 文档中的关于伪参数。 • AutomationAssumeRole：提供您为 Systems Manager Automation 创建的服务角色的 ARN。 7. 选择注册自动化任务。 **Important** 服务角色选项定义了维护时段运行任务所需的服务角色。但是，该角色不同于您之前为 Systems Manager Automation 创建的服务角色。	

Task	描述	所需技能
	有关更多信息和详细步骤，请参阅 Systems Manager 文档中的向维护窗口（控制台）分配任务。	

配置维护时段，以启动 Amazon RDS 数据库实例

Task	描述	所需技能
配置维护时段，以启动 Amazon RDS 数据库实例。	重复配置维护时段以停止 Amazon RDS 数据库实例操作说明的步骤，以配置其他维护时段，定期启动 Amazon RDS 数据库实例。 **Important** 在配置维护时段以启动数据库实例时，必须进行以下更改： • 为维护时段使用新名称（例如，StartRdsInstance“”）。 • 将 cron 表达式替换为用于启动数据库实例的 cron 表达式。 • 将 `AWS-StopRdsInstance` 运行手册替换为任务中的 `AWS-StartRdsInstance` 。	AWS 管理员

相关资源

- 使用 Systems Manager 自动化文档来管理实例并削减非工作时间成本（AWS 博客文章）

使用 Terraform 在 AWS Organizations 中集中分发软件包

由 Pradip kumar Pandey (AWS)、Aarti Rajput (AWS)、Chintamani Aphale (AWS)、T.V.R.L.Phani Kumar Dadi (AWS)、Mayuri Shinde (AWS) 和 Pratap Kumar Nanda (AWS) 创作

摘要

为了 AWS 区域 在工作负载之间建立强大的隔离屏障 AWS 账户 ，企业通常会维护多个分散在多个工作负载中的多个分布。为了保持安全和合规，他们的管理团队安装了基于代理的工具，例如CrowdStrikeSentinelOne、或用于安全扫描的TrendMicro工具，以及用于监控的 Amazon CloudWatch 代理、Datadog Agen t 或代AppDynamics 理。当这些团队想要在这个广阔的环境中自动化软件包管理和分发时，他们通常会面临挑战。

Dist@@ ri butor 是一项功能 AWS Systems Manager，它通过一个简化的界面，自动将软件打包和发布到云端和本地服务器上的托管 Microsoft Windows 和 Linux 实例。此模式演示了如何使用 Terraform 进一步简化管理软件安装的过程，并以最少的努力在大量实例和成员帐户中 AWS Organizations 运行脚本。

此解决方案适用于由 Systems Manager 管理的亚马逊、Linux 和 Windows 实例。

先决条件和限制

- 包含要安装的软件的分销商软件包
- Terraform 版本 0.15.0 或更高版本
- 由 Systems Manager 管理并具有访问目标账户中的亚马逊简单存储服务 (Amazon S3 EC2) 的基本权限的亚马逊弹性计算云 (Amazon S3) 实例
- 为您的组织设置的着陆区，该着陆区是使用以下方法设置的 AWS Control Tower
- （可选）适用于 Terraform 的 Account Factory (AFT)

架构

资源详细信息

此模式使用 Account F actory for Terraform (AFT) 创建所有必需的 AWS 资源，并使用代码管道在部署账户中部署资源。代码管道在两个存储库中运行：

- 全局自定义包含 Terraform 代码，该代码将在所有在 AFT 注册的账户中运行。

- 账户自定义包含将在部署账户中运行的 Terraform 代码。

您也可以在不使用 AFT 的情况下部署此解决方案，方法是在账户自定义文件夹中运行 Terraform 命令。

Terraform 代码部署了以下资源：

- AWS Identity and Access Management (IAM) 角色和策略
  - SystemsManager-AutomationExecutionRole 授予用户在目标账户中运行自动化的权限。
  - SystemsManager-AutomationAdministrationRole 授予用户在多个账户和组织单位中运行自动化的权限（OUs）。
- 压缩文件和软件包的 manifest.json
  - 在 Systems Manager 中，软件包至少包含一个包含软件或可安装资产的.zip 文件。
  - JSON 清单包含指向您的软件包代码文件的指针。
- S3 存储桶
  - 跨组织共享的分布式包安全地存储在 Amazon S3 存储桶中。
- AWS Systems Manager 文档（SSM 文档）
  - `DistributeSoftwarePackage`包含将软件包分发到成员账户中每个目标实例的逻辑。
  - `AddSoftwarePackageToDistributor`包含打包可安装软件资产并将其添加到 Automation 的逻辑，该功能为 AWS Systems Manager。
- Systems Manager 关联
  - Systems Manager 关联用于部署解决方案。

架构和工作流程

下图说明了以下步骤：

1. 要从集中式账户运行解决方案，您需要将软件包或软件以及部署步骤上传到 S3 存储桶。
2. 您的自定义包将显示在 Systems Manager 控制台的 “文档” 部分的 “我所有” 选项卡中。
3. State Manager 是 Systems Manager 的一项功能，用于在整个组织中创建、安排和运行软件包的关联。该关联规定，必须先在托管节点上安装并运行软件包，然后才能将其安装到目标节点上。
4. 该关联指示 Systems Manager 在目标节点上安装软件包。

5. 对于后续的任何安装或更改，用户可以定期运行相同的关联，也可以从单个位置手动运行相同的关联，以跨账户执行部署。
6. 在成员账户中，Automation 会向分销商发送部署命令。
7. 分销商跨实例分发软件包。

此解决方案使用其中的管理帐户 AWS Organizations，但您也可以指定一个帐户（委托管理员）来代表组织对其进行管理。

工具

Amazon Web Services

- Amazon Simple Storage Service(Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。这种模式使用 Amazon S3 来集中和安全地存储分发的软件包。
- AWS Systems Manager 可帮助您管理在 AWS Cloud中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。此模式使用以下 Systems Manager 功能：
  - Dist@@ ri butor 可以帮助您将软件打包并发布到 Systems Manager 托管实例。
  - 自动化简化了许多 AWS 服务的常见维护、部署和补救任务。
  - 文档在您的组织和账户中对您的 Systems Manager 托管实例执行操作。
- AWS Organizations是一项账户管理服务，可帮助您将多个 AWS 账户整合到一个由您创建和集中管理的组织中。

其他工具

- Terraform 是一款基础设施即代码 (IaC) 工具 HashiCorp ，可帮助您创建和管理云和本地资源。

代码存储库

此模式的说明和代码可在 GitHub 集中式包分发存储库中找到。

最佳实践

- 要为关联分配标签，请使用 AWS Command Line Interface (AWS CLI) 或AWS Tools for PowerShell。不支持使用 Systems Manager 控制台将标签添加到关联。有关更多信息，请参阅 Systems Manager 文档中的为 Systems Manager 资源添加标签。

- 要使用从其他账户共享的新版本的文档来运行关联，请将文档版本设置为`default`。
- 要仅标记目标节点，请使用一个标签密钥。如果要使用多个标签键来定位节点，请使用资源组选项。

操作说明

配置源文件和帐户

Task	描述	所需技能

克隆存储库。	1. 克隆 GitHub 集中式软件包分发存储库：
	`git clone https://github.com/aws-samples/aws-organization-centralised-package-distribution`
	2. Terraform 代码存储库需要两个由 AFT 管理的自定义文件夹。确认存储库的本地副本包含以下文件夹：
	`$ cd centralised-package-distribution`
	`$ ls`
	`global-customization account-customization`
		DevOps 工程师

更新全局变量。	更新`global-customization/variables.tf` 文件中的以下输入参数。这些变量适用于由 AFT 创建和管理的所有账户。 • `account_id` ：将在其中部署分销商解决方案的账户的 ID。	DevOps 工程师

Task	描述	所需技能
	• aws_region ：协会将部署到 AWS 区域 哪里。	
更新账户变量。	更新account-customization/variables.tf 文件中的以下输入参数。这些变量仅适用于由 AFT 创建和管理的特定账户。 • package_bucket_name ：包含软件包分发文件的 S3 存储桶的名称。 • package_name ：软件包分发文件的名称。 • package_version ：安装程序的软件包版本。	DevOps 工程师

自定义参数和部署文件

Task	描述	所需技能
更新状态管理器关联的输入参数。	更新account-customization/association.tf 文件中的以下输入参数以定义要在实例上保持的状态。如果默认参数值支持您的用例，则可以使用它们。 • targetAccounts ：AWS Organizati IDs ons 中的组织单位 (OU)，代表拥有要分配的目标实例的账户。OU 以 “ou” IDs 开头。 • targetRegions ：目标实例正在 AWS 区域 运行的	DevOps 工程师

Task	描述	所需技能

	（例如 “us-east-1” 或 “ap-southeast-2”）。
	• `action`：指定是安装还是卸载软件包。
	• `installationType` ：以下安装类型之一：
	  • `uninstall` ：软件包已卸载。
	  • `reinstall` ：在重新安装过程完成之前，应用程序将处于离线状态。
	  • `In-place update`：在安装中添加新的或更新的文件时，应用程序可用。
	• `name`：要安装或卸载的软件包的名称。
	• `version`：要安装或卸载的软件包的版本。如果未安装任何版本的软件包，系统将返回错误。
	• `bucketName` ：软件包已部署到的 S3 存储桶名称。此存储桶应仅包含软件包和清单文件。
	• `bucketPrefix` ：存储包资产的 S3 前缀。
	• `AutomationAssumeRole` ：的亚马逊资源名称 (ARN)。`SystemsManager-AutomationAdministrationRole`	

Task	描述	所需技能
准备压缩文件和软件包的`manifest.json` 文件。	此模式在文件夹中提供了带有 PowerShell 安装和卸载脚本的可安装文件示例（Windows 为.msi，Linux 为.rpm）。`account-customization/package` 1. 用您自己的文件替换 PowerShell 可安装文件，或者提供可安装文件、安装和卸载脚本以及清单文件，以便在您账户的`account-customization` 文件夹中创建软件包。 2. 根据您的要求自定义 Terraform 在`account-customization` 文件夹中生成的默认`manifest.json` 文件。	DevOps 工程师

运行 Terraform 命令来配置资源

Task	描述	所需技能

初始化 Terraform 配置。	要使用 AFT 自动部署解决方案，请将代码推送到 AWS CodeCommit：
	`$ git add *`
	`$ git commit -m "message"`
	`$ git push`
		DevOps 工程师

Task	描述	所需技能
	您也可以在不使用 AFT 的情况下通过从文件夹中运行 Terraform 命令来部署此解决方案。`account-customization` 要初始化包含 Terraform 文件的工作目录，请运行： `$ terraform init`	
预览更改。	要预览 Terraform 将对基础架构所做的更改，请运行以下命令： `$ terraform plan` 此命令评估 Terraform 配置，以确定已声明资源的所需状态。它还将所需状态与要在工作空间中配置的实际基础架构进行比较。	DevOps 工程师
应用更改。	运行以下命令以实现您对`variables.tf` 文件所做的更改： `$ terraform apply`	DevOps 工程师

验证资源

Task	描述	所需技能
验证 SSM 文档的创建。	1. 在 Systems Manager 控制台的左侧导航窗格中，选择“文档”。	DevOps 工程师

Task	描述	所需技能
	2. 选择 我拥有的选项卡。 您应该会看到`DistributeSoftwarePackage` 和`AddSoftwarePackageToDistributor` 软件包。	
验证自动化是否成功部署。	1. 在 Systems Manager 控制台的左侧导航窗格中，选择自动化。 2. 在自动化执行列表中，您应该看到最新的执行`DistributeSoftwarePackage` 和`AddSoftwarePackageToDistributor` 部署。 3. 选择 “执行 ID” 以验证他们是否成功完成。	DevOps 工程师
验证软件包是否已部署到目标成员账户实例。	1. 在 Systems Manager 控制台的导航窗格中，选择运行命令。 2. 在命令历史记录中，您将看到每次调用及其状态。 3. 选择任意命令 ID 以查看每个目标实例的部署历史记录。 4. 选择实例 ID，然后在 “输出” 部分查看分布。	DevOps 工程师

故障排除

事务	解决方案
状态经理关联失败或停留在待处理状态。	请参阅 AWS 知识中心中的疑难解答信息。
计划关联无法运行。	您的日程安排规格可能无效。状态管理器目前不支持在 cron 表达式中为关联指定月份。使用 cron 或费率表达式来确认日程安排。

相关资源

- 集中式软件包分发（GitHub 存储库）
- Account Factory for terraform (AFT)
- 用例和最佳实践（AWS Systems Manager 文档）

使用在 Amazon Logs 中为.NET 应用程序配置 CloudWatch 日志记录 NLog

创建者：Bibhuti Sahu（AWS）和 Rob Hill（AWS）（AWS）

摘要

此模式描述了如何使用 NLog 开源日志框架在 Amazon Log CloudWatch s 中记录.NET 应用程序的使用情况和事件。在 CloudWatch 控制台中，您可以近乎实时地查看应用程序的日志消息。您还可以设置指标并配置警报，以便在超过指标阈值时通知您。使用 A CloudWatch pplication Insights，您可以查看显示受监控应用程序潜在问题的自动或自定义仪表板。 CloudWatch Application Insights 旨在帮助您快速隔离应用程序和基础架构中持续存在的问题。

要将日志消息写入 CloudWatch 日志，请将`AWS.Logger.NLog` NuGet 软件包添加到.NET 项目中。然后，更新`NLog.config`文件以使用 CloudWatch 日志作为目标。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 一个 .NET Web 或控制台应用程序，它可以：
  - 使用支持的 .NET 框架或 .NET 核心版本。有关更多信息，请参阅产品版本。
  - 用于向 A NLog pplication Insights 发送日志数据。
- 为 Amazon Web Services 创建 IAM 角色的权限。有关更多信息，请参阅服务角色权限。
- 将角色传递给服务的权限。有关更多信息，请参阅向用户授予将角色传递给 Amazon Web Services 的权限。

产品版本

- .NET Framework 版本 3.5 或更高版本
- .NET Core 版本 1.0.1、2.0.0 或更高版本

架构

目标技术堆栈

- NLog

- Amazon CloudWatch 日志

目标架构

1. .NET 应用程序将日志数据写入 NLog 日志框架。
2. NLog 将日志数据写入 CloudWatch 日志。
3. 您可以使用 CloudWatch 警报和自定义仪表板来监控.NET 应用程序。

工具

Amazon Web Services

- Amazon App CloudWatch lication Insights 可帮助您观察应用程序和底层 AWS 资源的运行状况。
- Amazon CloudWatch Lo gs 可帮助您集中管理来自所有系统、应用程序和 AWS 服务的日志，以便您可以监控它们并安全地将其存档。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS 工具 PowerShell是一组 PowerShell 模块，可帮助您通过 PowerShell 命令行编写对 AWS 资源的操作的脚本。

其他工具

- 记录器。 NLog是将日志数据记录到 CloudWatch 日志的 NLog 目标。
- NLog是适用于.NET 平台的开源日志框架，可帮助您将日志数据写入目标，例如数据库、日志文件或控制台。
- PowerShell是一款在 Windows、Linux 和 macOS 上运行的微软自动化和配置管理程序。
- Visual Studio 是一个集成式开发环境（IDE），包括编译器、代码完成工具、图形设计器和其他支持软件开发的功能。

最佳实践

- 为目标日志组设置保留策略。这必须在 NLog 配置之外完成。默认情况下，日志数据无限期地存储在 CloudWatch 日志中。

- 坚持管理 AWS 访问密钥的最佳实践。

操作说明

设置访问权限和工具

Task	描述	所需技能
创建一个 IAM 策略。	按照 IAM 文档中使用 JSON 编辑器创建策略中的说明进行操作。输入以下 JSON 策略，该策略具有允许 CloudWatch 日志读取和写入日志所需的最低权限。 `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "logs:CreateLogGroup", "logs:CreateLogStream", "logs:GetLogEvents", "logs:PutLogEvents", "logs:DescribeLogGroups", "logs:DescribeLogStreams",`	AWS 管理员，AWS DevOps

Task	描述	所需技能

	<pre>"logs:PutRetention Policy"
	            ],
	            "Resource": [
	                "*"
	            ]
	        }
	    ]
	}</pre>	

创建一个 IAM 角色。	请按照 IAM 文档中的创建向 Amazon Web Services 委托权限的角色说明进行操作。选择您之前创建的策略。这是 Logs 在执行 CloudWatch 日志操作时扮演的角色。	AWS 管理员，AWS DevOps
为其设置 AWS 工具 PowerShell。	1. 按照安装适用的 AWS 工具中适用于您的操作系统的说明进行操作 PowerShell。 2. 使用适用于 PowerShell cmdlet 的 AWS 工具将您的访问密钥和密钥存储在配置文件中。有关说明，请参阅 AWS 工具中的管理配置 PowerShell 文件以获取文档。	常规 AWS

配置 NLog

Task	描述	所需技能
安装 NuGet 软件包。	1. 在 Visual Studio 中，选择文件，然后选择打开项目或解决方案。 2. 选择要安装的项目 NLog。 3. 在 Visual Studio 中，选择工具、NuGet 软件包管理器、软件包管理器控制台。 4. 输入以下命令安装`AWS.Logger.NLog` NuGet 软件包。 `Install-Package AWS.Logger.NLog -Version 3.1.0`	应用程序开发人员
配置日志目标。	1. 打开 `NLog.config` 文件。 2. 对于目标 `type`，输入 `AWSTarget` 。 3. 对于目标 `logGroup`，输入要使用的日志组的名称。如果该日志组尚不存在，则会自动创建一个具有所提供名称的新日志组。 4. 对于目标`region`，请输入配置 CloudWatch 日志的 AWS 区域。 5. 对于目标 `profile`，输入您之前创建的用于存储访问密钥和私有密钥的配置文件的名称。	应用程序开发人员

Task	描述	所需技能
	6. 保存并关闭 NLog.config 文件。 有关示例配置文件，请参阅此模式的其他信息部分。运行应用程序时，NLog 将写入日志消息并将其发送到 Lo CloudWatch gs。	

验证和监控日志

Task	描述	所需技能
验证日志记录。	按照“日志”文档中查看发送到 CloudWatch 日志的日志数据中的 CloudWatch 说明进行操作。验证是否正在记录 .NET 应用程序的日志事件。如果未记录日志事件，请参阅此模式中的故障排除部分。	常规 AWS
监控 .NET 应用程序堆栈。	根据您的用例 CloudWatch 的需要在中配置监控。您可以使用“CloudWatch 日志见解”、“CloudWatch 指标见解”和“CloudWatch 应用程序见解”来监控您的.NET 工作负载。您还可以配置警报以接收警报，还可以创建用于从单一视图监控工作负载的自定义控制面板。	常规 AWS

故障排除

事务	解决方案
日志数据不会显示在 CloudWatch 日志中。	确保将 IAM 策略附加到 L CloudWatch ogs 担任的 IAM 角色。有关说明，请参阅操作说明部分的设置访问权限和工具部分。

相关资源

- 使用日志组和日志流（CloudWatch 日志文档）
- Amazon CloudWatch 日志和.NET 日志框架（AWS 博客文章）

其他信息

以下为示例 `NLog.config` 文件。

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <configSections>
    <section name="nlog" type="NLog.Config.ConfigSectionHandler, NLog" />
  </configSections>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />
  </startup>
  <nlog>
    <extensions>
      <add assembly="NLog.AWS.Logger" />
    </extensions>
    <targets>
      <target name="aws" type="AWSTarget" logGroup="NLog.TestGroup" region="us-east-1"
 profile="demo"/>
    </targets>
    <rules>
      <logger name="*" minlevel="Info" writeTo="aws" />
    </rules>
  </nlog>
</configuration>
```

跨不同 Amazon Web Services account 和 AWS 区域 复制 AWS Service Catalog 产品

由 Sachin Vighe (AWS) 和 Santosh Kale (AWS) 编写

摘要

AWS Service Catalog 是一项区域性服务，这意味着 AWS Service Catalog 产品组合和产品仅在创建它们的 AWS 区域 可见。如果您在新区域设置 AWS Service Catalog 中心，则必须重新创建现有产品，这可能是一个耗时的过程。

此模式的方法介绍如何将源 Amazon Web Services account 或区域中 AWS Service Catalog 中心的产品复制到目标账户或区域的新中心，以简化进程。有关 AWS Service Catalog 中心和分支模型的更多信息，请参阅 AWS 管理与治理博客上的 AWS Service Catalog 中心和分支模型：如何自动向多个账户部署和管理 AWS Service Catalog

该模式还提供了跨账户或其他区域复制 AWS Service Catalog 产品所需单独代码包。通过使用此模式，您的组织可节省时间，在新的 AWS Service Catalog 中心中提供现有和以前的产品版本，最大限度地降低手动错误的风险，并跨多个账户或区域扩展该方法。

> **Note**
>
> 此模式的 Epics 部分提供了两个复制产品的选项。您可使用选项 1 跨账户复制产品，也可选择选项 2 跨区域复制产品。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 源账户或区域中的现有 AWS Service Catalog 产品。
- 目标账户或区域中的现有 AWS Service Catalog 中心。
- 如果您想跨账户复制产品，则必须共享包含产品的 AWS Service Catalog 产品组合，然后将其导入至目标账户。有关这方面的更多信息，请参阅 AWS Service Catalog 文档中的共享与导入产品组合。

限制

- 您想要跨区域或账户复制的 AWS Service Catalog 产品不得属于多个产品组合。

架构

下图显示了将 AWS Service Catalog 产品从源账户复制至目标账户的过程。

下图显示了将 AWS Service Catalog 产品从源区域复制到目标区域的过程。

技术堆栈

- Amazon CloudWatch
- AWS Identity and Access Management（IAM）
- AWS Lambda
- AWS Service Catalog

自动化和扩缩

您可使用 Lambda 函数扩展这种模式方法，该函数可以根据收到的请求数量或需要复制的 AWS Service Catalog 产品数量进行扩展。有关这方面的更多信息，请参见 AWS Lambda 文档中的 Lambda 函数扩展。

工具

- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Service Catalog 可帮助您集中管理获准在 AWS 上使用的 IT 服务目录。最终用户可在遵循组织设定约束的情况下快速部署他们所需已获得批准的 IT 服务。

代码

您可以使用 `cross-account-copy` 软件包（附件）跨账户复制 AWS Service Catalog 产品，也可以使用 `cross-region-copy` 软件包（附件）跨区域复制产品。

`cross-account-copy` 软件包包含以下文件：

- `copyconf.properties` — 包含用于跨账户复制产品的区域和 Amazon Web Services account ID 参数的配置文件。
- `scProductCopyLambda.py` — 用于跨账户复制产品的 Python 函数。
- `createDestAccountRole.sh` — 用于在目标账户中创建 IAM 角色的脚本。
- `createSrcAccountRole.sh` — 用于在源账户中创建 IAM 角色的脚本。
- `copyProduct.sh` — 创建和调用 Lambda 函数以跨账户复制产品的脚本。

`cross-region-copy` 软件包包含以下文件：

- `copyconf.properties` — 包含用于跨区域复制产品的区域和 Amazon Web Services account ID 参数的配置文件。
- `scProductCopyLambda.py` — 用于跨区域复制产品的 Python 函数。
- `copyProduct.sh` — 创建 IAM 角色和创建并调用 Lambda 函数以跨区域复制产品的脚本。

操作说明

选项 1 — 跨账户复制 AWS Service Catalog 产品

Task	描述	所需技能
更新配置文件。	1. 将 `cross-account-copy` 软件包（附件）下载至本地计算机。 2. 使用以下值更新 `copyconf.properties` 配置文件： • `srcRegion` — 提供包含产品的来源区域。 • `destRegion` — 提供产品的目标区域。	AWS 管理员、AWS 系统管理员、云管理员

Task	描述	所需技能
	• `sourceAccountId` — 提供您的源账户的 Amazon Web Services account ID。 • `destAccountId` — 提供目标账户的 Amazon Web Services account ID。	

在目标账户中配置 AWS CLI 的凭证。	通过运行 `aws configure` 命令并提供以下值，配置您的凭证以访问目标账户中的 AWS CLI：
	`$aws configure`
	`AWS Access Key ID [None]: <your_access_key_id>`
	`AWS Secret Access Key [None]: <your_secret_access_key>`
	`Default region name [None]: Region`
	`Default output format [None]:`
	有关更多信息，请参见 AWS Command Line Interface 文档中的配置基础知识。
		AWS 管理员、AWS 系统管理员、云管理员

Task	描述	所需技能

在源账户中配置 AWS CLI 的凭证。	通过运行 `aws configure` 命令并提供以下值，配置您的凭证以在您的源账户中访问 AWS CLI：
	
	`$aws configure`
	`AWS Access Key ID [None]: <your_access_key_id>`
	`AWS Secret Access Key [None]: <your_secret_access_key>`
	`Default region name [None]: Region`
	`Default output format [None]:`
	
	有关更多信息，请参见 AWS Command Line Interface 文档中的配置基础知识。
		AWS 管理员、AWS 系统管理员、云管理员

在目标账户中创建 Lambda 执行角色。	在您的目标账户中运行 `createDestAccountRole.sh` 脚本。该脚本实施以下操作： • 在目标账户中创建 Lambda 执行角色 • 为 Lambda 执行角色创建和附加 IAM policy	AWS 管理员、AWS 系统管理员、云管理员

Task	描述	所需技能
在您的源账户中创建跨账户 IAM 角色。	在您的源账户中运行 `createSrcAccountRole.sh` 脚本。该脚本实施以下操作： • 在您的源账户中创建跨账户 IAM 角色，该角色由目标账户中的 Lambda 执行角色代入，用于复制产品 • 为您的源账户中的跨账户角色创建和附加 IAM policy	AWS 管理员、AWS 系统管理员、云管理员
在您的目标账户中运行 copyProduct 脚本。	在您的目标账户中运行 `copyProduct.sh` 脚本。该脚本实施以下操作： • 创建和调用 Lambda 函数，以将产品从源账户复制到目标账户	AWS 管理员、AWS 系统管理员、云管理员

选项 2 — 将 AWS Service Catalog 产品从源区域复制到目标区域

Task	描述	所需技能
更新配置文件。	1. 将 `cross-region-copy` 软件包（附件）下载至本地计算机。 2. 使用以下值更新 `copyconf.properties` 配置文件： • `srcRegion` — 提供包含产品的来源区域。 • `destRegion` — 提供产品的目标区域。	AWS 系统管理员、云管理员、AWS 管理员

Task	描述	所需技能
	• `accountId` — 提供 Amazon Web Services account ID	

使用您的 AWS 凭证配置 CLI	通过运行 `aws configure` 命令并提供以下值，配置您的凭证以在您的环境中访问 AWS CLI：
	
	`$aws configure`
	`AWS Access Key ID`
	`[None]: <your_access_key_id>`
	`AWS Secret Access Key`
	`[None]: <your_secret_access_key>`
	`Default region name`
	`[None]: Region`
	`Default output format`
	`[None]:`
	
	有关更多信息，请参见 AWS Command Line Interface 文档中的配置基础知识。
		AWS 管理员、AWS 系统管理员、云管理员

运行 copyProduct 脚本。	在您的目标账户中运行 `copyProduct.sh` 脚本。该脚本实施以下操作： • 创建一个 Lambda 执行角色 • 为 Lambda 执行角色创建和附加 IAM policy • 创建和调用 Lambda 函数，以将产品从源区域复制到目标区域	AWS 管理员、AWS 系统管理员、云管理员

相关资源

- 创建 Lambda 执行角色（AWS Lambda 文档）
- 创建 Lambda 函数（AWS Lambda 文档）
- AWS Service Catalog API 参考
- AWS Service Catalog 文档

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

为云运营模式创建 RACI 或 RASCI 矩阵

由 Teddy Germade (AWS)、Jerome Descreux (AWS)、Josselin LE MINEUR (AWS) 和 Florian Leroux (AWS) 创建

摘要

云卓越中心 (CCoE) 或 CEE（云支持引擎）是一个有能力且负责任的团队，专注于云的运营就绪性。他们的工作重点是将信息 IT 组织从本地运营模式转变为云运营模式。 CCoE 应该是一个跨职能团队，包括来自基础架构、应用程序、运营和安全的代表。

云运营模式的关键组成部分之一是 RACI 矩阵或 RASCI 矩阵。这用于定义参与迁移活动和云运营的所有各方的角色和职责。矩阵名称源自矩阵中定义的责任类型：负责(R)、问责(A)、支持(S)、咨询(C) 和知情(I)。支持类型是可选的。如果将其包括在内，则称为 RASCI 矩阵，如果将其排除在外，则称为 RACI 矩阵。

从随附的模板开始，您的 CCo电子团队可以为您的组织创建 RACI 或 RASCI 矩阵。该模板包含云运营模式中常见的团队、角色和任务。该矩阵的基础是与操作集成和 CCo E 能力相关的任务。不过，您也可以自定义此模板，以满足贵组织的结构和用例需求。

实施 RACI 矩阵没有任何限制。这种方法适用于大型组织、初创企业以及介于两者之间的任何企业。对于小型组织来说，同一资源可以充当多个角色。

操作说明

创建矩阵

Task	描述	所需技能
确定关键利益相关者。	确定与您的云运营模式的战略目标相关的关键服务和团队经理。	项目经理
自定义矩阵模板。	下载附件部分中的模板，然后按如下方式更新 RACI 或 RASCI 矩阵： • 在 Cloud Teams 工作表中，根据需要更新组织的 CCo E	项目经理

Task	描述	所需技能

	直播名称、团队名称和团队描述。
	• 在 Cloud Roles 工作表上，根据贵组织的需要更新角色、团队名称和角色描述。
	• 在 RASCI 工作表中，根据贵组织的需要更新以下内容：
	  • 在第 1 行和 A 列中，更新 CCo E 流。
	  • 在第 2 行中，更新团队名称。
	  • 在第 3 行中，更新角色名称。
	  • 在 D 列和 E 列中，更新您希望包含在 RASCI 图表中的常规字段和活动。	

计划会议。	1. 向所有利益相关者传达 RASCI 目标。 2. 计划召开一次或多次会议，以便每个团队都有一名授权代表参加。	项目经理

Task	描述	所需技能
完成矩阵。	在与所有利益相关者的会议中，请执行以下操作： 1. 确认每个团队都有一名代表在场。团队参与是强制性的，这样您就可以准确地为每项任务分配责任类型。 2. 与参与者一起回顾什么是 RASCI 矩阵以及其目标。 3. 与参与者一起回顾责任共担模式，以便他们了解各自组织在云端安全方面的责任范围。 4. 在 RASCI 工作表上，对于每项任务或活动，填写 F 到 AN 列以分配以下责任类型： • 负责(R) - 此角色负责执行工作以完成任务。 • 问责(A) - 此角色负责确保任务完成。此角色还负责确保满足先决条件，并将任务委派给相关负责人。 • 支持(S) - 此角色会帮助负责人完成任务。此责任类型是可选的，您可以选择将其排除，以便创建更传统的 RACI 矩阵。 • 咨询(C) - 应向此角色咨询有关任务的意见或专业知识。根据任务的不同，可能不需要此责任类型。	项目经理

Task	描述	所需技能
	• 知情(I) - 此角色应随时了解任务的最新进度，并会在任务完成时收到通知。 • 空白 - 此角色不参与活动或任务。	
共享 RASCI 矩阵。	当 RACI 或 RASCI 矩阵完成时，请得到领导层的批准。将其保存在所有利益相关者均可访问的共享存储库或中心位置。我们建议您使用标准文档控制流程来记录和批准对矩阵的修订。	项目经理

相关资源

- AWS 责任共担模式

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 Amazon CloudWatch 异常检测为自定义指标创建警报

创建者：Ram Kandaswamy (AWS) 和 Raheem Jiwani (AWS)

摘要

在 Amazon Web Services (AWS) 云上，您可以使用亚马逊 CloudWatch 创建警报，用于监控指标并发送通知，或者在突破阈值时自动进行更改。

为避免受到静态阈值的限制，您可以根据过去的模式创建警报，并在特定指标超出正常操作窗口时通知您。例如，您可以从 Amazon API Gateway 监控您的 API 的响应时间，并接收有关妨碍您满足服务水平协议（SLA）的异常的通知。

此模式描述了如何对自定义指标使用 CloudWatch 异常检测。该模式向您展示了如何在 Amazon L CloudWatch ogs Insights 中创建自定义指标或使用 AWS Lambda 函数发布自定义指标，然后使用亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 设置异常检测和创建通知。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 现有 SNS 主题，可配置以用于发送电子邮件通知。有关此内容的更多信息，请参阅 Amazon SNS 文档中的 Amazon SNS 入门。
- 配置了CloudWatch 日志的现有应用程序。

限制

- CloudWatch 指标不支持毫秒的时间间隔。有关常规指标和自定义指标粒度的更多信息，请参阅 Amazon CloudWatch FAQs。

架构

图表显示了以下工作流：

1. 使用日志创建和更新的指标的 CloudWatch 日志将流式传输到 CloudWatch。
2. 警报根据阈值启动，并向 SNS 主题发送警报。
3. Amazon SNS 会向您发送一条电子邮件通知。

技术堆栈

- CloudWatch
- AWS Lambda
- Amazon SNS

工具

- Amazon CloudWatch 提供可靠、可扩展且灵活的监控解决方案。
- AWS Lambda 是一项计算服务，可帮助您在不预配置或管理服务器的情况下运行代码。
- Amazon Simple Notification Service (Amazon SNS) 是一项托管服务，提供从发布者至订阅用户的消息传输。

操作说明

为自定义指标设置异常情况检测

Task	描述	所需技能
选项 1 - 使用 Lambda 函数创建自定义指标。	下载`lambda_function.py`文件（附后），然后替换 AWS 文档中aws-lambda-developer-guide存储库中的示例`lambda_function.py`文件 GitHub。这为您提供了一个向日志发送自定义指标的 Lambda 函数示例。CloudWatch Lambda 函数使用 Boto3 API 进行集成。CloudWatch 运行 Lambda 函数后，您可以登录 AWS 管理控制台，打开控制台，发布的 CloudWatch 指标将在您发布的命名空间下可用。	DevOps 工程师，AWS DevOps

Task	描述	所需技能
选项 2-从 CloudWatch 日志组创建自定义指标。	登录 AWS 管理控制台，打开 CloudWatch 控制台，然后选择日志组。选择要为其创建警报的日志组。 选择操作，然后选择创建指标筛选条件。对于筛选条件模式，输入要使用的筛选条件模式。有关更多信息，请参阅 CloudWatch 文档中的过滤器和模式语法。 若要测试过滤模式，请在测试模式下输入一个或多个日志事件。每个日志事件必须位于一行内，因为换行符用于在日志事件消息框中分隔日志事件。测试模式后，您可以在指标详细信息下输入指标的名称和值。 有关创建自定义指标的更多信息和步骤，请参阅 CloudWatch 文档中的为日志组创建指标筛选器。	DevOps 工程师，AWS DevOps

Task	描述	所需技能
为您的自定义指标创建警报。	在 CloudWatch 控制台上，选择警报，然后选择创建警报。选择选择指标，然后在搜索框中输入您之前创建的指标的名称。选择图形化指标选项卡，然后根据您的要求配置选项。 在条件下，选择异常检测，而不是静态阈值。这会显示一个基于两个标准默认差的波段。您可以设置阈值并根据需要进行调整。 选择下一步。 **Note** 波段是动态的，取决于数据点的质量。当您开始聚合更多数据时，波段和阈值会自动更新。	DevOps 工程师，AWS DevOps

Task	描述	所需技能
设置 Amazon SNS 通知。	在通知下方，选择警报处于 ALARM、OK 或 INSUFFICIENT_DATA 状态时通知的 SNS 主题。 要使告警为相同告警状态或不同告警状态发送多个通知，请选择添加通知。选择下一步。输入警报的名称和说明。名称只能包含 ASCII 字符。然后选择下一步。 在预览和创建下方确认信息和条件符合您的要求，然后选择创建警报。	DevOps 工程师，AWS DevOps

相关资源

- 将自定义指标发布到 CloudWatch
- 使用 CloudWatch 异常检测
- 警报事件和 Amazon EventBridge
- 将自定义指标推送到 Cloud Watch 时应遵循哪些最佳做法？（视频）
- CloudWatch 应用洞察简介（视频）
- 使用 CloudWatch（视频）检测异常

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

创建使用具有默认加密的 Amazon EBS 卷的 AWS Cloud9 IDE

由 Janardhan Malyala (AWS) 和 Dhrubajyoti Mukherjee (AWS) 创作

摘要

注意： AWS Cloud9 不再向新客户开放。的现有客户 AWS Cloud9 可以继续照常使用该服务。了解更多

您可以使用默认加密来强制加密Amazon Web Services (AWS) 云上的 Amazon Elastic Block Store (Amazon EBS) 卷和快照副本。

您可以创建默认使用加密的 EBS 卷的 AWS Cloud9 集成式开发环境 (IDE)。但是，AWS Cloud9 的 AWS Identity and Access Management (IAM) 服务相关角色需要访问这些 EBS 卷的 AWS Key Management Service (AWS KMS) 密钥。如果未提供访问权限，则 AWS Cloud9 IDE 可能无法启动，调试也可能变得困难。

此模式提供了将 AWS Cloud9 的服务相关角色添加到 EBS 卷所使用的 AWS KMS 密钥的步骤。此模式描述的设置可帮助您成功创建和启动使用默认加密的 EBS 卷的 IDE。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- EBS 卷的默认加密已开启。有关默认加密的更多信息，请参阅亚马逊弹性计算云 (Amazon EC2) 文档中的 Amazon EBS 加密。
- 用于加密 EBS 卷的现有客户托管 KMS 密钥。

> **Note**
>
> 您无需为 AWS Cloud9 创建服务相关角色。当您创建 AWS Cloud9 开发环境时，AWS Cloud9 会为你创建服务链接角色。

架构

技术堆栈

- Amazon Cloud9
- IAM
- AWS KMS

工具

- AWS Cloud9 是一种集成式开发环境 (IDE)，可帮助您编写、构建、运行、测试和调试软件。它还可以帮助您将软件发布到 Amazon Web Services Cloud。
- 亚马逊弹性区块存储 (Amazon EBS) Block Store 提供块级存储卷，用于亚马逊弹性计算云 (Amazon) 实例。 EC2
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥，以帮助保护您的数据。

操作说明

查找默认加密密钥值

Task	描述	所需技能
记录 EBS 卷的默认加密密钥值。	登录 AWS 管理控制台并打开亚马逊 EC2 控制台。选择“EC2 控制面板”，然后在“帐户”属性中选择“数据保护和安全”。在 EBS 加密部分中，复制并记录默认加密密钥中的值。	云架构师、 DevOps 工程师

提供对 AWS KMS 密钥的访问权限

Task	描述	所需技能
为 AWS Cloud9 提供对 EBS 卷的 KMS 密钥的访问权限。	1. 打开 AWS KMS 控制台，然后选择客户托管密钥。选择用于 Amazon EBS 加密	云架构师、 DevOps 工程师

Task	描述	所需技能
	的 AWS KMS 密钥，然后选择查看密钥。 2. 在密钥政策选项卡上，确认您可以看到密钥政策的文本形式。如果您看不到文本表单，请选择切换到策略视图。 3. 选择编辑。将其他信息部分中的代码添加到策略中，然后选择保存更改。策略更改允许 AWS Cloud9 `AWSServiceRoleForAWSCloud9` 的服务相关角色访问密钥。 有关更新密钥政策的更多信息，请参阅如何更改密钥政策（AWS KMS 文档）。 **⚠ Important** AWS Cloud9 的服务相关角色是在您启动第一个 IDE 时自动创建的。有关更多信息，请参阅 AWS Cloud9 文档中的创建服务链接角色。	

创建并启动 IDE

Task	描述	所需技能
创建并启动 AWS Cloud9 IDE。	打开 AWS Cloud9 控制台，然后选择创建环境。 按照 AWS Cloud9 文档中创建 EC2 环境中的步骤根据您的要求配置 IDE。	云架构师、 DevOps 工程师

相关资源

- 加密 AWS Cloud9 使用的 EBS 卷
- 为 AWS Cloud9 创建服务链接相关角色
- 在 AWS Cloud9 中创建 EC2 环境

其他信息

AWS KMS 密钥政策更新

将 <aws_accountid> 替换为您的 Amazon Web Services account ID。

```
{
            "Sid": "Allow use of the key",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::<aws_accountid>:role/aws-service-role/
cloud9.amazonaws.com/AWSServiceRoleForAWSCloud9"
            },
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "kms:DescribeKey"
            ],
            "Resource": "*"
        },
        {
```

```
            "Sid": "Allow attachment of persistent resources",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::<aws_accountid>:role/aws-service-role/
cloud9.amazonaws.com/AWSServiceRoleForAWSCloud9"
            },
            "Action": [
                "kms:CreateGrant",
                "kms:ListGrants",
                "kms:RevokeGrant"
            ],
            "Resource": "*",
            "Condition": {
                "Bool": {
                    "kms:GrantIsForAWSResource": "true"
                }
            }
        }
```

使用跨账户密钥

如果要使用跨账户 KMS 密钥，则必须将授权与 KMS 密钥策略结合使用。这允许跨账户访问密钥。使用您用于创建 Cloud9 环境的同一帐户，在终端中运行以下命令。

```
aws kms create-grant \
 --region <Region where Cloud9 environment is created> \
 --key-id <The cross-account KMS key ARN> \
 --grantee-principal arn:aws:iam::<The account where Cloud9 environment is
 created>:role/aws-service-role/cloud9.amazonaws.com/AWSServiceRoleForAWSCloud9 \
 --operations "Encrypt" "Decrypt" "ReEncryptFrom" "ReEncryptTo" "GenerateDataKey"
 "GenerateDataKeyWithoutPlaintext" "DescribeKey" "CreateGrant"
```

运行此命令后，您可以使用其他账户中的密钥使用 EBS 加密来创建 Cloud9 环境。

自动创建基于标签的 Amazon CloudWatch 控制面板

由 Janak Vadaria (AWS)、RAJNEESH TYAGI (AWS) 和 Vinodkumar Mandalapu (AWS) 创作

摘要

手动创建不同的 Amazon CloudWatch 控制面板可能很耗时，尤其是在您必须创建和更新多个资源以自动扩展环境时。自动创建和更新 CloudWatch 仪表板的解决方案可以为您节省时间。这种模式可以帮助您部署一个全自动 AWS Cloud Development Kit (AWS CDK) 管道，该管道根据标签更改事件为您的 AWS 资源创建和更新 CloudWatch 仪表板，以显示黄金信号指标。

在站点可靠性工程 (SRE) 中，Golden Signals是指一组全面的指标，这些指标可以从用户或消费者的角度提供服务的广阔视角。这些指标包括延迟、流量、错误和饱和度。有关更多信息，请参阅什么是站点可靠性工程 (SRE)？ 在 AWS 网站上。

这种模式提供的解决方案是事件驱动的。部署后，它会持续监控标签变更事件，并自动更新 CloudWatch 仪表板和警报。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- AWS Command Line Interface (AWS CLI)，已安装并配置
- AWS CDK v@@ 2 的先决条件
- 已启动的环境已启用 AWS
- Python 版本 3
- AWS 适用于 Python 的 SDK (Boto3)，已安装
- Node.js 版本 18 或更高版本
- 节点包管理器 (npm)，已安装并配置为 AWS CDK
- 对和的熟悉程度中等（ 200 级） AWS CDK AWS CodePipeline

限制

该解决方案目前仅为以下 AWS 服务创建自动控制面板：

- Amazon Relational Database Service (Amazon RDS)

- AWS Auto Scaling
- Amazon Simple Notification Service (Amazon SNS)
- Amazon DynamoDB
- AWS Lambda

架构

目标技术堆栈

- CloudWatch 仪表板
- CloudWatch 警报

目标架构

1. 配置的应用程序 AWS 标签或代码更改的标签更改事件会启动管道， AWS CodePipeline 以构建和部署更新的 CloudWatch 仪表板。
2. AWS CodeBuild 运行 Python 脚本来查找已配置标签的资源，并将资源存储 IDs 在 CodeBuild 环境中的本地文件中。
3. CodeBuild 运行 cdk synth 以生成用于部署 CloudWatch 仪表板和警报的 AWS CloudFormation 模板。
4. CodePipeline 将 AWS CloudFormation 模板部署到指定的 AWS 账户 和区域。
5. 成功部署 AWS CloudFormation 堆栈后，您可以查看 CloudWatch 仪表板和警报。

自动化和扩缩

该解决方案已通过使用实现自动化 AWS CDK。你可以在亚马逊 CloudWatch存储库的 GitHub 黄金信号仪表板中找到代码。为了进一步扩展和创建自定义仪表板，您可以配置多个标签键和值。

工具

Amazon 服务

- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来，包括 AWS Lambda 函数、使用 API 目标的 HTTP 调用终端节点或其他来源的事件总线。 AWS 账户

- AWS CodePipeline帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件更改所需的步骤。
- AWS CodeBuild是一项完全托管的生成服务，可帮助您编译源代码、运行单元测试和生成可随时部署的工件。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您通过命令行外壳中的命令与 AWS 服务进行交互。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

最佳实践

作为安全最佳实践，您可以对连接到管道的源存储库使用加密和身份验证。有关其他最佳实践，请参阅 CodePipeline 文档中的CodePipeline 最佳实践和用例。

操作说明

配置和部署示例应用程序

Task	描述	所需技能
配置和部署示例应用程序。	1. 使用以下命令克隆 GitHub 示例代码存储库： `git clone https://github.com/aws-samples/golden-signals-dashboards-sample-app` 2. 导航到计算机上克隆的存储库，然后使用您选择的编辑器打开该`src/project-settings.ts` 文件。 3. 根据您的 AWS 资源标签和应用程序映射更	AWS DevOps

Task	描述	所需技能
	改projectSettings 常量值。 4. 设置AWS_ACCOUNT AWS_REGION 、和GS_DASHBOARD_INSTANCE 环境变量： • 设置AWS_ACCOUNT 为您账户的 AWS 账户 ID。 • 设置AWS_REGION 为要部署示例应用程序的区域。 • 根据您的开发环境prod，设置为GS_DASHBOARD_INSTANCE 、或。dev test（我们建议使用test此模式中描述的测试程序。） 5. AWS CLI 使用您的 AWS 凭据进行设置。有关更多信息，请参阅 AWS CLI 文档中的使用命令设置和查看配置设置。 6. 运行以下命令以部署 Golden Signals 仪表板示例应用程序： `sh deploy.sh`	

Task	描述	所需技能
自动创建仪表板和警报。	部署示例应用程序后，您可以使用预期的标签值创建此解决方案支持的任何资源，这将自动创建指定的仪表板和警报。 要测试此解决方案，请创建一个 AWS Lambda 函数： 1. 登录您部署示例应用程序 AWS 区域 的位置。 AWS Management Console 2. 打开 Lambda 控制台，网址为。https://console.aws.amazon.com/lambda/ 3. 选择 “创建函数”，然后输入函数名称。 4. 在 “高级设置” 窗格中，选择 “启用标签”，然后选择 “添加新标签”。输入以下键和值： • 键：`AutoDashboard` • 值：`True` 5. 选择 Create function (创建函数)。 Lambda 函数会立即启动代码管道，该管道会自动为该特定 Lambda 函数创建仪表板和警报。 6. 要查看自动控制面板和警报，请打开 CloudWatch 控制台，网址为https://console.aws.amazon.com/cloudwatch/。您可以查看您在`projectSettings` 常	AWS DevOps

Task	描述	所需技能
	量中指定的函数的自定义仪表板和警报（默认为 APP1-lambda）。 7. 选择 Lambda 函数的控制面板，查看作为此解决方案一部分创建的其他自动控制面板。 8. 对其他服务（例如 Amazon RDS、Amazon SNS 和 DynamoDB）重复这些步骤以生成相关的控制面板。AWS Auto Scaling有关 Amazon RDS 的示例，请参阅其他信息部分。	

移除示例应用程序

Task	描述	所需技能
移除`golden-signals-dashboard` 构造。	1. 要删除示例应用程序创建的所有 AWS CloudFormation 堆栈，必须重新配置`AWS_ACCOUNT` `AWS_REGION` 、和`GS_DASHBOARD_INSTANCE` 环境变量。该`destroy.sh` 命令需要这些配置。 • `AWS_ACCOUNT` 是您账户的 AWS 账户 ID。 • `AWS_REGION` 是您部署示例应用程序的区域。	AWS DevOps

Task	描述	所需技能
	• GS_DASHBOARD_INSTANCE 是devtest、或prod，基于您之前的设置。 2. AWS CLI 使用您的 AWS 凭据进行设置。 3. 运行以下命令删除示例应用程序和所有关联 AWS CloudFormation 堆栈： `sh destroy.sh`	

故障排除

事务	解决方案
未找到 Python 命令（参见第 8 行）。findresources.sh	检查你的 Python 安装版本。如果您安装了 Python 版本 3，请python替python3换为resources.sh 文件第 8 行，然后再次运行sh deploy.sh 命令来部署解决方案。

相关资源

- 引导（文档）AWS CDK
- 使用命名配置AWS CLI 文件（文档）
- AWS CDK 工作坊

其他信息

下图显示了作为本解决方案一部分创建的 Amazon RDS 控制面板示例。

记录您的 AWS 着陆区设计

由迈克尔·达纳特 (AWS)、弗洛里安·兰格 (AWS) 和迈克尔·洛德曼 (AWS) 创作

摘要

l anding zon e 是一个基于安全和合规最佳实践的架构良好的多账户环境。它是企业范围的容器，可容纳您的所有组织单位 (OUs) AWS 账户、用户和其他资源。landing zone 可以进行扩展，以满足任何规模的企业的需求。 AWS 有两种创建着陆区的选项：使用基于服务的着陆区AWS Control Tower或您构建的自定义着陆区。每个选项都需要不同的 AWS 知识水平。

AWS AWS Control Tower 旨在通过自动设置着陆区来帮助您节省时间。 AWS Control Tower 由管理 AWS 并使用最佳实践和指南来帮助您创建基础环境。 AWS Control Tower 使用集成服务（例如AWS Service Catalog和）在您的 AWS Organizationslanding zone 中配置账户并管理对这些账户的访问权限。

AWS landing zone 项目的要求、实施细节和操作措施项目各不相同。每个 landing zone 实施都需要处理一些自定义方面。这包括（但不限于）如何处理访问管理、使用哪个技术堆栈以及实现卓越运营的监控要求。此模式提供了一个模板，可帮助您记录 landing zone 项目。通过使用该模板，您可以更快地记录您的项目，并帮助您的开发和运营团队了解您的着陆区。

先决条件和限制

限制

这种模式并不能描述什么是着陆区或如何实现着陆区。有关这些主题的更多信息，请参阅相关资源部分。

操作说明

创建设计文档

Task	描述	所需技能
确定关键利益相关者。	确定与您的 landing zone 相关的关键服务和团队经理。	项目经理
自定义模板。	在 “附件” 部分下载模板，然后按如下方式更新模板：	项目经理

Task	描述	所需技能
	1. 移除所有不适用于贵组织的 landing zone 或流程的部分。 2. 添加您的组织独有的所有版块。	
完成模板。	在与利益相关者会面或使用write-and-review流程时，按如下方式填写模板： 1. 使用蓝色方框中的指南和信息完成每个部分。 2. 使用您组织的自定义值替换或删除任何黄色字段。 3. 使用您的自定义架构或流程图替换或删除任何图像字段。 4. 完成模板的 “修订历史记录和贡献者” 部分。	项目经理
共享设计文档。	landing zone 设计文档完成后，将其保存在共享存储库或所有利益相关者都可以访问的中心位置。我们建议您使用标准文档控制流程来记录和批准对设计文档的修订。	项目经理

相关资源

- AWS Control Tower 文档
  - 规划你的 AWS Control Tower 着陆区
  - AWS 您的 AWS Control Tower 着陆区（landing zone）的多账户策略
  - 设置着陆区的管理提示
  - 对 landing zone 配置的期望

- AWS Control Tower（AWS 解决方案库）的自定义
- 设置安全且可扩展的多账户 AWS 环境（AWS 规范性指南）

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

通过在多个 AWS 区域、账户中启用 Amazon DevOps Guru 并 OUs 使用 AWS CDK 来提高运营绩效

由 Rahul Sharad Gaikwad 博士 (AWS) 编写

摘要

此模式演示了使用中的 AWS Cloud Development Kit (AWS CDK) 在多个亚马逊网络服务 (AWSOUs) 区域、账户和组织单位 () 中启用 Amazon DevOps Guru 服务的步骤。 TypeScript您可以使用 AWS CDK 堆栈 CloudFormation StackSets 从管理员（主）AWS 账户部署 AWS，从而在多个账户中启用 Amazon DevOps Guru，而不必登录每个账户并为每个账户单独启用 DevOps Guru。

Amazon DevOps Guru 提供人工智能操作 (AIOps) 功能，可帮助您提高应用程序的可用性并更快地解决操作问题。 DevOps Guru 无需任何机器学习专业知识，即可应用由机器学习 (ML) 驱动的建议，从而减少您的手动工作。 DevOps Guru 会分析您的资源和运营数据。如果它检测到任何异常，它将提供指标、事件和建议来帮助您解决问题。

此模式描述了启用 Amazon DevOps Guru 的三个部署选项：

- 适用于多个账户及区域的所有资源
- 适用于所有堆栈资源 OUs
- 适用于跨多个账户和区域的具体资源

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 已安装和配置 AWS 命令行界面（AWS CLI）。（请参阅 AWS CLI 文档中的安装、更新和卸载 AWS CLI。）
- AWS CDK Toolkit，已安装并配置。(请参阅 AWS CDK 文档中的 AWS CDK Toolkit。)
- Node Package Manager (npm)，已为中的 AWS CDK 安装和配置。 TypeScript（请参阅 npm 文档中的下载和安装 Node.js 和 npm。）
- Python3，已安装并配置，用于运行 Python 脚本向示例无服务器应用程序注入流量。(请参阅 Python 文档中的 Python 设置和用法。)
- Pip，已安装并配置，安装 Python 请求库。（请参阅 PyPI 网站上的 pip 安装说明。）

产品版本

- AWS CDK Toolkit 版本 1.107.0 或更高版本
- npm 版本 7.9.0 或更高版本
- Node.js 版本 15.3.0 或更高版本

架构

技术

此模式的架构包含以下服务：

- Amazon DevOps Guru
- AWS CloudFormation
- Amazon API Gateway
- AWS Lambda
- Amazon DynamoDB
- Amazon CloudWatch
- AWS CloudTrail

AWS CDK 堆栈

该模式使用以下 AWS CDK 堆栈：

- `CdkStackSetAdminRole` — 创建 AWS Identity and Access management (IAM) 管理员角色以在管理员与目标账户之间建立信任关系。
- `CdkStackSetExecRole` — 创建 IAM 角色以信任管理员账户。
- `CdkDevopsGuruStackMultiAccReg`— 允许 DevOps Guru 跨多个 AWS 区域和账户访问所有堆栈，并设置亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 通知。
- `CdkDevopsGuruStackMultiAccRegSpecStacks`— 允许 DevOps Guru 跨多个 AWS 区域和账户访问特定堆栈，并设置 Amazon SNS 通知。
- `CdkDevopsguruStackOrgUnit`— 启用 DevOps Guru 跨越 OUs并设置 Amazon SNS 通知。
- `CdkInfrastructureStack` — 在管理员账户中部署示例无服务器应用程序组件，例如 API Gateway、Lambda 和 DynamoDB，以演示错误注入和见解生成。

应用程序架构示例

下图演示了跨多个账户和区域部署的示例无服务器应用程序的架构。该模式采用管理员账户部署所有 AWS CDK 堆栈。它还使用管理员帐户作为设置 DevOps Guru 的目标帐户之一。

1. 启用 DevOps Guru 后，它会首先为每个资源的行为设定基准，然后从 CloudWatch 提供的指标中提取操作数据。
2. 如果它检测到异常，则会将其与来自的事件关联起来 CloudTrail，并生成见解。
3. 该见解提供了相关事件序列以及规定的建议，使操作员能够识别罪魁祸首资源。
4. Amazon SNS 会向操作员发送通知消息。

自动化和扩缩

此模式提供的GitHub 存储库使用 AWS CDK 作为基础设施即代码 (IaC) 工具来创建该架构的配置。AWS CDK 可帮助您协调资源并在多个 DevOps AWS 账户、地区和. OUs

工具

Amazon Web Services

- AWS CDK — AWS Cloud Development Kit (AWS CDK) 可帮助您将云基础设施定义为使用五种支持的编程语言之一的代码： TypeScript、、 JavaScript Python、Java 和 C#。
- AWS CLI – AWS 命令行界面（AWS CLI）是一种统一工具，为与 Amazon Web Services 和资源的交互提供了一致的命令行界面。

代码

此模式的源代码可在 GitHub Amazon DevOps Guru CDK 示例存储库中找到。AWS CDK 代码是用编写的。 TypeScript要克隆和使用存储库，请按照下一节中的说明操作。

> **Important**
>
> 这种模式中的一些故事包括针对 Unix、Linux 和 macOS 进行格式化的 AWS CDK 和 AWS CLI 命令示例。对于 Windows，请将每行末尾的反斜杠 (\) 继续符替换为脱字号 (^)。

操作说明

准备 AWS 资源，以进行部署

Task	描述	所需技能

配置 AWS 命名配置文件。	按如下方式设置您的 AWS 命名配置文件，以在多账户环境中部署堆栈。
	
	对于管理员账户：
	
	`$aws configure --profile administrator`
	`AWS Access Key ID [****]: <your-administrator-access-key-ID>`
	`AWS Secret Access Key [****]: <your-administrator-secret-access-key>`
	`Default region name [None]: <your-administrator-region>`
	`Default output format [None]: json`
	
	对于目标账户：
	
	`$aws configure --profile target`
	`AWS Access Key ID [****: <your-target-access-key-ID>`
	`AWS Secret Access Key [****]: <your-target-secret-access-key>`
	`Default region name [None]: <your-target-region>`
		DevOps 工程师

Task	描述	所需技能
	`Default output format [None]: json` 有关更多信息，请参阅 AWS CLI 文档中的使用命名配置文件。	
验证 AWS 配置文件配置。	（可选）您可以按照 AWS CLI 文档中的设置和查看配置设置中的说明，验证 `credentials` 和 `config` 文件中的 AWS 配置文件配置。	DevOps 工程师
验证 AWS CDK 版本。	运行以下命令验证 AWS CDK Toolkit 的版本： `$cdk --version` 此模式需要版本 1.107.0 或更高版本。如果您使用 AWS CDK 早期版本，请按照 AWS CDK 文档中的说明对其进行更新。	DevOps 工程师
克隆项目代码。	使用以下命令克隆此模式的 GitHub 存储库： `$git clone https://github.com/aws-samples/amazon-devopsguru-cdk-samples.git`	DevOps 工程师

Task	描述	所需技能

安装软件包依赖关系并编译 TypeScript 文件。	通过运行以下命令安装软件包依赖关系并编译 TypeScript 文件：
	`$cd amazon-devopsguru-cdk-samples`
	`$npm install`
	`$npm fund`
	这些命令安装示例存储库内的所有软件包。
	Important
	如果您遇到任何关于丢失软件包的错误，请使用以下命令之一：
	`$npm ci`
	—或者—
	`$npm install -g @aws-cdk/<package-name>`
	您可以在 `/amazon-devopsguru-cdk-samples/package.json` 文件 `Dependencies` 部分找到软件包名称和版本的列表。有关更多信息，请参阅 npm 文档中的 npm ci 和 npm 安装。
		DevOps 工程师

构建(合成)AWS CDK 堆栈

Task	描述	所需技能
为 Amazon SNS 通知配置电子邮件地址。	请按照以下步骤为 Amazon SNS 通知提供电子邮件地址： 1. 编辑文件 `/amazon-devopsguru-cdk-samples/lib/cdk-devopsguru-multi-acc-reg-stack.ts` 和 `/amazon-devopsguru-cdk-samples/lib/cdk-devopsguru-org-uni-stack.ts` 。 2. 在 `DevOpsGuruTopic` 、`Subscription` 部分中，使用您的电子邮件地址更新 `Endpoint` 参数。 3. 保存并关闭文件。	DevOps 工程师

构建项目代码。	通过运行以下命令，构建项目代码并合成堆栈：
	`npm run build && cdk synth`
	您应该可以看到类似于如下所示的输出内容：
	`$npm run build && cdk synth`
	`> cdk-devopsguru@0.1.0 build`
	`> tsc`
		DevOps 工程师

Task	描述	所需技能
	`Successfully synthesized to ~/amazon-devopsguru-cdk-samples/cdk.out Supply a stack id (CdkDevopsGuruStackMultiAccReg,CdkDevopsGuruStackMultiAccRegSpecStacks, CdkDevopsguruStackOrgUnit, CdkInfrastructureStack, CdkStackSetAdminRole, CdkStackSetExecRole) to display its template.` 有关更多信息和步骤，请参阅 AWS CDK 文档中的[您的第一个 AWS CDK 应用程序](#)。	

列出 AWS CDK 堆栈。	运行以下命令以列出所有 AWS CDK 堆栈：
	
	`$cdk list`
	
	命令显示如下列表：
	
	`CdkDevopsGuruStackMultiAccReg CdkDevopsGuruStackMultiAccRegSpecStacks CdkDevopsguruStackOrgUnit CdkInfrastructureStack CdkStackSetAdminRole CdkStackSetExecRole`
		DevOps 工程师

选项 1-为多个账户的所有堆栈资源启用 DevOps Guru

Task	描述	所需技能
部署用于创建 IAM 角色的 AWS CDK 堆栈。	此模式使用 AWS CloudFormation StackSets 跨多个账户执行堆栈操作。如果您要创建第一个堆栈集，必须创建以下 IAM 角色才能在您的 Amazon Web Services account 中设置所需权限： • `AWSCloudFormationStackSetAdministrationRole` • `AWSCloudFormationStackSetExecutionRole` **Note** 角色必须有这些确切的名称。 1. 通过运行以下 CLI 命令在管理员（主）账户中创建 IAM `AWSCloudFormationStackSetAdministrationRole` 角色： `$cdk deploy CdkStackSetAdminRole --profile administrator` 2. 在您要运行堆栈实例的所有目标账户中创建 IAM	DevOps 工程师

Task	描述	所需技能

	`AWSCloudFormationStackSetExecutionRole` 角色。若要创建此角色，请运行以下 CLI 命令：
	
	`$cdk deploy CdkStackSetExecRole \`
	`  --parameters AdministratorAccountId=<administrator-account-ID> \`
	`  --profile administrator`
	
	`$cdk deploy CdkStackSetExecRole \`
	`  --parameters AdministratorAccountId=<administrator-account-ID> \`
	`  --profile target`
	
	有关更多信息，请参阅 AWS CloudFormation 文档中的授予自我管理权限。	

Task	描述	所需技能
部署 AWS CDK 堆栈，以便在多个账户中启用 DevOps Guru。	AWS CDK CdkDevopsGuruStackMultiAccReg 堆栈创建堆栈集，用于跨多账户和区域部署堆栈实例。若要部署堆栈，请使用指定参数运行以下 CLI 命令： `$cdk deploy CdkDevopsGuruStackMultiAccReg \ --profile administrator \ --parameters AdministratorAccountId=<administrator-account-ID> \ --parameters TargetAccountId=<target-account-ID> \ --parameters RegionIds="<region-1>,<region-2>"` 目前，Amazon DevOps Guru 已在 DevOps Gur u 常见问题解答中列出的 AWS 区域中提供。	DevOps 工程师

选项 2-为所有堆栈资源启用 DevOps Guru OUs

Task	描述	所需技能
提取 OU IDs。	在 AWS O rganizations 控制台上，确定要在其中启用 DevOps Guru 的组织单位。IDs	DevOps 工程师

Task	描述	所需技能
为启用服务管理权限。 OUs	如果您使用 AWS Organizations 进行账户管理，则必须授予服务托管权限才能启用 DevOps Guru。与其手动创建 IAM 角色，不如使用基于组织的可信访问权限和服务相关角色 ()。SLRs	DevOps 工程师
部署 AWS CDK 堆栈以启用 DevOps Guru。 OUs	AWS CDK `CdkDevopsguruStackOrgUnit` 堆栈支持 DevOps Guru 服务。OUs若要部署堆栈，请使用指定参数运行以下命令： `$cdk deploy CdkDevopsguruStackOrgUnit \ --profile administrator \ --parameters RegionIds="<region-1>,<region-2>" \ --parameters OrganizationalUnitIds="<OU-1>,<OU-2>"`	DevOps 工程师

选项 3-为跨多个账户的特定堆栈资源启用 DevOps Guru

Task	描述	所需技能
部署用于创建 IAM 角色的 AWS CDK 堆栈。	如果您未创建第一个选项中显示的必需 IAM 角色，请先执行以下操作： 1. 通过运行以下 CLI 命令在管理员（主）账户中创建 IAM `AWSCloudFormationS`	DevOps 工程师

Task	描述	所需技能

	tackSetAdministrationRole 角色：
	
	``
	$cdk deploy CdkStackSetAdminRole --profile administrator
	``
	
	2. 在您要运行堆栈实例的所有目标账户中创建 IAM AWSCloudFormationStackSetExecutionRole 角色。若要创建此角色，请运行 CLI 命令：
	
	``
	$cdk deploy CdkStackSetExecRole \
	  --parameters AdministratorAccountId=<administrator-account-ID> \
	  --profile administrator
	
	$cdk deploy CdkStackSetExecRole \
	  --parameters AdministratorAccountId=<administrator-account-ID> \
	  --profile target
	``
	
	有关更多信息，请参阅 AWS CloudFormation 文档中的授予自我管理权限。	

Task	描述	所需技能
删除现有堆栈。	如果您已经使用第一个选项为所有堆栈资源启用 DevOps Guru，则可以使用以下命令删除旧堆栈： `$cdk destroy CdkDevopsGuruStackMultiAccReg --profile administrator` 或者，您可以在重新部署堆栈时更改 `RegionIds` 参数，以避免出现堆栈已存在错误。	DevOps 工程师
通过堆栈列表更新 AWS CDK 堆栈。	1. 编辑 `/amazon-devopsguru-cdk-samples/lib/cdk-devopsguru-multi-acc-reg-spec-stack.ts` 文件。 2. 在`Resources`、`CloudFormation`、下`StackNames`，列出要为其启用 DevOps Guru 的堆栈。出于演示目的，该参数指定 `CdkInfrastructureStack` 堆栈，但您可以根据需要编辑此条目。 3. 保存并关闭文件。 4. 要合成和更新堆栈模板，请您运行： `$cdk synth`	数据工程师

Task	描述	所需技能
部署 AWS CDK 堆栈，为跨多个账户的特定堆栈资源启用 DevOps Guru。	AWS CDK CdkDevopsGuruStackMultiAccRegSpecStacks 堆栈允许 DevOps Guru 在多个账户中使用特定的堆栈资源。要部署堆栈，请运行以下命令： `$cdk deploy CdkDevopsGuruStackMultiAccRegSpecStacks \ --profile administrator \ --parameters AdministratorAccountId=<administrator-account-ID> \ --parameters TargetAccountId=<target-account-ID> \ --parameters RegionIds="<region-1>,<region-2>"` **Note** 如果您之前为选项 1 部署了此堆栈，请更改RegionIds 参数（确保从可用区域中进行选择），以避免出现堆栈已存在错误。	DevOps 工程师

部署 AWS CDK 基础设施堆栈

Task	描述	所需技能
部署示例无服务器基础设施堆栈。	AWS CDK `CdkInfrastructureStack` 堆栈部署了无服务器组件，例如 API Gateway、Lambda 和 DynamoDB 表，以演示 Guru 的见解。 DevOps 要部署堆栈，请运行以下命令： `$cdk deploy CdkInfrastructureStack --profile administrator`	DevOps 工程师

在 DynamoDB 内插入示例记录。	运行以下命令以在 DynamoDB 表中填充示记录。为 `populate-shops-dynamodb-table.json` 脚本提供正确的路径。
	`$aws dynamodb batch-write-item \ --request-items file://scripts/populate-shops-dynamodb-table.json \ --profile administrator`
	该命令将显示以下输出：
	`{ "UnprocessedItems": : {} }`
		DevOps 工程师

Task	描述	所需技能

在 DynamoDB 中验证插入记录。	要验证 DynamoDB 表中是否包含 `populate-shops-dynamodb-table.json` 文件中的示例记录，请访问作为 AWS CDK 堆栈输出发布的 `ListRestApiEndpointMonitorOperator` API 的网址。您也可以在`CdkInfrastructureStack` 堆栈的 AWS CloudFormation 控制台的 “输出” 选项卡中找到此 URL。AWS CDK 输出将类似于以下内容：
	
	`CdkInfrastructureStack.CreateRestApiMonitorOperatorEndpointD1D00045 = https://oure17c5vob.execute-api.<your-region>.amazonaws.com/prod/`
	
	`CdkInfrastructureStack.ListRestApiMonitorOperatorEndpointABBDB8D8 = https://cdff8icfrn4.execute-api.<your-region>.amazonaws.com/prod/`
		DevOps 工程师

Task	描述	所需技能
等待资源完成基准设定。	此无服务器堆栈有一些资源。我们建议您等待 2 小时后再执行后续步骤。如果您在生产环境中部署此堆栈，则最多可能需要 24 小时才能完成基准测试，具体取决于您选择在 Guru 中 DevOps 监控的资源数量。	DevOps 工程师

生成 DevOps 大师见解

Task	描述	所需技能

更新 AWS CDK 基础设施堆栈。	要试用 DevOps Guru 见解，您可以进行一些配置更改以重现典型的操作问题。
	1. 编辑 `/amazon-devopsguru-cdk-samples/lib/infrastructure-stack.ts` 文件。
	2. 在本 `DDB Table` 节中，将 DynamoDB 表的读取容量从 5 更改为 1。
	3. 保存并关闭文件。
	4. 运行以下命令以合成和部署更新后的 AWS CDK 基础设施堆栈：
	`$cdk synth`
	`$cdk deploy CdkInfrastructureStack --profile administrator`
		DevOps 工程师

Task	描述	所需技能
在 API 注入 HTTP 请求。	在 `ListRestApiMonitorOperatorEndpointxxxx` API 上以 HTTP 请求形式注入入入口流量： 1. 编辑 Python 脚本 `/amazon-devopsguru-cdk-samples/scripts/sendAPIRequest.py` 。 2. 使用 `ListRestApiMonitorOperatorEndpointxxxx` 的 API 链接更新 `url` 变量。您可以在 AWS CDK 部署命令的输出中找到此 URL，也可以在 AWS Cloudformation 控制台的堆栈输出选项卡中找到此 URL。 3. 保存并关闭文件。 4. 使用命令运行 Python 脚本： `$python sendAPIRequest.py` 5. 确保您得到一个 200 的状态码。 6. 您可能需要通过多个（最好是四个）终端运行脚本才能高速注入流量。 7. 脚本循环运行大约 10 分钟后，您可以在 DevOps Guru 控制台上看到操作见解。	DevOps 工程师

Task	描述	所需技能
查看 DevOps Guru 的见解。	在标准条件下， DevOps Guru 仪表板在持续见解计数器中显示为零。如果检测到异常，则会以洞察的形式发出警报。在导航窗格中，选择见解以查看异常的详细信息，包括概述、聚合指标、相关事件和建议。有关查看见解的更多信息，请参阅 “ AIOps 使用 Amazon DevOps Guru 获得运营见解” 博客文章。	DevOps 工程师

清理

Task	描述	所需技能

清理和删除资源。	完成此模式后，应移除您创建的资源，以免产生任何进一步费用。运行以下命令：
	
	`$cdk destroy CdkDevopsGuruStackMultiAccReg --profile administrator`
	`$cdk destroy CdkDevopsguruStackOrgUnit --profile administrator`
	`$cdk destroy CdkDevopsGuruStackMultiAccRegSpecStacks --profile administrator`
	`$cdk destroy CdkInfrastructureStack --profile administrator`
	`$cdk destroy CdkStackSetAdminRole --profile administrator`
		DevOps 工程师

Task	描述	所需技能
	`$cdk destroy CdkStackSetExecRole --profile administrator` `$cdk destroy CdkStackSetExecRole --profile target`	

相关资源

- AIOps 使用 Amazon DevOps Guru 获得运营见解
- 使用 AWS 在多个账户和地区轻松配置 Amazon DevOps Guru CloudFormation StackSets
- DevOps 大师工作坊

使用引导管道实现 Account Factory for Terraform (AFT)

由 Vinicius Elias (AWS) 和 Edgar Costa Filho (AWS) 创作

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

此模式为从的管理 AWS Control Tower 账户部署 Account Factory for Terraform (AFT) 提供了一种简单而安全的方法。 AWS Organizations该解决方案的核心是一个 AWS CloudFormation 模板，该模板通过创建Terraform管道来自动执行AFT配置，该管道的结构易于适应初始部署或后续更新。

安全和数据完整性是重中之重 AWS，因此，Terraform状态文件是跟踪托管基础设施和配置状态的关键组件，可以安全地存储在亚马逊简单存储服务 (Amazon S3) Simple Service 存储桶中。此存储桶配置了多种安全措施，包括服务器端加密和阻止公共访问的策略，以帮助确保您的 Terraform 状态免受未经授权的访问和数据泄露的侵害。

管理账户负责协调和监督整个环境，因此它是环境中的关键资源。 AWS Control Tower这种模式遵循 AWS 最佳实践，确保部署过程不仅高效，而且符合安全和治理标准，从而为在您的 AWS 环境中部署 AFT 提供了一种全面、安全和高效的方式。

有关 AFT 的更多信息，请参阅AWS Control Tower 文档。

先决条件和限制

先决条件

- 基本的 AWS 多账户环境，至少包含以下账户：管理账户、日志存档账户、审计账户，以及一个用于 AFT 管理的额外账户。
- 成熟的 AWS Control Tower 环境。应正确配置管理帐户，因为 CloudFormation 模板将在其中部署。
- AWS 管理账户中的必要权限。您需要足够的权限才能创建和管理资源，例如 S3 存储桶、 AWS Lambda 函数、 AWS Identity and Access Management (IAM) 角色和 AWS CodePipeline 项目。
- 熟悉 Terraform。了解 Terraform 的核心概念和工作流程很重要，因为部署涉及生成和管理 Terraform 配置。

限制

- 请注意您账户中的AWS 资源配额。部署可能会创建多个资源，遇到服务配额可能会阻碍部署过程。
- 该模板专为特定版本的 Terraform 和. AWS 服务升级或更改版本可能需要修改模板。
- 该模板不支持自我管理的版本控制系统 (VCS) 服务，例如 GitHub 企业版。

产品版本

- Terraform 版本 1.6.6 或更高版本
- AFT 版本 1.11 或更高版本

架构

目标技术堆栈

- AWS CloudFormation
- AWS CodeBuild
- AWS CodeCommit
- AWS CodePipeline
- Amazon EventBridge
- IAM
- AWS Lambda
- Amazon S3

目标架构

下图说明了这种模式中讨论的实现。

该工作流程包括三个主要任务：创建资源、生成内容和运行管道。

创建资源

此模式提供的CloudFormation 模板将创建和设置所有必需的资源，具体取决于您在部署模板时选择的参数。该模板至少会创建以下资源：

- 用于实现 AFT 的 CodePipeline 管道

- 一个 S3 存储桶，用于存储与 AFT 实现关联的 Terraform 状态文件
- 两个 CodeBuild 项目用于实施 Terraform 计划并在管道的不同阶段应用命令
- CodeBuild 和 CodePipeline 服务的 IAM 角色
- 第二个 S3 存储桶，用于存储管道运行时工件

根据您选择的 VCS 提供商（CodeCommit 或外部 VCS），模板会创建以下资源。

- 对于 CodeCommit：
  - 用于 CodeCommit 存储 AFT Terraform 引导程序代码的存储库
  - 捕获`main`分支上 CodeCommit 仓库变更的 EventBridge 规则
  - 该 EventBridge 规则的另一个 IAM 角色
- 对于任何其他外部 VCS 提供商，例如 GitHub：
  - 一个 AWS CodeConnections 连接

此外，当您选择 CodeCommit 作为 VCS 提供者时，如果将`Generate AFT Files`参数设置为`true`，则模板会创建以下额外资源来生成内容：

- 一个 S3 存储桶，用于存储生成的内容并用作 CodeCommit 存储库的来源
- 一个 Lambda 函数，用于处理给定参数并生成相应内容
- 用于运行 Lambda 函数的 IAM 函数
- 部署模板时运行 Lambda 函数的 CloudFormation 自定义资源

生成内容

为了生成 AFT 引导文件及其内容，该解决方案使用 Lambda 函数和 S3 存储桶。该函数在存储桶中创建一个文件夹，然后在该文件夹中创建两个文件：`main.tf`和`backend.tf`。该函数还处理提供的 CloudFormation 参数，并使用预定义的代码填充这些文件，替换相应的参数值。

要查看用作生成文件的模板的代码，请参阅解决方案的GitHub 存储库。基本上，文件按如下方式生成。

main.tf

```
module "aft" {
  source = "github.com/aws-ia/terraform-aws-control_tower_account_factory?
ref=<aft_version>"
```

```
  # Required variables
  ct_management_account_id  = "<ct_management_account_id>"
  log_archive_account_id    = "<log_archive_account_id>"
  audit_account_id          = "<audit_account_id>"
  aft_management_account_id = "<aft_management_account_id>"
  ct_home_region            = "<ct_home_region>"

  # Optional variables
  tf_backend_secondary_region = "<tf_backend_secondary_region>"
  aft_metrics_reporting       = "<false|true>"

  # AFT Feature flags
  aft_feature_cloudtrail_data_events      = "<false|true>"
  aft_feature_enterprise_support          = "<false|true>"
  aft_feature_delete_default_vpcs_enabled = "<false|true>"

  # Terraform variables
  terraform_version      = "<terraform_version>"
  terraform_distribution = "<terraform_distribution>"

  # VCS variables (if you have chosen an external VCS)
  vcs_provider                                = "<github|githubenterprise|gitlab|
gitlabselfmanaged|bitbucket>"
  account_request_repo_name                   = "<org-name>/aft-account-request"
  account_customizations_repo_name            = "<org-name>/aft-account-
customizations"
  account_provisioning_customizations_repo_name = "<org-name>/aft-account-provisioning-
customizations"
  global_customizations_repo_name             = "<org-name>/aft-global-
customizations"

}
```

backend.tf

```
terraform {
  backend "s3" {
    region = "<aft-main-region>"
    bucket = "<s3-bucket-name>"
    key    = "aft-setup.tfstate"
  }
}
```

在创建 CodeCommit 存储库期间，如果将`Generate AFT Files`参数设置为`true`，则模板将使用包含生成内容的 S3 存储桶作为`main`分支的来源，自动填充存储库。

运行管道

创建资源并配置引导程序文件后，管道将运行。第一阶段（来源）从存储库的主分支获取源代码，第二阶段（构建）运行 Terraform 计划命令并生成要查看的结果。在第三阶段（批准）中，管道等待手动操作批准或拒绝最后一个阶段（部署）。在最后阶段，管道使用前一个 Terraform `apply` 命令的结果作为输入来运行 Terraform `plan` 命令。最后，跨账户角色和管理账户中的权限用于在 AFT 管理账户中创建 AFT 资源。

> **Note**
>
> 如果您选择外部 VCS 提供商，则需要使用您的 VCS 提供商凭据授权连接。要完成设置，请按照 AWS 开发者工具控制台文档中更新待处理连接中的步骤进行操作。

工具

Amazon Web Services

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置这些资源，并在各个区域的整个生命周期中 AWS 账户 对其进行管理。
- AWS CodeBuild是一项完全托管的生成服务，可帮助您编译源代码、运行单元测试和生成可随时部署的工件。
- AWS CodeCommit是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。
- AWS CodePipeline帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件更改所需的步骤。
- AWS CodeConnections使 AWS 资源和服务（例如 CodePipeline）能够连接到外部代码存储库，例如 GitHub。
- AWS Lambda是一项计算服务，它运行您的代码以响应事件并自动管理计算资源，为创建现代化的无服务器生产应用程序提供了一种快速的方法。
- 适用于 Python (Boto3) 的 AWS SDK是一个软件开发套件，可帮助您将 Python 应用程序、库或脚本与集成 AWS 服务。

其他工具

- Terraform 是一款基础设施即代码 (IaC) 工具，可让您安全高效地构建、更改和版本化基础架构。这包括计算实例、存储和网络等低级组件；以及 DNS 条目和 SaaS 功能等高级组件。
- Python 是一种易于学习、功能强大的编程语言。它具有高效的高级数据结构，为面向对象的编程提供了一种简单但有效的方法。

代码存储库

此模式的代码可在 GitHub AFT bootstrap 管道存储库中找到。

有关官方的 AFT 存储库，请参阅中的 T erraform 的 Ac AWS Control Tower count Factory。 GitHub

最佳实践

使用提供的 CloudFormation 模板部署 AFT 时，我们建议您遵循最佳实践，以帮助确保安全、高效和成功的实施。实施和运营 AFT 的主要指导方针和建议包括以下内容。

- 全面审查参数：仔细检查并理解 CloudFormation 模板中的每个参数。准确的参数配置对于 AFT 的正确设置和运行至关重要。
- 定期更新模板：使用最新 AWS 功能和 Terraform 版本更新模板。定期更新可帮助您利用新功能并维护安全性。
- 版本控制：固定您的 AFT 模块版本，并在可能的情况下使用单独的 AFT 部署进行测试。
- 范围：仅使用 AFT 来部署基础设施护栏和定制。请勿使用它来部署应用程序。
- Linting 和验证：AFT 管道需要经过精心设计且经过验证的 Terraform 配置。在将配置推送到 AFT 存储库之前，运行 lint、验证和测试。
- Terraform 模块：将可重复使用的 Terraform 代码构建为模块，并始终指定 Terraform 和 AWS 提供者版本以满足组织的要求。

操作说明

设置和配置 AWS 环境

Task	描述	所需技能
准备 AWS Control Tower 环境。	在您的环境 AWS Control Tower 中进行设置和配置，以确保对您的 AWS 环境进行集中管理和治理 AWS 账	云管理员

Task	描述	所需技能
	户。有关更多信息，请参阅 AWS Control Tower 文档 AWS Control Tower中的入门。	
启动 AFT 管理账户。	使用 Ac AWS Control Tower count Factory 启动一个新的 AWS 账户 账户作为你的 AFT 管理账户。有关更多信息，请参阅 AWS Control Tower 文档中的使用 A ccount AWS Service Catalog Factory 配置账户。	云管理员

在管理账户中部署 CloudFormation 模板

Task	描述	所需技能
启动 CloudFormation 模板。	在这篇长篇故事中，您将部署此解决方案随附的 CloudFormation 模板，在您的 AWS 管理账户中设置 AFT 引导程序管道。该管道在你在上一篇故事中设置的 AFT 管理账户中部署 AFT 解决方案。 步骤 1：打开 AWS CloudFormation 控制台 • 登录 AWS Management Console 并打开AWS CloudFormation 控制台。确保您在正确 AWS Control Tower 的主区域内操作。 第 2 步：创建新堆栈	云管理员

Task	描述	所需技能
	1. 选择创建新堆栈。 2. 选择上传模板文件的选项，然后上传随此模式提供的CloudFormation 模板。 步骤 3：配置堆栈参数 • `VCS Provider`：选择要使用的版本控制系统 (VCS) 提供商。您可以选择外部 VCS GitHub，例如，CodeCommit 如果您的账户被允许使用该服务，则可以使用。 • `Repository Name` ：指定用于存储 AFT 引导模块的存储库名称。对于外部 VCS 提供商，请使用完整路径，包括组织名称（例如`my-github-org/my-repo` ）。 • `Branch Name`：指定源存储库分支。 • `CodeBuild Docker Image`：选择要用作 CodeBuild Docker 基础镜像的文件。 • 如果您将 VCS 提供程序设置为以外的选项 CodeCommit，请转到步骤 8。 第 4 步：决定文件生成	

Task	描述	所需技能

	• 如果您选择 CodeCommit 作为 VCS 提供商，则可以使用`Generate AFT Files`参数来控制默认 AFT 部署文件的生成。将此参数设置为：
	  • `true`自动创建 AFT 部署文件并将其存储在指定的存储库中。
	  • `false`如果你想手动处理文件创建或者文件已经准备就绪。
	• 如果您已选择`false`，请转至步骤 8；否则，请先执行步骤 5—7。
	
	第 5 步：填写 AWS Control Tower 和 AFT 账户详细信息
	
	• 如果将`Generate AFT Files`参数设置为`true`，请提供以下 AWS Control Tower 和 AFT 账户特定信息。
	  • `Log Archive Account ID`：中的日志存档账户 ID 的 ID AWS Control Tower。
	  • `Audit Account ID`：中审核账户的 ID AWS Control Tower。
	  • `AFT Management Account ID`: 您在第一部	

Task	描述	所需技能
	长篇故事中创建的 AFT 管理账户的 ID。 • `AFT Main Region`和`AFT Secondary Region`：AFT部署的主要和 AWS 区域 次要的。 步骤 6：配置 AFT 选项 • 设置指标报告： • `AFT Enable Metrics Reporting` ：启用或禁用 AFT 指标报告。有关更多信息，请参阅 AWS Control Tower 文档中的运营指标。 • 设置 AFT 功能选项： • `Enable AFT CloudTrail Data Events`：在所有 AFT 托管账户中启用 CloudTrail 数据事件。有关更多信息，请参阅 AWS Control Tower 文档中的AWS CloudTrail 数据事件。 • `Enable AFT Enterprise Support`：在所有 AFT 托管账户中启用企业支持。有关更多信息，请参阅 AWS Control Tower 文	

Task	描述	所需技能
	档中的 E AWS nterprise Support 计划。 • `Enable AFT Delete Default VPC`：仅删除 AFT 管理账户 VPCs 中的所有内容。有关更多信息，请参阅 AWS Control Tower 文档中的删除 AWS 默认 VPC。 步骤 7：指定版本 • `AFT Terraform Version`：选择要在 AFT 管道中使用的 Terraform 版本。 • `AFT Version`：定义要部署的 AFT 版本。保留默认设置 (`latest`) 以使用最新的 AFT 版本。 第 8 步：查看并创建堆栈 • 查看所有参数和设置。如果一切正常，请继续创建堆栈。 步骤 9：监控堆栈创建 • AWS CloudFormation 置备和配置您定义的资源。在 CloudFormation 控制台上监控堆栈创建过程。此过程可能需要几分钟。	

Task	描述	所需技能
	步骤 10：验证部署 • 当堆栈状态显示为 CREATE_COMPLET E 时，请确认所有资源均已正确创建。 • 在 “输出” 部分中，记下该`TerraformBackendBucketName` 值。	

填充并验证 AFT 引导存储库和管道

Task	描述	所需技能
选项 1：为外部 VCS 填充 AFT 引导存储库。	如果您将 VCS 提供程序设置为外部 VCS（不是 CodeCommit），请按照以下步骤操作。 （可选）部署 CloudFormation 模板后，您可以填充或验证新创建的 AFT bootstrap 存储库中的内容，并测试管道是否已成功运行。 步骤 1：更新连接 1. 在CodePipeline 控制台的导航窗格中，选择设置，连接。 2. 选择`aft-vcs-connection` 连接。它应该处于`Pending`状态。 3. 选择 “更新待定连接”，然后按照开发者工具控制台文档	云管理员

Task	描述	所需技能

	中更新待处理连接中的说明进行操作。
	4. 当连接处于Available 状态时，进入下一步。
	
	步骤 2：填充存储库
	
	1. 使用您的外部 VCS 凭据将您在模板中指定的存储库克隆到本地计算机。如果您保留默认名称，则会调用存储库aft-setup 。
	2. 在存储库中，创建一个名为的文件夹，里面terraform 有两个空文件：backend.tf 和main.tf。
	3. 打开backend.tf 文件并添加以下代码片段：
	
	`terraform {`
	`  backend "s3" {`
	`  region = "<aft-main-region>"`
	`  bucket = "<s3-bucket-name>"`
	`  key  = "aft-setup"`
	`  }`
	`}`
	
	在文件中：
	• <aft-main-region>替换为主 AFT 区域。这应该与 AWS	

Task	描述	所需技能
	Control Tower 主区域相匹配。 • `<s3-bucket-name>` 替换为 Terraform 后端存储桶的名称。你可以在之前部署的 CloudFormation 模板生成的`Terraform BackendBucketName` 输出中找到这一点。 4. 打开`main.tf`文件并使用 AFT 存储库中提供的示例之一来部署 AFT。例如，您可以与首选的 VCS 提供商（CodeCommit GitHub、或 Bitbucket）合作或自定义 AFT VPC。有关更多 AFT 输入选项，请参阅 AFT 存储库中的自述文件。 第 2 步：提交并推送您的更改 • 创建并填充文件夹和文件后，确认所做的更改，然后将代码上传到存储库。管道会自动启动，贯穿源代码和构建阶段，然后在部署阶段之前等待批准操作。	

Task	描述	所需技能
选项 2：填充的 AFT 引导存储库。 CodeCommit	如果您将 VCS 提供程序设置为 CodeCommit，请按照以下步骤操作。 （可选）部署 CloudFormation 模板后，您可以填充或验证新创建的 AFT bootstrap 存储库中的内容，并测试管道是否已成功运行。 如果将`Generate AFT Files`参数设置为`true`，请跳至下一个故事（验证管道）。 步骤 1：填充存储库 1. 打开AWS CodeCommit 控制台并选择新创建的存储库。如果保留默认名称，则存储库名称应为`aft-setup`。 2. 使用 SSH、HTTPS 或 HTTPS (GRC) 将存储库克隆到本地计算机，然后在编辑器中将其打开。 3. 创建一个名为的文件夹`terraform`，并在其中创建两个空文件：`backend.tf` 和`main.tf`。 4. 打开`backend.tf` 文件并添加以下代码片段： `terraform { backend "s3" {`	云管理员

Task	描述	所需技能

	<pre>region = "<aft-main-region>"
	bucket = "<s3-bucket-name>"
	key = "aft-setup"
	}
	}</pre>
	在文件中：
	• <aft-main-region>替换为主 AFT 区域。这应该与 AWS Control Tower 主区域相匹配。
	• <s3-bucket-name> 替换为 Terraform 后端存储桶的名称。你可以在之前部署的 CloudFormation 模板生成的TerraformBackendBucketName 输出中找到这一点。
	5. 打开main.tf文件并使用 AFT 存储库中提供的示例之一来部署 AFT。例如，您可以与首选版本控制系统 (VCS) 提供商（CodeCommit GitHub、或 Bitbucket）合作或自定义 AFT VPC。有关更多 AFT 输入选项，请参阅 AFT 存储库中的自述文件。
	
	第 2 步：提交并推送您的更改	

Task	描述	所需技能
	• 创建并填充文件夹和文件后，确认所做的更改，然后将代码上传到存储库。管道会自动启动，贯穿源代码和构建阶段，然后在部署阶段之前等待批准操作。	

Task	描述	所需技能
验证 AFT 引导管道。	步骤 1：查看管道 • 打开CodePipeline 控制台并检查`aft-bootstrap-pipeline` 管道是否已成功启动。它应该在运行 Terraform 计划或等待手动批准操作。 第 2 步：批准 Terraform 计划结果 • 您可以通过查看构建阶段的执行日志来查看 Terraform 计划的结果，然后在批准阶段批准或拒绝执行。如果您批准，管道将开始在提供的 AFT 管理账户中部署 AFT 资源。 步骤 3：等待部署 • 等待管道成功运行。这应该需要大约 30 分钟。您可能遇到的任何故障通常都是由 API 配额造成的。在这些情况下，您可以重新运行管道以继续部署。 步骤 4：检查已创建的资源 • 访问 AFT 管理账户并确认资源已创建。	云管理员

故障排除

事务	解决方案
CloudFormation 模板中包含的自定义 Lambda 函数在部署期间失败。	检查亚马逊 CloudWatch 日志中是否有 Lambda 函数以识别错误。这些日志提供了详细信息，可以帮助查明具体问题。确认 Lambda 函数具有必要的权限并且环境变量设置正确。
由于权限不足，您在创建或管理资源时会遇到故障。	查看附加到 Lambda 函数的 IAM 角色和策略 CodeBuild，以及部署中涉及的其他服务。确认他们拥有必要的权限。如果存在权限问题，请调整 IAM 策略以授予所需的访问权限。
您使用的 CloudFormation 模板版本已过时，且版本较新 AWS 服务 或 Terraform。	定期更新 CloudFormation 模板以使其与最新版本 AWS 和 Terraform 版本兼容。请查看发行说明或文档，了解任何特定于版本的更改或要求。
您在部署期间达到 AWS 服务 配额。	在部署管道之前，请检查 S3 存储桶、IAM 角色和 Lambda 函数等资源的 AWS 服务 配额。如有必要，请求会增加。有关更多信息，请参阅 AWS 网站上的AWS 服务 配额。
由于 CloudFormation 模板中的输入参数不正确，您会遇到错误。	仔细检查所有输入参数是否有错别字或值不正确。确认资源标识符（例如账户 IDs 和区域名称）准确无误。

相关资源

要成功实现此模式，请查看以下资源。这些资源提供了额外的信息和指导，对于通过使用来设置和管理AFT可能非常宝贵 AWS CloudFormation。

AWS文档：

- AWS Control Tower 《用户指南》提供了有关设置和管理的详细信息 AWS Control Tower。
- AWS CloudFormation 文档提供了对 CloudFormation 模板、堆栈和资源管理的见解。

IAM 策略和最佳实践：

- IAM 中的安全最佳实践解释了如何使用 IAM 角色和策略来帮助保护 AWS 资源。

Terraform 开启： AWS

- Terraform P AWS rovider 文档提供了有关将 Terraform 与配合使用的全面信息。 AWS

AWS 服务 配额：

- AWS 服务 q@@ u otas 提供有关如何查看 AWS 服务 配额以及如何请求增加配额的信息。

管理多个 Amazon Web Services account 和 AWS 区域中的 AWS Service Catalog 产品

创建者：Ram Kandaswamy (AWS)

摘要

Amazon Web Services (AWS) Service Catalog 简化并加速了企业基础设施即代码（IaC）模板的治理和分配。您可以使用 AWS CloudFormation 模板来定义产品所需的 AWS 资源（堆栈）集合。AWS CloudFormation StackSets 扩展了此功能，使您能够通过单个操作在多个账户和 AWS 区域中创建、更新或删除堆栈。

AWS Service Catalog 管理员使用开发人员编写的 CloudFormation 模板创建产品并进行发布。然后将这些产品与产品组合相关联，并对治理施加约束。为了向其他 AWS 账户或组织单位 (OUs) 中的用户提供您的产品，您通常会与他们共享您的产品组合。此模式描述了一种管理基于 AWS 的 AWS Service Catalog 产品的替代方法 CloudFormation StackSets。您可以使用堆栈集约束来设置可以部署和使用您的产品的 AWS 区域和账户，而不是共享产品组合。通过使用这种方法，您可以在多个账户和 AWS 区域中配置您的 AWS Service Catalog 产品，并从中心位置对其进行管理，同时满足您的监管要求。OUs

这种方法的优点：

- 该产品从主账户进行预置和管理，不与其他账户共享。
- 此方法提供了基于特定产品的所有预置产品（堆栈）的整合视图。
- 使用 Amazon Web Services 管理连接器进行配置更简单，因为它仅针对一个账户。
- 您可以更轻松地查询和使用 AWS Service Catalog 中的产品。

先决条件和限制

先决条件

- 适用于 IaC 和版本控制的 AWS CloudFormation 模板
- 用于预置和管理 AWS 资源的多账户设置和 AWS Service Catalog

限制

- 此方法使用 AWS CloudFormation StackSets，其局限性 StackSets 适用：

  - StackSets 不支持通过宏部署 CloudFormation 模板。如果您使用宏来预处理模板，则无法使用 StackSets基于基础的部署。
  - StackSets 提供了取消堆栈与堆栈集关联的功能，因此您可以瞄准特定的堆栈来修复问题。但是，已解除关联的堆栈不能再与堆栈集重新关联。
- AWS Service Catalog 会自动生成 StackSet 名称。当前不支持自定义。

架构

目标架构

1. 用户创建一个 AWS CloudFormation 模板来配置 AWS 资源，采用 JSON 或 YAML 格式。
2. 该 CloudFormation 模板在 AWS Service Catalog 中创建产品，然后将其添加到产品组合中。
3. 用户创建预配置产品，在目标账户中创建 CloudFormation 堆栈。
4. 每个堆栈都预置 CloudFormation 模板中指定的资源。

工具

Amazon Web Services

- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。
- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Service Catalog 可帮助您集中管理获准在 AWS 上使用的 IT 服务目录。最终用户可在遵循组织设定约束的情况下快速部署他们所需已获得批准的 IT 服务。

操作说明

跨账户预置产品

Task	描述	所需技能
创建产品组合。	产品组合是一个容器，其中包含一个或多个根据特定条件组合在一起的产品。使用产品组合可以帮助您在整个产品集中应用常见约束。 要创建产品组合，请按照 AWS Service Catalog 文档中的说明操作。如果您使用的是 AWS CLI，下面是一个示例命令： `aws servicecatalog create-portfolio --provider-name my-provider --display-name my-portfolio` 有关更多信息，请参阅 AWS CLI 文档。	AWS Service Catalog，IAM
创建 CloudFormation 模板。	创建描述资源的 CloudFormation 模板。在适用的情况下，应对资源属性值进行参数化。	AWS CloudFormation、JSON/YAML
使用版本信息创建产品。	当您在 AWS Service Catalog 中发布 CloudFormation 模板时，该模板就会变成产品。为可选的版本详细信息参数提供值，例如版本标题和描述；这将有助于以后查询产品。 要创建产品，请按照 AWS Service Catalog 文档中的说明	AWS Service Catalog

Task	描述	所需技能
	操作。如果您使用的是 AWS CLI，则示例命令为： `aws servicecatalog create-product --cli-input-json file://create-product-input.json` 其中 `create-product-input.json` 是传递产品参数的文件。有关此文件的示例，请参阅其他信息部分。有关更多信息，请参阅 AWS CLI 文档。	
应用约束。	将堆栈集约束应用于产品组合，以配置产品部署选项，例如多个 Amazon Web Services account 、区域和权限。有关说明，请参阅 AWS Service Catalog 文档。	AWS Service Catalog

Task	描述	所需技能
添加权限	为用户提供权限，以便他们可以启动产品组合中的产品。有关控制台说明，请参阅 AWS Service Catalog 文档。如果您使用的是 AWS CLI，下面是一个示例命令： `aws servicecatalog associate-principal-with-portfolio \ --portfolio-id port-2s6abcdefwdh4 \ --principal-arn arn:aws:iam::444455556666:role/Admin \ --principal-type IAM` 有关更多信息，请参阅 AWS CLI 文档。	AWS Service Catalog，IAM

Task	描述	所需技能
预置产品。	预置产品是产品的资源实例。基于 CloudFormation 模板配置产品会启动 CloudFormation 堆栈及其底层资源。 根据堆栈集约束，通过以适用的 AWS 区域和账户为目标来预置产品。在 AWS CLI 中，示例命令如下： `aws servicecatalog provision-product \ --product-id prod-abcdfz3syn2rg \ --provisioning-artifact-id pa-abc347 pcsccfm \ --provisioned-product-name "mytestpp name3"` 有关更多信息，请参阅 AWS CLI 文档。	AWS Service Catalog

相关资源

参考

- AWS Service Catalog 概述
- 使用 AWS CloudFormation StackSets

教程和视频

- AWS re: Invent 2019：实现一切自动化：选项和最佳实践（视频）

其他信息

使用该`create-product`命令时，`cli-input-json`参数指向一个文件，该文件指定了产品所有者、支持电子邮件和 CloudFormation 模板详细信息等信息。配置文件示例如下：

```
{
   "Owner": "Test admin",
      "SupportDescription": "Testing",
         "Name": "SNS",
            "SupportEmail": "example@example.com",
            "ProductType": "CLOUD_FORMATION_TEMPLATE",
               "AcceptLanguage": "en",
                  "ProvisioningArtifactParameters": {
                     "Description": "SNS product",
                        "DisableTemplateValidation": true,
                           "Info": {
                              "LoadTemplateFromURL": "<url>"
                     },
                           "Name": "version 1"
}
```

将 AWS 成员账户从 AWS Organizations 迁移至 AWS Control Tower

由 Rodolfo Jr. Cerrada (AWS) 创建

摘要

此示例介绍了如何将 Amazon Web Services (AWS) 账户从 AWS Organizations 迁移至 AWS Control Tower，该账户是由管理账户管理的成员账户。通过在 AWS Control Tower 中注册账户，您可以利用预防和侦查防护机制以及简化账户管理的功能。如果您的 AWS Organizations 管理账户遭到盗用，并且您想将成员账户转移至受 AWS Control Tower 管理的新组织，则可能还需要迁移您的成员账户。

AWS Control Tower 提供了一个框架，该框架结合并集成了其他几种 Amazon Web Services（包括 AWS Organizations）的功能，可确保您的多账户环境中一致的合规和治理。借助 AWS Control Tower，您可遵循一组用于扩展 AWS Organizations 能力的规定规则和定义。例如，您可以使用防防护机制确保创建安全日志和必要的跨账户存取权限，且不会对其进行更改。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 在 AWS Organizations 中的目标组织中设置 AWS Control Tower（有关说明，请参阅 AWS Control Tower 文档中的设置）
- AWS Control Tower 的管理员证书（该AWSControlTowerAdmins小组的成员）
- 源 Amazon Web Services account 的管理员凭证

限制

- AWS Organizations 中的源管理账户必须与 AWS Control Tower 中的目标管理账户不同。

产品版本

- AWS Control Tower 第 2.3 版（2020 年 2 月）或更高版本（参阅发行说明）

架构

下图阐明了迁移过程和参考架构。这种模式会将 Amazon Web Services account 从源组织迁移至受 AWS Control Tower 管理的目标组织。

注册过程包含以下步骤：

1. 该账户退出了 AWS Organizations 中的源组织。
2. 账户成为独立账户。这意味着它不属于任何组织，因此管理与账单由账户管理员独立管理。
3. 目标组织向该账户发送加入此组织的邀请。
4. 独立账户接受邀请，并成为目标组织成员。
5. 该账户已在 AWS Control Tower 中注册，并移至注册的组织单位 (OU)。（我们建议您查看 AWS Control Tower 控制面板，以确认注册。）此时，在注册的 OU 中启用的所有防护机制都将生效。

工具

Amazon Web Services

- AWS Organizations 是一项账户管理服务，可让您将多个 Amazon Web Services account 整合到您创建并集中管理的单个实体（组织）中。
- AWS Control Tower 集成了其他服务的功能，包括 AWS Organizations、AWS IAM Identity Center（AWS Single Sign-On 的后续任务）和 AWS Service Catalog，可帮助您在 Amazon Web Services Cloud 中的所有组织和账户中大规模执行和管理安全、运营和合规监管规则。

操作说明

从源组织删除成员账户

Task	描述	所需技能
验证成员账户是否可作为独立账户运行。	确认将离开来源组织的成员账户拥有独立账户运营所需信息。例如，如果成员账户无账单信息，则无法作为独立账户运行，因为 AWS 使用付款信息对账户未关联到组织期间发生的任何可计费 AWS 活动收费。	账户管理员

Task	描述	所需技能
	通常，如果您使用 AWS Organizations 控制台、API 或 AWS 命令行界面（CLI）命令创建成员账户，系统将不会自动收集独立账户所需信息。要添加此信息，请登录账户，并指定支持计划、联系信息以及付款方式。 有关从组织中删除账户之前需要了解的更多信息，请参阅 AWS Organizations 文档中的从组织中移除账户前。	

Task	描述	所需技能
将成员账户从其来源组织移除。	按照 AWS Organizations 文档说明，从组织中移除成员账户。您可以登录组织管理账户并移除成员账户，也可以登录成员账户并离开组织。 如果您没有管理员级证书可供删除或退出账户，请向组织管理员寻求帮助。 如果成员账户缺少支持计划、联系信息或者付款信息，系统将提示您提供并验证该信息。 当您离开组织时，系统将您重定向到 AWS Organizations 的开始页面，在其中可以查看您的账户加入其他组织的邀请。 **Important** 此时，您的账户是一个独立账户。如果您运行的工作负载不在 AWS Free Tier 范围内，则将会根据您为账户提供的付款和账单信息向您收费。	管理账户管理员或者账户管理员
验证成员账户是否不再是源组织的一部分。	在 AWS Organizations 控制台，您不应再看到离开组织按钮。相反，您应该看到来自其他组织的待处理邀请（如有）。	账户管理员

Task	描述	所需技能
从您离开的组织删除授予访问您账户的权限的 IAM 角色。	当您从来源组织中删除账户时，由 AWS Organizations 或管理员创建的 AWS Identity and Access Management (IAM) 角色不会自动删除。如果要终止从源组织管理账户访问的权限，则必须手动删除 IAM 角色。有关更多信息，请参阅 IAM 文档中的删除角色或实例配置文件。 当成员账户离开组织时，所有附加到该账户的标签都将被删除。独立账户不支持此标签。	账户管理员

邀请账户通过 AWS Control Tower 加入新组织

Task	描述	所需技能
登录 AWS Control Tower。	以管理员身份登录 AWS Control Tower 控制台。 目前，无法直接将 Amazon Web Services account 从源组织转移至受 AWS Control Tower 管理的 OU 中的组织。但是，当您将现有 Amazon Web Services account 注册至已由 AWS Control Tower 管理的 OU 时，您可以将 AWS Control Tower 的监管范围扩展到该账户。因此，您必须登录 AWS Control Tower 才算完成此步骤。	AWS Control Tower 管理员

Task	描述	所需技能
邀请成员账户。	1. 登录 AWS Organizations 控制台，然后导航至 Amazon Web Services account 页面。 2. 在添加 Amazon Web Services account 页面，选择邀请现有 Amazon Web Services account。 3. 填写账户信息，包括 12 位数的账号（不带破折号）以及可选的描述和标签，然后选择发送邀请。 ⚠ Important 确认账户转移不会影响任何应用程序或网络连接。 此操作中会发送一封邀请电子邮件，其中包含成员账户链接。当账户管理员点击链接并接受邀请时，成员账户将在 Amazon Web Services account 页面中显示。有关加入组织的更多信息，请参阅 AWS Organizations 文档中的邀请 Amazon Web Services account 加入您的组织。	AWS Control Tower 管理员

Task	描述	所需技能
测试应用程序和连接。	当成员账户在新组织中注册后，它会出现在根目录下的 OU 。它还会出现在 AWS Control Tower 控制台，被标记为未注册账户，因为它尚未在 AWS Control Tower 注册的 OU 中注册。 请验证以下内容： • 查看 AWS Control Tower 控制面板，查看是否存在任何防护机制违规行为。 • 检查网络连接（VPN 或 AWS Direct Connect），确保其不受传输的影响。 • （应用程序所有者）测试与此账户关联的应用程序，以验证它们是否按预期运行，并且依赖项没有受到账户转移的影响。	AWS Control Tower 管理员、成员账户管理员、应用程序所有者

为注册做好账户准备

Task	描述	所需技能
检查防护机制，并修复任何违规行为。	检查目标 OU 中定义的防护机制，特别是预防性防护机制，并修复任何违规行为。 设置 AWS Control Tower 登录区时，默认情况下启用许多强制性的预防防护机制。不得禁用。注册账户之前，您必须查看这些强制性防护机制并	AWS Control Tower 管理员、成员账户管理员

Task	描述	所需技能
	修复成员账户（手动或使用脚本）。 **Note** 预防性护栏可保持 AWS Control Tower 注册账户的合规性，并防止违反政策。任何违反预防性防护机制的行为都可能影响注册。成功注册后，如果检测到此行为，则将在 AWS Control Tower 控制面板显示侦查防护机制违规行为。不影响注册流程。有关更多信息，请参阅 AWS 文档中的 AWS Control Tower 中的防护机制。	
修复防护机制违规问题后，请检查是否存在连接问题。	在某些情况下，您可能必须关闭特定端口或禁用服务，才能修复防护机制违规行为。注册账户之前，请确保对使用这些端口和服务的应用程序进行修复。	应用程序所有者

注册 AWS Control Tower 账户

Task	描述	所需技能
登录 AWS Control Tower 控制台。	使用具有 AWS Control Tower 管理权限的登录凭证。请勿使	AWS Control Tower 管理员

Task	描述	所需技能
	用根用户（管理账户）证书注册 AWS Organizations 账户。将显示错误消息。	
注册账户。	1. 在 AWS Control Tower 的 Account Factory 页面，选择注册账户。 2. 填写详细信息，包括与您要注册的账户关联的电子邮件地址、AWS Control Tower 中显示的名称、IAM Identity Center 电子邮件地址、账户所有者的名字和姓氏、以及您要注册账户的 OU。IAM Identity Center 电子邮件地址是您的首选用户电子邮件地址。您可使用与账户电子邮件地址相同的电子邮件地址。 3. 选择注册账户。 有关更多信息，请参阅 AWS Control Tower 文档中的注册现有账户。	AWS Control Tower 管理员

注册后验证账户

Task	描述	所需技能
验证账户。	在 AWS Control Tower 中，选择账户。您刚刚注册的账户的初始状态为正在注册。注册完成后，其状态会变为已注册。	AWS Control Tower 管理员、成员账户管理员

Task	描述	所需技能
检查是否存在防护机制违规行为。	OU 中定义的防护机制将自动应用于已注册会员账户。监控 AWS Control Tower 控制面板是否存在违规行为，并进行相应的修复。有关更多信息，请参阅 AWS 文档中的 AWS Control Tower 中的防护机制。	AWS Control Tower 管理员、成员账户管理员

故障排除

事务	解决方案
您会收到错误消息：发生未知错误。请稍后重试，或联系 Amazon Web Services Support。	当您在 AWS Control Tower 中使用根用户证书（管理账户）注册新账户时，就会发生此错误。AWS Service Catalog 无法将 Account Factory 产品组合或产品映射至根用户，这会导致错误消息。若要纠正此错误，请使用非 root、具有完全访问权限的用户（管理员）凭证注册新账户。有关如何为管理用户分配管理访问权限的更多信息，请参阅 AWS IAM Identity Center（AWS Single Sign-On 后续任务）文档中的入门。
AWS Control Tower 活动页面显示了获取灾难性漂移操作。	此操作反映了对服务的偏差检查，不表示 AWS Control Tower 设置存在任何问题。无需采取行动。

相关资源

文档

- AWS Organizations 术语和概念（AWS Organizations 文档）
- 什么是 AWS Control Tower？（AWS Control Tower 文档）
- 从组织中移除成员账户（AWS Organizations 文档）

- 在 AWS Control Tower 中创建管理员账户（AWS Control Tower 文档）

教程和视频

- AWS Control Tower 研讨会（自定进度研讨会）
- 什么是 AWS Control Tower？（视频）
- 在 AWS Control Tower 中配置用户（视频）
- 为现有组织启用 AWS Control Tower（视频）

使用 AWS 服务监控 SAP RHEL Pacemaker 集群

由 Harsh Thoria (AWS)、Randy Germann (AWS) 和 RAVEENDRA Voore (AWS) 创作

摘要

此模式概述了使用亚马逊和 CloudWatch 亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 监控和配置适用于 SAP 应用程序和 SAP HANA 数据库服务的红帽企业 Linux (RHEL) Pacemaker 集群的警报的步骤。

该配置使您能够借助 CloudWatch 日志流、指标筛选器和警报监控 SAP SCS 或 ASCS、入队复制服务器 (ERS) 和 SAP HANA 集群资源处于 “已停止” 状态时。Amazon SNS 向基础设施或 SAP Basis 团队发送一封关于已停止集群状态的电子邮件。

您可以使用 AWS CloudFormation 脚本或 AWS 服务控制台为此模式创建 AWS 资源。此模式假设您使用的是控制台；它不提供 CloudFormation 脚本或涵盖 CloudWatch 和 Amazon SNS 的基础设施部署。Pacemaker 命令用于设置集群警报配置。

先决条件和限制

先决条件

- 一个活跃的 AWS 账户。
- Amazon SNS 设置为发送电子邮件或移动通知。
- ASCS/ERS for ABAP or SCS/ERS适用于 Java 的 SAP 和 SAP HANA 数据库 RHEL Pacemaker 集群。有关说明，请参阅：
  - SAP HANA 集群设置
  - SAP Netweaver abap/Java 集群设置

限制

- 该解决方案目前适用于 RHEL 版本 7.3 及更高版本的基于 Pacemaker 的集群。它尚未在 SUSE 操作系统上进行过测试。

产品版本

- RHEL 7.3 及更高版本

架构

目标技术堆栈

- RHEL Pacemaker 警报事件驱动型代理
- 亚马逊弹性计算云（亚马逊 EC2）
- CloudWatch 警报
- CloudWatch 日志组和指标筛选器
- Amazon SNS

目标架构

下图说明了此解决方案的组件和工作流程。

自动化和扩缩

- 您可以使用 CloudFormation 脚本自动创建 AWS 资源。您还可以使用其他指标筛选器来扩展和覆盖多个集群。

工具

Amazon Web Services

- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和运行的应用程序 AWS 的指标。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。

工具

- CloudWatch agent（unified）是一种工具，用于从 EC2 实例收集系统级指标、日志和跟踪，并从您的应用程序中检索自定义指标。
- Pacemaker 警报代理（适用于 RHEL 7.3 及更高版本）是一种在 Pacemaker 群集中发生变化时启动操作的工具，例如资源停止或重新启动时。

最佳实践

- 有关在上使用 SAP 工作负载的最佳实践 AWS，请参阅 Well-Architecte AWS d Framework 的 SAP 镜头。
- 考虑为 SAP HANA 集群设置 CloudWatch 监控所涉及的成本。有关更多信息，请参阅 CloudWatch 文档。
- 考虑使用寻呼机或工单机制处理 Amazon SNS 提醒。
- 请务必检查适用于个人电脑、起搏器和围栏代理的 RPM 软件包的 RHEL 高可用性 (HA) 版本。AWS

操作说明

设置 Amazon SNS

Task	描述	所需技能
创建 SNS 主题。	1. 登录 AWS Management Console 并在 v3/home 上打开亚马逊 SNS 控制台。https://console.aws.amazon.com/sns/ 2. 在 Amazon SNS 控制面板上的 Common actions（常用操作）下，选择 Create Topic（创建主题）。 3. 在“创建新主题”对话框中，为“类型”选择“标准”。 4. 在主题名称中，输入主题的名称（例如，`my-topic`）。 5. 选择创建主题。 这将创建一个 SNS 主题，其资源策略允许您发布通知。	AWS 管理员

Task	描述	所需技能
	6. 复制主题 ARN（例如）。`arn:aws:sns:us-east-1:111122223333:my-topic` 您将在后面的步骤中使用此 ARN。	

Task	描述	所需技能
修改 SNS 主题的访问策略。	1. 在 Amazon SNS 控制台的导航窗格中，选择主题，然后选择您创建的主题。 2. 选择编辑并转到访问策略部分。 3. 确保访问策略中包含 CloudWatch 允许发布到此主题的服务主体之一。例如： `{ "Sid": "Allow AWS CloudWatch to Publish to this SNS topic", "Effect": "Allow", "Principal": { "Service": [ "cloudwatch.amazonaws.com" ] }, "Action": "SNS:Publish", "Resource": "arn:aws:sns:us-east-1:111122223333:my-topic" }` 4. 选择 Save changes（保存更改）。	AWS 系统管理员

Task	描述	所需技能
订阅 SNS 主题。	1. 在 Amazon SNS 控制台的导航窗格中，选择订阅，创建订阅。 2. 对于主题 ARN，粘贴您在第一个任务中创建的 ARN。 3. 对于协议，选择电子邮件。 4. 在 En d point 中，输入负责 SAP Pacemaker 集群并应接收通知的人员或团队的电子邮件地址。例如，这可以是 SAP Basis 或基础架构团队的通讯组列表的电子邮件地址。 5. 选择创建订阅。 6. 在您的电子邮件应用程序中，打开 AWS 通知中的消息并确认您的订阅。 您的 Web 浏览器将显示来自 Amazon SNS 的确认响应。	AWS 系统管理员

确认集群的设置

Task	描述	所需技能
检查集群状态。	使用 pcs status 命令确认资源处于联机状态。	SAP Basis 管理员

配置 Pacemaker 警报

Task	描述	所需技能

在主群集实例上配置 Pacemaker 警报代理。	登录主集群中的 EC2 实例并运行以下命令：
	`install --mode=0755 /usr/share/pacemaker/alerts/alert_file.sh.sample`
	`touch /var/lib/pacemaker/alert_file.sh`
	`touch /var/log/pcmk_alert_file.log`
	`chown hacluster:haclient /var/log/pcmk_alert_file.log`
	`chmod 600 /var/log/pcmk_alert_file.log`
	`pcs alert create id=alert_file description="Log events to a file." path=/var/lib/pacemaker/alert_file.sh`
	`pcs alert recipient add alert_file id=my-alert_logfile value=/var/log/pcmk_alert_file.log`
		SAP Basis 管理员

在辅助群集实例上配置 Pacemaker 警报代理。	登录辅助群集中的辅助群集 EC2 实例并运行以下命令： `install --mode=0755 /usr/share/pacemaker/alerts/alert_file.sh.sample`	SAP Basis 管理员

Task	描述	所需技能
	touch /var/lib/pacemaker/alert_file.sh touch /var/log/pcmk_alert_file.log chown hacluster:haclient /var/log/pcmk_alert_file.log chmod 600 /var/log/pcmk_alert_file.log	

确认已创建 RHEL 警报资源。	使用以下命令确认警报资源已创建：
	`pcs alert`
	命令的输出将如下所示：
	`[root@xxxxxxx ~]# pcs alert`
	`Alerts:`
	` Alert: alert_file (path=/var/lib/pacemaker/alert_file.sh)`
	`  Description: Log events to a file.`
	`  Recipients:`
	`   Recipient: my-alert_logfile (value=/var/log/pcmk_alert_file.log)`
		SAP Basis 管理员

配置代 CloudWatch 理

Task	描述	所需技能

安装代 CloudWatch 理。	有几种方法可以在 EC2 实例上安装 CloudWatch 代理。要使用命令行，请执行以下操作：
	1. 下载 CloudWatch 代理软件包：
	`wget https://s3.<region>.amazonaws.com/amazoncloudwatch-agent-region/redhat/amd64/latest/amazon-cloudwatch-agent.rpm`
	其中`<region>`是 EC2 实例 AWS 区域 所在的位置（例如，`us-west-2` ）。
	2. （可选）验证包签名。有关说明，请参阅 CloudWatch 文档中的验证 CloudWatch 代理软件包的签名。
	3. 在第一个实例上安装软件包：
	`sudo rpm -U ./amazon-cloudwatch-agent.rpm`
	4. 对辅助实例重复此操作。
	
	有关更多信息，请参阅 CloudWatch 文档。
		AWS 系统管理员

Task	描述	所需技能
将 IAM 角色附加到 EC2 实例。	要使 CloudWatch 代理能够从实例发送数据，您必须将 IAM CloudWatchAgentServerRole角色附加到每个实例。或者，您可以将 CloudWatch 代理策略添加到现有的 IAM 角色中。有关更多信息，请参阅 CloudWatch 文档。	AWS 管理员
将 CloudWatch 代理配置为监视主群集实例上的 Pacemaker 警报代理日志文件。	1. 通过运行以下命令配置主集群实例： `sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-config-wizard` 2. 对于 Linux 选择 1，然后为您的监控策略选择选项。 3. 对于 “是否要监视任何日志文件” 的问题，选择 “是”，然后提供 p cs 警报命令中的 Pacemaker 日志文件的路径。就我们而言，确实如此`var/log/pcmk_alert_file.log`。 4. 提供日志组和日志流的名称。如果您未指定日志流，则使用 AWS 实例 ID 作为默认值。 5. 对辅助群集实例重复步骤 1-4。	AWS 管理员

Task	描述	所需技能
在主群集和辅助群集实例上启动 CloudWatch 代理。	要启动代理，请在主群集和辅助群集中的 EC2 实例上运行以下命令： `sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a fetch-config -m ec2 -s -c file:/opt/aws/amazon-cloudwatch-agent/bin/config.json`	AWS 管理员

设置 CloudWatch 资源

Task	描述	所需技能
设置 CloudWatch 日志组。	1. 在以下位置打开 CloudWatch 控制台 https://console.aws.amazon.com/cloudwatch/ 2. 在导航窗格中，选择日志组，创建日志组。 3. 输入日志组的名称，然后选择创建日志组。 CloudWatch 代理会将 Pacemaker 警报文件作为 CloudWatch 日志流传输到日志组。	AWS 管理员
设置 CloudWatch 指标筛选条件。	指标筛选器可帮助您搜索模式，例如`stop <cluster-resource-name>` 在	AWS 管理员、SAP Basis 管理员

Task	描述	所需技能
	CloudWatch 日志流中。识别出此模式后，指标筛选器会更新自定义指标。 1. 在 CloudWatch 控制台的导航窗格中，选择日志组。 2. 选择您在上一个任务中创建的日志组的名称。 3. 选择 Actions（操作）、Create metric filter（创建指标筛选条件）。 4. 在筛选模式中，输入要使用的筛选模式，例如，匹配名ABC_scs为的 SAP SCS 群集资源的停止事件。`stop ABC_scs` 有关更多信息，请参阅 CloudWatch 文档中的过滤模式语法。 5. （可选）要测试您的筛选条件模式，请在 Test Pattern（测试模式）下，输入一个或多个用于测试模式的日志事件。每个日志事件都必须单独指定一行，因为在“日志事件消息”框中使用换行符来分隔日志事件。 6. 选择 Next（下一步），然后为筛选条件输入名称。 7. 在指标详细信息下，在指标命名空间中，输入要发布指标的 CloudWatch 命名空间	

Task	描述	所需技能
	的名称（例如，sapcluster_monitoring ）。如果此命名空间尚不存在，请选择 “新建”。 8. 在指标名称中，输入新指标的名称（例如sapcluster_<sid> ，其中<sid>是 SAP 系统标识名称）。 9. 在 “指标值” 中，输入 1。 或者，您可以输入令牌，例如$size。对于包含 size 字段的每个日志事件，此操作会以 size 字段中的数值为增量增加该指标。 10.在默认值中，输入 0。 11选择 Create metric filter（创建指标筛选条件）。 当指标筛选器在步骤 4 中识别出模式时，它会将 CloudWatch 自定义指标的值更新sapcluster_abc 为 1。 CloudWatch 警报会SAP-Cluster-QA1-ABC 监控该指标，sapcluster_abc 并在指标值变为 1 时发出 SNS 通知。这表示群集资源已停止，需要采取措施。	

Task	描述	所需技能
为 SAP ASCS/SCS 和 ERS CloudWatch 指标设置指标警报。	要基于单个指标创建警报，请执行以下操作： 1. 在 CloudWatch 控制台的导航窗格中，选择警报，所有警报。 2. 选择创建警报。 3. 选择 Select Metric（选择指标）。 4. 搜索在上一个任务`sapcluster_monitoring` 中创建的自定义指标。 5. 选择 SAP SCS 的指标名称（例如，`sapcluster_<abc>` ），该名称也是在上一个任务中创建的。 6. 在 “图表化指标” 选项卡上，设置以下内容： • 对于 Statistic（统计数据），选择 Maximum（最大）。 • 对于周期，选择 1 分钟。 • 对于 “阈值” 类型，选择 “静态”，并将的阈值设置为**`sapcluster_<sid>`** 大于或等于 1 的值。 7. 选择下一步。 8. 在 “通知” 中，选择您在第一篇长篇故事中创建的 SNS 主题。	AWS 管理员

Task	描述	所需技能
	9. 在名称和描述中，提供警报名称和简短描述，然后选择下一步。 10选择创建警报。	
为 SAP HANA CloudWatch 指标设置指标警报。	重复上一个任务中设置 CloudWatch 指标警报的步骤，并进行以下更改： • 在步骤 5 中，选择 SAP HANA 的指标名称（例如，`sapcluster_db_<abc>` ）。 • 对于步骤 6，将的阈值设置`sapcluster_<sid>` 为大于 0 的值。	AWS 管理员

相关资源

- 集群事件的触发脚本（RHEL 文档）
- 使用@@ 向导创建 CloudWatch 代理配置文件（CloudWatch 文档）
- 在@@ 服务器上安装和运行 CloudWatch 代理（CloudWatch 文档）
- 基于静态阈值创建 CloudWatch 警报（CloudWatch 文档）
- 使用@@ 高可用性集群在 AWS HANA 上手动部署 SAP HANA（ AWS 网站上有 SAP 文档）
- SAP NetWeaver 指南（ AWS 网站上有 SAP 文档）

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

监控多个共享的 Amazon 系统映像的使用情况 AWS 账户

由 Naveen Suthar (AWS) 和 Sandeep Gawande (AWS) 创建

摘要

亚马逊系统映像 (AMIs) 用于在您的亚马逊网络服务（EC2) 环境中创建亚马逊弹性计算云 (Amazon AWS) 实例。您可以在单独的集中式 AWS 账户 AMIs 中创建，在这种模式中该账户称为创建者账户。然后，您可以跨多个相同 AWS 账户 的 AMI 共享 AWS 区域，在此模式中这些账户称为消费者账户。通过单 AMIs 一账户进行管理可提供可扩展性并简化治理。在消费者账户中，您可以在 Amazon A EC2 uto Scaling 启动模板和亚马逊 Elastic Kubernetes Service (Amazon EKS) 节点组中引用共享的 AMI。

当共享 AMI 被弃用、取消注册或取消共享时， AWS 服务 消费者账户中引用该 AMI 的用户无法使用此 AMI 启动新实例。任何自动扩缩事件或同一实例的重新启动都失败。这可能会导致生产环境中出现问题，例如应用程序停机或性能下降。当 AMI 共享和使用事件多次发生时 AWS 账户，可能很难监控此活动。

此模式可帮助您监控同一区域中账户之间的共享 AMI 使用情况和状态。它使用无服务器 AWS 服务，例如亚马逊 EventBridge、亚马逊 AWS Lambda DynamoDB 和亚马逊简单电子邮件服务 (Amazon SES) Service。您可以使用 HashiCorp Terraform 来配置基础设施即代码 (IaC)。当消费者账户中的服务引用已取消注册或未共享的 AMI 时，此解决方案会发出警报。

先决条件和限制

先决条件

- 两个或更多活跃账户 AWS 账户：一个创作者账户和一个或多个消费者账户
- 一个或多个 AMIs 从创作者账户共享给消费者账户
- 已安装 Terraform CLI（Terraform 文档）
- Terraform P AWS rovider，已配置（Terraform 文档）
- （可选，但推荐）Terraform 后端，已配置（Terraform 文档）
- Git，已安装

限制

- 此模式使用账户 ID 监视 AMIs 已共享给特定账户的内容。此模式不会监控 AMIs 已使用组织 ID 共享给组织的内容。

- AMIs 只能共享给同一个账号 AWS 区域。这种模式 AMIs 在单个目标区域内进行监视。要监控多个区域 AMIs 的使用情况，请在每个区域部署此解决方案。
- 此模式不会监视部署此解决方案之前共享的任何 AMIs 内容。如果您想监控之前共享的 AMIs，可以取消共享 AMI，然后将其重新共享给使用者账户。

产品版本

- Terraform 版本 1.2.0 或更高版本
- Terraform P AWS rovider 版本 4.20 或更高版本

架构

目标技术堆栈

以下资源通过 Terraform 预配为 IaC：

- Amazon DynamoDB 表
- 亚马逊 EventBridge 规则
- AWS Identity and Access Management (IAM) 角色
- AWS Lambda 函数
- Amazon SES

目标架构

图表显示了以下工作流：

1. 创建者账户中的 AMI 与该账户中的消费者账户共享 AWS 区域。
2. 共享 AMI 时，创建者账户中的 EventBridge 规则会捕获该`ModifyImageAttribute`事件并在创建者账户中启动 Lambda 函数。
3. Lambda 函数将与 AMI 相关的数据存储在创建者账户的 DynamoDB 表中。
4. 当消费者账户 AWS 服务 中的用户使用共享 AMI 启动 Amazon EC2 实例，或者当共享 AMI 与启动模板关联时，消费者账户中的 EventBridge 规则会捕获共享 AMI 的使用情况。
5. 该 EventBridge 规则在使用者账户中启动 Lambda 函数。Lambda 函数执行以下操作：

   a. Lambda 函数更新消费者账户的 DynamoDB 表中与 AMI 相关的数据。
   b. Lambda 函数在创建者账户中扮演一个 IAM 角色并更新创建者账户中的 Lambda 表。在 `Mapping` 表中，它创建一个项目，用于将实例 ID 或启动模板 ID 映射到其各自的 AMI ID。
6. 在创建者账户中集中管理的 AMI 已弃用、取消注册或取消共享。
7. 创建者账户中的 EventBridge 规则通过`remove`操作捕获`ModifyImageAttribute`或`DeregisterImage`事件并启动 Lambda 函数。
8. Lambda 函数检查 DynamoDB 表，以确定 AMI 是否在任何消费者账户中使用。如果`Mapping`表中没有与 AMI IDs 关联的实例 IDs 或启动模板，则该过程已完成。
9. 如果有任何实例 IDs 或启动模板 IDs 与`Mapping`表中的 AMI 关联，则 Lambda 函数将使用 Amazon SES 向配置的订阅者发送电子邮件通知。

工具

AWS 服务

- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如， AWS Lambda 函数、使用 API 目的地的 HTTP 调用端点或其他 AWS 账户目的地的事件总线。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用您的 AWS 资源，从而帮助您安全地管理对这些资源的访问。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Email Service (Amazon SES) 可帮助您使用自己的电子邮件地址和域发送和接收电子邮件。

其他工具

- HashiCorp Terraform 是一款基础设施即代码 (IaC) 工具，可帮助您使用代码来配置和管理云基础架构和资源。
- Python 是通用的计算机编程语言。

代码存储库

此模式的代码可在 GitHub cross-account-ami-monitoring-terraform- samples 存储库中找到。

最佳实践

- 遵循使用 AWS Lambda 函数的最佳实践。
- 遵循建筑的最佳实践 AMIs。
- 在创建 IAM 角色时，请遵循最低权限原则，并授予执行任务所需最小权限。有关更多信息，请参阅 IAM 文档中的授予最低权限和安全最佳实践。
- 为这些 AWS Lambda 功能设置监控和警报。有关更多信息，请参阅 Lambda 函数监控和故障排除。

操作说明

自定义 Terraform 配置文件

Task	描述	所需技能
创建已命 AWS CLI 名的配置文件。	为创建者账户和每个消费者账户创建一个 AWS Command Line Interface (AWS CLI) 命名的个人资料。有关说明，请参阅 AWS CLI在AWS 入门资源中心进行设置。	DevOps 工程师
克隆存储库。	输入以下命令。这将使用 SSH 从中克隆 cross-account-ami-monitoring-terraform-samples 存储库。 GitHub `git clone git@github.com:aws-samples/cross-account-ami-monitoring-terraform-samples.git`	DevOps 工程师
更新 provider.tf 文件。	1. 输入以下命令以导航到克隆存储库中的 `terraform` 文件夹。	DevOps 工程师

Task	描述	所需技能
	`cd cross-account-ami-monitoring/terraform` 2. 打开 `provider.tf` 文件。 3. 按如下方式更新创建者账户和消费者账户的 Terraform AWS 提供者配置： • 对于 `alias`，输入提供程序配置的名称。 • 对于`region`，输入 AWS 区域 要部署此解决方案的目标。 • 对于`profile`，输入用于访问该帐户的 AWS CLI 指定配置文件。 4. 如果要配置多个消费者账户，请为每个其他消费者账户创建一个配置文件。 5. 保存并关闭 `provider.tf` 文件。 有关配置提供程序的详细信息，请参阅 Terraform 文档中的多个提供程序配置。	

Task	描述	所需技能
更新 terraform.tfvars 文件。	1. 打开 `terraform .tfvars` 文件。 2. 在 `account_email_mapping` 参数中，为创建者账号和消费者账号配置告警，如下所示： • 对于 `account`，输入账户 ID。 • 对于 `email`，输入要发送警报的电子邮件地址。每个账户只能输入一个电子邮件地址。 3. 如果要配置多个消费者账户，请为每个其他消费者账户输入一个账户和电子邮件地址。 4. 保存并关闭 `terraform .tfvars` 文件。	DevOps 工程师

Task	描述	所需技能
更新 main.tf 文件。	仅当将此解决方案部署到多个消费者账户时，才完成这些步骤。如果仅将此解决方案部署到一个消费者账户，则无需修改此文件。 1. 打开 `main.tf`文件。 2. 对于每个额外的消费者账户，创建一个基于模板中 `consumer_account_A` 模块的新模块。对于每个消费者账户，对于 `provider`，该值应与您在 `provider.tf` 文件中输入的别名匹配。 3. 保存并关闭 `main.tf`文件。	DevOps 工程师

使用 Terraform 部署解决方案

Task	描述	所需技能

部署解决方案。	在 Terraform CLI 中，输入以下命令以在创建者和使用者账户中部署 AWS 资源：
	1. 输入以下命令，以初始化 Terraform。
	`terraform init`
	2. 输入以下命令以验证 Terraform 配置。
	`terraform validate`
		DevOps 工程师

Task	描述	所需技能

	3. 输入以下命令以创建 Terraform 执行计划。
	`terraform plan`
	4. 查看 Terraform 计划中的配置更改，并确认要实现这些更改。
	5. 输入以下命令以部署资源。
	`terraform apply`	

验证电子邮件地址身份。	在部署 Terraform 计划时，Terraform 为 Amazon SES 中的每个消费者账户创建了一个电子邮件地址身份。在向该电子邮件地址发送通知之前，您必须验证该电子邮件地址。有关说明，请参阅 Amazon SES 文档中的验证电子邮件地址身份。	常规 AWS

验证资源部署

Task	描述	所需技能
验证创建者账户中的部署。	1. 登录创建者账户。 2. 在导航栏中，确认正在查看目标 AWS 区域。如果您位于其他区域，请选择当前显示的区域的名称，然后选择目标区域。 3. 打开 DynamoDB 控制台。 4. 在导航窗格中，选择表。	DevOps 工程师

Task	描述	所需技能
	5. 在表列表中，验证 AmiShare 表是否存在。 6. 打开 Lambda 控制台。 7. 在导航窗格中，选择函数。 8. 在函数列表中，验证 ami-share 函数是否存在。 9. 打开 IAM 控制台。 10.在导航窗格中，选择角色。 11.在角色列表中，验证 external-ddb-role 角色是否存在。 12.打开 EventBridge 管理控制台。 13.在导航窗格中，选择规则。 14.在规则列表中，验证 modify_image_attribute_event 规则是否存在。 15.打开 Amazon SES 控制台。 16.在导航窗格中，选择已验证身份。 17.在身份列表中，验证是否已为每个消费者账户注册并验证了电子邮件地址身份。	

Task	描述	所需技能
验证消费者账户中的部署。	1. 登录消费者账户。 2. 在导航栏中，确认正在查看目标 AWS 区域。如果您位于其他区域，请选择当前显示的区域的名称，然后选择目标区域。 3. 打开 DynamoDB 控制台。 4. 在导航窗格中，选择表。 5. 在表列表中，验证 `Mapping` 表是否存在。 6. 打开 Lambda 控制台。 7. 在导航窗格中，选择函数。 8. 在函数列表中，验证 `ami-usage-function` 和 `ami-deregister-function` 函数是否存在。 9. 打开 EventBridge 管理控制台。 10. 在导航窗格中，选择规则。 11. 在规则列表中，验证 `ami_usage_events` 和 `ami_deregister_events` 规则是否存在。	DevOps 工程师

验证监控

Task	描述	所需技能
在创建者账户中创建 AMI。	1. 在创建者账户中，创建私有 AMI。有关说明，请参阅通过 Amazon EC2 实例创建 AMI。	DevOps 工程师

Task	描述	所需技能
	2. 与消费者账户之一共享新的 AMI。有关说明，请参阅与特定人员共享 AMI AWS 账户。	
使用消费者账户中的 AMI。	在消费者账户中，使用共享的 AMI 创建 Amazon EC2 实例或启动模板。有关说明，请参阅如何从自定义 AMI 启动亚马逊 EC2 实例（re AWS : Post 知识中心）或为 Auto S caling 组创建启动模板（Ama EC2 zon Auto Sc aling 文档）。	DevOps 工程师
验证监控和警报。	1. 登录创建者账户。 2. 打开 Amazon EC2 控制台。 3. 在导航窗格中，请选择 AMIs。 4. 在列表中选择您的 AMI，然后选择操作、然后选择编辑 AMI 权限。 5. 在共享账户部分中，选择消费者账户，然后选择删除所选项。 6. 选择保存更改。 7. 验证您为消费者账户定义的目标电子邮件地址是否收到 AMI 共享已取消的通知。	DevOps 工程师

（可选）停止监控共享 AMIs

Task	描述	所需技能
删除资源。	1. 输入以下命令以删除此模式部署的资源并停止监视共享 AMIs。 `terraform destroy` 2. 输入 `yes` 确认 `destroy` 命令。	DevOps 工程师

故障排除

事务	解决方案
我没有收到电子邮件提醒。	未发送 Amazon SES 电子邮件的原因可能有多种。请检查以下事项： 1. 在 Epics 部分中，使用验证资源部署长篇故事来确认基础架构已全部配置正确。 AWS 账户 2. 验证 Ama CloudWatch zon 日志中的 Lambda 函数事件。有关说明，请参阅 Lambda 文档中的使用 CloudWatch 控制台。确认不存在权限问题，例如任何基于身份或基于资源的策略中的显式拒绝。有关更多信息，请参阅 IAM 文档中的策略评估逻辑。 3. 在 Amazon SES 中，验证电子邮件地址身份的状态是否为已验证。有关更多信息，请参阅验证电子邮件地址身份。

相关资源

AWS 文档

- 使用 Python 构建 Lambda 函数（Lambda 文档）
- 创建 AMI（亚马逊 EC2 文档）
- 与特定人员共享 AMI AWS 账户（亚马逊 EC2 文档）
- 注销您的 AMI（亚马逊 EC2 文档）

Terraform 文档

- 安装 Terraform
- Terraform 后端配置
- Terraform AWS Provider
- Terraform 二进制下载

在 AWS Organizations 中设置程序账户关闭警报

由 Richard Milner-Watts (AWS)、Debojit Bhadra (AWS) 和 Manav Yadav (AWS) 编写

摘要

AWS Organizations 的 CloseAccount API 使您能够以编程方式关闭组织内的成员账户，而不必使用根证书登录账户。RemoveAccountFromOrganization API 从 AWS Organizations 中的组织中提取一个账户，使其成为独立账户。

这 APIs 可能会增加可以关闭或移除 AWS 账户的操作员数量。所有通过 AWS Organizations 管理账户中的 AWS Identity and Access Management (IAM) 访问该组织的用户都可以拨打这些 APIs电话，因此访问权限不限于账户根电子邮件的所有者以及任何关联的多因素身份验证 (MFA) 设备。

此模式会在调用`CloseAccount`和`RemoveAccountFromOrganization` APIs 时发出警报，因此您可以监控这些活动。对于警报，它使用 Amazon Simple Notification Service (Amazon SNS) 主题。您还可通过 webhook 设置 Slack 通知。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- AWS Organizations 中的组织
- 访问组织根目录下的组织管理账户，以创建所需资源

限制

- 如 AWS Organizations API参考中所述，该 `CloseAccount`API 仅允许在连续30天内关闭10%的活跃成员账户。
- 关闭 Amazon Web Services account 后，其状态将更改为已暂停。在此状态转换后的 90 天内，Amazon Web Services Support 可重新开放账户。90 天后，该账户将被永久删除。
- 有权访问 AWS Organizations 管理账户并且 APIs 可能还有禁用这些提醒的权限的用户。如果主要关注的是恶意行为而不是意外删除，请考虑使用 IAM 权限边界 保护由此模式创建的资源。
- `CloseAccount` 和 `RemoveAccountFromOrganization` 的 API 调用在美国东部（弗吉尼亚州北部）(`us-east-1`) 处理。因此，您必须在 `us-east-1` 中部署此解决方案才能观察事件。

架构

目标技术堆栈

- AWS Organizations
- AWS CloudTrail
- Amazon EventBridge
- AWS Lambda
- Amazon SNS

目标架构

下图显示此模式的解决方案架构。

1. AWS Organizations 处理 `CloseAccount` 或 `RemoveAccountFromOrganization` 请求
2. EventBridge Amazon 与 AWS 集成，可将这些事件传送 CloudTrail 到默认事件总线。
3. 自定义亚马逊 EventBridge 规则与 AWS Organizations 的请求相匹配，并调用 AWS Lambda 函数。
4. Lambda 函数向 SNS 主题发送消息，用户可以订阅该消息以接收电子邮件警报或进一步处理。
5. 如果启用 Slack 通知，Lambda 函数会向 Slack 网络挂钩发送一条消息。

工具

Amazon Web Services

- AWS CloudFormation 提供了一种方法，通过将基础设施视为代码，对一组相关的 AWS 和第三方资源进行建模，快速一致地配置这些资源，并在它们的整个生命周期中对其进行管理。
- Amazon EventBridge 是一项无服务器事件总线服务，可用于将应用程序与来自各种来源的数据连接起来。 EventBridge 接收事件（环境变化的指示器），并应用规则将事件路由到目标。规则根据事件的结构（称为事件模式）或计划将事件与目标匹配。
- AWS Lambda 是一项计算服务，支持无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。您只需按使用的计算时间付费。代码不运行时不会产生任何费用。

- AWS Organizations 可帮助您在增长和扩缩您的 AWS 资源时集中管理和治理您的环境。使用 AWS Organizations，您可通过编程方式创建新的 Amazon Web Services account 并分配资源，对账户进行分组以组织您的工作流，将策略应用于账户或群组进行管理，并通过对所有账户使用单一付款方式来简化账单。
- AWS CloudTrail 监控和记录您的 AWS 基础设施中的账户活动，并允许您控制存储、分析和补救措施。
- 亚马逊简单通知服务 (Amazon SNS) Simple Not ification Service 是一项完全托管的消息服务，用于 (A2A) application-to-application 和 (A2P) application-to-person 通信。

其他工具

- 适用于 Python 的 AWS Lambda Powertools 库 是一组实用程序，可为 Lambda 函数提供跟踪、日志、指标和事件处理功能。

代码

此模式的代码位于 GitHub AWS 账户关闭通知器存储库中。

该解决方案包括一个 CloudFormation 模板，用于部署此模式的架构。它使用 AWS Lambda Powertools for Python 库来提供日志记录和跟踪。

操作说明

部署架构

Task	描述	所需技能
启动解决方案堆栈的 CloudFormation 模板。	此模式的 CloudFormation 模板位于GitHub 存储库的主分支中。它部署 IAM 角色、EventBridge 规则、Lambda 函数和 SNS 主题。 要启动模板： 1. 克隆GitHub 存储库以获取解决方案代码的副本。	AWS 管理员

Task	描述	所需技能
	2. 打开 AWS Organizations 管理账户 AWS 管理控制台。 3. 选择美国东部（弗吉尼亚北部us-east-1 ）区域 ()，然后打开CloudFormation 控制台。 4. 使用 account-closure-notifier.yml 模板并指定以下值来创建堆栈： • 堆栈名称：aws-account-closure-notifier-stack • ResourcePrefix 参数：aws-account-closure-notifier • SlackNotification 参数：如果需要 Slack 通知，请将设置更改为 true。 • SlackWebhookEndpoint 参数：如果需要 Slack 通知，请指定 webhook URL。 有关启动 CloudFormation 堆栈的更多信息，请参阅 AWS 文档。	

Task	描述	所需技能
验证解决方案是否已成功启动。	1. 等待 CloudFormation 堆栈达到 CREATE_COMPLETE 状态。 2. 在中打开EventBridge 控制台us-east-1 。 3. 确认已使用 aws-account-closure-notifier-event-rule 名称创建了新规则。	AWS 管理员
订阅 SNS 主题。	（可选）如果您想订阅 SNS 主题： 1. 在 us-east-1 中打开 Amazon SNS 控制台，找到名为 aws-account-closure-notifier-sns-topic 的主题。 2. 选择主题名称，然后选择创建订阅。 3. 对于协议，选择电子邮件。 4. 对于端点，指定应接收通知的电子邮件地址，然后选择创建订阅。 5. 查看您的电子邮件收件箱中是否包含来自 AWS 通知的消息。使用电子邮件中的链接确认订阅。 有关设置 SNS 通知的更多信息，请参阅 Amazon SNS 文档。	AWS 管理员

验证解决方案

Task	描述	所需技能
向设置事件总线发送测试事件。	GitHub 存储库提供了一个示例事件，您可以将其发送到 EventBridge 默认事件总线进行测试。该 EventBridge 规则还会对使用自定义事件源`account.closure.notifier` 的事件做出反应。 **Note** 您不能使用 CloudTrail 事件源发送此事件，因为无法将事件作为 AWS 服务发送。 要发送测试事件： 1. 在中打开EventBridge 控制台`us-east-1` 。 2. 在导航窗格的总线，选择事件总线，然后选择默认的事件总线。 3. 选择发送事件。 4. 在事件源中，输入 `account.closure.notifier` 。 5. 在详细信息类型中，输入 `AWS API Call via CloudTrail` 。 6. 要了解事件详情，请将 GitHub 存储库中的内容	AWS 管理员

Task	描述	所需技能
	复制并粘贴到文本框中。`tests/dummy-event.json` 7. 选择发送以启动通知工作流。	
确认已收到了电子邮件通知。	查看订阅 SNS 主题的邮箱，以获取通知。您应该会收到一封电子邮件，其中包含已关闭账户和执行 API 调用的主体的详细信息。	AWS 管理员
验证是否已收到了 Slack 通知。	（可选）如果您在部署 CloudFormation 模板时为`SlackWebhookEndpoint` 参数指定了 Webhook 网址，请检查映射到 webhook 的 Slack 频道。它应显示一条消息，其中包含已关闭账户和执行 API 调用的主体的详细信息。	AWS 管理员

相关资源

- CloseAccount 操作（AWS Organiations API 参考）
- RemoveAccountFromOrganization 操作（AWS Organiations API 参考）
- AWS Lambda Powertools for Python

查看您的 Amazon Web Services account 或组织 EBS 快照详情

由 Arun Chandapillai (AWS) 和 Parag Nagwekar (AWS) 编写

摘要

此模式介绍了如何自动生成包含 Amazon Web Services (AWS) 账户或 AWS Organizations 组织单位 (OU) 中所有 Amazon Elastic Block Store (Amazon EBS) 快照的按需报告。

Amazon EBS 是一项 easy-to-use专为亚马逊弹性计算云 (Amazon EC2) 设计的可扩展、高性能的块存储服务。EBS 卷提供持久的永久存储，您可以将其连接到您的 EC2 实例。您可以使用 EBS 卷作为数据的主存储，并通过创建快照来 point-in-time备份 EBS 卷。您可使用 AWS 管理控制台或 AWS 命令行界面（AWS CLI）查看特定 EBS 快照的详细信息。此模式提供了一种编程方式，检索有关您的 Amazon Web Services account 或 OU 中所有 EBS 快照的信息。

您可以使用此模式提供的脚本生成逗号分隔值 (CSV) 文件，其中包含有关每个快照的如下信息：账户 ID、快照 ID、卷 ID 和大小、拍摄快照的日期、实例 ID 和描述。如果您的 EBS 快照已被标记，则报告还会包含拥有者和团队属性。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- AWS CLI 第 2 版，已安装和配置
- 具有相应权限的 AWS Identity and Access Management (IAM) 角色（如果您计划从 AWS Organizations 运行脚本，则可访问特定账户或组织单位中所有账户的访问权限）

架构

下图显示了生成 EBS 快照按需报告的脚本工作流，这些快照分布在 OU 中的多个 Amazon Web Services account。

工具

Amazon Web Services

- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。

- Amazon Elastic Block Store (Amazon EBS) 提供用于实例的块级存储卷。 EC2
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Organizations 是一项账户管理服务，使您可将多个 Amazon Web Services account 整合到您所创建的组织中并进行集中管理。

代码

此模式中使用的示例应用程序的代码可在 GitHubaws-ebs-snapshots-awsorganizations存储库中找到。按照下一节中的说明使用示例文件。

操作说明

下载脚本

Task	描述	所需技能
下载 Python 脚本。	从GitHub 存储库下载脚本 GetSnapshotDetailsAllAccountsOU.py。	常规 AWS

获取 Amazon Web Services account EBS 快照详情

Task	描述	所需技能
运行 Python 脚本。	运行命令： `python3 getsnapshotinfo.py --file <output-file>.csv --region <region-name>` 其中，`<output-file>` 是指您想要获取有关放置的 EBS 快照信息的 CSV 输出文件，`<region-name>` 是存储快照的 AWS 区域。例如：	常规 AWS

Task	描述	所需技能
	`python3 getsnapshotinfo.py --file snapshots.csv --region us-east-1`	

获取组织 EBS 快照详细信息

Task	描述	所需技能

运行 Python 脚本。	运行命令：
	`python3 getsnapshotinfo.py --file <output-file>.csv --role <IAM-role> --region <region-name>`
	其中，`<output-file>` 是指您想要获取有关放置的 EBS 快照信息的 CSV 输出文件，`<IAM-role>` 是提供访问 AWS Organizations 权限的角色，`<region-name>` 是存储快照的 AWS 区域。例如：
	`python3 getsnapshotinfo.py --file snapshots.csv --role <IAM role> --region us-west-2`
		常规 AWS

相关资源

- Amazon EBS 文档

- Amazon EBS 行动
- Amazon EBS API 参考
- 提高 Amazon EBS 性能
- Amazon EBS 资源
- EBS 快照定价

其他信息

EBS 快照类型

Amazon EBS 可根据所有权和访问权限提供三种类型的快照：

- 由您拥有 – 默认情况下，只有您可以从您拥有的快照创建卷。
- 公共快照 — 您可与所有其他 Amazon Web Services account 公开共享快照。要创建公共快照，您需要修改快照的权限，以便与您指定的 Amazon Web Services account 共享快照。然后，您将授权的用户可以通过创建自己的 EBS 卷来使用您共享的快照，而您的原始快照不受影响。您还可以将未加密的快照公开提供给所有 AWS 用户。但是，出于安全原因，您不能公开加密快照。由于公开快照有可能暴露个人与敏感数据，因此会带来重大的安全风险。我们强烈建议不要与所有 Amazon Web Services account 共享 EBS 快照。有关共享快照的更多信息，请参阅 AWS 文档。
- 私有快照 — 您可与您指定的单个 Amazon Web Services account 私下共享快照。若要与特定 Amazon Web Services account 私下共享快照，请按照 AWS 文档中的说明进行操作，然后为权限设置选择私有。您已授权的用户可以使用您共享的快照来创建自己的 EBS 卷，同时您的原始快照不受影响。

概述与程序

下表提供了指向有关 EBS 快照更多信息的链接，包括如何通过查找和删除未使用的快照降低 EBS 卷成本，以及如何归档不需要频繁或快速检索的、很少访问的快照。

有关信息	参见
快照、功能和限制	创建 Amazon EBS 快照
如何创建快照	控制台：创建快照 AWS CLI：create-snapsho 命令

	例如： `aws ec2 create-snapshot --volume-id vol-1234567890abcdef0 --description " volume snapshot"`
删除快照（一般信息）	删除 Amazon EBS 快照
如何删除快照	控制台：删除快照 AWS CLI：delete-snapshot 命令 例如： `aws ec2 delete-snapshot --snapshot-id snap-1234567890abcdef0`
存档快照（一般信息）	归档 Amazon EBS 快照 Amazon EBS 快照归档（博客文章）
如何归档快照	控制台：存档快照 AWS CLI：modify-snapshot-tier 命令
如何检索已存档快照	控制台：还原已归档快照 AWS CLI：restore-snapshot-tier 命令
快照定价	Amazon EBS 定价

常见问题解答

什么是最短归档期？

最短归档期为 90 天。

还原已存档快照需要多长时间？

将已归档快照从归档层还原到标准层最长可能需要 72 小时，具体时间取决于快照的大小。

已归档快照是完整的快照吗？

归档的快照始终是完整的快照。

用户可存档哪些快照？

您只能归档您在账户中拥有的快照。

您可以归档注册的亚马逊机器映像（AMI）的根设备卷的快照吗？

不，您不能归档注册 AMI 的根设备卷的快照。

共享快照包含哪些安全注意事项？

当您共享快照时，您将允许其他人访问快照上的所有数据。只能与您信任使用您的数据的人共享快照。

如何与其他 AWS 区域共享快照？

快照受限于在其中创建它们的区域。要与其他区域共享快照，请将快照复制到该区域，然后分享副本。

您可以共享加密的快照吗？

您不能共享使用默认 AWS 托管式密钥加密的快照。您只能共享使用客户托管密钥加密的快照。共享加密快照时，还必须共享用于加密快照的客户托管密钥。

那么未加密的快照呢？

您可以公开共享未加密的快照。

更多模式

- 使用着陆区域加速器自动创建账户 AWS
- 使用 Amazon Bedrock 实现 AWS 基础设施运营自动化
- 自动执行 AWS 资源评测
- 自动清点多个账户和地区的 AWS 资源
- 使用 AWS CDK 自动部署 AWS Service Catalog 产品组合与产品
- 使用和自动进行动态管道管理，以便在 Gitflow 环境中部署修补程序解决方案 AWS Service CatalogAWS CodePipeline
- 使用 Terraform 在亚马逊托管 Grafana 上自动提取和可视化亚马逊 MWAA 自定义指标
- 使用 Amazon Inspector 自动对跨账户工作负载进行安全扫描 AWS Security Hub
- 使用云托管人和 AWS CDK 将适用于 Systems Manager 的 AWS 托管策略自动附加到 EC2 实例配置文件
- 自动加密现有和新 Amazon EBS 卷
- 使用 Amazon CloudWatch 可观测性访问管理器进行集中监控
- 在启动时检查 EC2 实例是否有必需的标签
- 为 AWS IoT 环境中的安全事件配置日志记录和监控
- 使用 Amazon EFS 创建 Amazon ECS 任务定义并在 EC2 实例上挂载文件系统
- 使用 AWS CloudFormation Guard 策略创建 AWS Config 自定义规则
- 使用 AWS CDK 外观和逃生舱口自定义默认角色名称
- 使用和删除未使用的 Amazon EBS 卷 AWS ConfigAWS Systems Manager
- 使用 AWS CDK 和来部署和管理 AWS Control Tower 控件 CloudFormation
- 使用 Terraform 部署和管理 AWS Control Tower 控件
- 使用 AWS CodePipeline、AWS 和 AWS 在多个 AWS CodeCommit 区域部署代码 CodeBuild
- 使用 AWS CloudFormation 模板 GuardDuty 有条件地启用 Amazon
- 使用导出 AWS IAM 身份中心身份及其分配的报告 PowerShell
- 使用 Troposphere 生成包含 AWS Config 托管规则的 AWS CloudFormation 模板
- 为 SageMaker 笔记本实例提供对另一个 AWS 账户中 CodeCommit 存储库的临时访问权限
- 将 Stonebranch Universal Controller 与 AWS Mainframe Modernization 集成
- 使用 Step Functions 和 Lambda 代理函数在 AWS 账户上启动 CodeBuild 项目
- 使用 ACM 将 Windows SSL 凭证迁移到应用程序负载均衡器

- 监控 IAM 根用户活动
- 使用 AWS CDK 和 GitHub 操作工作流程优化多账户无服务器部署
- 从 AWS CodeCommit 事件中执行自定义操作
- 在非工作负载子网的多账户 VPC 设计中保留可路由的 IP 空间
- 通过部署角色自动售货机解决方案来配置权限最低的 IAM 角色
- 使用 Amazon SES 通过单个电子邮件地址注册多个 Amazon Web Services account
- 使用 AWS Lambda 自动化功能 AWS 账户 从 AWS Managed Microsoft AD 正面移除 Amazon EC2 条目
- 使用 AWS Lambda 自动化功能 AWS 账户 从 AWS Managed Microsoft AD 中移除 Amazon EC2 条目
- 在不重启容器的情况下轮换数据库凭证
- 使用 AWS Fargate 大规模运行事件驱动型和计划性工作负载
- 使用本地 SMTP 服务器和数据库邮件发送 Amazon RDS for SQL Server 数据库实例通知
- 为 AWS 设置一个 Grafana 监控控制面板 ParallelCluster
- 使用 AWS Organizations 自动标记中转网关连接
- 使用 BMC Discovery 查询提取迁移数据以进行迁移规划
- 使用验证 PCI DSS 4.0 的最佳操作实践 AWS Config
- 使用 Amazon 可视化所有 AWS 账户的 IAM 凭证报告 QuickSight

消息和通信

主题

- 在 Amazon MQ 中自动化 RabbitMQ 配置
- 提高 Amazon Connect 联系中心的座席工作站的通话质量
- 更多模式

在 Amazon MQ 中自动化 RabbitMQ 配置

由 Yogesh Bhatia(AWS) 和 Afroz Khan(AWS) 编写

摘要

Amazon MQ 是一项托管式消息代理服务，可兼容许多常见消息代理。将 Amazon MQ 与 RabbitMQ 配合使用可提供一个在 Amazon Web Services (AWS) Cloud 中管理的强大的 RabbitMQ 集群，其中包含多个代理和配置选项。Amazon MQ 提供了高度可用、安全且可扩展的基础设施，并且每秒可以轻松处理大量消息。多个应用程序可以使用具有不同虚拟主机、队列和交换器的基础设施。但是，管理这些配置选项或手动创建基础设施，可能需要时间和精力。此示例介绍了一种通过单个文件一步管理 RabbitMQ 配置的方法。您可将此模式提供的代码嵌入到任何持续集成（CI）工具（例如 Jenkins 或 Bamboo）中。

您可使用此模式来配置任何 RabbitMQ 集群。它所需要的只是连接至集群。尽管还有许多其他方法可以管理 RabbitMQ 配置，但此解决方案只需一步即可创建整个应用程序配置，因此您可轻松管理队列和其他细节。

先决条件和限制

先决条件

- AWS 命令行界面（AWS CLI）已安装并配置为指向您的 Amazon Web Services account（有关说明，请参阅 AWS CLI 文档）
- Ansible 已安装，因此您可以运行 playbook 来创建配置
- rabbitmqadmin 已安装 (有关说明，请参阅 RabbitMQ 文档)
- 亚马逊 MQ 中的 RabbitMQ 集群，使用健康的亚马逊指标创建 CloudWatch

其他要求

- 确保单独为虚拟主机和用户创建配置，而不是作为 JSON 的一部分。
- 确保配置 JSON 是存储库的一部分并且受版本控制。
- rabbitmqadmin CLI 的版本必须与 RabbitMQ 服务器的版本相同，因此最好的选择是从 RabbitMQ 控制台下载 CLI。
- 作为管道的一部分，请确保在每次运行之前验证 JSON 语法。

产品版本

- AWS CLI 版本 2.0
- Ansible 版本 2.9.13
- rabbitmqadmin 版本 3.9.13 (必须与 RabbitMQ 服务器版本相同)

架构

源技术堆栈

- 在现有本地虚拟机或 Kubernetes 集群（本地或云端）上运行的 RabbitMQ 集群

目标技术堆栈

- Amazon MQ 上针对 RabbitMQ 的自动化 RabbitMQ 配置

目标架构

配置 RabbitMQ 的方法有很多种。此模式使用导入配置功能，其单个 JSON 文件包含所有配置。此文件应用所有设置，并且可以由 Bitbucket 或 Git 等版本控制系统管理。此模式使用 Ansible 通过 rabbitmqadmin CLI 实现配置。

工具

工具

- rabbitmqadmin 是基于 RabbitMQ HTTP 的 API 的命令行工具。它用于管理和监控 RabbitMQ 节点和集群。
- Ansible 是一款用于自动化应用程序和 IT 基础设施的开源工具。
- AWS CLI 让您能够在命令行 Shell 中使用命令与 Amazon Web Services 进行交互。

Amazon Web Services

- Amazon MQ 是托管式消息代理服务，让您可轻松地在云中设置和操作消息代理。
- AWS 利用基础设施即代码， CloudFormation帮助您设置 AWS 基础设施并加快云配置速度。

代码

附件中提供了此模式中使用的 JSON 配置文件和示例 Ansible playbook。

操作说明

创建您的 AWS 基础设施

Task	描述	所需技能
在 AWS 中创建一个 RabbitMQ 集群。	如果您还没有 RabbitMQ 集群，则可以使用 AWS 在 AW S CloudFormation 上创建堆栈。或者，您可以使用 Ansible 中的 Cloudformation 模块创建堆栈。使用后一种方法，您可使用 Ansible 完成这两项任务：创建 RabbitMQ 基础设施和管理配置。	AWS CloudFormation、Ansible

创建 Amazon MQ for RabbitMQ 配置

Task	描述	所需技能
创建属性文件。	下载附件中的 JSON 配置文件 (`rabbitmqconfig.json`)，或者从 RabbitMQ 控制台将其导出。 修改它以配置队列、交换器和绑定。此配置文件演示了以下内容： - 创建两个队列：`sample-queue1` 和 `sample-queue2` - 创建两个交易所：`sample-exchange1` 和 `sample-exchange2` - 实现队列和交换之间的绑定	JSON

Task	描述	所需技能
	根据 rabbitmqadmin 的要求，这些配置在根 (/) 虚拟主机下执行。	
检索适用于 RabbitMQ 的 Amazon MQ 基础设施的详细信息。	检索 AWS 上的 RabbitMQ 基础设施的以下详细信息： • 代理名称 • RabbitMQ 主机 • RabbitMQ 用户名（在创建集群时创建的管理员用户） • RabbitMQ 密码 您可使用 AWS 管理控制台 或 AWS CLI 来检索此信息。这些详细信息使 Ansible playbook 能够连接到您的 Amazon Web Services account 并使用 RabbitMQ 集群来运行命令。 ⚠ Important 运行 Ansible 攻略手册的计算机必须能够访问您的 AWS 账户，并且 AWS CLI 必须已经配置，如先决条件部分所述。	AWS CLI、Amazon MQ

Task	描述	所需技能

创建 hosts_var 文件。	为 Ansible 创建 `hosts_var` 文件并确保文件中定义了所有变量。考虑使用 Ansible 保管库存储密码。您可按如下方式配置 `hosts_var` 文件（用您的信息替换星号）：
	
	`RABBITMQ_HOST: "**********.mq.us-east-2.amazonaws.com"`
	`RABBITMQ_VHOST: "/"`
	`RABBITMQ_USERNAME: "admin"`
	`RABBITMQ_PASSWORD: "*******"`
		Ansible

创建 Ansible Playbook。	有关示例 Playbook，请参阅附件中的 `ansible-rabbit-config.yaml` 。下载并保存此文件。Ansible playbook 导入并管理应用程序所需所有 RabbitMQ 配置，例如队列、交换和绑定。 遵循 Ansible playbook 的最佳实践，例如保护密码。使用 Ansible Vault 进行密码加密，并从加密文件中检索 RabbitMQ 密码。	Ansible

部署配置

Task	描述	所需技能
运行 Playbook。	运行您在上一部操作说明中创建的 Ansible Playbook。 `ansible-playbook ansible-rabbit-config.yaml` 您可在 RabbitMQ 控制台上验证新的配置。	RabbitMQ、Amazon MQ、Ansible

相关资源

- 从 RabbitMQ 迁移至 Amazon MQ（AWS Blog 文章）
- 管理命令行工具（RabbitMQ 文档）
- 创建或删除 AWS CloudFormation 堆栈（Ansible 文档）
- 将消息驱动的应用程序迁移至 Amazon MQ for RabbitMQ（AWS Blog 文章）

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

提高 Amazon Connect 联系中心的座席工作站的通话质量

由 Ernest Ozdoba (AWS) 编写

摘要

通话质量问题是联系中心最难解决的一些问题。为避免语音质量问题和复杂的故障排除程序，您必须优化座席的工作环境和工作站设置。此模式描述了 Amazon Connect 联系中心座席工作站的语音质量优化技术。它对以下领域提供了建议：

- 调节工作环境。座席的周境不会影响语音通过网络传输方式，但会影响通话质量。
- 座席工作站设置。联系中心工作站的硬件和网络配置会对通话质量产生显著影响。
- 浏览器设置 座席使用网络浏览器访问 Amazon Connect 联络控制面板 (CCP) 网站并与客户沟通，因此浏览器设置可能会影响通话质量。

以下组件也可能会影响通话质量，但它们不在工作站的范围之内，因此不在此模式中：

- 流量通过 AWS Direct Connect、全通道 VPN 或分隧道 VPN 流向 Amazon Web Services (AWS) Cloud
- 在公司办公室内外处理时的网络状况
- 公共交换电话网络 (PSTN) 连接
- 客户设备和电话运营商
- 虚拟桌面基础架构（VDI）设置

有关这些领域的更多信息，请参阅 Amazon Connect 文档中的常见联系人控制面板 (CCP) 问题以及使用端点测试实用程序。

先决条件和限制

先决条件

- 头戴式耳机和工作站必须符合 Amazon Connect 管理员指南中的指定要求。

限制

- 此模式中的优化技术可适用于软电话语音质量。当您在桌面电话模式下配置 Amazon Connect CCP 时，其不适用。但是，如果您的软电话设置无法为呼叫提供可接受语音质量，则可以使用桌面电话模式。

产品版本

- 有关支持的浏览器和版本，请参阅 Amazon Connect 管理员指南。

架构

这种模式与架构无关，因为它针对座席工作站设置。如下图所示，从座席到客户的语音路径受座席头戴式耳机、浏览器、操作系统、工作站硬件以及网络的影响。

在 Amazon Connect 联系中心，用户的音频连接是通过 WebRTC 建立的。语音使用 Opus 互动音频代码编码，并在传输过程中使用安全实时传输协议 (SRTP) 进行加密。其他网络架构也有可能，包括 VPN、私有 WAN/LAN 和 ISP 网络。

工具

- Amazon Connect 端点测试实用程序 — 该实用程序检查网络连接和浏览器设置。
- WebRTC 设置浏览器配置编辑器：
  - 对于 Firefox：about:config
  - 对于 Chrome：chrome://flags
- CCP Log Parser – 此工具可帮助您分析 CCP 日志以进行故障排除。

操作说明

调整工作环境

Task	描述	所需技能
减少背景噪音。	避开嘈杂环境。如果无法做到这一点，请使用以下隔音技巧优化环境：	座席，经理

Task	描述	所需技能
	• 使用窗帘、地毯以及软质家具等消音表面吸收噪音。 • 通过在办公桌设置屏障来阻挡噪音。 • 可以考虑使用主动降噪 (ANC) 解决方案，例如白噪声发生器，以集中注意力并确保隐私，或者使用降噪耳机。 • 防止来电中的回声。大而空的空间可能会产生回声效果或者放大噪音。覆盖能反弹声音的表面，将有助于减少回声。	

优化座席工作站设置

Task	描述	所需技能
选择适当的头戴式耳机。	• 如果环境嘈杂，请您选择立体声耳机。将声音传送至两只耳朵，可以帮助客服集中注意力，更好地听到客户的声音，并通过降低客服人员提高声音的可能性来减少整体噪音。 • 避免使用扬声器或内置计算机音频。为获得最佳音质，请使用联系中心专用的有线头戴式耳机。无线耳机很方便，但由于无线电干扰和转码，它们可能会导致额外音频延迟和音频质量降低。	座席，经理

Task	描述	所需技能
请正常使用头戴式耳机。	• 启用头戴式耳机的主动降噪和语音增强功能（如有）。查找 ANC 或 ANR 等设置。有关激活设置的说明，请参阅头戴式耳机的用户手册。 • 调节您的麦克风，这样您就可以直接对着麦克风说话。麦克风的最佳位置在下巴下方。正确放置可使声级相差 10 分贝 (dB)。多数头戴式耳机允许您旋转或弯曲麦克风臂（吊杆），因此在说话时请务必将其放在正确的位置。 • 有些头戴式耳机配备了多个麦克风与高级功能，例如语音波束成形，这有助于在不产生轰鸣声的情况下捕捉语音。要确保按制造商的预期使用主麦克风，请参见设备的用户手册。	座席
检查工作站资源。	确保座席计算机性能良好。如果他们使用消耗资源的第三方应用程序，他们的计算机可能无法满足 CCP 运行的最低硬件要求。如果客服遇到通话质量问题，请确保他们有足够的处理能力 (CPU)、磁盘空间、网络带宽以及内存可用于 CCP。工程师应关闭所有不必要应用程序和选项卡，以提高 CCP 性能和通话质量。	管理员

Task	描述	所需技能
配置操作系统声音设置。	麦克风音量和增强默认设置通常可以正常工作。如果您发现输出语音很安静或者麦克风拾音过多，则调整这些设置可能会有所帮助。麦克风设置可在您的计算机系统声音配置中找到（声音、MacOS 的输入、Windows 中的麦克风属性）。您可通过系统工具或第三方应用程序访问可能会影响语音质量的高级设置。以下是部分可以检查的设置： • 采样率 - 此值决定每秒探测声音的次数。默认设置通常是 44 或 48 千赫兹 (kHz)。Amazon Connect 的最佳值是 48 kHz。您可使用浏览器设置来覆盖默认值。有关更多信息，请参阅《Amazon Connect 管理员指南》中的故障排除部分。 • 增益 - 此值决定麦克风对声音的放大程度。如您调高增益，您的麦克风可能会吸收更多的背景噪音。 • 位深度 — 此数字分辨率值描述了可识别声振幅级别。位深度越高，声音听起来就越流畅。但是，多个传统电话网络使用脉冲码调制 (PCM) 标准，该标准仅支持 8 位分辨率。	座席、管理员

Task	描述	所需技能
	• 开启阈值 — 这是麦克风拣选的最小声幅度。 如果您遇到语音质量问题，请尝试将这些值还原为默认设置，然后再执行调查。 有关这些设置与其他可调设置的更多信息，请参阅您的设备手册。	

Task	描述	所需技能
使用有线网络。	通常，有线以太网延迟较低，因此更容易提供语音数据传输所需一致传输质量。 我们建议每次通话的带宽至少 100 KB。 • 如果客服人员在家办公，我们建议您通过无线连接进行有线连接。蜻蜓客户意见的时间不应超过 150 毫秒。您可以通过 Amazon Connect 端点测试实用程序访问 Amazon Connect 延迟测试。但是，此实用程序会衡量从浏览器至 Amazon Connect 区域的延迟，而不是对买家的延迟。150 毫秒的单向延迟建议可避免座席和客户互相交谈。该值从头到尾测量的，每个元素都会增加延迟，包含 Amazon Connect 地区与客户之间的通话部分。 • 如果座席在办公室办公，则只要参数在建议范围内，并且优先考虑实时传输协议 (RTP) 流量，就可接受企业 Wi-Fi。	网络管理员，座席

Task	描述	所需技能
更新硬件驱动程序。	当您使用自带固件 USB 或其他类型的头戴式耳机时，我们建议您将其更新为最新版本。辅助端口的简单头戴式耳机使用计算机的内置音频设备，因此请确保操作系统的硬件驱动程序是最新的。极少数情况下，音频驱动程序更新可能会导致音频问题，您可能需要将其还原。有关更改固件和驱动程序版本的更多信息，请参阅您的设备手册。	管理员
避免使用 USB 集线器与解密器。	连接头戴式耳机时，请避免使用其他设备，例如解密器、端口类型转换器、集线器以及延长线。 此设备可能会影响通话质量。改为将设备直接连接至电脑的端口。	座席

Task	描述	所需技能
查看 CCP 日志。	CCP Log Parser 提供了一种检查应用程序日志的简便方法。 1. 通话后下载 CCP 日志。 2. 打开 CCP Log Parser。 3. 拖放日志文件，以上传日志进行分析。 4. 分析完日志后，将默认选择快照和日志选项卡。选择旁边的指标选项卡以查看见解。 5. 在 WebRTC Metrics - audio_input 部分，检查以下内容： • 音频电平图，查看您接收到的音频电平是否高于 0。这表示已收到来电者音频。 • 任何丢失的数据包的数据包图表。如果此图表显示大幅增长，请联系 IT 支持团队。 6. WebRTC Metrics - audio_output 部分，检查以下内容： • 音频电平图，用于确认音频已从您的设备发出。 • 数据包图。如果您看到数据包丢失激增，请将其报告至您的 IT 支持团队。 • Jitter Buffer & RTT 图。往返时间（RTT）值超过 300，将影响通话体验。	座席（高级技能）

Task	描述	所需技能
	请将这些问题报告至您的 IT 支持团队。	

优化浏览器设置

Task	描述	所需技能
还原默认 WebRTC 设置。	必须启用 WebRTC 才能通过 CCP 软通话。建议您保留 WebRTC 相关功能默认设置。 • 在 Chrome 中，您可以通过导航至URL chrome://flags 以设置标记。在搜索框内输入 WebRTC，以查找可与 CCP 交互的设置，然后将其设置为默认。 • 在 Firefox 中，在地址栏中键入 about:config，然后在配置页面的搜索框中键入 WebRTC。非默认设置以粗体文本显示，可以更改为默认。	管理员
故障排除时，禁用浏览器扩展程序。	部分浏览器扩展程序可能会影响通话质量，甚至会阻止通话正常连接。在浏览器中使用隐身窗口或者私密模式，并禁用所有扩展程序。如果这样可以解决问题，请查看浏览器扩展程序并寻找可疑的插件，或者单独禁用它们。	座席、管理员
检查浏览器的采样率。	确认您的麦克风输入已设置为最佳 48 kHz 采样率。有关说	座席、管理员

Task	描述	所需技能
	明，请参阅 Amazon Connect 管理员指南。	

相关资源

如果您已按此模式中的步骤进行操作，但仍然遇到通话质量问题，请参阅以下资源以获取故障排除提示。

- 查看常见的联系人控制面板 (CCP) 问题。
- 使用端点测试实用程序检查连接。
- 如有任何其他问题，请按故障排除指南进行操作。

如果您的故障排除和调整不能解决通话质量问题，则根本原因可能是您的工作站外部原因。若要进一步排除故障，请联系您的 IT 支持团队。

更多模式

- 使用 CQRS 和事件溯源将整体分解为微服务
- 部署 ChatOps 解决方案来管理 SAST 扫描结果，方法是在聊天应用程序中使用 Amazon Q Developer 自定义操作和 AWS CloudFormation
- 将 Amazon API Gateway 与亚马逊 SQS 集成以处理异步 REST APIs
- 使用 Amazon SES 通过单个电子邮件地址注册多个 Amazon Web Services account
- 使用 AWS Fargate 大规模运行消息驱动型工作负载
- 使用自动化工作流程简化 Amazon Lex 机器人的开发和部署

安全、身份和合规性

主题

- 使用 Amazon Cognito 身份池 AWS 服务 从 ASP.NET 核心应用程序进行访问
- EC2 使用 AWS Directory Service 对亚马逊上的微软 SQL Server 进行身份验证
- 自动执行事件响应和取证
- 自动修复 AWS Security Hub 标准调查结果
- 使用 Amazon Inspector 自动对跨账户工作负载进行安全扫描 AWS Security Hub
- 自动审核允许从公有 IP 地址进行访问 AWS 的安全组
- 使用 AWS Confi CloudTrail g 中的自定义补救规则自动重新启用 AWS
- 自动修复未加密的 Amazon RDS 数据库实例和集群
- 使用 AWS Organizations 和 AWS Secrets Manager 大规模自动轮换 IAM 用户访问密钥
- 使用 CodePipeline IAM Access Analyzer 和 AWS CloudFormation 宏在 AWS 账户中自动验证和部署 IAM 策略和角色
- 将 AWS Security Hub 与 Jira 软件双向集成
- 使用 Image Builder 和 Terraform 为经过强化的容器 EC2 镜像构建管道
- 使用 Terraform 在 AWS Organizations 中集中管理 IAM 访问密钥
- 查看亚马逊 CloudFront 分配的访问日志、HTTPS 和 TLS 版本
- 检查和的安全组入口规则中是否有单主机网络条目 IPv4 IPv6
- 为企业应用程序选择 Amazon Cognito 身份验证流程
- 使用 AWS CloudFormation Guard 策略创建 AWS Config 自定义规则
- 创建一份包含多个 Prowler 安全调查结果的合并报告 AWS 账户
- 使用和删除未使用的 Amazon EBS 卷 AWS ConfigAWS Systems Manager
- 使用 AWS CDK 和来部署和管理 AWS Control Tower 控件 CloudFormation
- 使用 Terraform 部署和管理 AWS Control Tower 控件
- 部署可同时检测多个代码交付项中的安全问题的管道
- 使用以下方法为公共子网部署基于侦探属性的访问控制 AWS Config
- 为公有子网部署基于属性的预防性访问控制
- 使用 Terraform 为 AWS WAF 解决方案部署安全自动化
- 检测已过期的 CA 证书的 Amazon RDS 和 Aurora 数据库实例

- 使用 Step Functions 通过 IAM Access Analyzer 动态生成 IAM policy
- 使用 AWS CloudFormation 模板 GuardDuty 有条件地启用 Amazon
- 在 Amazon RDS for SQL Server 中启用透明数据加密
- 确保 AWS 负载均衡器使用安全侦听器协议 (HTTPS、SSL/TLS)
- 确保在发布时启用 Amazon EMR 静态数据加密
- 确保 IAM 配置文件与 EC2 实例关联
- 确保 Amazon Redshift 集群在创建时已加密
- 使用导出 AWS IAM 身份中心身份及其分配的报告 PowerShell
- 监控和修复 AWS KMS 密钥的计划删除
- 使用 Security Hub 识别 AWS Organizations 其中的公有 Amazon S3 存储桶
- 在 Microsoft Sentinel 中提取和分析 AWS 安全日志
- 使用以下 AWS IAM Identity Center 命令将权限集作为代码进行管理 AWS CodePipeline
- 使用 AWS Secrets Manager 管理凭证
- 监控 ElastiCache 集群中的安全组
- 在启动时监控 Amazon EMR 集群的传输中加密
- 监控 Amazon ElastiCache 集群的静态加密
- 使用 AWS Config 监控 EC2 实例密钥对
- 监控 IAM 根用户活动
- 在创建 IAM 用户时发送通知
- 使用服务控制策略阻止账户级别的互联网访问
- 使用，根据 IP 地址或地理位置限制访问 AWS WAF
- 使用 git-secrets 扫描 Git 存储库中的敏感信息及安全问题
- 将警报从 AWS Network Firewall 发送到 Slack 通道
- 使用 AWS Private CA 和 AWS RAM 简化私有证书管理
- 在多账户环境中关闭所有 Security Hub 成员账户的安全标准控件
- 使用从 AWS IAM 身份中心更新 AWS CLI 证书 PowerShell
- 使用 AWS Config 监控 Amazon Redshift 安全配置
- 使用 Network Firewall 从服务器名称指示中捕获出站流量的 DNS 域名
- 使用 Terraform 自动 GuardDuty 为组织启用亚马逊
- 使用验证 PCI DSS 4.0 的最佳操作实践 AWS Config

- 验证新的 Amazon Redshift 集群是否有所需 SSL 端点
- 验证新 Amazon Redshift 集群是否在 VPC 中启动
- 更多模式

使用 Amazon Cognito 身份池 AWS 服务 从 ASP.NET 核心应用程序进行访问

由 Bibhuti Sahu (AWS) 和 Marcelo Barbosa (AWS) 创建

摘要

此模式讨论如何配置 Amazon Cognito 用户池和身份池，然后在成功进行身份验证后启用 ASP.NET Core 应用程序访问 AWS 资源。

Amazon Cognito 可为您的网络和移动应用程序提供身份验证、授权和用户管理。Amazon Cognito 的两个主要组件是用户池和身份池。

用户池是 Amazon Cognito 中的用户目录。利用用户池，您的用户可以通过 Amazon Cognito 登录您的 Web 或移动应用程序。您的用户还可以通过社交身份提供商（例如 Google、Facebook、Amazon 或 Apple）以及 SAML 身份提供商登录。

借助 Amazon Cognito 身份池（联合身份），您能够为用户创建唯一的身份，并通过身份提供商对其进行联合身份验证。使用身份池，您可以获得临时的、权限有限的 AWS 凭证来访问其他人。AWS 服务在开始使用新的 Amazon Cognito 身份池之前，您必须分配一个或多个 AWS Identity and Access Management (IAM) 角色来确定您希望应用程序用户对您的资源的访问级别。 AWS 身份池定义了两种类型的身份：经过身份验证的身份和未经身份验证的身份。每个身份类型都可以在 IAM 中分配自己的角色。经过身份验证的身份属于通过公共登录提供商（Amazon Cognito 用户池、Facebook、Google、SAML 或任何 OpenID Connect 提供商）或开发人员提供商（自己的后端身份验证流程）验证身份的用户,而未认证身份的用户通常属于访客用户。当 Amazon Cognito 收到用户请求时，服务会确定该请求是通过身份验证的还是未经身份验证的，确定与该身份验证类型相关联的角色，然后使用附加到该角色的策略来响应请求。

先决条件和限制

先决条件

- AWS 账户 拥有亚马逊 Cognito 和 IAM 权限
- 访问您要使用的 AWS 资源
- ASP.NET 内核 2.0.0 或更高版本

架构

技术堆栈

- Amazon Cognito
- ASP.NET 内核

目标架构

工具

工具 SDKs、和 AWS 服务

- Visual Studio 或 Visual Studio 代码
- 亚马逊。 AspNetCore.Identity.Cognito (1.0.4) — 软件包 NuGet
- AWSSDK.S3 (3.3.110 .32) — 软件包 NuGet
- Amazon Cognito

代码

附加的 .zip 文件包含说明以下内容的示例文件：

- 如何检索已登录用户的访问令牌
- 如何使用访问令牌兑换 AWS 证书
- 如何使用凭证访问亚马逊简单存储服务 (Amazon S3) Service 服务 AWS

用于经过身份验证的身份的 IAM 角色

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "mobileanalytics:PutEvents",
```

```
        "cognito-sync:*",
        "cognito-identity:*",
        "s3:ListAllMyBuckets*"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

操作说明

创建 Amazon Cognito 用户池

Task	描述	所需技能
创建用户池。	1. 登录 AWS Management Console 并打开 Amazon Cognito 控制台。 2. 选择管理用户池。 3. 在页面右上角，选择创建用户池。 4. 为您的用户池提供一个名称，选择查看默认值，然后选择创建池。 5. 记下池 ID。	开发人员
添加应用程序客户端。	您可以创建一个应用程序，使用内置网页进行用户注册和登录。 1. 在用户池页面左侧的导航栏上，选择一般设置下的应用程序客户端，然后选择添加应用程序客户端。 2. 为应用程序命名，然后选择创建应用程序客户端。	开发人员

Task	描述	所需技能
	3. 记下应用程序客户端 ID 和客户端密钥(选择显示详细信息以查看客户端密钥)。	

创建一个 Amazon Cognito 身份池

Task	描述	所需技能
创建 身份池。	1. 在 Amazon Cognito 控制台上选择管理身份池，然后选择创建新身份池。 2. 为身份池键入名称。 3. 如果要启用未经身份验证的身份，请从未经身份验证的身份部分中选择该选项。 4. 在身份验证提供商部分，通过设置用户池 ID 和应用程序客户端 ID 来配置 Amazon Cognito 身份池，然后选择创建池。	开发人员
为身份池分配 IAM 角色。	您可以编辑经过身份验证和未经身份验证的用户的 IAM 角色，或保留默认值，然后选择允许。对于此模式，我们将编辑经过身份验证的 IAM 角色，并为 `s3:ListAllMyBuckets` 提供访问权限。有关示例代码，请参阅前面在工具部分中提供的 IAM 角色。	开发人员
复制身份池 ID。	当您在上一步中选择允许时，将显示Amazon Cognito 入门页	开发人员

Task	描述	所需技能
	面。在此页面上，您可以从获取 AWS 凭证部分复制身份池 ID，也可以选择右上角的编辑身份池，然后从显示的屏幕中复制身份池 ID。	

配置您的示例应用程序

Task	描述	所需技能
克隆示例 ASP.NET Core 网络应用程序。	1. 从 https://github.com/aws/aws-aspnet-cognito-identity-provider.git 中克隆示例.NET 核心 Web 应用程序。 2. 导航到 `samples` 文件夹并打开解决方案。在此项目中，您将配置 `appsettings.json` 文件并添加一个新页面，该页面将在成功登录后呈现所有 S3 存储桶。	开发人员
添加依赖项。	向 ASP.NET Core 应用程序添加 NuGet 依赖关系。`Amazon.AspNetCore.Identity.Cognito`	开发人员
将配置键和值添加到 `appsettings.json` 。	将附加的 `appsettings.json` 文件中的代码包含在 `appsettings.json` 文件中，然后将占位符替换为前面步骤中的值。	开发人员
创建新用户并登录。	在 Amazon Cognito 用户池中创建一个新用户，并验证该用	开发人员

Task	描述	所需技能
	户是否存在于用户池中的用户和组下。	
创建一个名`MyS3Buckets` 为的新 Razor 页面。	向您的示例应用程序添加新的 ASP.NET Core Razor 页面，并替换所附示例中 `MyS3Bucket.cshtml` 和 `MyS3Bucket.cshtml.cs` 的内容。在 `_Layout.cshtml` 页面的导航下添加新的 MyS3Bucket 页面。	开发人员

故障排除

事务	解决方案
打开 GitHub 存储库中的示例应用程序后，尝试将该 NuGet 包添加到 Samples 项目时会出现错误。	在 `src` 文件夹中，确保从 `Amazon.AspNetCore.Identity.Cognito` 文件中删除对 `Samples.sln` 项目的引用。然后，您可以毫无问题地将 NuGet 软件包添加到 Samples 项目中。

相关资源

- Amazon Cognito
- Amazon Cognito 用户群体
- Amazon Cognito 身份池
- 访问策略示例
- GitHub -AWS ASP.NET Cognito 身份提供商

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

EC2 使用 AWS Directory Service 对亚马逊上的微软 SQL Server 进行身份验证

由 Jagadish Kantubugata (AWS) 和 Oludahun Bade Ajidahun (AWS) 创建

摘要

此模式描述了如何创建 AWS Directory Service 目录并使用它在亚马逊弹性计算云 (亚马逊 EC2) 实例上对 Microsoft SQL Server 进行身份验证。

AWS Directory Service 提供多种方式将 Amazon Cloud Directory (AD) 和 Microsoft Active Directory 与其他 Amazon Web Services 一起使用。目录中存储有关用户、组和设备的信息，管理员使用这些信息来管理对信息和资源的访问。针对想要在云中使用其现有 Microsoft AD 或能够识别轻量目录访问协议 (LDAP) 的应用程序的用户，AWS Directory Service 提供了多种目录选择。它还为需要目录来管理用户、组、设备和访问权限的开发人员提供了同样的选择。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 具有两个私有子网和两个公有子网的虚拟私有云（VPC）
- 用于将服务器加入域的 AWS Identity and Access Management (IAM) 角色

架构

源技术堆栈

- 源可以是本地 Active Directory

目标技术堆栈

- AWS Directory Service for Microsoft Active Directory (AWS Managed Microsoft AD)

目标架构

工具

- SQL Server Management Studio (SSMS) 是一款用于管理Microsoft SQL Server 的工具，包含访问、配置和管理 SQL Server 组件。

操作说明

设置目录

Task	描述	所需技能
选择 AWS Managed Microsoft AD 作为目录类型。	在 AWS Directory Service 控制台，选择目录、设置目录、AWS Managed Microsoft AD、下一步。	DevOps
选择版本。	从 AWS Managed Microsoft AD 的可用版本中，选择标准版。	DevOps
指定目录 DNS 名称。	用于完全限定的域名。此名称将仅在您的 VPC 内解析。该名称不需要可公开解析。	DevOps
设置管理员密码。	为名为管理员的默认管理用户设置密码。	DevOps
选择 VPC 和子网。	选择将包含目录的 VPC 和域控制器的子网。如果您的 VPC 没有至少有两个子网，您必须创建一项。	DevOps
审核并启动目录。	查看目录的版本和价格信息，然后选择创建目录。	DevOps

在域中启动 SQL Server 的 EC2 实例

Task	描述	所需技能
为 SQL Server 选择 AMI。	这篇长篇故事中的步骤将 Windows EC2 实例无缝加入你的 AWS 托管微软 AD 目录。 在亚马逊 EC2 控制台上，选择启动实例，然后为 SQL Server 选择相应的亚马逊系统映像 (AMI)。	DevOps，DBA
配置实例详细信息	配置 Windows 实例，以满足你对 SQL Server 的要求。	DevOps，DBA
选择密钥对名称。	选择密钥对，然后启动实例。	DevOps，DBA
添加网络。	您可以选择在其中创建了目录的 VPC。	DevOps，DBA
选择 IAM role (IAM 角色)。	在高级设置中，选择附有 AWS 托管式策略 `AmazonSSMManagedInstanceCore` 和 `AmazonSSMDirectoryServiceAccess` 的 IAM 个人资料。	DevOps，DBA
添加子网。	在 VPC 中选择一个公有子网。选择的子网必须将所有外部流量都路由到互联网网关。否则将无法远程连接到实例。	DevOps，DBA
选择您的域。	选择您从域加入目录列表创建的域。	DevOps，DBA
启动实例。	选择启动实例。	数据库管理员

使用 Directory Service 对 SQL Server 进行身份验证

Task	描述	所需技能
以 Windows 管理员身份登录。	使用 Windows 管理员凭据登录 Windows EC2 实例。	数据库管理员
登录至 SQL Server。	启动 SQL Server Management Studio (SSMS)，并使用 Windows 身份验证方法登录至 SQL Server。	数据库管理员
为目录用户创建登录名。	在 SSMS 中，选择安全，然后选择新建登录。	数据库管理员
搜索登录名。	选择登录文本框旁的搜索按钮。	数据库管理员
选择位置。	在选择用户或组对话框，选择位置。	数据库管理员
输入网络凭证。	输入您在创建目录服务时使用的完全限定网络凭证，例如：`test.com\admin` 。	数据库管理员
选择目录。	选择 AWS 目录名称，然后选择确定。	数据库管理员
选择对象名称。	选择要为其创建登录名的用户。选择位置，选择整个目录，搜索用户然后添加登录信息。	数据库管理员
登录至 SQL Server 实例。	使用您的域凭据登录 SQL Server 的 Windows EC2 实例。	数据库管理员

Task	描述	所需技能
以域用户身份登录 SQL Server。	启动 SSMS，并使用 Windows 身份验证方法连接至数据库引擎。	数据库管理员

相关的资源

- AWS Directory Service 文档（AWS 网站）
- 创建 AWS Managed Microsoft AD 目录（AWS Directory Service 文档）
- 无缝加入 Windows EC2 实例（AWS Directory Service 文档）
- AWS 上的 Microsoft SQL Server（AWS 网站）
- SSMS 文档（Microsoft 网站）
- 在 SQL Server 中创建登录名（SQL Server 文档）

自动执行事件响应和取证

由 Lucas Kauffman (AWS) 和 Tomek Jakubowski (AWS) 编写

摘要

此模式部署了一组使用 AWS Lambda 函数提供以下内容的进程：

- 一种以最少知识启动事件响应流程的方法
- 符合《AWS 安全事件响应指南》的自动化、可重复的流程
- 分离账户来操作自动化步骤、存储构件并创建取证环境

自动事件响应和取证框架遵循标准数字取证流程，包括以下阶段：

1. 遏制
2. 收购
3. 检查
4. 分析

您可对静态数据（例如采集的内存或磁盘映像）以及独立系统上的实时动态数据进行调查。

有关更多详细信息，请参阅其他信息部分。

先决条件和限制

先决条件

- 二 AWS 账户：
  - 安全账户，可以是现有账户，但最好是新账户
  - 取证账户，最好是新账号
- AWS Organizations 设置
- 在 Organizations 的成员账户中：
  - 亚马逊弹性计算云 (Amazon EC2) 角色必须具有亚马逊简单存储服务 (Amazon S3) 的 “获取” 和 “列表” 权限，并且可以访问。 AWS Systems Manager我们建议使用`AmazonSSMManagedInstanceCore` AWS 托管角色。请注意，启动事件响应后，此角色将自

动附加到 Amazon EC2 实例。响应完成后， AWS Identity and Access Management (IAM) 将移除该实例的所有权限。

  - AWS 成员账户和事件响应和分析中的虚拟私有云 (VPC) 终端节点 VPCs。这些端点是：S3 网关、 EC2 消息、SSM 和 SSM 消息。
- AWS Command Line Interface (AWS CLI) 已安装在 Amazon EC2 实例上。如果尚未 AWS CLI 安装 Amazon EC2 实例，则需要访问互联网才能正常使用磁盘快照和内存采集。在这种情况下，脚本将通过互联网下载 AWS CLI 安装文件并将其安装在实例上。

限制

- 该框架无意生成可被视为可在法庭上提交的电子证据的构件。
- 目前，此模式仅支持在 x86 架构运行的基于 Linux 的实例。

架构

目标架构

除成员账户外，目标环境还包含两个主要账户：安全账户和取证账户。使用两个账户的原因如下：

- 将它们与任何其他客户账户分开，以减少取证分析失败时的影响范围
- 帮助确保隔离和保护正在分析的构件的完整性
- 为调查保密
- 为了避免出现这样的情况：威胁行为者可能 AWS 账户 通过达到服务配额并阻止您实例化 Amazon EC2 实例来执行调查，从而使用了您立即可用的所有资源。

此外，拥有单独的安全和取证账户允许创建单独的角色 - 用于获取证据的响应者和用于分析证据的调查者。每个角色都可访问其单独的账户。

下图仅展示了账户之间的交互。每个账户的详细信息将在后续图表中显示，并附上完整的图表。

下图显示了会员账户。

1. 将向 Slack 亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 主题发送一个事件。

下图显示了安全账户。

2. 安全账户中的 Amazon SNS 主题会启动取证事件。

下图显示了取证账户。

安全账户是为内存和磁盘映像采集创建两个主要 AWS Step Functions 工作流程的地方。工作流程运行后，他们将访问事件中涉及的 Amazon EC2 实例的成员账户，然后启动一组 Lambda 函数来收集内存转储或磁盘转储。然后，这些构件将存储在取证账户中。

取证账户将在分析项目 Amazon S3 存储桶中保存 Step Functions 工作流程收集的项目。取证账户还将有一个 Amazon I EC2 mage Builder 管道，用于生成取证实例的亚马逊系统映像 (AMI)。目前，该映像基于 SANS SIFT Workstation。

构建过程使用可连接至互联网的维护 VPC。稍后可以使用该图像来启动 Amazon EC2 实例，以分析分析 VPC 中收集的项目。

Analysis VPC 没有互联网连接。默认情况下，该模式创建三个私有分析子网。您最多可以创建 200 个子网，这是 VPC 中子网数量的配额，但是 VPC 终端节点需要添加这些子网才能在其中自动运行命令。 AWS Systems Manager Session Manager

从最佳实践的角度来看，我们建议使用 AWS CloudTrail 和 AWS Config 来执行以下操作：

- 跟踪您的取证账户的变更
- 监控存储和分析的构件的访问权限和完整性

Workflow (工作流程)

下图显示了工作流的关键步骤，其中包括从实例受损到分析和控制实例的过程和决策树。

1. `SecurityIncidentStatus` 标签是否已设置为分析值？ 如果是，则执行以下操作：
   a. 请附上 AWS Systems Manager 和 Amazon S3 的正确 IAM 配置文件。
   b. 在 Slack 中向 Amazon SNS 队列发送一封 Amazon SNS 消息。
   c. 向 `SecurityIncident` 队列发送 Amazon SNS 消息。
   d. 调用内存和磁盘采集状态机。
2. 是否已获取内存和磁盘？ 如否，则存在错误。

3. 使用标签为 Amazon EC2 实例`Contain`添加标签。
4. 附加 IAM 角色和安全组，以完全隔离实例。

自动化和扩缩

这种模式的目的是提供一种可扩展的解决方案，以便在单个 AWS Organizations 组织内的多个账户中执行事件响应和取证。

工具

AWS 服务

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在资源的整个生命周期中跨地区对其 AWS 账户 进行管理。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，用于 AWS 服务 通过命令行 shell 中的命令进行交互。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Key Management Service (AWS KMS) 帮助您创建和控制加密密钥以保护您的数据。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Storage Service (Amazon S3)是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Security Hub提供了您的安全状态的全面视图 AWS。它还可以帮助您根据安全行业标准和最佳实践检查您的 AWS 环境。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。
- AWS Step Functions是一项无服务器编排服务，可帮助您组合 AWS Lambda 功能和其他功能 AWS 服务 来构建关键业务应用程序。
- AWS Systems Manager 可帮助您管理在 AWS Cloud中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。

代码

有关代码以及具体的实施和使用指南，请参阅 GitHub 自动事件响应和取证框架存储库。

操作说明

部署 CloudFormation 模板

Task	描述	所需技能
部署 CloudFormation 模板。	CloudFormation 模板标记为 1 到 7，脚本名称的第一个单词表示需要在哪个帐户中部署模板。请注意，启动 CloudFormation 模板的顺序很重要。 • `1-forensic-AnalysisVPCnS3Buckets.yaml` ：部署在取证账户中。它会创建 Amazon S3 存储桶和分析 VPC，然后激活 CloudTrail。 • `2-forensic-MaintenanceVPCnEC2ImageBuilderPipeline.yaml` ：基于 SANS SIFT 部署维护 VPC 和映像生成器管道。 • `3-security_IR-Disk_Mem_automation.yaml` ：在安全账户中部署启用磁盘与内存采集的功能。 • `4-security_LiME_Volatility_Factory.yaml` ：启动构建函数以开始基于给定的 AMI IDs 创建内存模块。请注意，AMI 各 IDs 不相同 AWS 区域。无论何时需要新的内存模块，都可以使用新的 AMI IDs 重新	AWS 管理员

Task	描述	所需技能
	运行此脚本。考虑将其与您的黄金映像 AMI 构建器管道集成（如果在您的环境中使用）。 • `5-member-IR-automation.yaml` ：创建成员事件响应自动化功能，启动事件响应流程。它允许跨账户共享 Amazon Elastic Block Store (Amazon EBS) 卷、在事件响应过程中自动发布到 Slack 通道、启动取证过程以及在过程完成后隔离实例。 • `6-forensic-artifact-s3-policies.yaml` ：部署所有脚本后，此脚本修复了所有跨账户交互所需权限。 • `7-security-IR-vpc.yaml` ：配置用于事件响应量处理的 VPC。 要为特定 Amazon EC2 实例启动事件响应框架，请使用密钥`SecurityIncidentStatus` 和值创建一个标签`Analyze`。这将启动成员 Lambda 函数，该函数将自动启动隔离和内存以及磁盘获取。	

Task	描述	所需技能
操作框架。	Lambda 函数还将在`Contain`最后（或失败时）重新标记资产。这将启动遏制，从而将实例与无 INBOUND/OUTBOUND 安全组和不允许所有访问的 IAM 角色完全隔离。 按照GitHub 存储库中的步骤进行操作。	AWS 管理员

部署自定义 Security Hub 操作

Task	描述	所需技能
使用 CloudFormation 模板部署自定义 Security Hub 操作。	要创建自定义操作以便您可以使用 Security Hub 中的下拉列表，请部署`Modules/SecurityHub Custom Actions/SecurityHubCustomActions.yaml` CloudFormation 模板。然后修改每个成员账户中的 `IRAutomation` 角色，以允许运行该操作的 Lambda 函数代入 `IRAutomation` 角色。有关更多信息，请参阅GitHub 存储库。	AWS 管理员

相关资源

- AWS 安全事件响应指南

其他信息

通过使用此环境，安全运营中心 (SOC) 团队可以通过以下方式改进其安全事件响应流程：

- 能够在隔离环境中进行取证，以避免生产资源的意外泄露
- 拥有标准化、可重复、自动化的流程来进行遏制和分析。
- 使任何账户所有者或管理员都能够在不了解如何使用标签的情况下启动事件响应流程
- 拥有标准化、干净的环境来执行事件分析和取证，而不会受到更大环境的干扰
- 能够并行创建多项分析环境
- 将 SOC 资源集中在事件响应上，而非云取证环境的维护和文档记录上
- 从手动流程转向自动化流程，以实现可扩展性
- 使用 CloudFormation 模板来保持一致性并避免可重复的任务

此外，您可避免使用永久基础设施，且可以在需要时为资源付费。

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

自动修复 AWS Security Hub 标准调查结果

由 Chandini Penmetsa (AWS) 和 Aromal Raj Jayarajan (AWS) 编写

摘要

使用 AWS Security Hub，您可以启用对标准最佳实践的检查，例如：

- AWS 基础安全最佳实践
- 独联体 AWS 基金会基准
- 支付卡行业数据安全标准 (PCI DSS)

这些标准中的每一个都有预定义的控件。Security Hub 会检查给定对象中的控件 AWS 账户 并报告调查结果。

AWS Security Hub 默认情况下，会将所有调查结果发送给 EventBridge Amazon。此模式提供了一种安全控制，可部署 EventBridge 规则来识别 AWS 基础安全最佳实践标准调查结果。该规则根据基础安全最佳实践标准确定了自动扩展、虚拟私有云 (VPCs)、亚马逊弹性区块存储 (Amazon EBS) Block Store 和亚马逊关系数据库服务 (Amazon RDS) AWS 的以下发现：

- [AutoScaling.1] 与负载均衡器关联的 Auto Scaling 组应使用负载均衡器运行状况检查
- [EC2.2] VPC 默认安全组不应允许入站和出站流量
- [EC2.6] 应全部启用 VPC 流量记录 VPCs
- [EC2.7] 应启用 EBS 默认加密
- [RDS.1] RDS 快照应为私有快照
- [RDS.6] 应为 RDS 数据库实例和集群配置增强监控
- [RDS.7] RDS 集群应启用删除保护

该 EventBridge 规则将这些发现转发给一个 AWS Lambda 函数，该函数会对发现进行补救。Lambda 函数随后将包含补救信息的通知发送到 Amazon Simple Notification Service (Amazon SNS)主题。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 您希望在其中接收补救通知的电子邮件地址

- Security Hub 并在你打算部署控件 AWS 区域 的地方 AWS Config 启用
- 亚马逊简单存储服务 (Amazon S3) 存储桶，与用于上传代码的控件位于同一区域 AWS Lambda

限制

- 此安全控件会自动修复安全控制部署后报告的新调查发现。若要补救现有调查发现，请在 Security Hub 控制台上手动选择调查发现。然后，在 “操作” 下，选择在部署过程中创建的AFSBPRemedy自定义操作 AWS CloudFormation。
- 此安全控制是区域性的，必须部署在您 AWS 区域 要监控的中。
- 对于 EC2 .6 补救措施，要启用 VPC 流日志，将创建一个`/VpcFlowLogs/vpc_id`格式化的 Amazon CloudWatch 日志组。如果存在同名日志组，则将使用现有的日志组。
- 对于 EC2 .7 补救措施，要启用 Amazon EBS 默认加密，需要使用默认 AWS Key Management Service (AWS KMS) 密钥。此更改可防止使用某些不支持加密的实例。

架构

目标技术堆栈

- Lambda 函数
- Amazon SNS 主题
- EventBridge 规则
- AWS Identity and Access Management Lambda 函数、VPC 流日志和 Amazon RDS 增强监控的 (IAM) 角色

目标架构

自动化和扩缩

如果您正在使用 AWS Organizations，则可以使用AWS CloudFormation StackSets将此模板部署到您希望它监控的多个账户中。

工具

- AWS CloudFormation是一项通过使用基础架构即代码来帮助您建模和设置 AWS 资源的服务。

- Amazon EventBridge 提供来自您自己的应用程序、软件即服务 (SaaS) 应用程序的实时数据流 AWS 服务，并将这些数据路由到 Lambda 函数等目标。
- AWS Lambda支持在不预置或管理服务器的情况下运行代码。
- Amazon Simple Storage Service (Amazon S3) 是一项高度可扩展的对象存储服务，可用于各种存储解决方案，包括网站、移动应用程序、备份和数据湖。
- 亚马逊简单通知服务 (Amazon SNS) Simp le Notification Service 协调和管理发布者与客户之间的消息传送或发送，包括网络服务器和电子邮件地址。订阅用户接收所有发布至他们所订阅主题的消息，并且一个主题的所有订阅用户收到的消息都相同。

最佳实践

- 九种 AWS Security Hub 最佳实践
- AWS 基础安全最佳实践标准

操作说明

部署安全控件

Task	描述	所需技能
定义 Amazon S3 存储桶。	在 Amazon S3 控制台上，选择或创建具有唯一名称且不包含前导斜杠的 Amazon S3 存储桶。Amazon S3 存储桶名称是全球唯一的，命名空间由所有人共享 AWS 账户。您的 Amazon S3 存储桶必须与正在评估的 Security Hub 调查结果位于同一区域。	云架构师
将 Lambda 代码上传到亚马逊 S3 存储桶。	将 “附件” 部分中提供的 Lambda 代码.zip 文件上传到定义的 Amazon S3 存储桶。	云架构师
部署 AWS CloudFormation 模板。	部署作为该模式附件提供的 AWS CloudFormation 模板。	云架构师

Task	描述	所需技能
	在下一个操作说明中，提供参数的值。	

填写 AWS CloudFormation 模板中的参数

Task	描述	所需技能
提供 Amazon S3 存储桶名称。	输入您在第一个长篇故事中创建的 Amazon S3 存储桶的名称。	云架构师
提供 Amazon S3 前缀。	提供 Lambda 代码.zip 文件在您的 Amazon S3 存储桶中的位置，不要使用前导斜杠（例如，）。`<directory>/<file-name>.zip`	云架构师
提供亚马逊 SNS 主题的 ARN。	如果您想使用现有的 Amazon SNS 主题来发送补救通知，请提供亚马逊 SNS 主题的亚马逊资源名称 (ARN)。如果您想使用新的 Amazon SNS 主题，请将该值保留为`None`（默认值）。	云架构师
提供电子邮箱地址。	提供您想要接收补救通知的电子邮件地址（仅在您想要 AWS CloudFormation 创建 Amazon SNS 主题时才需要）。	云架构师
定义日志记录级别。	定义 Lambda 函数的日志记录级别和频率。`Info` 指明有关应用程序进度的详细信息消息。`Error` 指明仍允许应用程序继	云架构师

Task	描述	所需技能
	续运行的错误事件。Warning 指明潜在的有害情况。	
为 VPC 流日志提供 IAM 角色的 ARN。	提供用于 VPC 流日志的 IAM 角色的 ARN。如果您输入 None，则 AWS CloudFormation 创建一个 IAM 角色并使用它。	云架构师
为 Amazon RDS 增强监控提供 IAM 角色的 ARN。	提供用于 Amazon RDS 增强监控的 IAM 角色的 ARN。如果您输入None，则 AWS CloudFormation 创建一个 IAM 角色并使用它。	云架构师

确认订阅

Task	描述	所需技能
确认 Amazon SNS 订阅。	成功部署模板后，如果创建了新的 Amazon SNS 主题，则会向您提供的电子邮件地址发送订阅消息。您必须确认此订阅电子邮件消息，才能开始接收补救通知。	云架构师

相关资源

- 在 AWS CloudFormation 控制台上创建堆栈
- AWS Lambda 网站
- AWS Security Hub 文档

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 Amazon Inspector 自动对跨账户工作负载进行安全扫描 AWS Security Hub

由 Ramya Pulipaka (AWS) 和 Mikesh Khanal (AWS) 编写

摘要

此模式描述了如何自动扫描 Amazon Web Services (AWS) 云上跨账户工作负载中的漏洞。

该模式有助于为按标签分组的亚马逊弹性计算云 (Amazon EC2) 实例的基于主机的扫描或基于网络的 Amazon Inspector 扫描创建计划。 AWS CloudFormation 堆栈会将所有必需的 AWS 资源和服务部署到您的 AWS 账户。

Amazon Inspector 的调查结果将导出到 AWS Security Hub 您的账户 AWS 区域、虚拟私有云 (VPCs) 和亚马逊 EC2 实例中，并提供对这些漏洞的见解。您可以通过电子邮件接收这些调查发现，也可以创建一个 Amazon Simple Notification Service (Amazon SNS) 主题，使用 HTTP 端点将调查发现发送到票务工具、安全信息和事件管理 (SIEM) 软件或其他第三方安全解决方案。

先决条件和限制

先决条件

- 用于托管跨账户工作负载（包括中央审计账户）的活跃 AWS 账户 。
- 用于接收 Amazon SNS 电子邮件通知的现有电子邮件地址。
- 票证工具、SIEM 软件或其他第三方安全解决方案使用的现有 HTTP 端点。
- Security Hub，已启用并配置。您可以在没有 Security Hub 的情况下使用此模式，但我们建议使用 Security Hub，因为它会生成见解。有关更多信息，请参阅 Sec urity Hub 文档中的设置 Security Hub。
- 您要扫描的每个 EC2 实例上都必须安装 Amazon Inspector 代理。您可以使用 AWS Syst ems Manager 运行命令在多个 EC2 实例上安装 Amazon Inspector 代理。

技能

- 在中体验堆栈集的使用`self-managed`和`service-managed`权限 CloudFormation。如果您想使用`self-managed`权限将堆栈实例部署到特定区域的特定账户，则必须创建所需的 AWS Identity and Access Management (IAM) 角色。如果您想使用`service-managed`权限将堆栈实例部署到由特定区域管理 AWS Organizations 的账户，则无需创建所需的 IAM 角色。有关更多信息，请参阅 CloudFormation 文档中的创建堆栈集。

限制

- 如果没有对账户中的亚马逊 EC2 实例应用标签，则 Amazon Inspector 会扫描该账户中的所有实例。
- CloudFormation 堆栈集和`onboard-audit-account.yaml`文件（附件）必须部署在同一个区域。
- 默认情况下，Amazon Inspector Classic 不支持汇总调查发现。Security Hub是查看多个账户评估的推荐解决方案，或者 AWS 区域.
- 尽管限制因地区而异，但这种模式的方法可以在美国东部（弗吉尼亚北部）地区`us-east-1`的 Amazon SNS 主题每秒 30,000 笔交易 (TPS) 的发布配额之下进行扩展。为了更有效地扩展并避免数据丢失，我们建议在亚马逊 SNS 主题之前使用亚马逊简单队列服务 (Amazon SQS) Simple Queue Service。

架构

下图说明了自动扫描 Amazon EC2 实例的工作流程。

工作流程由以下步骤组成：

1. 亚马逊 EventBridge 规则使用 cron 表达式按特定计划自行启动并启动 Amazon Inspector。
2. Amazon Inspector 会扫描账户中已标记的亚马逊 EC2 实例。
3. Amazon Inspector 将调查发现发送至 Security Hub，后者会生成有关工作流、优先级划分和补救的见解。
4. Amazon Inspector 还将评估的状态发送到审计账户中的亚马逊 SNS 主题。如果将`findings reported`事件发布到 Amazon SNS 主题，则会调用一个 AWS Lambda 函数。
5. Lambda 函数提取和格式化调查结果，并将其发送到审计账户中的另一个 Amazon SNS 主题。
6. 调查结果将发送到订阅 Amazon SNS 主题的电子邮件地址。完整的详细信息和建议将以 JSON 格式发送到订阅的 HTTP 端点。

工具

- AWS CloudFormation帮助您对 AWS 资源进行建模和设置，这样您就可以花更少的时间管理这些资源，而将更多的时间集中在应用程序上。
- AWS CloudFormation StackSets通过允许您通过一次操作在多个账户和地区创建、更新或删除堆栈，扩展了堆栈的功能。

- AWS Control Tower创建了一个抽象层或编排层，该层结合并集成了其他几个层的功能 AWS 服务，包括。 AWS Organizations
- Amazon EventBridge 是一项无服务器事件总线服务，可以轻松地将您的应用程序与来自各种来源的数据连接起来。
- AWS Lambda是一项计算服务，可帮助您在不预置或管理服务器的情况下运行代码。
- AWS Security Hub让您全面了解自己在 AWS 中的安全状态，并帮助您根据安全行业标准和最佳实践检查您的环境。
- Amazon Simple Notification Service (Amazon SNS) 是一项托管服务，提供从发布者至订阅用户的消息传输。

操作说明

部署 CloudFormation 模板

Task	描述	所需技能
在审计账户中部署 CloudFormation 模板。	下载 `onboard-audit-account.yaml` 文件（附件）并将其保存到计算机上的本地路径。 登录您的审计账户，打开CloudFormation 控制台，然后选择创建堆栈。 AWS Management Console 选择先决条件部分中的准备模板，然后选择模板已就绪。选择指定模板部分中的模板源，然后选择模板已就绪。上传 `onboard-audit-account.yaml` 文件，然后根据您的要求配置其余选项。 请务必配置以下输入参数：	开发人员、安全工程师

Task	描述	所需技能
	• `DestinationEmailAddress` — 输入电子邮件地址以接收调查发现。 • `HTTPEndpoint` — 为您的票务或 SIEM 工具提供了 HTTP 端点。 **Note** 也可以使用 AWS Command Line Interface (AWS CLI) 部署 CloudFormation 模板。有关这方面的更多信息，请参阅 CloudFormation 文档中的创建堆栈。	
确认 Amazon SNS 订阅。	检查您的电子邮件收件箱，然后从 Amazon SNS 中的电子邮件中选择 Confirm subscription(确认订阅)。这会打开 Web 浏览器窗口并显示订阅确认信息。	开发人员、安全工程师

创建 CloudFormation 堆栈集以自动执行 Amazon Inspector 的扫描计划

Task	描述	所需技能
在审计账户创建堆栈集。	下载 `vulnerability-management-program.yaml` 文件（附件）到计算机上的本地路径。	开发人员、安全工程师

Task	描述	所需技能
	在CloudFormation 控制台上，选择 “查看堆栈集”，然后选择 “创建”。 StackSet选择模板已就绪，选择上传模板文件，然后上传该 **vulnerability-management-program.yaml** 文件。 如果要使用`self-managed` 权限，请按照 CloudFormation 文档中创建具有自我管理权限的堆栈集中的说明进行操作。这将在个人账户中创建堆栈集。 如果要使用`service-managed` 权限，请按照 CloudFormation 文档中使用服务管理权限创建堆栈集中的说明进行操作。这会在您的整个组织或指定的组织单位中创建堆栈集（OUs）。 确保为堆栈集配置了以下输入参数： • `AssessmentSchedule` — EventBridge 使用 cron 表达式的时间表。 • `Duration` – Amazon Inspector 评测运行的持续时间（以秒为单位）。 • `CentralSNSTopicArn` — 亚马逊 SNS 中心主题的亚马逊资源名称 (ARN)。	

Task	描述	所需技能
	• `Tagkey` — 与资源组关联的标签密钥。 • `Tagvalue` — 与资源组关联的标签值。 如果您想扫描审计账户中的 Amazon EC2 实例，则必须在审计账户中将`vulnerability-management-program.yaml` 文件作为 CloudFormation 堆栈运行。	
验证解决方案。	检查您是否按照您为 Amazon Inspector 指定的时间表通过电子邮件或 HTTP 端点收到调查发现。	开发人员、安全工程师

相关资源

- 使用 Amazon Inspector 扩展您的安全漏洞测试（AWS 博客文章）
- 自动修复 Amazon Inspector 的安全发现（AWS 博客文章）
- 如何使用 Amazon 和 Amazon EC2 Inspector 简化安全评估设置（AWS 博客文章） AWS Systems Manager

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

自动审核允许从公有 IP 地址进行访问 AWS 的安全组

由 Eugene Shifer (AWS) 和 Stephen DiCato (AWS) 创作

摘要

作为一种安全最佳实践，必须最大限度地减少 AWS 资源暴露于绝对必要的资源。例如，为公众提供服务的 Web 服务器需要允许从 Internet 进行入站访问，但应将对其他工作负载的访问限制在特定的网络上，以减少不必要的泄露。Amazon Virtual Private Cloud（亚马逊 VPC）中的@@ 安全组是一种有效的控制措施，可帮助您限制资源访问权限。但是，评估安全组可能是一项繁琐的任务，尤其是在多账户架构中。 AWS Config 规则和AWS Security Hub 控制可以帮助您识别允许从公共互联网 (0.0.0.0/0) 访问特定网络通信协议（例如安全外壳 (SSH)、HTTP、HTTPS 和 Windows 远程桌面协议 (RDP)）的安全组。但是，如果服务在非标准端口上运行或者仅限访问某些公有 IP 地址，则这些规则和控制不适用。例如，当 Web 服务与 TCP 端口 8443 而不是标准的 TCP 端口 443 关联时，可能会发生这种情况。当开发人员可以从其家庭网络访问服务器时（例如出于测试目的），也可能发生这种情况。

要解决这个问题，您可以使用此模式中提供的基础设施即代码 (IaC) 解决方案来识别允许从任何非私有（RFC 1918 不合规）IP 地址访问您或组织中的任何工作负载的安全组。 AWS 账户 AWS 该AWS CloudFormation模板提供了自定义 AWS Config 规则、AWS Lambda函数和必要的权限。您可以将其作为堆栈部署到单个账户中，也可以将其作为堆栈集部署到整个组织中，通过进行管理 AWS Organizations。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 使用经验 GitHub
- 如果你要部署到单个 AWS 账户：
  - 创建 CloudFormation 堆栈的@@ 权限
  - AWS Config 在目标账户中设置
  - （可选）在目标账户中设置 Sec urity Hub
- 如果您要部署到 AWS 组织中：
  - 创建 CloudFormation 堆栈集的@@ 权限
  - 通过 AWS Organizations 集成设置的 Security Hub
  - AWS Config 在您要部署此解决方案的账户中设置

- 指定一个作 AWS 账户 为 Security Hub AWS Config 的委托管理员

限制

- 如果您要部署到未启用 Security Hub 的个人账户，则可以使用 AWS Config 来评估结果。
- 如果您要部署到的组织没有 AWS Config 和 Security Hub 的委托管理员，则必须登录个人成员账户才能查看调查结果。
- 如果您使用 AWS Control Tower 管理和治理组织中的账户，请使用定制 AWS Control Tower (cfc T) 按此模式部署 IaC。使用 CloudFormation 控制台会导致配置偏离 AWS Control Tower 防护栏，并要求您重新注册组织单位 (OUs) 或托管账户。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅AWS 服务 按地区划分。有关特定终端节点，请参阅服务终端节点和配额页面，然后选择服务的链接。

架构

部署到个人中 AWS 账户

以下架构图显示了 AWS 资源在单个架构中的部署 AWS 账户。您可以直接通过 CloudFormation 控制台使用 CloudFormation 模板来配置资源。如果启用了 Security Hub，则可以在 AWS Config 或 Security Hub 中查看结果。如果未启用 Security Hub，则只能在中查看结果 AWS Config。

图表显示了以下工作流：

1. 你创建了一个 CloudFormation 堆栈。这将部署一个 Lambda 函数和 AWS Config 一条规则。规则和函数均使用在中发布资源评估 AWS Config 和日志所需的 AWS Identity and Access Management (IAM) 权限进行设置。
2. 该 AWS Config 规则在侦探评估模式下运行，每 24 小时调用一次 Lambda 函数。
3. Lambda 函数对安全组进行评估并将更新发送到。 AWS Config
4. Security Hub 会收到所有 AWS Config 调查结果。
5. 您可以在 Security Hub 或中查看调查结果 AWS Config，具体取决于您在账户中设置的服务。

部署到 AWS 组织中

下图显示了通过 AWS Organizations 和管理的多个账户部署该模式 AWS Control Tower。您可以通过 cfCT 部署 CloudFormation 模板。评估结果集中在 Security Hub 的委派管理员帐户中。图中的 AWS CodePipeline 工作流程部分显示了 cfCT 部署期间发生的背景步骤。

图表显示了以下工作流：

1. 在管理账户中，您可以将 iaC 模板的压缩 (ZIP) 文件上传到 cfcT 部署的亚马逊简单存储服务 (Amazon S3) 存储桶。
2. cfcT 管道解压缩文件，运行 cfn-nag (GitHub) 检查，并将模板部署为堆栈集。 CloudFormation
3. 根据您在 cfcT 清单文件中指定的配置，将堆栈 CloudFormation StackSets 部署到单个账户或指定账户。 OUs这将在目标账户中部署一个 Lambda 函数和 AWS Config 一条规则。规则和函数均使用在中发布资源评估 AWS Config 和日志所需的 IAM 权限进行设置。
4. 该 AWS Config 规则在侦探评估模式下运行，每 24 小时调用一次 Lambda 函数。
5. Lambda 函数对安全组进行评估并将更新发送到。 AWS Config
6. AWS Config 将所有发现结果转发给 Security Hub。
7. Security Hub 的发现结果汇总到委派的管理员帐户中。
8. 您可以使用委托管理员帐户在 Security Hub 中查看汇总结果。

工具

AWS 服务

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。
- AWS Config提供了您的资源 AWS 账户 及其配置方式的详细视图。它可以帮助您确定资源之间的关联方式，以及它们的配置如何随时间变化。 AWS Config 规则定义了您理想的资源配置设置，并 AWS Config 可以评估您的 AWS 资源是否符合规则中的条件。
- AWS Control Tower按照规范性最佳实践，帮助您设置和管理 AWS 多账户环境。AWS Control Tower (cfcT) 的自定义可帮助您自定义 AWS Control Tower 着陆区并与 AWS 最佳实践保持一致。此解决方案的定制是通过 CloudFormation 模板和 AWS Organizations 服务控制策略 (SCPs) 实现的。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。

- AWS Organizations是一项账户管理服务，可帮助您将多个账户整合 AWS 账户 到一个由您创建和集中管理的组织中。
- AWS Security Hub提供了您的安全状态的全面视图 AWS。它还可以帮助您根据安全行业标准和最佳实践检查您的 AWS 环境。

其他工具

- Python 是通用的计算机编程语言。

代码存储库

此模式的代码可在 GitHub 检测易受攻击的安全组存储库中找到。

最佳实践

我们建议您遵循以下资源中的最佳实践：

- 组织单位的最佳实践 AWS Organizations（AWS 云运营和迁移博客）
- 使用 AWS Control Tower on AWS（AWS 解决方案库）建立初始基础的指南
- 创建和修改 AWS Control Tower 资源的指南（AWS Control Tower 文档）
- cfCT 部署注意事项（AWS Control Tower 文档）
- 应用最低权限权限（IAM 文档）

操作说明

查看 CloudFormation 模板

Task	描述	所需技能
确定您的部署策略。	查看解决方案和代码以确定您的 AWS 环境的部署策略。确定您是部署到单个账户还是 AWS 组织中。	应用程序所有者，常规 AWS
克隆存储库。	输入以下命令克隆检测易受攻击的安全组存储库：	应用程序开发人员、应用程序所有者

Task	描述	所需技能
	`git clone https://github.com/aws-samples/detect-public-security-groups.git`	
验证 Python 版本。	1. 导航到克隆存储库中的顶级目录： `cd detect-public-security-groups` 2. 打开安全组-公共评估.yaml。 3. 在`SgPublicAccessCheckLambdaFunction` 资源中，确认 Python 版本与您的目标兼容 AWS 区域。默认情况下，此函数使用 Python 3.12。有关更多信息，请参阅AWS Lambda 添加对 Python 3.12 的支持。如有必要，请更新 Python 版本。 4. 保存并关闭安全组-公共评估.yaml。	AWS 管理员、应用程序开发人员

部署 CloudFormation 模板

Task	描述	所需技能
部署 CloudFormation 模板。	将 CloudFormation 模板部署到您的 AWS 环境中。请执行以下操作之一：	应用程序开发者、AWS 管理员、常规 AWS

Task	描述	所需技能
	• 如果您要部署到单个堆栈 AWS 账户，请按照创建堆栈中的说明进行操作。 • 如果您要部署到不受管理的组织 AWS Control Tower，请按照创建堆栈集中的说明进行操作。 • 如果您要部署到由管理的组织 AWS Control Tower，请参阅构建自己的自定义项中的说明。	
验证部署。	在CloudFormation 控制台中，验证堆栈或堆栈集是否已成功部署。	AWS 管理员、应用程序所有者

查看调查结果

Task	描述	所需技能
查看 AWS Config 规则调查结果。	在 Security Hub 中，执行以下操作以查看单个发现结果的列表： 1. 打开 S ecurity Hub 控制台。 2. 在导航窗格中，选择 调查发现。 3. 在 “添加筛选器” 框中，添加以下筛选器： • 合规状态为 FAILED • 标题是 SgPublicAccessCheck	AWS 管理员、AWS 系统管理员、云管理员

Task	描述	所需技能
	4. 选择应用。 在 Security Hub 中，执行以下操作以查看按以下分组的全部发现结果列表 AWS 账户： 1. 打开 S ecurity Hub 控制台。 2. 在导航窗格中，选择 Insights。 3. 选择 Create insight (创建见解)。 4. 要为见解选择分组属性，请执行以下操作： a. 选择搜索框以显示筛选选项。 b. 选择 Group by (分组依据)。 c. 选择 AwsAccountId。 d. 选择应用。 5. 在 “添加筛选器” 框中，添加以下筛选器： • 标题是 `SgPublicAccessCheck` • 合规状态为 `FAILED` 6. 选择 Create insight (创建见解)。 7. 输入洞察名称，然后选择创建洞察。 在中 AWS Config，要查看调查结果列表，请按照 AWS	

Task	描述	所需技能
	Config 文档中查看合规信息和评估结果中的说明进行操作。	

故障排除

事务	解决方案
创建或删除 CloudFormation 堆栈集失败。	部署后 AWS Control Tower ，它会强制执行必要的护栏，并控制 AWS Config 聚合器和规则。这包括防止通过 CloudFormation进行任何直接更改。要正确部署或删除此 CloudFormation 模板（包括所有相关资源），必须使用 cfcT。
cfcT 无法删除 CloudFormation 模板。	如果即使在清单文件中进行了必要的更改并删除了 CloudFormation 模板文件之后，模板仍然存在，请确认清单文件中是否包含`enable_stack_set_deletion` 参数并且该值已设置为`false`。有关更多信息，请参阅 cfCT 文档中的删除堆栈集。

相关资源

- AWS Config 自定义规则（AWS Config 文档）

使用 AWS Confi CloudTrail g 中的自定义补救规则自动重新启用 AWS

由 Manigandan Shri (AWS) 创建

摘要

可视化 Amazon Web Services (AWS) Account 中的活动是一项重要的安全和运营最佳实践。AWS CloudTrail 可帮助您对账户进行治理、合规以及运营和风险审计。

为了确保您的账户 CloudTrail 保持启用状态，AWS Config 提供了`cloudtrail-enabled`托管规则。如果已关闭， CloudTrail 则该`cloudtrail-enabled`规则会使用自动修复功能自动将其重新启用。

但是， CloudTrail 如果您使用自动修复，则必须确保遵循安全最佳实践。这些最佳实践包括在所有 AWS 区域 CloudTrail 中启用、记录读取和写入工作负载、启用见解以及使用 AWS 密钥管理服务 (AWS KMS) 托管密钥 (SSE-KMS) 托管密钥 (SSE-KMS) 使用服务器端加密对日志文件进行加密。

此模式通过提供自定义补救操作来自动 CloudTrail 在您的账户中重新启用，从而帮助您遵循这些安全最佳实践。

> **Important**
>
> 我们建议使用服务控制策略 (SCPs) 来防止任何篡 CloudTrail改。有关这方面的更多信息，请参阅 AWS 安全博客上的 “如何使用 AWS 组织大规模简化安全” 中的 “防止篡改 AWS” CloudTrail 部分。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 创建 AWS Systems Manager Automation 运行手册的权限
- 您账户中的现有跟踪

限制

此模式不支持以下操作：

- 设置存储位置的 Amazon Simple Storage Service (Amazon S3) 前缀密钥
- 发布至 Amazon Simple Notification Service (Amazon SNS) 主题
- 配置 Amazon CloudWatch 日志以监控您的 CloudTrail 日志

架构

技术堆栈

- AWS Config
- CloudTrail
- Systems Manager
- Systems Manager Automation

工具

- AWS Config 可提供您账户中 AWS 资源配置的详细视图。
- AWS CloudTrail 可帮助您实现账户的治理、合规以及运营和风险审计。
- AWS Key Management Service (AWS KMS)是一项加密和密钥管理服务。
- AWS Systems Manager 可帮助您查看和控制您在 AWS 上的基础设施。
- AWS Systems Manager Automation Automation 简化了亚马逊弹性计算云 (亚马逊 EC2) 实例和其他 AWS 资源的常见维护和部署任务。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

代码

cloudtrail-remediation-action.yml 文件（附后）可帮助您创建 Systems Manager 自动化运行手册，以便 CloudTrail 使用安全最佳实践进行设置和重新启用。

操作说明

配置 CloudTrail

Task	描述	所需技能
创建 S3 存储桶。	登录 AWS 管理控制台，打开 Amazon S3 控制台，然后创建一个 S3 存储桶来存储 CloudTrail 日志。有关更多信息，请参阅 Amazon S3 文档中的创建 S3 存储桶	系统管理员
添加存储桶策略 CloudTrail 以允许将日志文件传送到 S3 存储桶。	CloudTrail 必须具有将日志文件传输到 S3 存储桶所需的权限。在 Amazon S3 控制台上，选择您之前创建的 S3 存储桶，然后选择权限。使用 CloudTrail 文档中的 Amazon S3 存储桶策略创建 CloudTrail 3 存储桶策略。 有关如何向 S3 存储桶添加策略的步骤，请参阅 Amazon S3 文档中的使用 Amazon S3 控制台添加存储桶策略。 **Important** 如果您在中创建跟踪时指定了前缀 CloudTrail，请确保将其包含在 S3 存储桶策略中。前缀是 S3 对象键的可选附加内容，可在 S3 存储桶中创建类似于文件夹的组织结构。有关这方面的更多信息，请参	系统管理员

Task	描述	所需技能
	阅 CloudTrail 文档中的创建跟踪。	
创建 KMS 密钥。	为创建 AWS KMS 密钥 CloudTrail 以加密对象，然后再将其添加到 S3 存储桶。有关本故事的帮助，请参阅文档中的使用 AWS KMS 托管密钥 (SSE-KMS) 加密 CloudTrail 日志文件。 CloudTrail	系统管理员
为 KMS 密钥添加密钥政策。	附加 KMS 密钥策略 CloudTrail 以允许使用 KMS 密钥。有关本故事的帮助，请参阅文档中的使用 AWS KM S 托管密钥 (SSE-KMS) 加密 CloudTrail 日志文件。 CloudTrail **Important** CloudTrail 不需要`Decrypt`权限。	系统管理员
AssumeRole 为 Systems Manager 创建运行手册	为 Systems Manager Automation 创建运行手册的`AssumeRole` 。有关这方面的说明和更多信息，请参阅 Systems Manager 文档中的设置自动化。	系统管理员

创建并测试 Systems Manager 自动化运行手册

Task	描述	所需技能
创建 Systems Manager Automation 运行手册	使用 `cloudtrail-remediation-action.yml` 文件（附件）创建 Systems Manager Automation 运行手册。有关这方面的更多信息，请参阅 Systems Manager 文档中的创建 Systems Manager 文档。	系统管理员
测试运行手册。	在 Systems Manager 控制台上，测试您之前创建的 Systems Manager 自动化运行手册。有关这方面的更多信息，请参阅 Systems Manager 文档中的运行简单的自动化。	系统管理员

在 AWS Config 中设置自动修复规则

Task	描述	所需技能
添加 CloudTrail启用规则。	在 AWS Config 控制台上，选择规则，然后选择添加规则。在 Add rule 页面，选择 Add custom rule。在配置规则页面上，输入名称和说明，并添加 `cloudtrail-enabled` 规则。有关更多信息，请参阅 AWS Config 文档中的管理 AWS Config 规则。	系统管理员
添加自动修复操作。	从操作下拉列表，选择管理修复。选择“自动修复”，然后	系统管理员

Task	描述	所需技能
	选择您之前创建的 Systems Manager 运行手册。 以下是必需的输入参数 CloudTrail： • `CloudTrailName` • `CloudTrailS3BucketName` • `CloudTrailKmsKeyId` • `AssumeRole` （可选） 以下输入参数默认设置为 true： • `IsMultiRegionTrail` • `IsOrganizationTrail` • `IncludeGlobalServiceEvents` • `EnableLogFileValidation` 保留速率限制参数和资源 ID 参数的默认值。选择保存。 有关更多信息，请参阅 AWS Config 文档中的使用 AWS Config 规则修复不合规 AWS 资源。	

Task	描述	所需技能
测试自动修复规则。	要测试自动修复规则，请打开 CloudTrail 控制台，选择 Tra il s，然后选择跟踪。选择停止日志记录，以关闭该跟踪的日志记录。当系统提示您确认时，选择停止记录。 CloudTrail 停止记录该跟踪的活动。 按照 AWS Config 文档中评估您的资源中的说明进行操作，确保该功能 CloudTrail 已自动重新启用。	系统管理员

相关资源

配置 CloudTrail

- 创建 S3 存储桶
- 适用于 Amazon S3 存储桶政策 CloudTrail
- 使用 Amazon S3 控制台添加存储桶策略
- 创建跟踪
- 设置自动化
- 使用 AWS KMS 托管密钥 (SSE-KMS) 加密 CloudTrail 日志文件

创建和测试 Systems Manager Automation 运行手册

- 创建 Systems Manager 文档
- 运行简单的自动化

在 AWS Config 中设置自动修复规则

- 管理 AWS Config 规则
- 按照 AWS Config 规则修复不合规的 AWS 资源

其他资源

- AWS CloudTrail -安全最佳实践
- AWS Systems Manager 入门
- AWS Config 入门
- 开始使用 AWS CloudTrail

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

自动修复未加密的 Amazon RDS 数据库实例和集群

由 Ajay Rawat (AWS) 和 Josh Joy (AWS) 创建

摘要

此模式描述了如何使用 AWS Config、 AWS Systems Manager 运行手册和 () 密钥自动修复亚马逊网络服务 () 上未加密的亚马逊关系数据库服务 (Amazon RDS AWS) 数据库实例和集群 AWS Key Management Service (AWS KMS)。

RDS 加密的数据库实例通过保护您的数据免受未经授权的访问来为基础存储提供额外一层数据保护。您可以使用 Amazon RDS 加密来增强中部署的应用程序的数据保护 AWS Cloud，并满足静态加密的合规性要求。您可以在创建 RDS 数据库实例时为其启用加密，而不能在创建数据库实例之后启用加密。但是，您可以对未加密的 RDS 数据库实例添加加密，方法是创建数据库实例快照，然后创建此快照的加密副本。然后，您可以从加密快照还原数据库实例，从而获得原始数据库实例的加密副本。

此模式 AWS Config 规则 用于评估 RDS 数据库实例和集群。它使用 AWS Systems Manager 运行手册（定义要对不合规的 Amazon RDS 资源执行的操作）和加密数据库快照的 AWS KMS 密钥来应用补救。然后，它会强制执行服务控制策略 (SCPs)，以防止在未加密的情况下创建新的数据库实例和集群。

中提供了此模式的代码GitHub。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 此模式的GitHub 源代码存储库中的文件已下载到您的计算机
- 未加密 RDS 数据库实例或集群
- 用于加密 RDS 数据库实例和集群的现有 AWS KMS 密钥
- 访问以更新 KMS 密钥资源策略
- AWS Config 已在您的中启用 AWS 账户 （请参阅 AWS 文档 AWS Config中的入门）

限制

- 您只能在创建 RDS 数据库实例时为其启用加密，而不能在创建数据库实例之后启用加密。

- 您无法拥有未加密数据库实例的加密只读副本或加密数据库实例的未加密只读副本。
- 您不能将未加密的备份或快照还原到加密的数据库实例。
- Amazon RDS 加密适用于大多数数据库实例类。有关例外情况列表，请参阅 Amazon RDS 文档中的加密 Amazon RDS 资源。
- 要将加密快照从一个快照复制 AWS 区域 到另一个快照，您必须在目标中指定 KMS 密钥 AWS 区域。这是因为 KMS 密钥特定于 AWS 区域 在中创建它们的。
- 源快照在复制过程中保持加密状态。Amazon RDS 使用信封加密在复制过程中保护数据。有关更多信息，请参阅 AWS KMS 文档中的信封加密。
- 您无法对加密数据库实例取消加密。但是，您可以从加密的数据库实例中导出数据，然后将数据导入未加密的数据库实例。
- 只有当您确定不再需要使用 KMS 密钥时，才能将其删除。如果不确定，请考虑禁用 KMS 密钥，而不是将其删除。如果您稍后需要再次使用已禁用的 KMS 密钥，您可以重新启用，但您无法恢复已删除的 KMS 密钥。
- 如果您不选择保留自动备份，则会删除与数据库实例 AWS 区域 相同的自动备份。删除数据库实例后，无法恢复。
- 您的自动备份将保留您在删除数据库实例时对其设定的保留期。无论您是否选择创建最终数据库快照，都会出现此设置的保留期。
- 如果启用了自动修复，则此解决方案将加密所有具有相同 KMS 密钥的数据库。

架构

下图说明了 AWS CloudFormation 实现的架构。请注意，您也可以使用来实现此模式 AWS Cloud Development Kit (AWS CDK)。

工具

工具

- AWS CloudFormation帮助您自动设置 AWS 资源。其允许您使用模板文件创建并删除资源集合作为单一单元（堆栈）。
- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，用于在代码中定义您的云基础架构，并使用熟悉的编程语言对其进行配置。

Amazon Web Services 和特征

- AWS Config跟踪您的 AWS 资源配置及其与其他资源的关系。它还可以评估这些 AWS 资源的合规性。该服务使用可以配置的规则，根据所需的配置来评估 AWS 资源。您可以为常见的合规性场景使用一组 AWS Config 托管规则，也可以为自定义场景创建自己的规则。当发现 AWS 资源不合规时，您可以通过 AWS Systems Manager 运行手册指定补救措施，也可以选择通过亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 主题发送警报。换句话说，您可以将补救操作与关联 AWS Config 规则 并选择自动运行这些操作以解决不合规的资源，而无需手动干预。如果在自动修复后资源仍然不合规，则可以设置规则以再次尝试自动修复。
- Amazon Relational Database Service (Amazon RDS) 让您可以轻松地在云中设置、操作和扩展关系数据库。Amazon RDS 的基本构建块是数据库实例，它是中的一个隔离数据库环境 AWS Cloud。Amazon RDS 提供不同的实例类型，这些实例类型经过优化，适合不同的关系数据库用例。实例类型包括 CPU、内存、存储和网络容量的不同组合，便于您灵活选择适合数据库的资源组合。每种实例类型都包含多个实例大小，您可以根据目标工作负载的要求扩展数据库。
- AWS Key Management Service (AWS KMS) 是一项托管服务，可让您轻松创建和控制 AWS KMS keys，对您的数据进行加密。KMS 密钥是根密钥的逻辑表示形式。KMS 密钥包含元数据，如密钥 ID、创建日期、描述和密钥状态。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- 服务控制策略 (SCPs) 提供对组织中所有账户的最大可用权限的集中控制。 SCPs 帮助您确保您的帐户符合组织的访问控制准则。 SCPs 不要影响管理账户中的用户或角色。它们仅影响组织中的成员账户。我们强烈建议您在未彻底测试该政策对账户的影响之前，不要将其附加 SCPs 到组织的根目录上。您可以改为创建一个组织单位 (OU)，并将您的账户一次移入一个，或至少每次以少量移入，以确保您不会无意中阻止用户使用关键服务。

代码

此模式的源代码和模板可在GitHub 存储库中找到。该模式提供了两个实现选项：您可以部署 AWS CloudFormation 模板来创建用于加密 RDS 数据库实例和集群的修复角色，或者使用。 AWS CDK存储库为这两个选项设置了单独文件夹。

Epics 部分提供了部署 CloudFormation 模板的 step-by-step说明。如果要使用 AWS CDK，请按照 GitHub 存储库中`README.md`文件中的说明进行操作。

最佳实践

- 启用静态数据加密和传输中数据加密。

- AWS Config 在所有账户中启用和 AWS 区域。
- 记录所有资源类型的配置更改。
- 定期交替 IAM 凭据。
- 利用标记 AWS Config，这样可以更轻松地管理、搜索和筛选资源。

操作说明

创建 IAM 补救角色和 Systems Manager 运行手册

Task	描述	所需技能
下载 CloudFormation 模板。	从GitHub 存储库下载`unencrypted-to-encrypted-rds.template.json` 文件。	DevOps 工程师
创建 CloudFormation 堆栈。	1. 登录 AWS Management Console 并打开CloudFormation 控制台。 2. 启动 `unencrypted-to-encrypted-rds.template.json` 模板以创建新堆栈。 有关部署模板的更多信息，请参阅AWS CloudFormation 文档。	DevOps 工程师
查看 CloudFormation 参数和值。	1. 查看堆栈详细信息并根据您的环境要求更新值。 2. 选择创建堆栈，以部署模板。	DevOps 工程师
查看资源。	创建堆栈后，状态将变为 CREATE_COMPLETE。在 CloudFormation 控制台中	DevOps 工程师

Task	描述	所需技能
	查看已创建的资源（IAM 角色、Systems Manager 运行手册）。	

更新密 AWS KMS 钥政策

Task	描述	所需技能
更新 KMS 密钥政策。	1. 确保密钥别名 `alias/RDSEncryptionAtRestKMSAlias` 存在。 2. 密钥政策声明应包含 IAM 修复角色。（请查看您在上一篇长篇故事中部署的 CloudFormation 模板创建的资源。） 3. 在以下密钥政策中，更新以粗体显示的部分，以使其与您的账户和创建的 IAM 角色相匹配。 `{ "Sid": "Allow access through RDS for all principals in the account that are authorized to use RDS", "Effect": "Allow", "Principal": { "AWS": "arn:aws:iam:: <your-AWS-account-ID>”:role/<your-IAM-remediation-role>" },`	DevOps 工程师

Task	描述	所需技能

	"Action": [
	"kms:Encrypt",
	"kms:Decrypt",
	"kms:ReEn
	crypt*",
	"kms:Gene
	rateDataKey*",
	"kms:Crea
	teGrant",
	"kms:List
	Grants",
	"kms:Desc
	ribeKey"
	],
	"Resource": "*",
	"Condition": {
	"StringEquals":
	{
	"kms:ViaS
	ervice": "rds.us-e
	ast-1.amazonaws.com",
	"kms:Call
	erAccount": "<your-AW
	S-account-ID>"
	}
	}
	}	

查找和修复不合规的资源

Task	描述	所需技能
查看不合规资源。	1. 要查看不合规资源的列表，请打开AWS Config 控制台。 2. 在导航窗格中，选择规则，然后选择 `rds-storage-encrypted` 规则。	DevOps 工程师

Task	描述	所需技能
	AWS Config 控制台中列出的不合规资源将是实例，而不是集群。修复自动化可加密实例与集群，并创建新加密实例或新建集群。但是，请务必不要同时修复属于同一集群的多个实例。 在修复任何 RDS 数据库实例或卷前，请确保 RDS 数据库实例未在使用中。确认创建快照时未进行写入操作，以确保快照包含原始数据。考虑强制执行维护时段，在此期间进行修复。	
修复不合规的资源。	1. 准备就绪且维护时段生效后，选择要修复的资源，然后选择修复。 现在操作状态列应显示操作执行已排队。 2. 在 Systems Manager 中查看修复进度和状态。打开 Systems Manager 控制台。在导航窗格中，选择自动化，然后选择相应的自动化的执行 ID，以查看更多详细信息。	DevOps 工程师

Task	描述	所需技能
验证 RDS 数据库实例是否可用。	自动化完成后，新加密的 RDS 数据库实例将变为可用。加密 RDS 数据库实例将具有前缀 `encrypted` ，其后其原始名称。例如，如果未加密的 RDS 数据库实例名称为 `database-1` ，则新加密的 RDS 数据库实例将为 `encrypted-database-1` 。	DevOps 工程师
终止未加密实例。	修复完成且新加密的资源经过验证后，您可以终止未加密的实例。确保终止任何资源之前已确认新加密资源与未加密资源相匹配。	DevOps 工程师

强制执行 SCPs

Task	描述	所需技能
强制执行 SCPs。	强制执行 SCPs ，以防止将来在未加密的情况下创建数据库实例和集群。为此，请使用GitHub 存储库中提供的`rds_encrypted.json` 文件，并按照AWS 文档中的说明进行操作。	安全工程师

相关资源

参考

- 设置 AWS Config
- AWS Config 自定义规则

- AWS KMS 概念
- AWS Systems Manager 文档
- 服务控制策略

工具

- AWS CloudFormation
- AWS Cloud Development Kit (AWS CDK)

指南和模式

- 使用中的自定义修正规则自动重新启用 AWS CloudTrailAWS Config

其他信息

如何运 AWS Config 作？

在您使用时 AWS Config，它会首先发现您账户中存在的受支持 AWS 资源，然后为每种资源生成一个配置项目。 AWS Config 当资源配置发生变化时，它还会生成配置项目，并且它会维护自启动配置记录器之时起的资源配置项目的历史记录。默认情况下， AWS Config 会为中每个支持的资源创建配置项目 AWS 区域。如果您不 AWS Config 想为所有支持的资源创建配置项目，则可以指定希望它跟踪的资源类型。

与之有何 AWS Config 规则 关系 AWS Config 和关系 AWS Security Hub？

AWS Security Hub 是一项安全与合规服务，将安全与合规状态管理作为一项服务提供。它使用 AWS Config 和 AWS Config 规则 作为其主要机制来评估 AWS 资源配置。 AWS Config 规则 也可以用来直接评估资源配置。其他 AWS 服务，例如 AWS Control Tower 和 AWS Firewall Manager，也使用 AWS Config 规则。

使用 AWS Organizations 和 AWS Secrets Manager 大规模自动轮换 IAM 用户访问密钥

由 Tracy Hickey (AWS)、Gaurav Verma (AWS)、Laura Seletos (AWS)、Michael Davie (AWS) 和 Arvind Patel (AWS) 创建

摘要

> **Important**
>
> 作为最佳实践，AWS 建议您使用 AWS Identity and Access Management (IAM) 角色，而不是拥有访问密钥等长期证书的 IAM 用户。此模式中记录的方法仅适用于需长期使用的 AWS API 凭证的旧版实施。对于这些实施，我们仍然建议您考虑使用短期证书的选项，例如使用亚马逊弹性计算云 (Amazon EC2) 实例配置文件或 IAM Anywhere 角色。本文所述方法仅适用于您无法立即改为使用短期凭证、并且需要按计划轮换长期凭证的情况。借助此方法，您有责任定期更新旧版应用程序代码或配置，以使用轮换后的 API 凭证。

访问密钥是 IAM 用户的长期凭证。定期轮换您的 IAM 凭证，有助于防止被盗用的 IAM 访问密钥集访问您的 Amazon Web Services Account中的组件。轮换 IAM 凭证也是 IAM 安全最佳实践的重要组成部分。

此模式可帮助您使用 IAM 密钥轮换存储库中提供的 AWS CloudFormation 模板自动轮换 GitHub IAM 访问密钥。

此模式支持在单个或多个账户中部署。如果您使用的是 AWS Organizations，则此解决方案可识别您组织 IDs 中的所有 AWS 账户，并随着账户的删除或新账户的创建而动态扩展。集中式 AWS Lambda 函数使用代入 IAM 角色，以在您选择的多个账户中于本地运行轮换函数。

- 现有访问密钥过期 90 天时将生成新的 IAM 访问密钥。
- 新访问密钥以密钥形式存储至 AWS Secrets Manager。基于资源的策略仅允许指定IAM 主体访问和检索密钥。如果选择将密钥存储至管理账户，则所有账户的密钥均应存储至管理账户。
- 分配至创建新访问密钥的 Amazon Web Services Account 的所有者的电子邮件地址会收到通知。
- 先前访问密钥在 100 天后停用，然后在 110 天后删除。
- 将向 Amazon Web Services account 拥有者发送一封集中式电子邮件通知。

Lambda 函数和 Amazon CloudWatch 会自动执行这些操作。然后，您可以检索新的访问密钥对，并在代码或应用程序中将其替换。轮换、删除和停用期限可自定义。

先决条件和限制

- 至少一个活跃 Amazon Web Services Account。
- AWS Organizations，已配置并设置（请参阅教程）。
- 从管理账户查询 AWS Organizations 的权限。有关更多信息，请参阅 AWS Organizations 文档中的 AWS Organizations 和服务相关角色。
- 有权启动 AWS CloudFormation 模板和相关资源的 IAM 委托人。有关更多信息，请参阅 AWS CloudFormation 文档中的授予自我管理权限。
- 用于部署资源的现有 Amazon Simple Storage Service (Amazon S3)存储桶。
- Amazon Simple Email Service (Amazon SES) 已移出沙盒。有关更多信息，请参阅 Amazon SES 文档中的脱离 Amazon SES 沙盒。
- 如果您选择在虚拟私有云 (VPC) 中运行 Lambda，则应在运行主 CloudFormation 模板之前创建以下资源：
  - VPC。
  - 子网。
  - Amazon SES、AWS Systems Manager、AWS Security Token Service (AWS STS)、Amazon S3 和 AWS Secrets Manager 的端点。（您可以运行 GitHub IAM 密钥轮换存储库中提供的终端节点模板来创建这些终端节点。）
- 存储在 AWS Systems Manager 参数（SSM 参数）中的 Simple Mail Transfer Protocol (SMTP) 用户和密码。参数必须与主 CloudFormation 模板参数相匹配。

架构

技术堆栈

- Amazon CloudWatch
- Amazon EventBridge
- IAM
- AWS Lambda
- AWS Organizations
- Amazon S3

架构

下图显示了此模式的组件和工作流程。该解决方案支持两种凭证存储方案：储存至成员账户和储存至管理账户。

选项 1：将凭证存储至成员账户

选项 2：将凭证存储至管理账户

图表显示了以下工作流程：

1. 一个 EventBridge 事件每 24 小时启动一个 Lamb `account_inventory` da 函数。
2. 此 Lambda 函数向 AWS Organizations 查询所有 AWS 账户 IDs、账户名称和账户电子邮件的列表。
3. `account_inventory` Lambda 函数为每个 Amazon Web Services Account ID 启动一个 `access_key_auto_rotation` Lambda 函数，并将元数据传递至该函数进行额外处理。
4. `access_key_auto_rotation` Lambda 函数使用代入 IAM 角色访问 Amazon Web Services Account ID。Lambda 脚本对账户中的所有用户及其 IAM 访问密钥进行审计。
5. 如果 IAM 访问密钥使用期限未超过最佳实践阈值，则 Lambda 函数将不采取进一步的行动。
6. 如果 IAM 访问密钥使用期限已超过最佳实践阈值，则 `access_key_auto_rotation` Lambda 函数将决定要执行的轮换操作。
7. 当需要执行操作时，如果生成了新密钥，则 `access_key_auto_rotation` Lambda 函数会在 AWS Secrets Manager 中创建并更新密钥。还创建了仅允许指定的 IAM 主体访问和检索密钥的基于资源的策略。对于选项 1，将凭证存储至相应账户的 Secrets Manager。对于选项 2（如果 `StoreSecretsInCentralAccount` 标志设置为 True），则将凭证存储至管理账户的 Secrets Manager。
8. 启动 `notifier` Lambda 函数，以通知轮换活动的账户所有者。此函数接收 Amazon Web Services Account 的 ID、账户名称、账户电子邮件以及已执行轮换操作。
9. `notifier` Lambda 函数在部署 S3 存储桶中查询电子邮件模板，并使用相关活动元数据动态更新该模板。然后，电子邮件将发送至账户所有者的电子邮件地址。

备注：

- 此解决方案支持多个可用区的弹性。但是，其不支持多个 AWS 区域的弹性。要在多个区域获得支持，您可以在第二个区域部署解决方案并禁用密钥轮换 EventBridge 规则。然后，当您想在第二个区域运行解决方案时，您可以启用该规则。
- 您可以在审核模式下运行此解决方案。在审核模式下，不会对 IAM 访问密钥进行任何修订，但会发送一封电子邮件通知用户。若要在审计模式下运行解决方案，请在运行密钥轮换模板时或在 `access_key_auto_rotation` Lambda 函数的环境变量中将 `DryRunFlag` 标志设置为 True。

自动化和扩缩

自动执行此解决方案的 CloudFormation 模板在 GitHub IAM 密钥轮换存储库中提供，并列在 “代码” 部分中。在 AWS Organiz CloudFormation StackSetsations 中，您可以使用在多个账户中部署`ASA-iam-key-auto-rotation-iam-assumed-roles.yaml` CloudFormation 模板，而不必将解决方案单独部署到每个成员账户。

工具

Amazon Web Services

- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和在 AWS 上运行的应用程序的指标。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Organizations 是一项账户管理服务，使您可将多个 Amazon Web Services account 整合到您所创建的组织中并进行集中管理。
- AWS Secrets Manager 帮助您将代码中的硬编码凭证（包括密码）替换为对 Secrets Manager 的 API 调用，以便以编程方式检索密钥。
- Amazon Simple Storage Service(Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- Amazon Simple Email Service (Amazon SES) 帮助您通过使用您自己的电子邮件地址和域发送和接收电子邮件。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。

- Amazon VPC 终端节点提供了一个接口，用于连接由 AWS 提供支持的服务 PrivateLink，包括许多 AWS 服务。对于您在 VPC 中指定的每个子网，我们将在子网中创建一个端点网络接口，并为其分配子网地址范围内的私有 IP 地址。

代码

所需的 AWS CloudFormation 模板、Python 脚本和运行手册文档可在 GitHub IAM 密钥轮换存储库中找到。模板按以下方式部署。

模板	部署	备注
`ASA-iam-key-auto-rotation-and-notifier-solution.yaml`	部署账户	这是此解决方案的主要模板。
`ASA-iam-key-auto-rotation-iam-assumed-roles.yaml`	您想要轮换凭证的单个或多个成员账户	您可以使用 CloudFormation 堆栈集在多个账户中部署此模板。
`ASA-iam-key-auto-rotation-list-accounts-role.yaml`	Central/管理账户	使用此模板保存 AWS Organizations 的账户清单。
`ASA-iam-key-auto-rotation-vpc-endpoints.yaml`	部署账户	仅在 VPC 中运行 Lambda 函数时才使用此模板自动创建端点 (在主模板中将 `RunLambdaInVPC` 参数设置为 True)。

操作说明

设置解决方案

Task	描述	所需技能
选择您的部署 S3 存储桶。	登录您的账户的 AWS 管理控制台，打开 Amazon S3 控制台，然后选择要部署的	云架构师

Task	描述	所需技能
	S3 存储桶。如果想在 AWS Organizations 中为多个账户实施该解决方案，请登录组织管理账户。	
克隆存储库。	将 GitHub IAM 密钥轮换存储库克隆到您的本地桌面。	云架构师
将文件上传至 S3 存储桶。	将已克隆文件上传至 S3 存储桶。使用以下默认文件夹结构复制并粘贴所有已克隆文件和目录：`asa/asa-iam-rotation` **Note** 您可以在 CloudFormation 模板中自定义此文件夹结构。	云架构师
修改电子邮件模板。	根据您的要求修改 `iam-auto-key-rotation-enforcement.html` 电子邮件模板 (位于 `template` 文件夹中)。用您的部门名称替换模板末尾的 `[Department Name Here]`。	云架构师

部署解决方案

Task	描述	所需技能
启动密钥轮换 CloudFormation 模板。	1. 在部署账户中启动 `ASA-iam-key-auto-rotation-and-notifi`	云架构师

Task	描述	所需技能
	er-solution.yaml 模板。有关更多信息，请参阅 CloudFormation 文档中的选择堆栈模板。 2. 为参数指定值，包括： • CloudFormation S3 存储桶名称 (S3BucketName)-包含您的 Lambda 代码的部署 S3 存储桶的名称。 • CloudFormation S3 存储桶前缀 (S3BucketPrefix)-S3 存储桶的前缀。 • 假定 IAM 角色名称 (IAMRoleName)- key-rotation Lambda 函数在轮换密钥时将采用的假定角色名称。 • IAM 执行角色名称 (ExecutionRoleName)- key-rotation Lambda 函数使用的 IAM 执行角色的名称。 • 库存执行角色名称 (Inventory Execution RoleName)- account_inventory Lambda 函数使用的 IAM 执行角色的名称。 • Dry Run Flag (Audit Mode) (DryRunFlag)	

Task	描述	所需技能
	- 设置为 True 可开启审核模式（默认）。设置为 False 以开启强制模式。 • 列出组织账户的账户（`OrgListAccount`）- 用于列出组织中的账户的中央/管理账户的账户 ID。 • 列出账户角色名称（`OrgListRole`）- 用于列出组织中账户的角色名称。 • 中央账户的 Secrets Store 标志（`StoreSecretsInCentralAccount`）– 设置为 True 可将密钥存储至中央账户。设置为 False 以将密钥存储至相应账户。 • 要复制凭证的区域（`CredentialReplicationRegions`）– 以逗号分隔的要复制凭证的 AWS 区域 (Secrets Manager)；例如 `us-east-2,us-west-1,us-west-2`。跳过您要在其中创建堆栈的区域。 • 在 VPC 中运行 Lambda（`RunLambdaInVpc`）– 设置为 True，以在指定 VPC 中运行 Lambda 函	

Task	描述	所需技能
	数。您必须创建 VPC 端点，并将 NAT 网关连接至包含 Lambda 函数的子网。有关更多信息，请参阅介绍此选项的 re:Post 文章。 • Lambda 函数的 VPC ID (VpcId)、适用于安全组规则的 VPC CIDR (VpcCidr) 和 Lambda 函数的子网 ID (SubnetId) – 如果您将 RunLambdaInVpc 设置为 True，请提供有关 VPC、CIDR 和子网的信息。 • 管理员电子邮件地址 (AdminEmailAddress) - 用于向其发送通知的有效电子邮件地址。 • AWS Organization ID (AWSOrgID) – 您的组织的唯一 ID。此 ID 以 o- 开头，其后为 10-32 个小写字母或数字。 • 电子邮件模板文件名【审核模式】 (EmailTemplateAudit) 和 【强制模式】 (EmailTemplateEnforce) - notifier 模块在审核模式和强制模式下发送的电子邮件 HTML 模板的文件名。	

Task	描述	所需技能
	• SMTP 用户 SSM 参数名称 (`SMTPUserParamName`) 和 SMTP 密码 SSM 参数名称 (`SMTPPasswordParamName`) - Simple Mail Transfer Protocol (SMTP) 的用户和密码信息。	

Task	描述	所需技能
启动代入角色的 CloudFormation 模板。	1. 在 AWS CloudFormation 控制台中，为要轮换密钥的每个账户启动`ASA-iam-key-auto-rotation-iam-assumed-roles.yaml` 模板。如果您有多个账户，则可以将管理账户中的主 CloudFormation 模板部署为堆栈，并将包含堆 CloudFormation 栈集的`ASA-iam-key-auto-rotation-iam-assumed-roles.yaml` 模板部署到所有必需的账户。有关更多信息，请参阅 CloudFormation 文档 CloudFormation StackSets中的使用 AWS。 2. 为以下参数指定值： • 假定 IAM 角色名称 (`IAMRoleName`) - Lambda `access_key_auto_rotation` 函数将采用的假定 IAM 角色名称。您可以保留默认值。 • IAM 执行角色名称 (`ExecutionRoleName`) – 将担任子账户角色以运行 Lambda 函数的 IAM 角色。 • 主要 Amazon Web Services Account ID (`PrimaryAccountID`) – 将在其中部署主模板的	云架构师

Task	描述	所需技能
	Amazon Web Services Account ID。 • IAM 豁免组 (`IAMExemptionGroup`) – 便于从自动密钥轮换中排除 IAM 账户的 IAM 组名称。	

Task	描述	所需技能
启动账户库存 CloudFormation 模板。	1. 在管理/中央账户中启动 `ASA-iam-key-auto-rotation-list-accounts-role.yaml` 模板 2. 为以下参数指定值： • 假定 IAM 角色名称 (`IAMRoleName`) - Lambda `access_key_auto_rotation` 函数将采用的假定 IAM 角色名称。 • 账户 Lambda 的 IAM 执行角色名称 (`AccountExecutionRoleName`) – Lambda `notifier` 函数将担任的 IAM 角色的名称。 • 轮换 Lambda 的 IAM 执行角色名称 (`RotationExecutionRoleName`) – Lambda `access_key_auto_rotation` 函数将担任的 IAM 角色的名称。 • 主要 Amazon Web Services Account ID (`PrimaryAccountID`) – 将在其中部署主模板的 Amazon Web Services Account ID。	云架构师

Task	描述	所需技能
启动 VPC 终端节点的 CloudFormation 模板。	此任务是可选的。 1. 在部署账户中启动 `ASA-iam-key-auto-rotation-vpc-endpoints.yaml` 模板。 2. 为以下参数指定值： • VPC ID (`pVpcId`)、子网 ID (`pSubnetId`) 和 VPC 的 CIDR 范围 (`pVPCCidr`) - 提供有关 VPC、CIDR 和子网的信息。 • 将每个 VPC 端点参数设置为 True。如果已有端点，则可选择 False。	云架构师

相关资源

- IAM 中的安全最佳实践（IAM 文档）
- AWS Organizations 和服务相关角色（AWS Organizations 文档）
- 选择堆栈模板（CloudFormation 文档）
- 使用 AWS CloudFormation StackSets（CloudFormation 文档）

使用 CodePipeline IAM Access Analyzer 和 AWS CloudFormation 宏在 AWS 账户中自动验证和部署 IAM 策略和角色

由 Helton Ribeiro (AWS) 和 Guilherme Simoes (AWS) 创作

摘要

注意： CodeCommit AWS 不再向新客户开放。AWS 的现有客户 CodeCommit 可以继续照常使用该服务。了解更多。

此模式描述了创建部署管道的步骤并提供了代码，允许您的开发团队在您的 Amazon Web Services (AWS) Account 中创建 AWS Identity and Access Management (IAM) 策略和角色。此方法可帮助您的组织减少运营团队开销并加快部署进程。其亦有助于开发人员创建与您现有治理和安全控制兼容的 IAM 角色和策略。

此模式的方法使用 AWS Identity and Access Management Access Analyzer 来验证您要附加到 IAM 角色的 IAM 策略，并使用 AWS CloudFormation 来部署 IAM 角色。但是，您的开发团队不会直接编辑 AWS CloudFormation 模板文件，而是创建 JSON 格式的 IAM 策略和角色。在开始部署之前，AWS CloudFormation 宏会将这些 JSON 格式的策略文件转换为 AWS CloudFormation IAM 资源类型。

部署管道 (`RolesPipeline`) 分为源、验证和部署阶段。在源代码阶段，您的开发团队会将包含 IAM 角色和策略定义的 JSON 文件推送到 AWS CodeCommit 存储库。 CodeBuild 然后，AWS 运行脚本来验证这些文件，并将它们复制到亚马逊简单存储服务 (Amazon S3) 存储桶。由于您的开发团队无法直接访问存储在单独的 S3 存储桶中的 AWS CloudFormation 模板文件，因此他们必须遵循 JSON 文件的创建和验证流程。

最后，在部署阶段，AWS CodeDeploy 使用 AWS CloudFormation 堆栈来更新或删除账户中的 IAM 策略和角色。

> **Important**
>
> 此模式的工作流程是概念验证 (POC)，我们建议您仅在测试环境中使用该工作流程。如果想在生产环境中使用此模式方法，请参阅 IAM 文档中的 IAM 安全最佳实践，并对您的 IAM 角色和 Amazon Web Service 进行必要的更改。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 用于 `RolesPipeline` 管道的新或现有 S3 存储桶。确保正在使用的访问凭证有权将对象上传至此存储桶。
- 已安装和配置 AWS 命令行界面（AWS CLI）。有关这方面的更多信息，请参阅 AWS CLI 文档中的安装、更新和卸载 AWS CLI。
- AWS Serverless Application Model (AWS SAM)CLI,已安装并配置。有关这方面的更多信息，请参阅 AWS SAM 文档中的安装 AWS SAM CLI。
- Python 3，已安装于本地计算机。有关这方面的更多信息，请参阅 Python 文档。
- Git 客户端，已安装并配置。
- GitHub `IAM roles pipeline`存储库，克隆到您的本地计算机。
- 现有 JSON 格式的 IAM policy 与角色。有关这方面的更多信息，请参阅 Github `IAM roles pipeline` 存储库中的ReadMe文件。
- 您的开发团队不得拥有编辑此解决方案的 AWS CodePipeline CodeBuild、和 CodeDeploy 资源的权限。

限制

- 此模式的工作流程是概念验证 (POC)，我们建议您仅在测试环境中使用该工作流程。如果想在生产环境中使用此模式方法，请参阅 IAM 文档中的 IAM 安全最佳实践，并对您的 IAM 角色和 Amazon Web Service 进行必要的更改。

架构

下图向您展示了如何使用 CodePipeline IAM Access Analyzer 和 AW CloudFormation S 宏自动验证 IAM 角色和策略并将其部署到账户。

图表显示了以下工作流：

1. 开发人员编写包含 IAM policy 与角色定义的 JSON 文件。开发人员将代码推送到 CodeCommit 存储库 CodePipeline ，然后启动`RolesPipeline`管道。
2. CodeBuild 使用 IAM 访问分析器验证 JSON 文件。如果存在任何与安全或错误相关的调查发现，则部署进程将停止。
3. 如果无与安全或错误相关的调查发现，则将 JSON 文件发送至 `RolesBucket` S3 存储桶。

4. 然后，作为 AWS Lambda 函数实现的 AWS CloudFormation 宏从`RolesBucket`存储桶中读取 JSON 文件并将其转换为 AWS CloudFormation IAM 资源类型。
5. 预定义的 AWS CloudFormation 堆栈会安装、更新或删除账户中的 IAM 策略和角色。

自动化和扩缩

IA GitHub M 角色管道存储库中提供了自动部署此模式的 AWS CloudFormation 模板。

工具

- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- IAM Access Analyzer帮助您标识企业和账户中与外部实体共享的资源，例如 S3 存储桶或 IAM 角色。这可以帮助您识别对资源和数据的意外访问。
- AWS Serverless Application Model（AWS SAM）是一个开源框架，帮助您在 Amazon Web Services Cloud 中构建无服务器应用程序。

代码

此模式的源代码和模板可在 GitHub IAM 角色管道存储库中找到。

操作说明

克隆存储库

Task	描述	所需技能
克隆示例存储库。	将 GitHub IAM 角色管道存储库克隆到您的本地计算机。	应用程序开发人员，常规 AWS

部署 RolesPipeline 管道

Task	描述	所需技能
部署管道。	1. 导航到包含已克隆存储库的目录。 2. **Important** 运行`make deploy bucket=<bucket_name>` 命令。:您必须替换`<bucket_name>` 为现有 S3 存储桶的存储桶名称。 3. 运行 `aws codepipeline get-pipeline –name RolesPipeline` 命令，以检查您的部署是否成功。	应用程序开发人员，常规 AWS
克隆管道存储库。	1. A `RolesPipeline` WS CloudFormation 堆栈创建`roles-pipeline-repo` CodeCommit 存储库。 2. 登录 AWS 管理控制台，打开 AWS CodeCommit 控制台，然后复制 CodeCommit 存储库的 URL 以将其克隆到您的本地计算机。有关这方面的更多信息，请参阅 AWS CodeCommit 文档中的 Connect 到 AWS CodeCommit 存储库。	应用程序开发人员，常规 AWS

测试 RolesPipeline 管道

Task	描述	所需技能
使用有效的 IAM 策略和角色测试 RolesPipeline 管道。	1. 为 IAM policy 和角色创建 JSON 文件。您可以使用 GitHub `IAM roles pipeline`存储库中`role-example` 目录中的示例。 2. **Important** 使用所需配置定义 IAM policy 和角色。：请确保遵循 GitHub `IAM roles pipeline`存储库中`ReadMe`文件中描述的格式。 3. 将修改内容推送到`roles-pipeline-repo` CodeCommit 存储库中。 4. 验证 `RolesPipeline` 管道的实施。 5. 确保在账户中适当部署 IAM policy 和角色。 6. 验证是否存在与 IAM policy 或者角色关联的权限边界。有关这方面的更多信息，请参阅 IAM 文档中的IAM 实体的权限边界。	应用程序开发人员，常规 AWS
使用无效的 IAM 策略和角色测试 RolesPipeline 管道。	1. 修改`roles-pipeline-repo` CodeCommit 存储库并添加无效的 IAM 角色或策略。例如，您可以使用不存	应用程序开发人员，常规 AWS

Task	描述	所需技能
	在的操作或无效 IAM policy 版本。 2. 验证管道的实施。如果 IAM Access Analyzer 检测到无效 IAM policy 或角色，则将在验证阶段停止管道。	

清除资源

Task	描述	所需技能
准备清理。	清空 S3 存储桶，然后运行 `destroy` 命令。	应用程序开发人员，常规 AWS
删除 RolesStack 堆栈。	1. 该`RolesPipeline` 管道创建了一个用于部署 IAM 策略和角色的 A `RolesStack` WS CloudFormation 堆栈。您必须先删除堆栈，然后才能删除 `RolesPipeline` 管道。 2. 登录 AWS 管理控制台，打开 AWS CloudFormation 控制台，然后选择`RolesStack` 堆栈并选择删除。	应用程序开发人员，常规 AWS
删除 RolesPipeline 堆栈。	要删除 `RolesPipeline` AWS CloudFormation 堆栈，请按照 Github `IAM roles pipeline` 存储库中ReadMe文件的说明进行操作。	应用程序开发人员，常规 AWS

相关资源

- IAM Access Analyzer - 策略验证（AWS 新闻博客）
- 使用 AW CloudFormation S 宏对模板执行自定义处理（AWS CloudFormation 文档）
- 使用 Python 构建 Lambda 函数（AWS Lambda 文档）

将 AWS Security Hub 与 Jira 软件双向集成

创建者：Joaquin Manuel Rinaudo (AWS)

摘要

该解决方案支持 AWS Security Hub 与 Jira 之间的双向集成。使用此解决方案，您可根据 Security Hub 的调查发现自动手动创建和更新 JIRA 票证。安全团队可以使用此集成通知开发团队需要采取行动的严重安全调查发现。

此解决方案允许您：

- 选择哪些 Security Hub 控件自动在 Jira 中创建或更新票证。
- 在 Security Hub 控制台上，使用 Security Hub 自定义操作在 Jira 中手动升级票证。
- 根据 AWS Organizations 中定义的 Amazon Web Services account 标签，在 Jira 中自动分配票证。如未定义此标签，则使用默认受让人。
- 自动抑制 Jira 中标记为误报或已接受风险的 Security Hub 调查发现。
- 当 Jira 票证的相关调查发现存档在 Security Hub 中时，自动关闭 Jira 票证。
- 当 Security Hub 调查发现再次发生时，重新打开 Jira 票证。

Jira 工作流

此解决方案使用自定义 Jira 工作流，允许开发人员管理和记录风险。当问题在工作流中移动时，双向集成可确保 Jira 票证和 Security Hub 调查发现的状态在两个服务的工作流中同步。该工作流程是 Dinis Cruz 的《SecDevOps 风险工作流程》的衍生作品，已获得 CC BY 4.0 许可。我们建议添加 Jira 工作流条件，以便仅您的安全团队成员才能更改票证状态。

有关此解决方案自动生成的 Jira 票证的示例，请参阅此模式的其他信息部分。

先决条件和限制

先决条件

- 如果您要在多账户 AWS 环境中部署此解决方案，请执行以下操作：
  - 您的多账户环境处于活动状态，则由 AWS Organizations 管理。

  - 您的 Amazon Web Services account 已启用 Security Hub。
  - 在 AWS Organizations 中，您已经指定 Security Hub 管理员账户。
  - 您的跨账户 IAM 角色拥有 AWS Organizations 管理账户的 `AWSOrganizationsReadOnlyAccess` 权限。
  - （可选）您已使用 `SecurityContactID` 标记您的 Amazon Web Services account。此标签用于将 Jira 票证分配至定义的安全联系人。
- 如果您想在单个 Amazon Web Services account 中部署此解决方案：
  - 您已经有一个有效的 Amazon Web Services account。
  - Amazon Web Services account 已启用 Security Hub。
- Jira Server 实例

> **Important**
>
> 此解决方案支持使用 Jira Cloud。但是，Jira Cloud 不支持导入 XML 工作流，因此需要在 Jira 中手动重新创建工作流。

- Jira 中的管理员权限
- 以下 Jira 令牌之一：
  - 对于 Jira Enterprise 来说，这是个人访问令牌 (PAT)。有关更多信息，请参阅使用个人访问令牌（Atlassian 支持）。
  - 对于 Jira Cloud 来说，这是 Jira API 令牌。有关更多信息，请参阅管理 API 令牌（Atlassian 支持）。

架构

本节说明了各种场景下的解决方案架构，例如当开发人员和安全工程师决定接受风险或决定解决问题时。

场景 1：开发人员解决问题

1. Security Hub 针对指定的安全控制措施生成调查发现，例如 AWS 基础安全防御最佳实践标准。
2. 与调查结果和`CreateJIRA`操作关联的亚马逊 CloudWatch 事件会启动 AWS Lambda 函数。
3. Lambda 函数使用其配置文件和调查发现的 `GeneratorId` 字段来评估是否应升级调查发现。
4. Lambda 函数决定应上报调查发现，它从 AWS 管理账户中的 AWS Organizations 的 `SecurityContactID` 账户获取账户标签。此 ID 与开发人员关联，用作 Jira 票证的受让人 ID。

5. Lambda 函数使用存储在 AWS Secrets Manager 中的凭证在 Jira 中创建票证。Jira 会通知开发人员。
6. 开发人员解决了底层的安全调查发现，并在 Jira 中将票证状态更改为 `TEST FIX`。
7. Security Hub 将调查发现更新为 `ARCHIVED`，并生成新事件。此事件将导致 Lambda 函数自动关闭 Jira 票证。

场景 2：开发人员决定接受风险

1. Security Hub 针对指定的安全控制措施生成调查发现，例如 AWS 基础安全防御最佳实践标准。
2. 与发现和`CreateJIRA`操作关联 CloudWatch 的事件会启动 Lambda 函数。
3. Lambda 函数使用其配置文件和调查发现的 `GeneratorId` 字段来评估是否应升级调查发现。
4. Lambda 函数决定应上报调查发现，它从 AWS 管理账户中的 AWS Organizations 的 `SecurityContactID` 账户获取账户标签。此 ID 与开发人员关联，用作 Jira 票证的受让人 ID。
5. Lambda 函数使用存储在 Secrets Manager 中的凭证在 Jira 中创建票证。Jira 会通知开发人员。
6. 开发人员决定接受风险，并在 Jira 中将票证的状态更改为 `AWAITING RISK ACCEPTANCE`。
7. 安全工程师审查请求并找到适当业务理由。安全工程师将 Jira 票证的状态更改为 `ACCEPTED RISK`。这将关闭 Jira 票证。
8. CloudWatch 每日事件会启动刷新 Lambda 函数，该函数可识别已关闭的 JIRA 票证并将其相关的 Security Hub 发现结果更新为。`SUPPRESSED`

工具

- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。
- Amazon CloudWatch Events 使用规则匹配事件并将其路由到函数或流，从而帮助您监控 AWS 资源的系统事件。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Organizations 是一项账户管理服务，使您可将多个 Amazon Web Services account 整合到您所创建的组织中并进行集中管理。

- AWS Secrets Manager 帮助您将代码中的硬编码凭证（包括密码）替换为对 Secrets Manager 的 API 调用，以便以编程方式检索密钥。
- AWS Security Hub 向您提供在 AWS 中安全状态的全面视图。您可使用它根据安全行业标准和最佳实践检查您的环境。

代码存储库

此模式的代码可在 aws-securityhub-jira-software-integration GitHub 存储库中找到。它包括该解决方案的示例代码与 Jira 工作流。

操作说明

配置 Jira

Task	描述	所需技能
导入工作流。	作为 Jira 的管理员，将 `issue-workflow.xml` 文件导入至 Jira Server 实例。该文件可以在的 aws-securityhub-jira-software-integration 存储库中 GitHub找到。有关说明，请参阅使用 XML 创建工作流（Jira 文档）。	Jira 管理员
激活和分配工作流。	在您将工作流分配至工作流方案之前，工作流处于非活动状态。然后，将工作流方案分配至项目。 1. 对于您的项目，请确保您已为此项目确定了问题类型方案。您可创建新的问题类型或从现有问题类型中进行选择，例如 Bug。 2. 根据激活工作流（Jira 文档）中的说明，将导入的工作流分配给工作流方案。	Jira 管理员

Task	描述	所需技能
	3. 根据将工作流方案与项目关联（Jira 文档）中的说明，将工作流架构分配至项目。	

设置解决方案参数

Task	描述	所需技能
配置解决方案参数。	1. 在 conf 文件夹中，打开 `params_prod.shfile` 。 2. 为以下参数提供值： • `ORG_ACCOUNT_ID` – 您的 AWS Organizations 管理账户的账户 ID。该解决方案读取账户标签，并将票证分配至在 Amazon Web Services account 标签中定义的特定安全联系人。 • `ORG_ROLE` – 用于访问 AWS Organizations 管理账户的 IAM 角色的名称。此角色必须具有 `OrganizationsReadOnlyAccess` 权限。 • `EXTERNAL_ID` – 一个可选参数，指明是否使用外部 ID 代入 `ORG_ROLE` 中定义的 IAM 角色。有关更多信息，请参阅如何使用外部 ID（IAM 文档）。	AWS 系统管理员

Task	描述	所需技能
	• JIRA_DEFAULT_ASSIGNEE – 这是所有安全问题的默认受让人的 Jira ID。如果账户未正确标记或无法代入角色，则使用此默认分配。 • JIRA_INSTANCE – 您的 Jira 服务器的 HTTPS 地址采用以下格式：team-<team-id>.atlassian.net/ • JIRA_PROJECT_KEY – 用于创建票证的 Jira 项目密钥的名称，例如 SEC 或 TEST。此项目必须已在 Jira 中。 • ISSUE_TYPE – 在 Jira 中分配给项目的议题类型方案的名称，例如 Bug 或 Security Issue。 • REGIONS – 您要在其中部署此解决方案的 AWS 区域 代码列表，例如 eu-west-1 。 3. 保存和关闭解决方案参数文件。	

Task	描述	所需技能
确定要自动执行的调查发现。	1. 打开 Security Hub 控制台 https://console.aws.amazon.com/securityhub/ 2. 在 Security Hub 导航窗格中，选择调查发现。 3. 选择调查发现标题。 4. 选择调查发现 ID。这将显示调查发现的完整 JSON。 5. 在 JSON 中，复制 `GeneratorId` 字段中的字符串。此值采用 AWS Security 调查发现格式 (ASFF)。例如，`aws-foundational-security-best-practices/v/1.0.0/S3.1` 对应于来自安全控件应启用 S3.1 S3 阻止公有访问设置的调查发现。 6. 重复这些步骤，直到复制了所有要自动执行的调查发现的 `GeneratorID` 值。	

Task	描述	所需技能
将调查发现添加到配置文件中。	1. 在 src/code 中，打开文件 `config.jsonconfig` 。 2. 将您在上一个情节中检索到的 `GeneratorID` 值粘贴到 `default` 参数，并使用逗号分隔每个 ID。 3. 保存并关闭配置文件。 以下代码示例显示了自动执行 `aws-foundational-security-best-practices/v/1.0.0/SNS.1` 和 `aws-foundational-security-best-practices/v/1.0.0/S3.1` 调查发现。 `{ "Controls" : { "eu-west-1": [ "arn:aws:securityhub:::ruleset/cis-aws-foundations-benchmark/v/1.2.0/rule/1.22" ], "default": [ aws-foundational-security-best-practices/v/1.0.0/SNS.1, aws-foundational-security-best-practices/v/1.0.0/S3.1 ] } }`	AWS 系统管理员

Task	描述	所需技能
	Note 您可以选择为每个 AWS 区域自动执行不同的调查结果。为了防止重复的调查发现，可选择一个区域来自动创建与 IAM 相关的控件。	

部署集成

Task	描述	所需技能
部署集成。	在命令行终端中，输入以下命令： `./deploy.sh prod`	AWS 系统管理员
将 Jira 凭证上传至 AWS Secrets Manager。	1. 打开 Secrets Manager 控制台，网址为https://console.aws.amazon.com/secretsmanager/。 2. 在“密钥”下方，选择存储新密钥。 3. 对于密钥类型，请选择其他密钥类型。 4. 如果您使用 Jira Enterprise，对于键/值对，请执行以下操作： • 在第一行，在密钥框中输入 `auth`，然后在值框中输入 `token_auth` 。	AWS 系统管理员

Task	描述	所需技能
	• 添加第二行，在密钥框中输入 `token`，然后在值框中输入您的个人访问令牌。 如果您使用 Jira Cloud，对于键/值对，请执行以下操作： • 在第一行，在密钥框中输入 `auth`，然后在值框中输入 `basic_auth` 。 • 添加第二行，在密钥框中输入 `token`，然后在值框中输入您的 API 令牌。 • 添加第三行，在密钥框中输入 `email`，然后在值框中输入您的电子邮件地址。 5. 选择下一步。 6. 对于密钥名称，输入 `Jira-Token` ，然后在页面底部选择下一步。 7. 在“密钥轮换”页面上，保持禁用自动轮换，然后在该页面底部，选择下一步。 8. 在“查看”页面上，查看密钥详细信息，然后选择存储。	

Task	描述	所需技能
创建 Security Hub 自定义操作。	1. 对于每个 AWS 区域，在 AWS 命令行界面 (AWS CLI) Line CLI 中，使用create-action-target命令创建名为的 Security Hub 自定义操作`CreateJiraIssue` 。 `aws securityhub create-action-target --name "CreateJiraIssue" \ --description "Create ticket in JIRA" \ --id "CreateJiraIssue" --region $<aws-region>` 2. 打开 Security Hub 控制台，网址为https://console.aws.amazon.com/securityhub/。 3. 在 Security Hub 导航窗格中，选择调查发现。 4. 在调查发现列表中，选择要上报的调查发现。 5. 在操作菜单中，选择 `CreateJiraIssue` 。	AWS 系统管理员

相关资源

- 适用于 Jira Service Management 的 AWS 服务管理连接器
- AWS 基础安全防御最佳实践标准

其他信息

Jira 票证示例

当出现指定的 Security Hub 调查发现，此解决方案会自动创建一个 Jira 票证。该票证包含以下信息：

- 标题 – 标题采用以下格式标识安全问题：

```
AWS Security Issue :: <AWS account ID> :: <Security Hub finding title>
```

- 描述 – 票证的描述部分描述了与调查发现相关的安全控制，包括指向 Security Hub 控制台中调查发现的链接，并简要描述了如何处理 Jira 工作流中的安全问题。

以下是一个自动生成的 Jira 票证示例。

标题	AWS Security Issue :: 012345678912 :: Lambda.1 Lambda 函数策略应禁止公有访问。
描述	问题在哪里？ 我们在您负责的 Amazon Web Services account 012345678912 中检测到安全调查发现。 此控件检查附加到 Lambda 资源的 AWS Lambda 函数策略是否禁止公有访问。如果 Lambda 函数策略允许公有访问，则控件失败。 <链接至 Security Hub 调查发现> 我需要怎么处理该票证？ • 访问此账户并验证配置。将其移至已分配待修复，以确认正在处理票证。修复后，移至测试修复，以便 Security 可验证问题是否得到解决。 • 如果您认为应该接受风险，请将其移至等待风险接受。这将需要安全工程师审查。

- 如果您认为是误报，请将其转换为标记为误报。这将由安全工程师进行审查并相应地重新打开/关闭。

使用 Image Builder 和 Terraform 为经过强化的容器 EC2 镜像构建管道

创建者：Mike Saintcross (AWS) 和 Andrew Ranes (AWS)

摘要

这种模式构建了一个 EC2 Image Builder 管道，用于生成经过强化的 Amazon Linux 2 基础容器映像。Terraform 用作基础设施即代码（IaC）工具，它可配置和预调配基础设施用于创建经过强化的容器映像。该配方可帮助您部署基于 Docker 的 Amazon Linux 2 容器映像，该映像已根据 Red Hat Enterprise Linux (RHEL) 7 STIG 版本 3 第 7 版—Medium 进行了强化。（参见 Im EC2 age Builder 文档的 Linux STIG 组件部分中的 2022.2.1 STIG-Build-Linux-Medium 版。）这被称为黄金容器映像。

该版本包括两 EventBridge 条 Amazon 规则。一条规则是，当 Amazon Inspector 调查发现为高或严重时，将启动容器映像管线，以便替换不安全的映像。这条规则要求同时启用 Amazon Inspector 和 Amazon Elastic Container Registry (Amazon ECR) 增强型扫描。另一条规则在成功将映像推送到 Amazon ECR 存储库后，向 Amazon Simple Queue Service (Amazon SQS) 队列发送通知，以帮助您使用最新的容器映像。

> **Note**
>
> 亚马逊 Linux 2 的支持已接近终止。欲了解更多信息，请参阅亚马逊 Linux 2 FAQs。

先决条件和限制

先决条件

- 一个 Amazon Web Services account，您可在其中部署基础设施。
- AWS 命令行界面（AWS CLI）已安装用于设置您的 AWS 凭证以供本地部署。
- 已按照 Terraform 文档中的说明下载 Terraform 并设置。
- Git（如果您从本地计算机进行预调配）。
- Amazon Web Services account 中的角色，可用于创建 AWS 资源。
- .tfvars 文件中定义的所有变量。 或者，您可在应用 Terraform 配置时定义所有变量。

限制

- 该解决方案创建了一个 Amazon Virtual Private Cloud (Amazon VPC) 基础设施，其中包括一个 NAT 网关和一个用于从其私有子网连接互联网的互联网网关。您不能使用 VPC 终端节点，因为 AWS Task Orchestrator 和 Executor () 的引导过程会从AWSTOE互联网上安装 AWS CLI 版本 2。

产品版本

- Amazon Linux 2
- AWS CLI 版本 1.1 或更高版本

架构

目标技术堆栈

这种模式创建 43 项资源，包括：

- 两个 Amazon Simple Storage Service (Amazon S3) 存储桶：一个用于管线组件文件，一个用于服务器访问和 Amazon VPC 流日志
- Amazon ECR 存储库
- 包含一个公有子网、一个私有子网、一个私有子网、路由表、一个 NAT 网关以及一个互联网网关的虚拟私有云（VPC）
- I EC2 mage Builder 管道、配方和组件
- 容器映像
- AWS Key Management Service (AWS KMS) 密钥，用于映像加密
- SQS 队列
- 三个角色：一个用于运行 EC2 Image Builder 管道，一个用于 EC2 Image Builder 的实例配置文件，一个用于 EventBridge 规则
- 两 EventBridge 条规则

Terraform 模块结构

有关源代码，请参阅 GitHub 存储库 Terraform EC2 Image Builder 容器强化管道。

```
### components.tf
### config.tf
### dist-config.tf
### files
```

```
#   ###assumption-policy.json
### hardening-pipeline.tfvars
### image.tf
### infr-config.tf
### infra-network-config.tf
### kms-key.tf
### main.tf
### outputs.tf
### pipeline.tf
### recipes.tf
### roles.tf
### sec-groups.tf
### trigger-build.tf
### variables.tf
```

模块详细信息

- components.tf 包含用于上传 /files 目录内容的 Amazon S3 上传资源。您也可在此处以模块化方式添加自定义组件 YAML 文件。
- /files 包含用来定义 components.tf 中所用组件的 .yml 文件。
- image.tf 包含基本映像操作系统的定义。在这里，您可修改不同基础映像管线的定义。
- infr-config.tf 和 dist-config.tf 包含启动和分发映像所需最低 AWS 基础设施所需资源。
- infra-network-config.tf 包含要将容器映像部署到的最低 VPC 基础设施。
- hardening-pipeline.tfvars 包含要在应用时所用的 Terraform 变量。
- pipeline.tf在 Terraform 中创建和管理 EC2 Image Builder 管道。
- recipes.tf 是您可以指定不同的组件混合物来创建容器配方的位置。
- roles.tf包含亚马逊弹性计算云 (Amazon) 实例配置文件和管道部署角色的 AWS Identity and Access Management (IAM EC2) 策略定义。
- trigger-build.tf包含 EventBridge 规则和 SQS 队列资源。

目标架构

该图说明了以下工作流程：

1. EC2 Image Builder 使用定义的配方构建容器映像，该配方安装操作系统更新并将 RHEL Medium STIG 应用于 Amazon Linux 2 基础映像。

2. 经过强化的映像将发布到私有 Amazon ECR 注册表，成功发布映像后， EventBridge 规则会向 SQS 队列发送一条消息。
3. 如果 Amazon Inspector 配置为增强扫描，它将扫描 Amazon ECR 注册表。
4. 如果 Amazon Inspector 为图像生成了 “临界” 或 “高” 严重性发现， EventBridge 则规则会触发 EC2 Image Builder 管道再次运行并发布经过强化处理的新图像。

自动化和扩缩

- 此模式描述了如何在计算机上预调配基础设施并构建管线。但是它旨在大规模使用。与其在本地部署 Terraform 模块，不如在多账户环境中使用它们，例如带有 Account Factory for Terraform 环境的 AWS Control Tower。在这种情况下，您应该使用后端状态 S3 存储桶管理 Terraform 状态文件，而不是在本地管理配置状态。
- 为了扩大使用范围，可将解决方案从 Control Tower 或登录区账户模型部署到一个中央账户，例如共享服务或公共服务账户，并授予消费者账户访问 Amazon ECR 存储库和 AWS KMS 密钥的权限。有关设置的更多信息，请参阅 re:Post 文章如何允许辅助账户在我的 Amazon ECR 映像存储库中推送或拉取图片？ 例如，在账户自动售卖机或 Account Factory for Terraform 中，向每个账户基准或账户自定义基准添加权限，以提供对该 Amazon ECR 存储库和加密密钥的访问权限。
- 部署容器镜像管道后，您可以使用 Im EC2 age Builder 功能（例如组件）对其进行修改，这些功能可以帮助您将更多组件打包到 Docker 版本中。
- 用于加密容器映像的 AWS KMS 密钥应在要使用该映像的账户之间共享。
- 您可通过复制整个 Terraform 模块并修改以下 `recipes.tf` 属性来添加对其他映像的支持：
  - 将 `parent_image = "amazonlinux:latest"` 修改为其他映像类型。
  - 将 `repository_name` 修改为指向现有的 Amazon ECR 存储库。这将创建另一个管线，该管线将不同的父映像类型部署到您现有的 Amazon ECR 存储库。

工具

工具

- Terraform（IaC 预调配）
- Git（如果在本地预调配）
- AWS CLI 版本 1 或版本 2（如果在本地预调配）

代码

此模式的代码位于 GitHub 存储库 Terraform EC2 Image Builder 容器强化管道中。要使用示例代码，请按照下一部分中的说明进行操作。

操作说明

预调配基础设施

Task	描述	所需技能

设置本地凭证。	设置您的 AWS 临时凭证。
	1. 查看 AWS CLI 是否已安装：
	`$ aws --version aws-cli/1.16.249 Python/3.6.8...`
	• AWS CLI 版本应为 1.1 或更高版本。
	• 如果找不到该命令，安装 AWS CLI。
	2. 运行 `aws configure` 并提供以下值：
	`$ aws configure AWS Access Key ID [**********xxxx]: <Your AWS access key ID> AWS Secret Access Key [**********xxxx]: <Your AWS secret access key> Default region name: [us-east-1]: <Your desired Region for deployment>`
		AWS DevOps

Task	描述	所需技能
	`Default output format [None]: <Your desired output format>`	

克隆存储库。	1. 克隆随此模式提供的存储库。您可使用 HTTPS 或 Secure Shell (SSH)。
	
	HTTPS：
	
	`git clone https://github.com/aws-samples/terraform-ec2-image-builder-container-hardening-pipeline`
	
	SSH：
	
	`git clone git@github.com:aws-samples/terraform-ec2-image-builder-container-hardening-pipeline.git`
	2. 导航到包含此解决方案的本地目录：
	
	`cd terraform-ec2-image-builder-container-hardening-pipeline`
		AWS DevOps

Task	描述	所需技能

更新变量。	更新 `hardening-pipeline.tfvars` 文件中的变量以匹配您的环境和所需配置。您必须自己提供 `account_id` 。但是，您还应该修改其余变量，以适应所需部署。所有变量均为必需项。
	
	`account_id = "<DEPLOYMENT-ACCOUNT-ID>"`
	`aws_region = "us-east-1"`
	`vpc_name = "example-hardening-pipeline-vpc"`
	`kms_key_alias = "image-builder-container-key"`
	`ec2_iam_role_name = "example-hardening-instance-role"`
	`hardening_pipeline_role_name = "example-hardening-pipeline-role"`
	`aws_s3_ami_resources_bucket = "example-hardening-ami-resources-bucket-0123"`
	`image_name = "example-hardening-al2-container-image"`
	`ecr_name = "example-hardening-container-repo"`
	`recipe_version = "1.0.0"`
	`ebs_root_vol_size = 10`
		AWS DevOps

Task	描述	所需技能
	以下是关于每个变量的描述： • `account_id` – 您要将解决方案部署到的 AWS 账号。 • `aws_region` – 您要将解决方案部署到的 AWS 区域。 • `vpc_name` – 您的 VPC 基础设施的名称。 • `kms_key_alias` – Im EC2 age Builder 基础设施配置要使用的 AWS KMS 密钥名称。 • `ec2_iam_role_name` – 将用作 EC2 实例配置文件的角色的名称。 • `hardening_pipeline_role_name` – 将用于部署强化管线的角色的名称。 • `aws_s3_ami_resources_bucket` – S3 存储桶的名称，该存储桶将托管构建管线和容器映像所需所有文件。 • `image_name` – 容器映像名称。此值必须介于 3 到 50 个字符间，并且只能包含字母数字字符和连字符。 • `ecr_name` – 用于存储容器映像的 Amazon ECR 注册表的名称。	

Task	描述	所需技能
	• `recipe_version` – 映像配方的版本。默认值为 1.0.0。 • `ebs_root_vol_size` – Amazon Elastic Block Store (Amazon EBS) 根卷的大小（以 GB 为单位）。默认值为 10 GB。	
初始化 Terraform。	更新变量值后，您可初始化 Terraform 配置目录。初始化配置目录会下载并安装配置中定义的 AWS 提供程序。 `terraform init` 您应该看到一条消息，指出 Terraform 已成功初始化并标识了已安装的提供程序的版本。	AWS DevOps
部署基础设施并创建容器映像。	使用以下命令通过使用 `.tfvars` 文件中定义的变量来初始化、验证 Terraform 模块并将其应用于环境： `terraform init && terraform validate && terraform apply -var-file *.tfvars -auto-approve`	AWS DevOps

Task	描述	所需技能
自定义容器。	在 EC2 Image Builder 部署管道和初始配方后，您可以创建容器配方的新版本。 您可以添加 EC2 Image Builder 中可用的 31 个以上组件中的任何一个来自定义容器构建。有关更多信息，请参阅 I EC2 mage Builder 文档中创建新版本的容器配方的 “组件” 部分。	AWS 管理员

验证资源

Task	描述	所需技能
验证 AWS 基础设施预调配。	成功完成第一个 Terraform `apply` 命令后，如果您在本地预调配，则应在本地计算机的终端中看到以下片段： `Apply complete! Resources: 43 added, 0 changed, 0 destroyed.`	AWS DevOps
验证各个 AWS 基础设施资源。	要验证已部署的各个资源，如果您在本地预调配，则可以运行以下命令： `terraform state list` 此命令将返回 43 项资源列表。	AWS DevOps

删除资源

Task	描述	所需技能
移除基础设施和容器映像。	使用 Terraform 配置后，可运行以下命令来移除资源： `terraform init && terraform validate && terraform destroy -var-file *.tfvars -auto-approve`	AWS DevOps

故障排除

事务	解决方案
验证提供商凭证时出错	在本地计算机上运行 Terraform `apply` 或 `destroy` 命令时，可能会遇到类似以下内容的错误： `Error: configuring Terraform AWS Provider: error validating provider credentials: error calling sts:GetCallerIdentity: operation error STS: GetCallerIdentity, https response error StatusCode: 403, RequestID: 123456a9-fbc1-40ed-b8d8-513d0133ba7f, api error InvalidClientTokenId: The security token included in the request is invalid.` 此错误是由本地计算机配置中使用的凭证安全令牌过期引起的。 要解决该错误，请参阅 AWS CLI 文档中的设置和查看配置设置。

相关资源

- Terraform EC2 Image Builder 容器强化管道（存储库）GitHub
- EC2 Image Builder 文档
- 适用于 Terraform 的 AWS Control Tower Account Factory（AWS Blog 文章）
- 后端状态 S3 存储桶（Terraform 文档）
- 安装或更新最新版本的 AWS CLI（AWS CLI 文档）
- 下载 Terraform

使用 Terraform 在 AWS Organizations 中集中管理 IAM 访问密钥

由 Aarti Rajput (AWS)、Chintamani Aphale (AWS)、T.V.R.L.Phani Kumar Dadi (AWS)、Pradip kumar Pandey (AWS)、Mayuri Shinde (AWS) 和 Pratap Kumar Nanda (AWS) 创作

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多

强制执行密钥和密码的安全规则是每个组织的一项基本任务。一项重要规则是定期轮换 AWS Identity and Access Management (IAM) 密钥以加强安全性。每当团队想要通过 AWS 命令行界面 (AWS CLI) 或 AWS 之外的应用程序访问 AWS 时，通常都会在本地创建和配置 AWS 访问密钥。为了维护整个组织的强大安全性，必须在满足要求后或定期更改或删除旧的安全密钥。管理组织中多个账户的密钥轮换的过程既耗时又乏味。这种模式可帮助您使用适用于 Terraform 的 Account Factory (AFT) 和 AWS 服务自动执行轮换流程。

该模式具有以下优点：

- 从一个中心位置管理组织中所有账户的访问密钥 IDs 和私有访问密钥。
- 自动轮换`AWS_ACCESS_KEY_ID`和`AWS_SECRET_ACCESS_KEY`环境变量。
- 如果用户凭证遭到泄露，则强制续订。

该模式使用 Terraform 来部署 AWS Lambda 函数、 EventBridge 亚马逊规则和 IAM 角色。EventBridge 规则定期运行并调用 Lambda 函数，该函数根据用户访问密钥的创建时间列出所有用户访问密钥。如果之前的密钥早于您定义的轮换周期（例如，45 天），其他 Lambda 函数会创建新的访问密钥 ID 和私有访问密钥，并使用亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 和亚马逊简单电子邮件服务 (Amazon SES) Service (Amazon SES) 通知安全管理员。在 AWS Secrets Manager 中为该用户创建密钥，旧的私有访问密钥存储在 Secrets Manager 中，并配置访问旧密钥的权限。为了确保不再使用旧的访问密钥，在非活动期（例如 60 天，也就是我们的示例中密钥轮换后 15 天）后将其禁用。在缓冲期处于非活动状态之后（例如，在我们的示例中为 90 天或密钥轮换后 45 天），旧的访问密钥将从 AWS Secrets Manager 中删除。有关详细的架构和工作流程，请参阅 “架构” 部分。

先决条件和限制

- 使用 AWS Control Tower（版本 3.1 或更高版本）为贵组织构建的着陆区

- Account F@@ actory for Terraform (AFT) 配置了三个账户：
  - 组织管理账户从一个中心位置管理整个组织。
  - AFT 管理账户托管 Terraf orm 管道，并将基础设施部署到部署账户中。
  - 部署账户部署了这个完整的解决方案，并从一个中心位置管理 IAM 密钥。
- Terraform 0.15.0 或更高版本，用于在部署账户中配置基础架构。
- 在亚马逊简单电子邮件服务 (Amazon SES) 中配置的电子邮件地址。
- （推荐）为了增强安全性，请在虚拟私有云 (VPC) 的私有子网（部署账户）内部署此解决方案。在自定义变量时，您可以提供 VPC 和子网的详细信息（参见 Epics 部分中的自定义代码管道参数）。

架构

AFT 存储库

此模式使用 Account Factory for Terraform (AFT) 创建所有必需的 AWS 资源，并使用代码管道将资源部署到部署账户中。代码管道在两个存储库中运行：

- 全局自定义包含 Terraform 代码，该代码将在所有在 AFT 注册的账户中运行。
- 账户自定义包含将在部署账户中运行的 Terraform 代码。

资源详细信息

AWS CodePipeline 任务在部署账户中创建以下资源：

- AWS EventBridge 规则和配置的规则
- `account-inventory` Lambda 函数
- `IAM-access-key-rotation` Lambda 函数
- `Notification` Lambda 函数
- 包含电子邮件模板的亚马逊简单存储服务 (Amazon S3) Service 存储桶
- 必需的 IAM 政策

架构

该图阐释了以下内容：

1. 一条 EventBridge 规则每 24 小时调用一次 `account-inventory` Lambda 函数。
2. `account-inventory`Lambda 函数向 AWS Organizations 查询所有 AWS 账户 IDs、账户名称和账户电子邮件的列表。
3. `account-inventory`Lambda 函数为每个 AWS 账户启动一个 `IAM-access-key-auto-rotation` Lambda 函数，并将元数据传递给该账户进行额外处理。
4. `IAM-access-key-auto-rotation`Lambda 函数使用假设的 IAM 角色来访问 AWS 账户。Lambda 脚本对账户中的所有用户及其 IAM 访问密钥进行审计。
5. 部署 `IAM-access-key-auto-rotation` Lambda 函数时，IAM 密钥轮换阈值（轮换周期）被配置为环境变量。如果修改了轮换周期，则会使用更新的环境变量重新部署 `IAM-access-key-auto-rotation` Lambda 函数。您可以配置参数来设置轮换周期、旧密钥的非活动时间以及将删除旧密钥的非活动缓冲区（请参阅 Epic s 部分中的自定义代码管道参数）。
6. `IAM-access-key-auto-rotation`Lambda 函数根据访问密钥的配置来验证访问密钥的使用年限。如果 IAM 访问密钥的使用期限未超过您定义的轮换期，则 Lambda 函数将不采取任何进一步的操作。
7. 如果 IAM 访问密钥的使用期限已超过您定义的轮换周期，`IAM-access-key-auto-rotation`Lambda 函数将创建一个新密钥并轮换现有密钥。
8. Lambda 函数将旧密钥保存在 Secrets Manager 中，并将权限限制为访问密钥偏离安全标准的用户。Lambda 函数还会创建基于资源的策略，该策略仅允许指定的 IAM 委托人访问和检索密钥。
9. `IAM-access-key-rotation`Lambda 函数调用 Lambda `Notification` 函数。
10. `Notification`Lambda 函数查询 S3 存储桶以获取电子邮件模板，并动态生成包含相关活动元数据的电子邮件。
11. `Notification`Lambda 函数调用 Amazon SES 以采取进一步行动。
12. Amazon SES 会向账户所有者的电子邮件地址发送包含相关信息的电子邮件。

工具

Amazon Web Services

- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。此模式需要 IAM 角色和权限。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Secrets Manager 帮助您将代码中的硬编码凭证（包括密码）替换为对 Secrets Manager 的 API 调用，以便以编程方式检索密钥。

- Amazon Simple Email Service (Amazon SES) 可帮助您使用自己的电子邮件地址和域发送和接收电子邮件。

其他工具

- Terraform 是一款基础设施即代码 (IaC) 工具 HashiCorp ，可帮助您创建和管理云和本地资源。

代码存储库

此模式的说明和代码可在 GitHub IAM 访问密钥轮换存储库中找到。您可以在 AWS Control Tower 中央部署账户中部署代码，以便从中央位置管理密钥轮换。

最佳实践

- 对于 IAM，请参阅 IAM 文档中的安全最佳实践。
- 有关密钥轮换的信息，请参阅 IAM 文档中的访问密钥更新指南。

操作说明

设置源文件

Task	描述	所需技能

克隆存储库。	1. 克隆 I AM 访问密钥轮换 GitHub 存储库：
	`$ git clone https://github.com/aws-samples/centralized-iam-key-management-aws-organizations-terraform.git`
	2. 确认存储库的本地副本包含三个文件夹：
	`$ cd Iam-Access-keys-Rotation`
	`$ ls`
		DevOps 工程师

Task	描述	所需技能

	`org-account-customization`
	`global-account-customization`
	`account-customization`	

配置账户

Task	描述	所需技能

配置引导账户。	作为 AFT 引导过程的一部分，你应该在本地计算机`aft-bootstrap` 上有一个名为的文件夹。
	1. 手动将所有 Terraform 文件从本地 GitHub org-account-customization文件夹复制到您的`aft-bootstrap` 文件夹。
	2. 运行 Terraform 命令在 AWS Control Tower 管理账户中配置全局跨账户角色：
	`$ cd aft-bootstrap`
	`$ terraform init`
	`$ terraform apply –auto-approve`
		DevOps 工程师

配置全局自定义。	作为 AFT 文件夹设置的一部分，您应该在本地计算机`aft-global-customizations` 上有一个名为的文件夹。	DevOps 工程师

Task	描述	所需技能

	1. 手动将所有 Terraform 文件从本地 GitHub global-account-customization文件夹复制到您的`aft-global-customizations/terraform` 文件夹。
	2. 将代码推送到 AWS CodeCommit：
	`$ git add *`
	`$ git commit -m "message"`
	`$ git push`	

Task	描述	所需技能

配置账户自定义。	作为 AFT 文件夹设置的一部分，您的本地计算机`aft-account-customizations` 上有一个名为的文件夹。
	1. 使用您出售的账号创建一个文件夹。
	2. 手动将所有 Terraform 文件从本地 GitHub 帐户自定义文件夹复制到您的文件夹。`aft-account-customizations/<vended account>/terraform`
	3. 将代码推送到 AWS CodeCommit：
	`$ git add *`
	`$ git commit -m "message"`
	`$ git push`
		DevOps 工程师

自定义代码管道的参数

Task	描述	所需技能
为所有账户自定义非 TerraForm 代码管道参数。	在`aft-global-customizations/terraform/` 文件夹`input.auto.tfvars` 中创建一个名为的文件并提供所需的输入数据。有关默认值，请参阅 GitHub 存储库中的文件。	DevOps 工程师

Task	描述	所需技能
为部署账户自定义代码管道参数。	在`aft-account-customizations/<AccountName>/terraform/` 文件夹`input.auto.tfvars` 中创建一个名为的文件，然后将代码推送到 AWS CodeCommit。将代码推送到 AWS CodeCommit 会自动启动代码管道。 根据组织要求指定参数值，包括以下内容（默认值请参阅 Github 存储库中的文件）： • `s3_bucket_name` — 电子邮件模板的唯一存储桶名称。 • `s3_bucket_prefix` — S3 存储桶内的文件夹名称。 • `admin_email_address` — 应接收通知的管理员的电子邮件地址。 • `org_list_account` — 管理账户的账号。 • `rotation_period` — 密钥应在多长时间后从活动状态轮换到非活动状态。 • `inactive_period` — 应在多长天后停用轮换的密钥。此值必须大于的值`rotation_period` 。 • `inactive_buffer` — 密钥轮换和停用之间的宽限期。	DevOps 工程师

Task	描述	所需技能
	• `recovery_grace_period` — 从停用密钥到删除密钥之间的宽限期。 • `dry_run_flag` — 如果您想在不轮换密钥的情况下向管理员发送通知以进行测试，则设置为 true。 • `store_secrets_in_central_account` — 如果要将密钥存储在部署帐户中，则设置为 true。如果该变量设置为 false（默认），则密钥将存储在成员账户中。 • `credential_replication_region` — 您要在其中部署 Lambda 函数的 AWS 区域和电子邮件模板的 S3 存储桶。 • `run_lambda_in_vpc` — 设置为 true 可在 VPC 内运行 Lambda 函数。 • `vpc_id`— 如果您想在 VPC 内运行 Lambda 函数，则为部署账户的 VPC ID。 • `vpc_cidr`— 部署账户的 CIDR 范围。 • `subnet_id` — 部署账户 IDs 的子网。 • `create_smtp_endpoint` — 如果要启用电子邮件端点，则设置为 true。	

验证密钥轮换

Task	描述	所需技能
验证解决方案。	1. 从 AWS 管理控制台登录部署账户。 2. 打开 IAM 控制台，检查用户证书（访问密钥 IDs 和私有密钥）是否按指定轮换。 3. 轮换 IAM 密钥后，请确认以下内容： • 旧值存储在 AWS Secrets Manager 中。 • 机密名称的格式为`Account_<account ID>_User_<username>_AccessKey` 。 • 您在`admin_email_address` 参数中指定的用户会收到一封有关密钥轮换的电子邮件通知。	DevOps 工程师

扩展解决方案

Task	描述	所需技能
自定义电子邮件通知日期。	如果您想在禁用访问密钥之前的特定日期发送电子邮件通知，则可以使用以下更改更新`IAM-access-key-auto-rotation` Lambda 函数： 1. 定义一个名为的变量`notify-period` 。	DevOps 工程师

Task	描述	所需技能
	2. 在停用密钥main.py之前，请在中添加一个if条件： `If (keyage>rotation-period-notify-period){ send_to_notifier(context, aws_account_id, account_name, resource_owner, resource_actions[resource_owner], dryrun, config.emailTemplateAudit) }`	

故障排除

事务	解决方案
account-inventory Lambda 任务在列出账户AccessDenied 时失败。	如果遇到此问题，则必须验证权限： 1. 登录新出售的账户，打开 Amazon CloudWatch 控制台，然后查看 CloudWatch 日志组/aws/lambda/account-inventory-lambda 。 2. 在最新的 CloudWatch 日志中，确定导致访问被拒绝问题的账号。 3. 登录 AWS Control Tower 管理账户并确认该角色allow-list-account 已创建。 4. 如果该角色不存在，请使用命令重新运行 Terraform 代码。terraform apply 5. 选择“可信账户”选项卡，并验证同一个账户是否可信。

相关资源

- Terraform 推荐做法（Terraform 文档）
- IAM 中的安全最佳实践（IAM 文档）
- 密钥轮换最佳实践（IAM 文档）

查看亚马逊 CloudFront 分配的访问日志、HTTPS 和 TLS 版本

由 SaiJeevan Devireddy (AWS) 和 Bijesh Bal (AWS) 创建

摘要

此模式会检查 Amazon CloudFront 分配以确保其使用 HTTPS、使用传输层安全 (TLS) 版本 1.2 或更高版本，并且已启用访问日志记录。 CloudFront 是亚马逊网络服务 (AWS) 提供的一项服务，可加快向用户分发静态和动态网页内容（例如.html、.css、.js 和图像文件）的速度。 CloudFront 通过名为边缘位置的全球数据中心网络交付您的内容。当用户请求与您一起提供的内容时 CloudFront，该请求会被路由到延迟（时间延迟）最低的边缘站点，以便以最佳性能交付内容。

此模式提供了一个 AWS Lambda 函数，该函数在亚马逊 CloudWatch 事件检测到 CloudFront API 调用CreateDistributionCreateDistributionWithTags、或时启动。UpdateDistributionLambda 函数中的自定义逻辑会评估在 AWS 账户中创建或更新的所有 CloudFront 分配。如果检测到以下违规行为，它会使用 Amazon Simple Notification Service (Amazon SNS) 发送违规通知：

- 全局性检查：
  - 自定义证书不使用 TLS 1.2 版
  - 已禁用分发日志记录
- Origin 检查
  - Origin 未配置 TLS 1.2 版
  - 允许在 HTTPS 以外的协议上与源站进行通信
- 行为检查：
  - 允许在 HTTPS 以外的协议上进行默认行为通信
  - 允许在 HTTPS 以外的协议上进行自定义行为通信

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 您希望接收违规通知的电子邮件地址

限制

- 除非对发行版进行了更新，否则此安全控制不会检查现有的 Cloudfront 发行版。
- CloudFront 被视为一项全球服务，与特定的 AWS 区域无关。但是，全球服务的 Amazon CloudWatch 日志和 AWS Cloudtrail API 日志记录发生在美国东部（弗吉尼亚北部）区域 (us-east-1)。因此， CloudFront 必须在中部署和维护此安全控制措施us-east-1。此单一部署可监视所有发行版 CloudFront。请勿在任何其他 AWS 区域部署安全控件。（在其他区域部署将导致无法启动 CloudWatch 事件和 Lambda 函数，并且不会有 SNS 通知。）
- 该解决方案已经过 CloudFront 网络内容分发的广泛测试。它不包含实时消息协议 (RTMP) 流式处理分发。

架构

目标技术堆栈

- Lambda 函数
- SNS 主题
- 亚马逊 EventBridge 规则

目标架构

自动化和扩缩

- 如果您使用的是 AWS Organizations，则可以使用 AWS Cloudformation StackSets 在要监控的多个账户中部署所附的模板。

工具

Amazon Web Services

- AWS CloudFormation — CloudFormation 是一项通过使用基础设施即代码来帮助您建模和设置 AWS 资源的服务。
- Amazon EventBridge — EventBridge 提供来自您自己的应用程序、软件即服务 (SaaS) 应用程序和 AWS 服务的实时数据流，并将这些数据路由到 Lambda 函数等目标。
- AWS Lambda — 您可以运行代码，而无需预置或管理服务器。

- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项高度可扩展的对象存储服务，可用于各种存储解决方案，包括网站、移动应用程序、备份和数据湖。
- Amazon SNS — Amazon SNS 可协调和管理发布者和客户端之间消息的传送或发送，包括 Web 服务器和电子邮件地址。订阅用户接收所有发布至他们所订阅主题的消息，并且一个主题的所有订阅用户收到的消息都相同。

代码

随附代码包括：

- 包含 Lambda 代码的 .zip 文件 (index.py)
- 用于部署 Lambda 代码的 CloudFormation 模板（.yml 文件）

操作说明

上传安全控件

Task	描述	所需技能
为 Lambda 代码创建 S3 存储桶	在 Amazon S3 控制台上，创建一个 S3 存储桶，其具有一个不包含前导斜杠的唯一名称。S3 存储桶名称是全局唯一的，并且命名空间由所有 Amazon Web Services account 共享。您的 S3 存储桶必须位于您计划部署 Lambda 代码的区域。	云架构师
将 Lambda 代码上传至 S3 存储桶。	将附件部分中提供的 Lambda 代码（cloudfront_ssl_log_lambda.zip 文件）上传到您在上一步中创建的 S3 存储桶。	云架构师

部署 CloudFormation 模板

Task	描述	所需技能
部署 CloudFormation 模板。	在 AWS CloudFormation 控制台上，在与 S3 存储桶相同的 AWS 区域中，部署 “附件” 部分中提供的 CloudFormation 模板 (cloudfront-ssl-logging.yml)。	云架构师
指定 S3 存储桶名称。	对于 S3 存储桶参数，请指定您在第一个操作说明中创建的 S3 存储桶的名称。	云架构师
为 Lambda 文件指定 Amazon S3 密钥名称。	对于 S3 密钥参数，请指定 S3 存储桶中 Lambda 代码 .zip 文件的 Amazon S3 位置。不要包含前导斜杠（例如，您可以输入 lambda.zip 或 controls/lambda.zip ）。	云架构师
提供通知电子邮件地址。	在通知电子邮件参数，提供您想要接收违规通知的电子邮件地址。	云架构师
定义日志记录级别。	在 Lambda 日志级别参数，定义您的 Lambda 函数的日志级别。选择以下任一值： • 信息，用于获取有关应用程序进度的详细信息消息。 • 错误，用于获取有关仍允许应用程序继续运行的错误事件的信息。 • 警告，用于获取有关潜在有害情况的信息。	云架构师

确认订阅

Task	描述	所需技能
确认订阅。	成功部署 CloudFormation 模板后，将创建一个新的 SNS 主题，并将订阅消息发送到您提供的电子邮件地址。您必须确认此电子邮件订阅才能接收违规通知。	云架构师

相关资源

- AWS CloudFormation 信息
- 在 AWS CloudFormation 控制台上创建堆栈（CloudFormation 文档）
- CloudFront 记录（CloudFront 文档）
- Amazon S3 信息
- AWS Lambda 信息

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

检查和的安全组入口规则中是否有单主机网络条目 IPv4 IPv6

由 SaiJeevan Devireddy (AWS)、Ganesh Kumar (AWS) 和 John Reynolds (AWS) 创作

摘要

此模式提供了一种安全控制，当 Amazon Web Services (AWS) 资源不符合您的规格时，它会通知您。它提供了一个 AWS Lambda 函数，用于在 Internet 协议版本 4 (IPv4) 和 IPv6 安全组源地址字段中查找单主机网络条目。当 Amazon Events 检测到亚马逊弹性计算云 (亚马 CloudWatch 逊 EC2) AuthorizeSecurityGroupIngressAPI 调用时，就会启动 Lambda 函数。Lambda 函数中的自定义逻辑评估安全组入口规则的 CIDR 块子网掩码。如果确定子网掩码不是 /32 (IPv4) 或 /128 ()，则 Lambda 函数使用亚马逊简单通知服务 (Amazon SNSIPv6) Service 发送违规通知。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 您希望接收违规通知的电子邮件地址

限制

- 此安全监控解决方案是区域性的，必须部署在要监控的每个 AWS 区域中。

架构

目标技术堆栈

- Lambda 函数
- SNS 主题
- 亚马逊 EventBridge 规则

目标架构

自动化和扩缩

- 如果您使用的是 AWS Organizations，则可以使用 AWS Cloudformation StackSets 在要监控的多个账户中部署此模板。

工具

Amazon Web Services

- AWS CloudFormation 是一项通过使用基础设施即代码来帮助您建模和设置 AWS 资源的服务。
- Amazon EventBridge 提供来自您自己的应用程序、软件即服务 (SaaS) 应用程序和 AWS 服务的实时数据流，并将这些数据路由到 Lambda 函数等目标。
- AWS Lambda 支持无需预置或管理服务器即可运行代码。
- Amazon Simple Storage Service (Amazon S3) 是一项高度可扩展的对象存储服务，可用于各种存储解决方案，包括网站、移动应用程序、备份和数据湖。
- Amazon SNS 可协调和管理发布者和客户端之间消息的传送或发送，包括 Web 服务器和电子邮件地址。订阅用户接收所有发布至他们所订阅主题的消息，并且一个主题的所有订阅用户收到的消息都相同。

代码

随附代码包括：

- 包含 Lambda 安全控制代码的.zip 文件 (`index.py`)
- 用于部署 Lambda 代码的 CloudFormation 模板（`security-control.yml`文件）

操作说明

上传安全控件

Task	描述	所需技能
为 Lambda 代码创建 S3 存储桶	在 Amazon S3 控制台上，创建一个 S3 存储桶，其具有一个不包含前导斜杠的唯一名称。S3 存储桶名称是全局唯一的，并且命名空间由所有 Amazon Web Services	云架构师

Task	描述	所需技能
	account 共享。您的 S3 存储桶必须位于要部署安全组入口检查的 AWS 区域。	
将 Lambda 代码上传至 S3 存储桶。	将附件部分中提供的 Lambda 代码（`security-control-lambda.zip` 文件）上传到您在上一步中创建的 S3 存储桶。	云架构师

部署 CloudFormation 模板

Task	描述	所需技能
更改 Python 版本。	下载 “附件” 部分中提供的 CloudFormation 模板 (`security-control.yml`)。打开文件并修改 Python 版本以反映 Lambda 支持的最新版本（当前为 Python 3.9）。 例如，您可以在代码中搜索 `python` ，并将 `Runtime` 的值从更改 `python3.6` 为 `python3.9` 。 有关支持 Python 运行时版本的最新信息，请参阅 AWS Lambda 文档。	云架构师
部署 AWS CloudFormation 模板。	在 AWS CloudFormation 控制台上，在与 S3 存储桶相同的 AWS 区域中，部署 CloudFormation 模板	云架构师

Task	描述	所需技能
	(`security-control.yml`)。	
指定 S3 存储桶名称。	对于 S3 存储桶参数，请指定您在第一个操作说明中创建的 S3 存储桶的名称。	云架构师
为 Lambda 文件指定 Amazon S3 密钥名称。	对于 S3 密钥参数，请指定 S3 存储桶中 Lambda 代码 .zip 文件的 Amazon S3 位置。请勿包含前导斜杠（例如，您可以输入 `lambda.zip` 或 `controls/lambda.zip`）。	云架构师
提供通知电子邮件地址。	在通知电子邮件参数，提供您想要接收违规通知的电子邮件地址。	云架构师
定义日志记录级别。	在 Lambda 日志级别参数，定义您的 Lambda 函数的日志级别。选择以下任一值： • 信息，用于获取有关应用程序进度的详细信息消息。 • 错误，用于获取有关仍允许应用程序继续运行的错误事件的信息。 • 警告，用于获取有关潜在有害情况的信息。	云架构师

确认订阅

Task	描述	所需技能
确认订阅。	成功部署 CloudFormation 模板后，将创建一个新的 SNS 主题，并将订阅消息发送到您提供的电子邮件地址。您必须确认此电子邮件订阅才能接收违规通知。	云架构师

相关资源

- AWS CloudFormation 信息
- 在 AWS CloudFormation 控制台上创建堆栈（AWS CloudFormation 文档）
- 您的 VPC 的安全组（Amazon VPC 文档）
- Amazon S3 信息
- AWS Lambda 信息

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

为企业应用程序选择 Amazon Cognito 身份验证流程

由 Michael Daehnert (AWS) 和 Fabian Jahnke (AWS) 创作

摘要

Amazon Cognito 为网络和移动应用程序提供身份验证、授权和用户管理。它为联合身份的身份验证提供了有益的功能。要使其启动并运行，技术架构师需要决定如何使用这些功能。

Amazon Cognito 支持用于身份验证请求的多个流程。这些流程定义了您的用户如何验证其身份。决定使用哪种身份验证流程取决于您的应用程序的特定要求，并且可能会变得复杂。此模式可帮助您确定哪种身份验证流程最适合您的企业应用程序。它假设你已经掌握了 Amazon Cognito、OpenID Connect (OIDC) 和联合身份验证的基本知识，它会指导你详细了解不同的联合身份验证流程。

该解决方案适用于技术决策者。它可以帮助您了解不同的身份验证流程，并将它们映射到您的应用程序要求。技术主管应收集所需的见解，以启动 Amazon Cognito 集成。由于企业组织主要关注 SAML 联合，因此此模式包括对具有 SAML 联合的 Amazon Cognito 用户池的描述。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 具有 Amazon Cognito 完全访问权限的 AWS 身份和访问管理 (IAM) 角色和权限
- （可选）访问您的身份提供商 (IdP)，例如微软 Entra ID、Active Directory 联合身份验证服务 (AD FS) 或 Okta
- 为您的应用提供高水平的专业知识
- 亚马逊 Cognito、OpenID Connect (OIDC) 和联邦的基础知识

限制

- 这种模式侧重于 Amazon Cognito 用户池和身份提供者。有关 Amazon Cognito 身份池的信息，请参阅其他信息部分。

架构

使用下表来帮助您选择身份验证流程。本节提供了有关每个流程的更多信息。

你需要machine-to-machine身份验证吗？	您的应用程序是否是基于 Web 的应用程序，其前端是在服务器上呈现的？	您的应用程序是单页应用程序 (SPA) 还是基于移动的前端应用程序？	您的应用程序是否需要刷新令牌才能使用 “让我保持登录状态” 功能？	前端是否提供基于浏览器的重定向机制？	推荐的亚马逊 Cognito 流程
是	否	否	否	否	客户凭证流程
否	是	否	是	是	授权码流程
否	否	是	是	是	带有验证密钥的授权码流，用于代码交换 (PKCE)
否	否	否	否	否	资源所有者密码流*

* 只有在绝对必要时才应使用资源所有者密码流程。有关更多信息，请参阅此模式中的 “资源所有者密码流程” 部分。

客户凭证流程

客户凭证流程是 Amazon Cognito 流程中最短的一个。如果系统或服务在没有任何用户交互的情况下相互通信，则应使用它。发出请求的系统使用客户端 ID 和客户端密钥来检索访问令牌。由于这两个系统都无需用户交互即可运行，因此无需额外的同意步骤。

该图阐释了以下内容：

1. 应用程序 1 向 Amazon Cognito 终端节点发送包含客户端 ID 和客户端密钥的身份验证请求，然后它会检索访问令牌。
2. 应用程序 1 在随后对应用程序 2 的每次调用中都使用此访问令牌。
3. 应用程序 2 使用 Amazon Cognito 验证访问令牌。

应使用此流程：

- 用于应用程序之间的通信，无需用户交互

不应使用此流程：

- 适用于任何可能进行用户互动的通信

授权码流程

授权码流程适用于基于 Web 的经典身份验证。在此流程中，后端处理所有的令牌交换和存储。基于浏览器的客户端看不到实际的令牌。此解决方案用于在.NET Core、Jakarta Faces 或 Jakarta 服务器页面 (JSP) 等框架中编写的应用程序。

授权码流程是一个基于重定向的流程。客户端必须能够与 Web 浏览器或类似的客户端进行交互。客户端被重定向到身份验证服务器并使用该服务器进行身份验证。如果客户端成功通过身份验证，则会将其重定向回服务器。

该图阐释了以下内容：

1. 客户端向 Web 服务器发送请求。
2. 网络服务器使用 HTTP 302 状态码将客户端重定向到 Amazon Cognito。客户端会自动跟随此重定向到配置的 IdP 登录。
3. IdP 会检查 IdP 端是否存在现有浏览器会话。如果不存在，则用户会收到一条通过提供用户名和密码进行身份验证的提示。IdP 向 Amazon Cognito 使用 SAML 代币进行回应。
4. Amazon Cognito 使用 JSON 网络令牌 (JWT) 返回成功，特别是代码令牌。用于将代码令牌交换calls /oauth2/token为访问令牌的 Web 服务器。网络服务器将客户端 ID 和客户端密钥发送到 Amazon Cognito 进行验证。
5. 访问令牌用于对其他应用程序的每次后续调用。
6. 其他应用程序使用 Amazon Cognito 验证访问令牌。

应使用此流程：

- 用户是否能够与 Web 浏览器或客户端进行交互。应用程序代码在服务器上运行和呈现，以确保浏览器不会泄露任何秘密。

不应使用此流程：

- 适用于单页应用程序 (SPAs) 或移动应用程序，因为它们是在客户端上呈现的，不应使用客户端密钥。

PKCE 的授权码流程

对于单页应用程序和移动应用程序，应使用带有代码交换校验密钥 (PKCE) 的授权码流。它是隐式流程的继任者，由于使用 PKCE，因此更加安全。PKCE 是 OAuth 2.0 授权码授予公共客户端的扩展。PKCE 可防止兑换拦截的授权代码。

该图阐释了以下内容：

1. 应用程序创建代码验证器和代码质询。这些是定义明确的唯一值，发送到 Amazon Cognito 以供将来参考。
2. 该应用程序调用 Amazon Cognito 的the /oauth2/authorization终端节点。它会自动将用户重定向到配置的 IdP 登录名。
3. IdP 会检查是否存在现有会话。如果不存在，则用户会收到一条通过提供用户名和密码进行身份验证的提示。IdP 向 Amazon Cognito 使用 SAML 代币进行回应。
4. 在 Amazon Cognito 使用代码令牌返回成功后，Web 服务器calls /oauth2/token将使用该代码令牌交换访问令牌。
5. 访问令牌用于对其他应用程序的每次后续调用。
6. 其他应用程序使用 Amazon Cognito 验证访问令牌。

应使用此流程：

- 适用于 SPAs 我们的移动应用程序

不应使用此流程：

- 如果应用程序后端处理身份验证

资源所有者密码流程

资源所有者密码流程适用于没有重定向功能的应用程序。它是通过在您自己的应用程序中创建登录表单来构建的。在 Amazon Cognito 上，登录是通过 CLI 或 SDK 调用进行检查的，而不是依赖重定向流程。由于联合需要基于浏览器的重定向，因此无法在此身份验证流程中进行联合。

该图阐释了以下内容：

1. 用户在应用程序提供的登录表单上输入其凭据。
2. AWS 命令行接口 (AWS CLI) Line Cognito 打电话给 Ama admin-initiated-authzon Cognito。

> **Note**
>
> 或者，您可以使用 AWS SDKs 代替 AWS CLI。

3. 亚马逊 Cognito 会返回访问令牌。
4. 访问令牌用于对其他应用程序的每次后续调用。
5. 其他应用程序使用 Amazon Cognito 验证访问令牌。

应使用此流程：

- 将使用直接身份验证逻辑（例如基本访问身份验证或摘要访问身份验证）的现有客户端迁移到 OAuth 通过将存储的凭据转换为访问令牌来迁移时

不应使用此流程：

- 如果你想使用联合身份
- 如果您的应用程序支持重定向

工具

Amazon Web Services

- Amazon Cognito 为您的 Web 和移动应用程序提供身份验证、授权和用户管理。

其他工具

- JSON 网络令牌 (JWT) 调试器是一个基于 Web 的 JWT 验证工具。

操作说明

评估您的申请

Task	描述	所需技能
定义身份验证要求。	根据您的特定身份验证要求评估您的应用程序。	应用程序开发者、应用程序架构师
使要求与身份验证流程保持一致。	在架构部分，使用决策表和每个流程的说明来选择您的 Amazon Cognito 身份验证流程。	应用程序开发人员、常规 AWS、应用程序架构师

设置 Amazon Cognito 用户池

Task	描述	所需技能
创建用户池。	1. 登录 AWS 管理控制台，然后打开 A mazon Cognito 控制台。 2. 创建一个新的 Cognito 用户池。有关说明，请参阅 Amazon Cognito 用户池。 3. 根据需要更新用户池设置和属性。例如，为用户池设置密码策略。暂时不要创建应用程序客户端。	常规 AWS
（可选）配置身份提供商。	1. 在 Amazon Cognito 用户池中创建 SAML 身份提供商。有关说明，请参阅在用户池中添加和管理 SAML 身份提供商。 2. 配置您的第三方 SAML 身份提供商，使其与 Amazon	常规 AWS，联合管理员

Task	描述	所需技能
	Cognito 用户池的联合身份验证配合使用。有关更多信息，请参阅配置您的第三方 SAML 身份提供商。如果您使用的是 AD FS，请参阅使用 Amazon Cognito 用户池为您的 Web 应用程序构建 AD FS 联合（AWS 博客文章）。	
创建应用程序客户端。	1. 为用户池创建应用程序客户端。有关说明，请参阅创建应用程序客户端。请注意以下几点： • 根据需要更改设置，例如令牌到期时间。 • 如果您的身份验证流程不需要客户端密钥，请清除“生成客户端密钥”复选框。 2. 选择应用程序客户端设置，将其集成更改为用户池登录（用户名和密码）或通过基于 SAML 的 IdP 进行联合登录。 3. 通过根据需要定义 URLs 和定义 OAuth 流程或范围来启用您的 IdP。	常规 AWS

将应用程序与 Amazon Cognito 集成

Task	描述	所需技能
交易所亚马逊 Cognito 集成详情。	根据您的身份验证流程，与应用程序共享 Amazon Cognito 信息，例如用户池 ID 和应用程序客户端 ID。	应用程序开发人员，常规 AWS
实施 Amazon Cognito 身份验证。	这取决于你选择的身份验证流程、你的编程语言和你使用的框架。有关一些入门链接，请参阅 “相关资源” 部分。	应用程序开发人员

相关资源

AWS 文档

- 用户池身份验证流程
- 验证 JSON 网络令牌
- 使用 Amazon Cognito 身份池从 ASP.NET 核心应用程序访问 AWS 服务
- 框架和 SDKs：
  - 亚马逊 Amplify 身份验证
  - 亚马逊 Cognito 身份提供商示例（适用于 Java 的 AWS 开发工具包 2.x 文档）
  - 使用 Amazon Cognito 对用户进行身份验证（适用于.NET 的 AWS 开发工具包文档）

AWS Blog 文章

- 使用 Cookie 授权 @Edge：保护您的亚马逊 CloudFront 内容不被未经身份验证的用户下载
- 使用 Amazon Cognito 用户池为您的 Web 应用程序构建 AD FS 联合

实施伙伴

- 提供身份验证解决方案的 AWS 合作伙伴

其他信息

常见问题解答

为什么不推荐使用隐式流程？

自 OAuth 2.1 框架发布以来，出于安全考虑，隐式流程被标记为已弃用。作为替代方案，请使用 “架构” 部分中描述的 PKCE 授权码流程。

如果 Amazon Cognito 没有提供我需要的某些功能怎么办？

AWS 合作伙伴为身份验证和授权解决方案提供不同的集成。有关更多信息，请参阅 AWS 合作伙伴以获取身份验证解决方案。

那么 Amazon Cognito 身份池流程呢？

Amazon Cognito 用户池和联合身份用于身份验证。Amazon Cognito 身份池用于通过请求临时 AWS 证书来授权 AWS 资源的访问权限。本模式中未讨论身份池的 ID 令牌和访问令牌交换。有关更多信息，请参阅 Amazon Cognito 用户池和身份池有什么区别和常见的 Amazon Cognito 场景。

后续步骤

此模式概述了 Amazon Cognito 身份验证流程。下一步，需要选择应用程序编程语言的详细实现。提供多种语言 SDKs 和框架，您可以将其与 Amazon Cognito 配合使用。有关有用的参考资料，请参阅相关资源部分。

使用 AWS CloudFormation Guard 策略创建 AWS Config 自定义规则

由 Andrew Lok (AWS)、Kailash Havildar (AWS)、Nicole Brown (AWS) 和 Tanya Howell (AWS) 创作

摘要

AWS Config规则可帮助您评估 AWS 资源及其目标配置状态。有两种类型的 AWS Config 规则：托管规则和自定义规则。您可以使用 AWS Lambda 函数或 AWS CloudFormation Guard(GitHub)（一种 policy-as-code语言）创建自定义规则。

使用 Guard 创建的规则比托管规则提供更精细的控制，而且它们通常比完全自定义 Lambda 规则更易于配置。这种方法使工程师和架构师无需了解 Python、NodeJS 或 Java 即可构建规则，这些都是通过 Lambda 部署自定义规则所必需的。

此模式提供了可行的模板、代码示例和部署方法，可帮助您使用 Guard 采用自定义规则。通过使用此模式，管理员可以使用 AWS Config 来构建具有配置项目属性的自定义合规性规则。例如，开发人员可以对 AWS Config 配置项目使用防护策略来持续监控已部署 AWS 和非AWS 资源的状态，检测违反规则的行为，并自动启动补救。

目标

阅读此模式后，您应该能够：

- 了解 Guard 策略代码如何与 AWS Config 服务交互。
- 部署场景 1，这是一项 AWS Config 自定义规则，它使用 Guard 语法来验证加密卷的合规性。此规则验证驱动器是否在使用中，并验证驱动器类型是否为 gp3。
- 部署场景 2，这是一个使用防护语法来验证 Amazon GuardDuty 合规性的 AWS Config 自定义规则。该规则验证 GuardDuty 录像机是否启用了亚马逊简单存储服务 (Amazon S3) 保护和亚马逊弹性 Kubernetes Service (Amazon EK S) 保护。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- AWS Config，在你的 AWS 账户

限制

- Guard 自定义规则只能查询目标配置项 JSON 记录中的键值对

架构

您可以将 Guard 语法作为自定义策略应用于 AWS Config 规则。 AWS Config 捕获每个指定资源的分层 JSON。 AWS Config 配置项的 JSON 包含键值对。这些属性在 Guard 语法中用作分配给其相应值的变量。

以下是对 Guard 语法的解释。使用配置项目 JSON 中的变量，并在前面加上一个字符。%

```
# declare variable
let <variable name> = <'value'>

# create rule and assign condition and policy
    rule <rule name> when
        <CI json key> == <"CI json value"> {
            <top level CI json key>.<next level CI json key> == %<variable name>
        }
```

场景 1：亚马逊 EBS 卷

场景 1 部署了一个 AWS Config 自定义规则，该规则使用 Guard 语法来验证加密卷的合规性。此规则验证驱动器是否在使用中，并验证驱动器类型是否为 gp3。

以下是方案 1 的 AWS Config 配置项目示例。 此配置项中有三个键值对用作警卫策略中的变量：volumestatusvolumeencryptionstatus、和。volumetype此外，该resourceType密钥还可用作 Guard 策略中的过滤器。

```
{
  "version": "1.3",
  "accountId": "111111111111",
  "configurationItemCaptureTime": "2023-01-15T19:04:45.402Z",
  "configurationItemStatus": "ResourceDiscovered",
  "configurationStateId": "4444444444444",
  "configurationItemMD5Hash": "",
  "arn": "arn:aws:ec2:us-west-2:111111111111:volume/vol-222222222222",
  "resourceType": "AWS::EC2::Volume",
  "resourceId": "vol-222222222222",
  "awsRegion": "us-west-2",
  "availabilityZone": "us-west-2b",
```

```
  "resourceCreationTime": "2023-01-15T19:03:22.247Z",
  "tags": {},
  "relatedEvents": [],
  "relationships": [
    {
      "resourceType": "AWS::EC2::Instance",
      "resourceId": "i-33333333333333333",
      "relationshipName": "Is attached to Instance"
    }
  ],
  "configuration": {
    "attachments": [
      {
        "attachTime": "2023-01-15T19:03:22.000Z",
        "device": "/dev/xvda",
        "instanceId": "i-33333333333333333",
        "state": "attached",
        "volumeId": "vol-222222222222",
        "deleteOnTermination": true,
        "associatedResource": null,
        "instanceOwningService": null
      }
    ],
    "availabilityZone": "us-west-2b",
    "createTime": "2023-01-15T19:03:22.247Z",
    "encrypted": false,
    "kmsKeyId": null,
    "outpostArn": null,
    "size": 8,
    "snapshotId": "snap-55555555555555555",
    "state": "in-use",
    "volumeId": "vol-222222222222",
    "iops": 100,
    "tags": [],
    "volumeType": "gp2",
    "fastRestored": null,
    "multiAttachEnabled": false,
    "throughput": null,
    "sseType": null
  },
  "supplementaryConfiguration": {}
}
```

以下是使用 Guard 语法定义场景 1 中的变量和规则的示例。在以下示例中：

- 前三行使用`let`命令定义变量。 它们被分配一个源自配置项目属性的名称和值。
- `compliancecheck`规则块添加了一个 when 条件依赖关系，用于查找`resourceType`匹配的键值对。`AWS::EC2::Volume`如果找到匹配项，则规则会继续执行其余的 JSON 属性，并根据以下三个条件查找匹配项：`stateencrypted`、和`volumeType`。

```
let volumestatus = 'available'
let volumetype = 'gp3'
let volumeencryptionstatus = true

    rule compliancecheck when
        resourceType == "AWS::EC2::Volume" {
            configuration.state == %volumestatus
            configuration.encrypted == %volumeencryptionstatus
            configuration.volumeType == %volumetype
        }
```

有关实现此自定义规则的完整 Guard 自定义策略，请参阅代码存储库中的 awsconfig-guard-cft.yaml 或 awsconfig-guard-tf-ec2vol.json。 GitHub 有关在 Guard 中部署此自定义策略的 HashiCorp Terraform 代码，请参阅代码存储库中的 awsconfig-guard-tf-example.json。

场景 2： GuardDuty 合规性

方案 2 部署了使用防护语法来验证 Amazon GuardDuty 合规性的 AWS Config 自定义规则。此规则验证 GuardDuty 录像机是否启用了 Amazon S3 保护和 Amazon EKS 保护。它还会验证每隔 15 分钟发布一次 GuardDuty 调查结果。此场景可以部署在所有 AWS 账户 组织和组织 AWS 区域 中（ 中 AWS Organizations ）。

以下是方案 2 的 AWS Config 配置项目示例。此配置项中有三个键值对用作警卫策略中的变量：`FindingPublishingFrequency`、`S3Logs`和。`Kubernetes`此外，该`resourceType`密钥还用作策略中的过滤器。

```
{
  "version": "1.3",
  "accountId": "111111111111",
  "configurationItemCaptureTime": "2023-11-27T13:34:28.888Z",
  "configurationItemStatus": "OK",
  "configurationStateId": "7777777777777",
  "configurationItemMD5Hash": "",
```

```
  "arn": "arn:aws:guardduty:us-
west-2:111111111111:detector/66666666666666666666666666666666",
  "resourceType": "AWS::GuardDuty::Detector",
  "resourceId": "66666666666666666666666666666666",
  "resourceName": "66666666666666666666666666666666",
  "awsRegion": "us-west-2",
  "availabilityZone": "Regional",
  "resourceCreationTime": "2020-02-17T02:48:04.511Z",
  "tags": {},
  "relatedEvents": [],
  "relationships": [],
  "configuration": {
    "Enable": true,
    "FindingPublishingFrequency": "FIFTEEN_MINUTES",
    "DataSources": {
      "S3Logs": {
        "Enable": true
      },
      "Kubernetes": {
        "AuditLogs": {
          "Enable": true
        }
      }
    },

    "Id": "66666666666666666666666666666666",
    "Tags": []
  },
  "supplementaryConfiguration": {
    "CreatedAt": "2020-02-17T02:48:04.511Z"
  }
}
```

以下是使用 Guard 语法定义场景 2 中的变量和规则的示例。在以下示例中：

- 前三行使用`let`命令定义变量。 它们被分配一个源自配置项目属性的名称和值。
- `compliancecheck`规则块添加了一个 when 条件依赖关系，用于查找`resourceType`匹配的键值对。`AWS::GuardDuty::Detector`如果找到匹配项，则规则会继续执行其余的 JSON 属性，并根据以下三个条件查找匹配项：`S3Logs.Enable`、`Kubernetes.AuditLogs.Enable`、和`FindingPublishingFrequency`。

```
let s3protection = true
let kubernetesprotection = true
let publishfrequency = 'FIFTEEN_MINUTES'

    rule compliancecheck when
        resourceType == "AWS::GuardDuty::Detector" {
            configuration.DataSources.S3Logs.Enable == %s3protection
            configuration.DataSources.Kubernetes.AuditLogs.Enable ==
 %kubernetesprotection
            configuration.FindingPublishingFrequency == %publishfrequency
        }
```

有关实现此自定义规则的完整 Guard 自定义策略，请参阅 GitHub 代码库中的 awsconfig-guard-cft-gd.yaml。有关在 Guard 中部署此自定义策略的 HashiCorp Terraform 代码，请参阅代码存储库中的 awsconfig-guard-tf-gd.json。

工具

AWS 服务

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在资源的整个生命周期中跨地区对其 AWS 账户 进行管理。
- AWS Config提供了您的资源 AWS 账户 及其配置方式的详细视图。它可以帮助您确定资源之间的关联方式，以及它们的配置如何随时间变化。

其他工具

- HashiCorp Terraform 是一款基础设施即代码 (IaC) 工具，可帮助您使用代码来配置和管理云基础架构和资源。

代码存储库

此模式的代码可在 with AWS CloudFormation Guard存储库中 GitHub AWS Config 找到。此代码存储库包含此模式中描述的两种场景的示例。

操作说明

创建 AWS Config 自定义规则

Task	描述	所需技能
（可选）为规则选择键值对。	如果您要定义自定义 Guard 策略，请完成以下步骤。如果您正在使用方案 1 或 2 的示例策略之一，请跳过这些步骤。 1. 登录 AWS Management Console 并打开AWS Config控制台。 2. 在左侧导航栏中，选择 “资源”。 3. 在资源清单中，选择要为其创建 AWS Config 自定义规则的资源类型。 4. 请选择查看详细信息。 5. 选择查看配置项目 (JSON)。此部分扩展为以 JSON 格式显示配置项目。 6. 确定要为其构建 AWS Config 自定义规则的键值对。	AWS 管理员、安全工程师
创建自定义规则。	使用您之前确定的键值对或使用提供的示例警卫策略之一，按照创建 AWS Config 自定义策略规则中的说明创建自定义规则。	AWS 管理员、安全工程师
验证自定义规则。	执行以下任一操作来验证自定义 Guard 规则：	AWS 管理员、安全工程师

Task	描述	所需技能
	• 在 AWS Command Line Interface (AWS CLI) 中输入以下命令。 `cfn-guard validate -r guard-s3.guard -d s3bucket-prod-pass.json` • 按照 Detect ive 模式下的“使用 AWS Config 规则评估您的资源”中的说明在中部署规则 AWS Config。确认 Guard 语法与目标账户或文件中的相应资源正确匹配。	

故障排除

事务	解决方案

在之外测试警卫策略 AWS Config	单元测试可以在本地设备上完成，也可以在集成开发环境 (IDE)（例如 AWS Cloud9 IDE）中完成。要执行单元测试，请执行以下操作：
	
	1. 安装 C AWS CloudFormation Guard LI 及其依赖项。
	2. 将 JSON 格式的 CI 示例作为.json 文件保存到您的工作站。
	3. 将 GuardDuty 策略作为.guard 文件保存到您的工作站。
	4. 在 Guard CLI 中，输入以下命令以使用防护策略验证示例 JSON 文件。
	
	`cfn-guard validate \`
	`-r guard-s3.guard \`

事务	解决方案
	`-d s3bucket-prod-pass.json`
调试 AWS Config 自定义规则	在您的警卫策略中，将该`EnableDebugLogDelivery` 值更改为`true`。默认值为 `false`。日志消息存储在 Amazon 中 CloudWatch。

相关资源

AWS 文档

- 创建 AWS Config 自定义策略规则（AWS Config 文档）
- 编写 AWS CloudFormation Guard 规则（警卫文档）

AWS 博客文章和研讨会

- AWS CloudFormation Guard 2.0 简介（AWS 博客文章）

其他资源

- AWS CloudFormation Guard (GitHub)
- AWS CloudFormation Guard CLI 文档 (GitHub)

创建一份包含多个 Prowler 安全调查结果的合并报告 AWS 账户

由 Mike Virgilio (AWS)、Andrea Di Fabio (AWS) 和 Jay Durga (AWS) 创作

摘要

Prowler (GitHub) 是一个开源命令行工具，可以帮助您评估、审计和监控您的 Amazon Web Services (AWS) 账户是否符合安全最佳实践。在这种模式下，您可以将 Prowler 部署到组织 AWS 账户 中由管理的集中管理中 AWS Organizations，然后使用 Prowler 对组织中的所有帐户进行安全评估。

虽然部署和使用 Prowler 进行评测的方法有很多，但该解决方案旨在实现快速部署、全面分析组织中的所有账户或已定义的目标客户，以及安全调查发现的可访问报告。按此解决方案，当 Prowler 完成对组织中所有账户的安全评测时，它会合并结果。它还会过滤掉任何预期错误消息，例如与限制 Prowler 无法扫描通过 AWS Control Tower配置的账户中的 Amazon Simple Storage Service (Amazon S3) 存储桶相关的错误。筛选后的合并结果将在此操作说明中包含的 Microsoft Excel 模板中报告。您可以使用此报告确定组织中安全控件的潜在改进。

该解决方案的设计考虑了以下内容：

- 这些 AWS CloudFormation 模板减少了在这种模式下部署 AWS 资源所需的工作量。
- 您可以在部署时调整 CloudFormation 模板和 prowler_scan.sh 脚本中的参数，以便为您的环境自定义模板。
- 通过并行处理 AWS 账户、汇总结果、合并报告和建议的补救措施以及自动生成的可视化效果来优化 Prowler 的评估和报告速度。
- 用户不需要监控扫描进度。评测完成后，Amazon Simple Notification Service (Amazon SNS) 主题将通知用户，以便他们可检索报告。
- 报告模板可帮助您仅阅读和评测整个组织相关结果。

先决条件和限制

先决条件

- AWS 账户 用于托管安全服务和工具，作为中组织的成员帐户进行管理 AWS Organizations。在这种模式中，此账户被称为 安全账户。
- 在安全账户中，必须具有具有出站互联网访问权限的私有子网。有关说明，请参阅 Amazon Virtual Private Cloud (Amazon VPC) 文档中的私有子网和 NAT 中的 服务器 VPC。您可以使用在公有子网中配置的 NAT 网关建立互联网接入。

- 对 AWS Organizations 管理账户或拥有委托管理员权限的账户的访问权限 CloudFormation。有关说明，请参阅 CloudFormation 文档中的注册委托管理员。
- 在 AWS Organizations 和之间启用可信访问 CloudFormation。有关说明，请参阅 CloudFormation 文档 AWS Organizations中的使用启用可信访问。

限制

- 目标 AWS 账户 必须作为一个组织在中进行管理 AWS Organizations。如果您不使用 AWS Organizations，则可以更新适用于您的环境的 IAM-ProwlerExecRole .yaml CloudFormation 模板和 prowler_scan.sh 脚本。相反，您可以提供要在其中运行脚本的 AWS 账户 IDs 和区域的列表。
- 该 CloudFormation 模板旨在将亚马逊弹性计算云 (Amazon EC2) 实例部署到具有出站互联网访问权限的私有子网中。 AWS Systems Manager 代理（SSM 代理）需要出站访问权限才能访问 AWS Systems Manager 服务端点，并且您需要出站访问权限才能克隆代码存储库和安装依赖项。如果要使用公有子网，则必须修改 p rowler-resources.yaml 模板以将弹性 IP 地址与实例相关联。 EC2

产品版本

- Prowler 版本 4.0 或更高版本

架构

此图显示以下流程：

1. 使用会话管理器（一种功能） AWS Systems Manager，用户对 EC2 实例进行身份验证并运行 prowler_scan.sh 脚本。此 Shell 脚本执行步骤 2-8。
2. 该 EC2 实例担任 `ProwlerEC2Role` IAM 角色，该角色授予访问 S3 存储桶和在组织中其他账户中担任 `ProwlerExecRole` IAM 角色的权限。
3. 该 EC2 实例在组织的管理账户中担任 `ProwlerExecRole` IAM 角色，并生成组织中的账户列表。
4. 该 EC2 实例在组织的成员账户（架构图中称为工作负载账户）中担任 `ProwlerExecRole` IAM 角色，并在每个账户中执行安全评估。调查结果以 CSV 和 HTML 文件的形式存储在 EC2 实例上。

> **Note**
>
> HTML 文件是 Prowler 评估的结果。由于 HTML 的性质，它们不在此模式中进行串联、处理或直接使用。但是，这些可能对个人账户报告审核很有用。

5. 该 EC2 实例处理所有 CSV 文件以删除已知的预期错误，并将剩余的发现结果合并到一个 CSV 文件中。
6. 该 EC2 实例将个人账户结果和汇总结果打包到一个 zip 文件中。
7. EC2 实例将 zip 文件上传到 S3 存储桶。
8. EventBridge 规则会检测文件上传，并使用 Amazon SNS 主题向用户发送电子邮件，通知他们评估已完成。
9. 用户从 S3 存储桶下载 zip 文件。用户将结果导入 Excel 模板，并查看结果。

工具

AWS 服务

- 亚马逊弹性计算云 (Amazon EC2) 在中提供可扩展的计算容量 AWS Cloud。您可以根据需要启动任意数量的虚拟服务器，并快速扩展或缩减它们。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如， AWS Lambda 函数、使用 API 目的地的 HTTP 调用端点或其他 AWS 账户目的地的事件总线。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Organizations是一项账户管理服务，可帮助您将多个账户整合 AWS 账户 到一个由您创建和集中管理的组织中。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Systems Manager 可帮助您管理在 AWS Cloud中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。此模式使用会话管理器，这是 Systems Manager 的一项功能。

其他工具

- Prowler 是一款开源命令行工具，可帮助您评估、审计和监控您的账户是否符合 AWS 安全最佳实践以及其他安全框架和标准。

代码存储库

此模式的代码可通过 Prowler 存储库在 GitHub 多账户安全评估中找到。代码存储库包含以下文件：

- prowler_scan.sh — 此 bash 脚本用于并行启动多个 AWS 账户 Prowler 安全评估。按照 Prowler-Resources.yaml 中的定义 CloudFormationtemplate，此脚本会自动部署到实例上的文件夹。`usr/local/prowler` EC2
- Prowler-resources.yaml — 您可以使用此 CloudFormation 模板在组织的安全账户中创建堆栈。此模板部署了该账户所需所有资源，以支持此解决方案。此堆栈必须在 IAM-ProwlerExecRole .yaml 模板之前部署。我们不建议您将这些资源部署至托管关键生产工作负载的账户中。

> **Note**
>
> 如果此堆栈被删除并重新部署，则必须重新构建`ProwlerExecRole`堆栈集，以便重建 IAM 角色之间的跨账户依赖关系。

- IAM-ProwlerExecRole .yaml — 您可以使用此 CloudFormation 模板创建堆栈集，用于在组织中的所有账户（包括管理账户）中部署 `ProwlerExecRole` IAM 角色。
- prowler-report-template.xlsm — 您可以使用此 Excel 模板来处理 Prowler 的调查结果。报告中的数据透视表提供搜索功能、图表与综合调查发现。

操作说明

准备部署

Task	描述	所需技能
克隆代码存储库。	1. 在命令行界面中，将工作目录更改为要存储示例文件的位置。 2. 输入以下命令：	AWS DevOps

Task	描述	所需技能
	`git clone https://github.com/aws-samples/multi-account-security-assessment-via-prowler.git`	
审核模板。	1. 在克隆的存储库中，打开 Prowler-resources.yaml 和 IAM-.yaml 文件。ProwlerExecRole 2. 查看由这些模板创建的资源，并按您的环境需要调整模板。有关更多信息，请参阅 CloudFormation 文档中的使用模板。 3. 保存并关闭 Prowler-resources.yaml 和 IAM-.yaml 文件。 ProwlerExecRole	AWS DevOps

创建堆 CloudFormation 栈

Task	描述	所需技能
在安全账户中预置资源。	使用 prowler-resources.yaml 模板，您可以创建一个 CloudFormation 堆栈，用于在安全账户中部署所有必需的资源。有关说明，请参阅 CloudFormation 文档中的创建堆栈。部署此模板时应注意以下几点：	AWS DevOps

Task	描述	所需技能
	1. 在指定模板页面，选择模板已就绪，然后上传 prowler-resources.yaml 文件。 2. 在指定堆栈详细信息页面上，在堆栈名称框里输入 `Prowler-Resources` 。 3. 在参数部分中，输入以下内容： • VPCId - 在账户中选择一个 VPC。 • SubnetId -- 选择可以访问 Internet 的私有子网。 注意：如果您选择公有子网，则不会为该 EC2 实例分配公有 IP 地址，因为默认情况下， CloudFormation 模板不会预配置和附加弹性 IP 地址。 • `InstanceType` — 根据并行评测的数量选择实例大小： • 对于 10，请选择 `r6i.large` 。 • 对于 12，请选择 `r6i.xlarge` 。 • 对于 14-18，请选择 `r6i.2xlarge` 。 • `InstanceImageId` — 保留 Amazon Linux 的默认设置。	

Task	描述	所需技能
	• KeyPairName - 如果您使用 SSH 进行访问，请指定现有密钥对的名称。 • PermittedSSHInbound - 如果您使用 SSH 进行访问，请指定允许的 CIDR 块。如果不使用 SSH，请保留默认值 127.0.0.1 。 • BucketName – 默认值为 prowler-output-<accountID>-<region> 。您可以根据需要进行修改。如果指定自定义值，则账户 ID 和区域会自动追加到指定值。 • EmailAddress — 指定电子邮件地址，以用于 Prowler 完成评测，以及 .zip 文件上传至 S3 存储桶时收到 Amazon SNS 通知。 注意：必须在 Prowler 完成评测之前确认 SNS 订阅配置，否则将无法发送通知。 • IAMProwlerEC2Role - 保留默认值，除非您的命名约定要求此 IAM 角色使用不同的名称。 • IAMProwlerExecRole — 除非在部署 IAM-	

Task	描述	所需技能
	ProwlerExecRole .yaml 文件时使用其他名称，否则请保留默认值。 • `Parallelism` — 指定要执行的并行评测的数量。确保 `InstanceType` 参数中的值支持此数量的并行评测。 • `FindingOutput` — 如果要排除及格结果，请选择 `FailOnly`。这大大缩小了输出大小，并将重点放在可能需要解决的检查。如果要纳入合格结果，请选择 `FailAndPass`。 4. 在 “查看” 页面上，选择 “以下资源需要能力：[AWS::IAM::Role]”，然后选择 “创建堆栈”。 5. 成功创建堆栈后，在 CloudFormation 控制台的输出选项卡上，复制 `ProwlerEC2Role` Amazon 资源名称 (ARN)。您稍后在部署 IAM-ProwlerExecRole .y aml 文件时会使用此 ARN。	

Task	描述	所需技能
在成员账户中预置 IAM 角色。	在 AWS Organizations 管理账户或具有委托管理员权限的账户中 CloudFormation，使用 IAM-ProwlerExecRole .yaml 模板创建 CloudFormation 堆栈集。堆栈集在组织中的所有成员账户中部署 `ProwlerExecRole` IAM 角色。有关说明，请参阅 CloudFormation 文档中的使用服务管理权限创建堆栈集。部署此模板时应注意以下几点： 1. 在 “准备模板” 下，选择 “模板已准备就绪”，然后上传 IAM-ProwlerExecRole .yaml 文件。 2. 在 “指定 StackSet 详细信息” 页面上，为堆栈集命名`IAM-ProwlerExecRole` 。 3. 在参数部分中，输入以下内容： • `AuthorizedARN` - 输入您在创建 `Prowler-Resources` 堆栈时复制的 `ProwlerEC2Role` ARN。 • `ProwlerExecRoleName` — 除非在部署 Prowler-Resources.yaml 文件时使用了其他名称，否则保留默认 `ProwlerExecRole` 值。	AWS DevOps

Task	描述	所需技能
	4. 在权限下方，选择服务托管权限。 5. 在设置部署选项页面，在部署目标下，选择部署到组织并接受所有默认选项。 注意：如果要将堆栈同时部署至所有成员账户，请将最大并发账户数和容错率设置为较高的值，例如 100。 6. 在 “部署区域” 下，选择 Prowler EC2 实例的部署 AWS 区域 位置。由于 IAM 资源是全局资源而不是区域资源，因此这会在所有活动区域中部署 IAM 角色。 7. 在 “审阅” 页面上，选择 “我确认 AWS CloudFormation 可能会使用自定义名称创建 IAM 资源”，然后选择 “创建” StackSet。 8. 监控堆栈实例选项卡（用于单个账户状态）和操作选项卡（用于整体状态）以确定部署何时完成。	

Task	描述	所需技能
在管理账户中预置 IAM 角色。	使用 IAM-ProwlerExecRole .yaml 模板，您可以创建一个 CloudFormation 堆栈，用于在组织的管理账户中部署 `ProwlerExecRole` IAM 角色。您之前创建的堆栈集不在管理账户中部署 IAM 角色。有关说明，请参阅 CloudFormation 文档中的创建堆栈。部署此模板时应注意以下几点： 1. 在 “指定模板” 页面上，选择 “模板已准备就绪”，然后上传 IAM-ProwlerExecRole .yaml 文件。 2. 在指定堆栈详细信息页面上的堆栈名称框中，输入 `IAM-ProwlerExecRole` 。 3. 在参数部分中，输入以下内容： • `AuthorizedARN` - 输入您在创建 `Prowler-Resources` 堆栈时复制的 `ProwlerEC2Role` ARN。 • `ProwlerExecRoleName` — 除非在部署 Prowler-Resources.yaml 文件时使用了其他名称，否则保留默认 `ProwlerExecRole` 值。 4. 在 “查看” 页面上，选择 “以下资源需要能力：	AWS DevOps

Task	描述	所需技能
	[AWS::IAM::Role]”，然后选择 “创建堆栈”。	

执行 Prowler 安全评测

Task	描述	所需技能
运行扫描。	1. 登录到组织中的安全账户。 2. 使用会话管理器连接到您之前 EC2 配置的 Prowler 实例。有关说明，请参阅使用会话管理器连接至您的 Linux 实例。如果您无法连接，请参阅此模式的故障排除部分。 3. 导航到 `usr/local/prowler`，然后打开 prowler_scan.sh 文件。 4. 根据您的环境需要，查看和修改此脚本中的可调参数和变量。有关自定义选项的更多信息，请参阅脚本开头的注释。 例如，您可以修改脚本以指定要扫描的 AWS 账户 IDs 或，也可以引用包含这些参数的外部文件 AWS 区域 ，而不是从管理账户获取组织中所有成员账户的列表。 5. 保存和关闭 prowler_scan.sh 文件。 6. 输入以下命令。这会运行 prowler_scan.sh 脚本。	AWS 管理员

Task	描述	所需技能
	``` sudo -i screen cd /usr/local/ prowler ./prowler_scan.sh ``` 请注意以下几点： • 屏幕命令允许脚本在连接超时或失去控制台访问权限时继续运行。 • 扫描开始后，您可以通过 Ctrl+A D 强制分离屏幕。屏幕已分离，您可以关闭实例连接并允许评测继续进行。 • 要恢复已分离的会话，请连接至实例，输入 `sudo -i`，然后输入 `screen -r`。 • 若要监控个人账户评测的进度，您可以导航至 `usr/local/prowler` 目录并输入命令 `tail -f output/stdout-<account-id>` 。 7. 等待 Prowler 完成所有账户扫描。该脚本可同时评测多个账户。当所有账户的评测都完成后，如果您在部署 Prowler-Resources.yaml 文件时指定了电子邮件地址，则会收到通知。	

Task	描述	所需技能
检索 Prowler 调查发现。	1. 从 `prowler-output-<accountID>-<region>` 存储桶下载 `prowler-output-<assessDate>.zip` 文件。有关说明，请参阅 Amazon S3 文档中的下载对象。 2. 删除存储桶中的所有对象，包含您下载的文件。这是成本优化的最佳实践，也是为了确保您可以随时删除`Prowler-Resources` CloudFormation 堆栈。有关说明，请参阅 Amazon S3 文档中的删除对象。	常规 AWS
停止实 EC2 例。	为防止在实例空闲时计费，请停止运行 Prowler 的 EC2 实例。有关说明，请参阅 Amazon EC2 文档中的停止和启动您的实例。	AWS DevOps

创建调查发现报告

Task	描述	所需技能
导入调查发现。	1. 在 Excel 中，打开 prowler-report-template.xlsx 文件，然后选择 Pro wler CSV 工作表。 2. 删除所有示例数据，其中包括标题行。如果系统询问您是否要删除与数据相关的查询，请选择否。删除查询可	常规 AWS

Task	描述	所需技能
	能会影响 Excel 模板中数据透视表的功能。 3. 提取从 S3 存储桶所下载的 zip 文件的内容。 4. 在 Excel 中，打开 prowler-fullorgresults-accessdeniedfiltered.txt。我们建议您使用此文件，因为已经删除了最常见的、不可操作的错误，例如与尝试扫描资源相关的`Access Denied`错误。AWS Control Tower 如果您想要未经熟悉爱你的结果，请改为打开 prowler-fullorgresults.txt 文件。 5. 选择 A 列。 6. 如果使用 Windows 系统，请输入 Ctrl+C；或如果您使用MacOS，请输入 Cmd+C，这将所有数据复制到剪贴板。 7. 在 Excel 报告模板的 Prowler CSV 工作表中，选择单元格 A1。 8. 如果使用 Windows 系统，请输入 Ctrl+V；或如果您使用MacOS，请输入 Cmd+V，这将调查发现复制到剪贴板。 9. 确认选中所有包含粘贴数据的单元格。如否，请选择 A 列。	

Task	描述	所需技能
	10.在数据选项卡，选择文本到列。 11.在向导中，执行以下操作： • 在步骤 1 中，选择分隔。 • 在步骤 2 中，对于分隔符，选择分号。在数据预览窗格，确认数据已分成列。 • 在步骤 3，选择完成。 12.确认文本数据跨多列分隔。 13.以新名称保存 Excel 报告。 14.搜索并删除调查发现中的任何 `Access Denied` 错误。有关如何以编程方式删除这些错误的说明，请参阅其他信息部分中的以编程方式删除错误。	

Task	描述	所需技能
完成报告。	1. 选择调查发现工作表，然后选择单元格 A17。此单元格是数据透视表标头 2. 在功能区的 “PivotTable 工具” 下，选择 “分析”，然后在 “刷新” 下选择 “全部刷新”。这将使用新数据集更新数据透视表。 3. 默认情况下，Excel 无法正确显示 AWS 账户 数字。若要修复数字格式，请执行以下操作： • 在调查发现工作表上，打开 A 列的上下文（右键单击）菜单，然后选择设置单元格格式。 • 选择数字，然后在小数位中输入 0。 • 选择确定。 注意：如果 AWS 账户 数字以一个或多个零开头，Excel 会自动删除零。如果您在报告中看到的账号少于 12 位数，则在号码开头以零补充缺失位数。 4. （可选）您可以折叠字段，以使结果更易于阅读。执行以下操作： • 在调查发现工作表，如果将光标移到第 18 行和第 19 行之间的行（关键标题和第一个查找结果之间的	常规 AWS

Task	描述	所需技能
	间距），则光标图标将变为指向下方的小箭头。 • 单击选择所有调查发现字段。 • 打开上下文（右键单击）菜单，找到展开/折叠，然后选择折叠。 5. 有关评测的详细信息，请查看调查发现、严重性和通过失败工作表。	

（可选）更新 Prowler 或代码存储库中的资源

Task	描述	所需技能

更新 Prowler。	如果要将 Prowler 更新至最新版本，请执行以下操作：
	1. 使用会话管理器连接到 Prowler 的 EC2 实例。有关说明，请参阅使用会话管理器连接至您的 Linux 实例。
	2. 输入以下命令。
	`sudo -i`
	`pip3 install --upgrade prowler`
		常规 AWS

更新 prowler_scan.sh 脚本。	如果要将 prowler_scan.sh 脚本更新至存储库中的最新版本，请执行以下操作： 1. 使用会话管理器连接到 Prowler 的 EC2 实例。有关	常规 AWS

Task	描述	所需技能

	说明，请参阅使用会话管理器连接至您的 Linux 实例。
	2. 输入以下命令。
	`sudo -i`
	3. 导航至 Prowler 脚本目录。
	`cd /usr/local/prowler`
	4. 输入以下命令，以存储本地脚本，以便您可以将自定义更改合并到最新版本中。
	`git stash`
	5. 输入以下命令，下载最新版本的脚本。
	`git pull`
	6. 输入以下命令，将自定义脚本与最新版本的脚本合并。
	`git stash pop`
	Note
	您可能会收到与任何不在 GitHub 存储库中的本地生成的文件相关的警告，例如查找报告。只要 prowler_scan.sh 显示本地隐藏的更改已	

Task	描述	所需技能
	合并，就可以忽略这些内容。	

(可选)清除

Task	描述	所需技能
删除所有已部署资源。	您可以将资源保留在账户中。如果您在 EC2 实例未使用时将其关闭，并保持 S3 存储桶为空，则可以降低维护资源以备将来扫描的成本。 如果要取消预配所有资源，请执行以下操作： 1. 删除管理账户中预置的 `IAM-ProwlerExecRole` 堆栈。有关说明，请参阅 CloudFormation 文档中的删除堆栈。 2. 删除在组织的管理账户或委托管理员账户中预置的 `IAM-ProwlerExecRole` 堆栈集。有关说明，请参阅 CloudFormation 文档中的删除堆栈集。 3. 删除 `prowler-output` S3 存储桶中的所有对象。有关说明，请参阅 Amazon S3 文档中的删除对象。 4. 删除安全账户中预置的 `Prowler-Resources` 堆栈。有关说明，请参阅	AWS DevOps

Task	描述	所需技能
	CloudFormation 文档中的删除堆栈。	

故障排除

事务	解决方案
无法使用会话管理器连接到 EC2 实例。	SSM 代理必须能够与 Systems Manager 端点通信。执行以下操作： 1. 验证部署 EC2 实例的子网是否可以访问互联网。 2. 重启实 EC2 例。
部署堆栈集时，CloudFormation 控制台会提示您这样做`Enable trusted access with AWS Organizations to use service-managed permissions` 。	这表示未在 AWS Organizations 和之间启用可信访问 CloudFormation。部署服务托管堆栈集需要可信访问权限。选择该按钮以启用受信任的访问。有关更多信息，请参阅 CloudFormation 文档中的启用可信访问。

相关资源

AWS 文档

- 在 AWS（AWS 规范性指南）上实施安全控制

其他资源

- Prowler () GitHub

其他信息

以编程方式删除错误

如果结果包含 `Access Denied` 错误，则应将其从调查发现中删除。这些错误通常是由于外部影响权限导致 Prowler 无法评测特定资源造成的。例如，在查看通过配置的 S3 存储桶时，某些检查会失败。AWS Control Tower您可通过编程方式提取这些结果，并将筛选后的结果另存为新文件。

以下命令删除包含单个文本字符串（一种模式）行，然后将结果输出至新文件中。

- 适用于 Linux 或 macOS (Grep)

```
grep -v -i "Access Denied getting bucket" myoutput.csv > myoutput_modified.csv
```

- 适用于 Windows (PowerShell)

```
Select-String -Path myoutput.csv -Pattern 'Access Denied getting bucket' -NotMatch >
 myoutput_modified.csv
```

以下命令删除与多个文本字符串匹配的行，然后将结果输出至新文件。

- 对于 Linux 或 macOS（在字符串之间使用转义管道）

```
grep -v -i 'Access Denied getting bucket\|Access Denied Trying to Get' myoutput.csv >
 myoutput_modified.csv
```

- 适用于 Windows（在字符串之间使用逗号）

```
Select-String -Path myoutput.csv -Pattern 'Access Denied getting bucket', 'Access
 Denied Trying to Get' -NotMatch > myoutput_modified.csv
```

报告示例

下图是 Prowler 合并调查发现报告中的调查发现工作表示例。

下图是 Prowler 合并调查发现报告中的 Pass Fail 工作表的示例。（默认情况下，输出中不包括通过结果。）

下图是 Prowler 合并调查发现报告中严重性工作表的示例。

使用和删除未使用的 Amazon EBS 卷 AWS ConfigAWS Systems Manager

由 Sankar Sangubotla (AWS) 创建

摘要

Amazon Elastic Block Store (Amazon EBS) 卷的生命周期通常独立于其所连接的亚马逊弹性计算云 (Amazon EC2) 实例的生命周期。除非您在启动时选择 “终止时删除” 选项，否则终止 Amazon EC2 实例会分离 Amazon EBS 卷，但不会将其删除。尤其是在通常启动和终止 Amazon EC2 实例的开发和测试环境中，这可能会导致大量未使用的 Amazon EBS 卷。无论是否正在使用 Amazon EBS 交易量 AWS 账户，都会在您的账户中累积费用。删除这些卷可以帮助您优化成本 AWS 账户。此外，删除未使用的 Amazon EBS 卷是一种最佳安全实践，可防止访问这些卷中任何未使用的、可能很敏感的数据。

AWS Config 可以帮助您手动或自动修复不合规的资源。此模式描述了如何配置 AWS Config 规则和自动修复操作，以删除账户中未使用的 Amazon EBS 卷。补救操作是自动化的预定义运行手册，其 AWS Systems Manager功能为。您可将运行手册配置为在删除卷之前创建快照。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- AWS Identity and Access Management (IAM) 运行自动化`AWSConfigRemediation-DeleteUnusedEBSVolume`运行手册的权限，这是 Systems Manager 的一项功能。有关更多信息，请参阅中的 “必需 IAM 权限” AWSConfigRemediation-DeleteUnusedEBSVolume。
- 一个或多个未使用 Amazon EBS 卷。

限制

- 未使用的 Amazon EBS 卷必须处于 `available` 状态。

架构

目标架构

1. 该 AWS Config 规则会评估 Amazon EBS 卷。
2. 该规则返回合规与不合规资源的列表。处于该`available`状态的 Amazon EBS 卷（即未使用的卷）被确定为不合规。
3. AWS Config 自动启动自动化运行手册。
4. 如已配置，Systems Manager 会在删除未使用的卷前创建快照。
5. Systems Manager 会删除未使用的亚马逊 EBS 卷。

自动化和扩缩

您可以将此解决方案应用至组织中的所有账户。有关更多信息，请参阅 AWS Config 文档中的管理组织中所有账户的规则。

工具

- AWS Config提供了您的资源 AWS 账户 及其配置方式的详细视图。它可以帮助您确定资源之间的关联方式，以及它们的配置如何随时间变化。
- AWS Systems Manager 可帮助您管理在 AWS Cloud中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。
- AWS Systems Manager 自动化简化了许多人的常见维护、部署和补救任务 AWS 服务。

操作说明

配置 AWS Config 规则

Task	描述	所需技能
为自动化运行手册创建角色。	创建名为 `AssumeRole` 的角色。Systems Manager Automation 使用此角色运行手册。有关说明，请参阅 Systems Manager 文档中的为自动化配置服务角色（担任角色）访问权限。	AWS 系统管理员

Task	描述	所需技能
打开 AWS Config 录音机。	按照 AWS Config 文档中AWS Config 使用控制台设置中的说明进行操作，确保控制台 AWS Config 正在运行并且已配置为记录 Amazon EBS 卷。	AWS 系统管理员
运行规则。	1. 按照 AWS Config 文档中评估资源中的说明运行ec2-volume-inuse-check 规则。等待评估完成。 2. 在规则页面，选择 ec2-volume-inuse-check 规则，然后在范围内资源中选择不合规。 3. 确认评估结果中包含一个或多个未使用 Amazon EBS 卷。	AWS 系统管理员

配置对未使用的 Amazon EBS 卷自动修复

Task	描述	所需技能
添加自动修复操作。	1. 在规则页面上，选择ec2-volume-inuse-check 规则。 2. 按照 AWS Config 文档中设置自动修复中的说明进行操作。请注意以下几点： 3. 在修正操作详细信息部分，选择 AWSConfig Remediation-DeleteUnusedEBSVolume 。	AWS 系统管理员

Task	描述	所需技能

	• 选择资源 ID 参数，然后在列表中选择VolumeId。在运行时，此参数将替换为不合规的 Amazon EBS 卷的 ID。
	• 在参数部分，为以下参数提供值：
	  • `CreateSnapshot` —（可选）如果设置为`true`，则自动化会在删除 Amazon EBS 卷之前创建 Amazon EBS 卷的快照。
	  • `AutomationAssumeRole` – 输入您之前创建的 `AssumeRole` 服务角色的 Amazon 资源名称（ARN）。	

测试 AWS Config 规则的自动修复。	1. 在 AWS Config 控制台的 “规则” 页面上，选择`ec2-volume-inuse-check` 规则。 2. 在操作菜单中，选择重新评估。 3. 允许规则评估不合规资源，然后确认未使用的 Amazon EBS 卷已被删除。	AWS 系统管理员

故障排除

事务	解决方案
AWS Config 不能准确反映资源状态。	有时，AWS Config 不会更新资源的状态。关闭录音机，然后在 AWS Config “设置” 页面上将其重新打开。记录器捕获资源状态。对于新创建或删除的资源，记录器可能需要一些时间才可反映当前状态。有关 Amazon EBS 卷状态的更多信息，请参阅 Amazon EBS 文档中的卷状态。

相关资源

- AWSConfigRemediation-DeleteUnusedEBSVolume 运行手册
- ec2-volume-inuse-check 规则
- 使用规则修复不合规的 AWS 资源 AWS Config

使用 AWS CDK 和来部署和管理 AWS Control Tower 控件 CloudFormation

由 Iker Reina Fuente (AWS) 和 Ivan Girardi (AWS) 编写

摘要

此模式描述了如何使用 AWS CloudFormation 、实施和管理预防、侦查和主动 AWS Control Tower 控制作为基础架构即代码 (IaC)。 AWS Cloud Development Kit (AWS CDK) 控制（也称为护栏）是一条高级规则，可为您的整体 AWS Control Tower 环境提供持续的治理。例如，您可以使用控件要求记录您的信息， AWS 账户 然后在发生特定的安全相关事件时配置自动通知。

AWS Control Tower 帮助您实施预防、侦查和主动控制，以管理您的 AWS 资源并监控多个资源的合规性 AWS 账户。每个控件都会强制执行一条规则。在此模式中，您使用提供的 IaC 模板来指定要在环境中部署哪些控件。

AWS Control Tower 控件适用于整个组织单位 (OU)，而控制会影响组织单位 AWS 账户 内的每个单位。因此，当用户在您的登录区中的任何账户中执行任何操作时，该操作将受到管理 OU 的控制的约束。

实施 AWS Control Tower 控制措施有助于为您的 AWS 着陆区（Landing zone）奠定坚实的安全基础。通过使用这种模式将控件部署为 IaC 直至 CloudFormation 和 AWS CDK，您可以标准化着陆区域中的控件，并更有效地部署和管理它们。此解决方案在部署期间使用 cdk_nag 来扫描应用程序。 AWS CDK 此工具会检查应用程序是否符合 AWS 最佳实践。

要将 AWS Control Tower 控件部署为 IaC，也可以使用 HashiCorp Terraform 来代替。 AWS CDK有关更多信息，请参阅使用 Terraform 部署和管理 AWS Control Tower 控件。

目标受众

建议有使用 AWS Control Tower、 CloudFormation AWS CDK、和经验的用户使用此模式 AWS Organizations。

先决条件和限制

先决条件

- 作为一个组织在 AWS Organizations 和一个 AWS Control Tower 着陆区进行主动 AWS 账户 管理。有关说明，请参阅 AWS Control Tower 文档中的入门。

- AWS Command Line Interface (AWS CLI)，已安装并配置。
- 节点包管理器 (npm)，已安装并配置为。 AWS CDK
- 的@@ 先决条件 AWS CDK。
- 在部署账户中担任现有 AWS Identity and Access Management (IAM) 角色的权限。
- 在组织管理账户中担任可用于引导 AWS CDK的 IAM 角色的权限。该角色必须具有修改和部署 CloudFormation 资源的权限。有关更多信息，请参阅文档中的引导。 AWS CDK
- 在组织的管理账户中创建 IAM 角色和策略的权限。有关更多信息，请参阅 IAM 文档中的访问 IAM 资源所需权限。
- 使用标识符 CT.CLOUDFORMATION.PR.1 应用基于服务控制策略（SCP）的控制。必须激活 SCP 才能部署主动控制。有关说明，请参阅不允许在 AWS CloudFormation 注册表中管理资源类型、模块和挂钩。

限制

- 此模式提供了跨 AWS 账户部署此解决方案（从部署账户到组织管理账户）的说明。出于测试目的，您可直接在管理账户中部署此解决方案，但未明确提供此配置的说明。
- 对于 AWS Control Tower 控件，此模式需要使用以下格式的全局标识符：

```
arn:<PARTITION>:controlcatalog:::control/<CONTROL_CATALOG_OPAQUE_ID>
```

此模式的先前版本使用了不再支持的区域标识符。我们建议您从区域标识符迁移到全球标识符。全局标识符可帮助您管理控件并扩大可以使用的控件数量。

> **Note**
>
> 在大多数情况下，的值<PARTITION>为aws。

产品版本

- AWS Control Tower 版本 3.2 或更高版本
- Python 版本 3.9 或更高版本
- npm 版本 8.9.0 或更高版本

架构

本部分概括介绍此解决方案，以及由示例代码建立的架构。下图显示了跨 OU 中的各个账户部署的控件。

AWS Control Tower 控制是根据其行为和指导进行分类的。

控制行主要包含三种类型：

1. 预防性控制旨在防止行动发生。这些策略是通过中的服务控制策略 (SCPs) 或资源控制策略 (RCPs) 来实现的 AWS Organizations。预防性控制的状态为强制实施或未启用。所有方面都支持预防性控制 AWS 区域。
2. Det@@ ective 控件旨在在特定事件发生时对其进行检测并记录操作 AWS CloudTrail。这些都是通过AWS Config 规则实现的。检测性控制的状态为合规、违规或未启用。Detective 控件仅适用于 AWS 区域 支持的控件 AWS Control Tower。
3. 主动控制会扫描将由配置的资源， AWS CloudFormation 并检查它们是否符合贵公司的政策和目标。不合规的资源将不会被配置。这些都是用AWS CloudFormation 钩子实现的。主动控制的状态为通过、失败或跳过。

控制指南是指有关如何将每种控制应用于您的控制的推荐做法 OUs。 AWS Control Tower 提供三类指导：必修指导、强烈推荐指导和选修指导。控制指导与其行为无关。有关更多信息，请参阅控制行为和指导。

工具

AWS 服务

- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，可帮助您在代码中定义和配置 AWS Cloud 基础架构。AWS CDK 工具包是与您的 AWS CDK 应用程序交互的主要工具。
- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。
- AWS Config提供了您的资源 AWS 账户 及其配置方式的详细视图。它可以帮助您确定资源之间的关联方式，以及它们的配置如何随时间变化。
- AWS Control Tower按照规范性最佳实践，帮助您设置和管理 AWS 多账户环境。
- AWS Organizations是一项账户管理服务，可帮助您将多个账户整合 AWS 账户 到一个由您创建和集中管理的组织中。

其他工具

- cdk_nag 是一个开源工具，它使用规则包的组合来检查 AWS CDK 应用程序是否符合最佳实践。
- npm 是在 Node.js 环境中运行的软件注册表，用于共享或借用软件包以及管理私有软件包的部署。
- Python 是通用的计算机编程语言。

代码存储库

此模式的代码可在使用 AWS CDK存储库 GitHub 部署 AWS Control Tower 控件中找到。你使用 cdk.json 文件与 AWS CDK 应用程序进行交互，然后使用 pack age.json 文件来安装 npm 软件包。

最佳实践

- 遵循最低权限原则（IAM 文档）。此模式中提供的示例 IAM 策略和信任策略包括所需的最低权限，在管理账户中创建的 AWS CDK 堆栈受这些权限的限制。
- 遵循AWS Control Tower 管理员最佳实践（AWS Control Tower 文档）。
- 遵循 AWS CDK（AWS CDK 文档）中开发和部署云基础架构的最佳实践。
- 引导时 AWS CDK，请自定义引导模板以定义策略和应能够读取和写入管理账户中任何资源的受信任账户。有关更多信息，请参阅自定义引导。
- 使用代码分析工具（例如 cfn_nag ）来扫描生成的模板。 CloudFormation cfn-nag 工具在 CloudFormation 模板中寻找可能表明基础架构不安全的模式。你也可以使用 cdk-nag 通过 cloudformation-include 模块来检查你的 CloudFormation 模板。

操作说明

准备启用控件

Task	描述	所需技能
在管理账户中创建 IAM 角色。	1. 使用其他信息部分的 IAM policy 中定义的权限，在管理账户创建 IAM policy。有关说明，请参阅 IAM 文档中的创建 IAM policy。记下该策略的 Amazon 资源名称	DevOps 工程师，通用 AWS

Task	描述	所需技能

	（ARN）。以下是 ARN 示例。
	
	`arn:aws:iam::<MANAGEMENT-ACCOUNT-ID>:policy/<POLICY-NAME>`
	
	2. 在管理账户中创建 IAM 角色，附加您在上一步中创建的 IAM 权限策略，并将自定义信任策略附加到其他信息部分的信任策略。有关说明，请参阅 IAM 文档中的使用自定义信任策略创建角色。以下是新角色的 ARN 示例：
	
	`arn:aws:iam::<MANAGEMENT-ACCOUNT-ID>:role/<ROLE-NAME>`	

Task	描述	所需技能
Bootstrap AWS CDK。	1. 在管理账户中，扮演一个拥有引导 AWS CDK权限的角色。 2. 输入以下命令，更换以下命令： • <MANAGEMENT-ACCOUNT-ID> 是组织的管理账户的 ID。 • <AWS-CONTROL-TOWER-REGION> 是部署在 AWS 区域 AWS Control Tower 哪里。有关区域代码的完整列表，请参阅 AWS 一般参考中的区域终端节点。 • <DEPLOYMENT-ACCOUNT-ID> 是部署账户的 ID。 • <DEPLOYMENT-ROLE-NAME> 是您正在使用部署账户的 IAM 角色的名称。 • <POLICY-NAME> 是您在管理账户中创建的策略名称。 `$ npx cdk bootstrap aws://<MANAGEMENT-ACCOUNT-ID>/<AWS-CONTROL-TOWER-REGION> \ --trust arn:aws:iam::<DEPLOYMENT-ACCOUNT-ID>:role/<DEPLOYMENT-ROLE-NAME> \`	DevOps 工程师，通用 AWS，Python

Task	描述	所需技能
	`--cloudformation-execution-policies arn:aws:iam::<MANAGEMENT-ACCOUNT-ID>:policy/<POLICY-NAME>`	
克隆存储库。	在 bash Shell 中输入以下命令：这将使用来自 GitHub的 AWS CDK存储库克隆 Deploy AWS Control Tower 控件。 `git clone https://github.com/aws-samples/aws-control-tower-controls-cdk.git`	DevOps 工程师，通用 AWS

Task	描述	所需技能
编辑 AWS CDK 配置文件。	1. 在克隆存储库中，打开 constants.py 文件。 2. 在`ACCOUNT_ID` 参数中，输入您的管理账户 ID。 3. 在`<AWS-CONTROL-TOWER-REGION>` 参数中，输入部署 AWS 区域 位置 AWS Control Tower 。 4. 在`ROLE_ARN` 参数中，输入您在管理账户中创建的角色的 ARN。 5. 打开 AWS Control Tower 文档中的所有全局标识符。 6. 在 JSON 格式的列表中，找到要实现的控件，然后复制其全局标识符（也称为 {CONTROL_CATALOG_OPAQUE_ID} 值）。例如，AWS-GR_AUDIT_BUCKET_ENCRYPTION_ ENABLED 控件的全局标识符为。`k4izcjxhukijhajp6ks5mjxk` 7. 在该`GUARDRAILS_CONFIGURATION` 部分的`Enable-Control` 参数中，输入您复制的全局标识符。以双引号输入标识符，并用逗号分隔多个标识符。 8. 在该 `GUARDRAILS_CONFIGURATION` 部分的 `OrganizationalUnitIds` 参数	DevOps 工程师，通用 AWS

Task	描述	所需技能
	中，输入要在其中启用控件的组织单位的 ID，例如 `ou-1111-11111111` 。用双引号输入 ID，并 IDs 用逗号分隔多个标识。有关如何检索 OU 的更多信息 IDs，请参阅查看 OU 的详细信息。 9. 保存并关闭 constants.py 文件。有关已更新的 constants.py 文件示例，请参阅此模式的其他信息部分。	

在管理账户中启用控件

Task	描述	所需技能
承担部署账户中的 IAM 角色。	在部署账户中，扮演有权在管理账户中部署 AWS CDK 堆栈的 IAM 角色。有关在中担任 IAM 角色的更多信息 AWS CLI，请参阅中的使用 IAM 角色 AWS CLI。	DevOps 工程师，通用 AWS
激活 环境。	如果您使用 Linux 或 macOS： 1. 输入以下命令以创建虚拟环境。 `$ python3 -m venv .venv` 2. 创建虚拟环境后，输入以下命令将其激活。	DevOps 工程师，通用 AWS

Task	描述	所需技能

	`$ source .venv/bin/activate`
	
	如果您使用的是 Windows：
	
	1. 输入以下命令，以激活虚拟环境。
	
	`% .venv\Scripts\activate.bat`	

安装依赖项。	激活虚拟环境后，输入以下命令运行 install_deps.sh 脚本。此命令安装必需的依赖项。 `$ ./scripts/install_deps.sh`	DevOps 工程师，通用 AWS，Python

部署 堆栈。	输入以下命令以合成和部署 CloudFormation 堆栈。
	
	`$ npx cdk synth`
	`$ npx cdk deploy`
		DevOps 工程师，通用 AWS，Python

相关资源

AWS 文档

- 关于控件（AWS Control Tower 文档）
- 控件库（AWS Control Tower 文档）
- AWS CDK 工具包命令（AWS CDK 文档）
- 使用 Terraform（AWS 规范性指南）部署和管理 AWS Control Tower 控件

其他资源

- Python

其他信息

constants.py 文件示例

下面是更新后的 constants.py 文件的示例。此示例启用 AWS-GR_ENCRYPTED_VOLUMES 控件（全局 ID:）和 AWS-GR_SUBNET_ AUTO_ASSIGN_PUBLIC_IP_DISABLES 控件（全局 ID:**503uicglhjkokaajywfpt6ros**）。 50z1ot237wl8u1lv5ufau6qqo有关全局标识符的列表 IDs，请参阅 AWS Control Tower 文档中的所有全局标识符。

```
ACCOUNT_ID = 111122223333
AWS_CONTROL_TOWER_REGION = us-east-2
ROLE_ARN = "arn:aws:iam::111122223333:role/CT-Controls-Role"
GUARDRAILS_CONFIGURATION = [
    {
        "Enable-Control": {
            "503uicglhjkokaajywfpt6ros",
            ...
        },
        "OrganizationalUnitIds": ["ou-1111-11111111", "ou-2222-22222222"...],
    },
    {
        "Enable-Control": {
            "50z1ot237wl8u1lv5ufau6qqo",
            ...
        },
        "OrganizationalUnitIds": ["ou-2222-22222222"...],
    },
]
```

IAM 策略

以下示例策略允许在将 AWS CDK 堆栈从部署账户部署到管理账户时启用或禁用 AWS Control Tower 控制所需的最少操作。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": [
                "controltower:EnableControl",
                "controltower:DisableControl",
                "controltower:GetControlOperation",
                "controltower:ListEnabledControls",
                "organizations:AttachPolicy",
                "organizations:CreatePolicy",
                "organizations:DeletePolicy",
                "organizations:DescribeOrganization",
                "organizations:DescribeOrganizationalUnit",
                "organizations:DetachPolicy",
                "organizations:ListAccounts",
                "organizations:ListAWSServiceAccessForOrganization",
                "organizations:ListChildren",
                "organizations:ListOrganizationalUnitsForParent",
                "organizations:ListParents",
                "organizations:ListPoliciesForTarget",
                "organizations:ListRoots",
                "organizations:UpdatePolicy",
                "ssm:GetParameters"
            ],
            "Resource": "*"
        }
    ]
}
```

信任策略

以下自定义信任策略允许部署账户中特定 IAM 角色承担管理账户中的 IAM 角色。替换以下内容：

- <DEPLOYMENT-ACCOUNT-ID> 是部署账户的 ID
- <DEPLOYMENT-ROLE-NAME> 是部署账户中允许在管理账户中担任该角色的角色的名称

```
{
    “Version”: “2012-10-17”,
    “Statement”: [
        {
            “Effect”: “Allow”,
            “Principal”: {
                “AWS”: “arn:aws:iam::<DEPLOYMENT-ACCOUNT-ID>:role/<DEPLOYMENT-ROLE-
NAME>”
```

```
            },
            “Action”: “sts:AssumeRole”,
            “Condition”: {}
        }
    ]
}
```

使用 Terraform 部署和管理 AWS Control Tower 控件

由 Iker Reina Fuente (AWS) 和 Ivan Girardi (AWS) 编写

摘要

此模式描述了如何使用 AWS Control Tower 控件、 HashiCorp Terraform 和基础设施即代码 (IaC) 来实施和管理预防、侦查和主动安全控制。控制（也称为护栏）是一条高级规则，可为您的整体 AWS Control Tower 环境提供持续的治理。例如，您可以使用控件要求记录您的信息， AWS 账户 然后在发生特定的安全相关事件时配置自动通知。

AWS Control Tower 帮助您实施预防、侦查和主动控制，以管理您的 AWS 资源并监控多个资源的合规性 AWS 账户。每个控件都会强制执行一条规则。在此模式中，您使用提供的 IaC 模板来指定要在环境中部署哪些控件。

AWS Control Tower 控件适用于整个组织单位 (OU)，而控制会影响组织单位 AWS 账户 内的每个单位。因此，当用户在您的登录区中的任何账户中执行任何操作时，该操作将受到管理 OU 的控制的约束。

实施 AWS Control Tower 控制措施有助于为您的 AWS 着陆区（Landing zone）奠定坚实的安全基础。通过使用此模式通过 Terraform 将控件部署为 IaC，可以标准化登录区中的控件，并更有效地部署和管理它们。

要将 AWS Control Tower 控件部署为 IaC，也可以使用 AWS Cloud Development Kit (AWS CDK) 代替 Terraform。有关更多信息，请参阅使用 AWS CDK 和部署和管理 AWS Control Tower 控件 AWS CloudFormation。

目标受众

建议有使用 Terraform 和经验的 AWS Control Tower用户使用此模式。 AWS Organizations

先决条件和限制

先决条件

- 作为一个组织在 AWS Organizations 和一个 AWS Control Tower 着陆区进行主动 AWS 账户 管理。有关说明，请参阅 AWS Control Tower 文档中的入门。
- AWS Command Line Interface (AWS CLI)，已安装并配置。

- 管理账户中有权部署此模式的 AWS Identity and Access Management (IAM) 角色。有关所需权限和示例策略的更多信息，请参阅此模式的其他信息部分中的 IAM 角色最低权限权限。
- 在管理账户中承担 IAM 角色的权限。
- 使用标识符 CT.CLOUDFORMATION.PR.1 应用基于服务控制策略（SCP）的控制。必须激活 SCP 才能部署主动控制。有关说明，请参阅不允许在 AWS CloudFormation 注册表中管理资源类型、模块和挂钩。
- Terraform CLI，已安装（Terraform 文档）。
- Terraform P AWS rovider，已配置（Terraform 文档）。
- Terraform 后端，已配置（Terraform 文档）。

限制

- 对于 AWS Control Tower 控件，此模式需要使用以下格式的全局标识符：

```
arn:<PARTITION>:controlcatalog:::control/<CONTROL_CATALOG_OPAQUE_ID>
```

此模式的先前版本使用了不再支持的区域标识符。我们建议您从区域标识符迁移到全球标识符。全局标识符可帮助您管理控件并扩大可以使用的控件数量。

> **Note**
>
> 在大多数情况下，的值<PARTITION>为aws。

产品版本

- AWS Control Tower 版本 3.2 或更高版本
- 对于 460.39 及以上版本：
- Terraform P AWS rovider 版本 4.67 或更高版本

架构

本部分概括介绍此解决方案，以及由示例代码建立的架构。下图显示了跨 OU 中的各个账户部署的控件。

AWS Control Tower 控制是根据其行为和指导进行分类的。

控制行主要包含三种类型：

1. 预防性控制旨在防止行动发生。这些策略是通过中的服务控制策略 (SCPs) 或资源控制策略 (RCPs) 来实现的 AWS Organizations。预防性控制的状态为强制实施或未启用。所有方面都支持预防性控制 AWS 区域。
2. Det@@ ective 控件旨在在特定事件发生时对其进行检测并记录操作 AWS CloudTrail。这些都是通过AWS Config 规则实现的。检测性控制的状态为合规、违规或未启用。Detective 控件仅适用于 AWS 区域 支持的控件 AWS Control Tower。
3. 主动控制会扫描将由配置的资源， AWS CloudFormation 并检查它们是否符合贵公司的政策和目标。不合规的资源将不会被配置。这些都是用AWS CloudFormation 钩子实现的。主动控制的状态为通过、失败或跳过。

控制指导是如何将每种控制应用于您的控制的推荐做法 OUs。 AWS Control Tower 提供三类指导：必修指导、强烈推荐指导和选修指导。控制指导与其行为无关。有关更多信息，请参阅控制行为和指导。

工具

AWS 服务

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在资源的整个生命周期中跨地区对其 AWS 账户 进行管理。
- AWS Config提供了您的资源 AWS 账户 及其配置方式的详细视图。它可以帮助您确定资源之间的关联方式，以及它们的配置如何随时间变化。
- AWS Control Tower按照规范性最佳实践，帮助您设置和管理 AWS 多账户环境。
- AWS Organizations是一项账户管理服务，可帮助您将多个账户整合 AWS 账户 到一个由您创建和集中管理的组织中。

其他工具

- HashiCorp Terraform 是一款基础设施即代码 (IaC) 工具，可帮助您使用代码来配置和管理云基础架构和资源。

代码存储库

此模式的代码可在使用 Terraform 存储库 GitHub 部署和管理 AWS Control Tower 控件中找到。

最佳实践

- 用于部署此解决方案的 IAM 角色应遵守最低权限原则（IAM 文档）。
- 遵循AWS Control Tower 管理员最佳实践（AWS Control Tower 文档）。

操作说明

在管理账户中启用控件

Task	描述	所需技能
克隆存储库。	在 bash Shell 中输入以下命令：这将使用来自的 Terraform 存储库克隆部署和管理 AWS Control Tower 控件。 GitHub `git clone https://github.com/aws-samples/aws-control-tower-controls-terraform.git`	DevOps 工程师
编辑 Terraform 后端和配置文件。	1. 在克隆的存储库中，打开 backend.tf 文件。 2. 编辑文件，以设置 Terraform 后端配置。您在此文件中定义的配置取决于环境。有关更多信息，请参阅后端配置（Terraform 文档）。 3. 保存并关闭 backend.tf 文件。	DevOps 工程师，Terraform
编辑 Terraform 提供程序配置文件。	1. 在克隆的存储库中，打开 provider.tf 文件。	DevOps 工程师，Terraform

Task	描述	所需技能
	2. 编辑文件，以设置 Terraform 提供程序配置。有关更多信息，请参阅提供程序配置（Terraform 文档）。将设置 AWS 区域 为可用 AWS Control Tower API 的区域。 3. 保存并关闭 provider.tf 文件。	

Task	描述	所需技能
编辑配置文件。	1. 在克隆的存储库中，打开 variables.tfvars 文件。 2. 打开 AWS Control Tower 文档中的所有全局标识符。 3. 在 JSON 格式的列表中，找到要实现的控件，然后复制其全局标识符（也称为 {CONTROL_CATALOG_OPAQUE_ID} 值）。例如，AWS-GR_AUDIT_BUCKET_ENCRYPTION_ ENABLED 控件的全局标识符为。`k4izcjxhukijhajp6ks5mjxk` 4. 在该`controls`部分的`control_names` 参数中，输入您复制的全局标识符。 5. 在该 `controls` 部分的 `organizational_unit_ids` 参数中，输入要在其中启用控件的组织单位的 ID，例如 `ou-1111-11111111` 。用双引号输入 ID，并 IDs 用逗号分隔多个标识。有关如何检索 OU 的更多信息 IDs，请参阅查看 OU 的详细信息。 6. 保存并关闭 variables.tfvars 文件。有关已更新的 variables.tfvars 文件示例，请参阅此模式的其他信息部分。	DevOps 工程师，通用 AWS，Terraform

Task	描述	所需技能
在管理账户中配置 IAM 角色。	在管理账户中，承担有权部署 Terraform 配置文件的 IAM 角色。有关所需权限和示例策略的更多信息，请参阅其他信息部分中的 IAM 角色的最低权限权限。有关在中担任 IAM 角色的更多信息 AWS CLI，请参阅中的使用 IAM 角色 AWS CLI。	DevOps 工程师，通用 AWS

部署配置文件。	1. 输入以下命令，以初始化 Terraform。
	`$ terraform init -upgrade`
	2. 输入以下命令，以预览与当前状态对比的更改。
	`$ terraform plan -var-file="variables.tfvars"`
	3. 查看 Terraform 计划中的配置更改，并确认您想要在组织中实施这些更改。
	4. 输入以下命令以部署资源。
	`$ terraform apply -var-file="variables.tfvars"`
		DevOps 工程师，通用 AWS，Terraform

（可选）禁用 AWS Control Tower 管理账户中的控件

Task	描述	所需技能
运行 `destroy` 命令。	输入以下命令，以删除此模式部署的资源。 `$ terraform destroy -var-file="variables.tfvars"`	DevOps 工程师，通用 AWS，Terraform

故障排除

事务	解决方案
`Error: creating ControlTower Control ValidationException: Guardrail <control ID> is already enabled on organizational unit <OU ID>` 错误	您尝试启用的控件已经在目标 OU 中启用。如果用户通过、通过 AWS Control Tower 或通过手动启用控件 AWS Management Console，则可能会发生此错误 AWS Organizations。若要部署 Terraform 配置文件，您可以使用以下任一选项。 选项 1：更新 Terraform 当前状态文件 您可以将资源导入至 Terraform 当前状态文件。当您重新运行 `apply` 命令时，Terraform 将跳过这个资源。执行以下操作，将资源导入至当前 Terraform 状态： 1. 在 AWS Control Tower 管理账户中，输入以下命令以检索组织根目录的 Amazon 资源名称 (ARNs) 列表。 OUs `<root-ID>` 有关检索此 ID 的更多信息，请参见查看根的详细信息。

事务	解决方案

	`aws organizations list-organizational-units-for-parent --parent-id <root-ID>`
	2. 对于上一步中返回的每个 OU，输入以下命令，其中 <OU-ARN> 是 OU 的 ARN。
	`aws controltower list-enabled-controls --target-identifier <OU-ARN>`
	3. 复制 ARNs 并在所需模块中执行 Terraform 导入，使其包含在 Terraform 状态中。有关说明，请参阅导入（Terraform 文档）。
	4. 重复操作说明部分部署配置中的步骤。
	
	选项 2：禁用控件
	
	如果您在非生产环境中工作，可以禁用控制台中的控件。通过重复 Epics 部分部署配置中的步骤来重新启用它。不建议在生产环境中使用此方法，因为有一段时间该控件会被禁用。如果要在生产环境中使用此选项，则可以实施临时控制，例如在中 AWS Organizations临时应用 SCP。

相关资源

AWS 文档

- 关于控件（AWS Control Tower 文档）
- 控件库（AWS Control Tower 文档）
- 使用 AWS CDK 和 AWS CloudFormation（AWS 规范性指导）部署和管理 AWS Control Tower 控件

其他资源

- Terraform
- Terraform CLI 文档

其他信息

Example variables.tfvars 文件

以下是更新的 variables.tfvars 文件的示例。此示例启用 AWS-GR_ENCRYPTED_VOLUMES 控件（全局 ID:）和 AWS-GR_SUBNET_ AUTO_ASSIGN_PUBLIC_IP_DISABLES 控件（全局 ID:**`503uicglhjkokaajywfpt6ros`**）。 `50z1ot237wl8u1lv5ufau6qqo`有关全局标识符的列表 IDs，请参阅 AWS Control Tower 文档中的所有全局标识符。

```
controls = [
    {
        control_names = [
            "503uicglhjkokaajywfpt6ros",
            ...
        ],
        organizational_unit_ids = ["ou-1111-11111111", "ou-2222-22222222"...],
    },
    {
        control_names = [
            "50z1ot237wl8u1lv5ufau6qqo",
            ...
        ],
        organizational_unit_ids = ["ou-1111-11111111"...],
    },
]
```

IAM 角色的最低权限

这种模式要求您在管理账户中担任 IAM 角色。最佳做法是承担具有临时权限的角色，并根据最低权限原则限制权限。以下示例策略允许执行启用或禁用 AWS Control Tower 控件所需的最少操作。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "controltower:EnableControl",
```

```
                "controltower:DisableControl",
                "controltower:GetControlOperation",
                "controltower:ListEnabledControls",
                "organizations:AttachPolicy",
                "organizations:CreatePolicy",
                "organizations:DeletePolicy",
                "organizations:DescribeOrganization",
                "organizations:DetachPolicy",
                "organizations:ListAccounts",
                "organizations:ListAWSServiceAccessForOrganization",
                "organizations:ListChildren",
                "organizations:ListOrganizationalUnitsForParent",
                "organizations:ListParents",
                "organizations:ListPoliciesForTarget",
                "organizations:ListRoots",
                "organizations:UpdatePolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

部署可同时检测多个代码交付项中的安全问题的管道

由本杰明·莫里斯（AWS）、狄娜·奥杜姆（AWS）、以赛亚·席斯勒（AWS）、Sapeksh Madan（AWS）和蒂姆·哈恩（AWS）创作

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多。

简单代码扫描管道 (SCSP) 提供双击创建代码分析管道，该管道可并行运行行业标准的开源安全工具。这使开发人员能够检查其代码的质量和安全性，而无需安装工具，甚至无需了解如何运行它们。这可以帮助您减少代码交付中的漏洞和错误配置。它还可以减少您的组织在安装、研究和配置安全工具上花费的时间。

在 SCSP 之前，使用此特定工具套件扫描代码需要开发人员查找、手动安装和配置软件分析工具。即使是本地安装的，诸如自动安全助手 (ASH) 之类的 all-in-one工具也需要配置 Docker 容器才能运行。但是，在 SCSP 中，一套行业标准的代码分析工具会自动在中运行。 AWS Cloud使用此解决方案，您可以使用 Git 推送代码可交付成果，然后收到可视化输出，其中包含哪些安全检查失败的 at-a-glance 见解。

先决条件和限制

- 活跃的 AWS 账户
- 您要扫描的一个或多个代码可交付成果，以发现安全问题
- AWS Command Line Interface (AWS CLI)，已安装并配置
- 已安装 Python 3.0 或更高版本以及 pip 版本 9.0.3 或更高版本
- Git，已安装
- 安装git-remote-codecommit在您的本地工作站上

架构

目标技术堆栈

- AWS CodeCommit 存储库
- AWS CodeBuild 项目
- AWS CodePipeline 管道

- Amazon Simple Storage Service (Amazon S3)桶
- AWS CloudFormation 模板

目标架构

用于静态代码分析的 SCSP 是一个旨在为可交付代码提供安全反馈的 DevOps 项目。

1. 在中 AWS Management Console，登录到目标 AWS 账户。确认您位于要部署管道 AWS 区域 的位置。
2. 使用代码仓库中的 CloudFormation 模板部署 SCSP 堆栈。这将创建一个新的 CodeCommit 存储库和 CodeBuild 项目。

> **Note**
>
> 作为替代部署选项，您可以通过在 CodeCommit 堆栈部署期间提供存储库的 Amazon 资源名称 (ARN) 作为参数来使用现有存储库。

3. 将存储库克隆到您的本地工作站，然后将所有文件添加到克隆存储库中相应的文件夹。
4. 使用 Git 将文件添加、提交和推送到 CodeCommit 存储库。
5. 推送到 CodeCommit 存储库会启动作 CodeBuild 业。该 CodeBuild 项目使用安全工具来扫描代码可交付成果。
6. 查看管道的输出。发现错误级别问题的安全工具将导致管道中的操作失败。修复这些错误或将其禁止为误报。在管道的 S3 存储桶中 CodePipeline 或其中的操作详细信息中查看工具输出的详细信息。

工具

AWS 服务

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在资源的整个生命周期中跨地区对其 AWS 账户 进行管理。
- AWS CodeBuild是一项完全托管的生成服务，可帮助您编译源代码、运行单元测试和生成可随时部署的工件。
- AWS CodeCommit是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。

其他工具

有关 SCSP 用于扫描代码交付件的工具的完整列表，请参阅中的 SCSP 自述文件。 GitHub

代码存储库

此模式的代码可在中的简单代码扫描管道 (SCSP) 存储库中 GitHub找到。

操作说明

部署 SCSP

Task	描述	所需技能
创建 CloudFormation 堆栈。	1. 登录到 AWS Management Console。 2. 在控制台中，确认您位于要部署解决方案的目标区域。有关更多信息，请参阅选择区域。 3. 选择以下链接。这将在中打开快速创建堆栈向导 CloudFormation。 https://console.aws.amazon.com/cloudformation/家？ #/ stacks/create/review?templateURL=https://proservetools.s3.amazonaws.com/cft/scsp-pipeline-stack.template.json&stackName=SimpleCodeScanPipeline 4. 在快速创建堆栈向导中，查看堆栈的参数设置，并根据用例的需要进行任何修改。 5. 选择 “我确认 AWS CloudFormation 可能会创建	AWS DevOps，AWS 管理员

Task	描述	所需技能
	IAM 资源”，然后选择 “创建堆栈”。 这将创建一个 CodeCommit 存储库、一个 CodePipeline 管道、多个 CodeBuild 任务定义和一个 S3 存储桶。生成运行和扫描结果将复制到此存储桶中。完全部署 CloudFormation 堆栈后，SCSP 就可以使用了。	

使用管道

Task	描述	所需技能
检查扫描结果。	1. 在 Amazon S3 控制台的 Buckets 中，选择 s impeline-scanpipeline-deleteresourcespipelinereso 存储桶。 2. 选择 scan_resul ts 目录，然后选择带有最新扫描日期戳的文件夹。 3. 查看此文件夹中的日志文件，查看管道中使用的安全工具检测到的所有问题。发现错误级别问题的安全工具将导致管道中的`failed`操作。如果它们是误报，则需要修复或抑制它们。	AWS 应用程序开发人员 DevOps

Task	描述	所需技能
	Note 您还可以在 CodePipeline 控制台的操作详细信息部分中查看工具输出的详细信息（扫描通过和失败）。	

故障排除

事务	解决方案
HashiCorp 未扫描 Terraform 或 AWS CloudFormation 文件。	确保 Terraform (.tf) 和 CloudFormation（.yml、.yaml 或.json）文件放在克隆存储库的相应文件夹中。 CodeCommit
`git clone`命令失败。	请确保您已安装`git-remote-codecommit` 并且您的 CLI 可以访问有权读取 CodeCommit 存储库的 AWS 证书。
并发错误，例如。`Project-level concurrent build limit cannot exceed the account-level concurrent build limit of 1`	在CodePipeline 控制台中选择“发布更改”按钮，重新运行管道。这是一个已知问题，在管道运行的最初几次似乎最为常见。

相关资源

提供有关 SCSP 项目的反馈。

其他信息

常见问题解答

SCSP 项目和自动安全助手 (ASH) 一样吗？

不是。 如果您想要一个使用容器运行代码扫描工具的 CLI 工具，请使用 ASH。A@@ utomated Security Helper (ASH) 是一款旨在降低新代码、基础设施或 IAM 资源配置中出现安全违规可能性的工具。ASH 是一个可以在本地运行的命令行实用程序。本地使用需要在系统上安装并运行容器环境。

如果您想要比 ASH 更简单的设置管道，请使用 SCSP。SCSP 不需要本地安装。SCSP 旨在在管道中单独运行检查并按工具显示结果。SCSP 还避免了设置 Docker 的大量开销，而且它与操作系统 (OS) 无关。

SCSP 仅适用于安全团队吗？

不，任何人都可以部署管道来确定其代码的哪些部分未通过安全检查。例如，非安全用户可以使用 SCSP 检查自己的代码，然后再与安全团队一起审查。

如果我使用的是其他类型的存储库，例如、或 Bitbucket，我能否使用 SCSP？ GitLab GitHub

您可以将本地 git 存储库配置为指向两个不同的远程存储库。例如，您可以克隆现有 GitLab 存储库，创建一个 SCSP 实例（如果需要 CloudFormation，指定 Terraform 和 AWS Config 规则开发套件 (AWS RDK) 文件夹），然后也可以使用`git remote add upstream <SCSPGitLink>`将本地存储库指向 SCS CodeCommit P 存储库。这允许先将代码更改发送到 SCSP 并进行验证，然后在进行任何其他更新以解决发现问题之后，将其推送到 GitLab GitHub、或 Bitbucket 存储库。有关多个遥控器的更多信息，请参阅将提交推送到其他 Git 存储库（AWS 博客文章）。

> **Note**
>
> 要小心漂移，例如避免通过 Web 界面进行更改。

贡献和添加你自己的动作

SCSP 设置作为一个 GitHub 项目进行维护，其中包含 SCSP AWS Cloud Development Kit (AWS CDK) 应用程序的源代码。要向管道添加其他检查，需要更新 AWS CDK 应用程序，然后合成或部署到管道运行的目标 AWS 账户 中。为此，首先克隆 SCSP GitHub 项目，然后在`lib`文件夹中找到堆栈定义文件。

如果你想添加额外的检查，那么 AWS CDK 代码中的`StandardizedCodeBuildProject`类可以非常简单地添加动作。提供名称、描述和 `install` /或`build`命令。 AWS CDK 使用合理的默认值创建 CodeBuild 项目。除了创建构建项目外，您还需要将其添加到构建阶段的 CodePipeline 操作中。在设

计新检查时，FAIL如果扫描工具检测到问题或无法运行，则应采取行动。PASS如果扫描工具未检测到任何问题，则应执行该操作。有关配置工具的示例，请查看Bandit操作代码。

有关预期输入和输出的更多信息，请参阅存储库文档。

如果添加自定义操作，则需要使用cdk deploy或cdk synth + CloudFormation deploy部署SCSP。这是因为快速创建堆栈 CloudFormation 模板由存储库所有者维护。

使用以下方法为公共子网部署基于侦探属性的访问控制 AWS Config

由阿尔贝托·梅嫩德斯 (AWS) 创作

摘要

分布式边缘网络架构依赖于与其虚拟私有云中的工作负载一起运行的网络边缘安全（VPCs）。与更常见的集中式方法相比，这提供了前所未有的可扩展性。尽管在工作负载账户中部署公有子网可以带来好处，但它也会带来新的安全风险，因为它增加了攻击面。我们建议您仅在这些资源的公有子网中部署 Elastic Load Balancing 资源，例如应用程序负载均衡器或 NAT 网关。 VPCs在专用公有子网中使用负载均衡器和 NAT 网关有助于实现对入站和出站流量的精细控制。

我们建议您同时实施预防控制和检测控制，以限制可在公有子网中部署的资源类型。有关使用基于属性的访问控制 (ABAC) 为公有子网部署预防性控制的更多信息，请参阅为公有子网部署基于属性的预防性访问控制。尽管这些预防性控制措施对大多数情况都有效，但可能无法解决所有可能的用例。因此，此模式建立在 ABAC 方法的基础上，可帮助您配置有关部署在公共子网中的不合规资源的警报。该解决方案检查弹性网络接口是否属于公有子网中不允许使用的资源。

为此，此模式使用AWS Config 自定义规则和 ABAC。无论何时创建或修改弹性网络接口的配置，自定义规则都会对其进行处理。简而言之，此规则执行两个操作来确定网络接口是否合规：

1. 为了确定网络接口是否在规则范围内，该规则会检查子网是否具有表明其为公有子网的特定AWS 标记。例如，这个标签可能是`IsPublicFacing=True`。
2. 如果网络接口部署在公有子网中，则该规则将检查 AWS 服务 创建此资源的原因。如果该资源不是 Elastic Load Balancing 资源或 NAT 网关，则它会将该资源标记为不合规。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- AWS Config，在工作负载帐户中设置
- 在工作负载账户中部署所需资源的权限
- 具有公有子网的 VPC
- 正确应用标签以识别目标公有子网
- （可选）中的组织 AWS Organizations

- （可选）中央安全账户，它是 AWS Config 和的委派管理员 AWS Security Hub

架构

目标架构

该图阐释了以下内容：

1. 部署或修改 elastic network interface 资源 (`AWS::EC2::NetworkInterface`) 时，会 AWS Config 捕获事件和配置。
2. AWS Config 将此事件与用于评估配置的自定义规则进行匹配。
3. 调用与此自定义规则关联的 AWS Lambda 函数。该函数评估资源并应用指定的逻辑来确定资源配置是否为`COMPLIANT`、`NON_COMPLIANT`或`NOT_APPLICABLE`。
4. 如果确定资源为，则通过亚马逊简单通知服务 (Amazon SNS) Simple Notification Service AWS Config 发送提醒。`NON_COMPLIANT`

> **Note**
>
> 如果此账户是中的成员账户 AWS Organizations，则可以通过 AWS Config 或将合规性数据发送到中央安全账户 AWS Security Hub。

Lambda 函数评估逻辑

下图显示了 Lambda 函数应用的逻辑，用于评估弹性网络接口的合规性。

自动化和扩缩

这种模式是一种侦探解决方案。您还可以使用补救规则对其进行补充，以自动解决任何不合规的资源。有关更多信息，请参阅使用规则修复不合规的 AWS Config 资源。

您可以通过以下方式扩展此解决方案：

- 强制应用您为识别面向公众的子网而建立的相应 AWS 标签。有关更多信息，请参阅 AWS Organizations 文档中的标签策略。

- 配置一个中央安全帐户，该帐户将 AWS Config 自定义规则应用于组织中的每个工作负载帐户。有关更多信息，请参阅 AWS（AWS 博客文章）中的大规模自动配置合规性。
- AWS Config 与集成 AWS Security Hub ，以便大规模捕获、集中和通知。有关更多信息，请参阅 AWS Security Hub 文档 AWS Config中的配置。

工具

- AWS Config提供了您的资源 AWS 账户 及其配置方式的详细视图。它可以帮助您确定资源之间的关联方式，以及它们的配置如何随时间变化。
- 弹性负载均衡在多个目标上分配传入的应用程序或网络流量。例如，您可以跨亚马逊弹性计算云 (Amazon EC2) 实例、容器以及一个或多个可用区中的 IP 地址分配流量。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。
- Amazon Virtual Private Cloud（亚马逊 VPC）可帮助您将 AWS 资源启动到您定义的虚拟网络中。该虚拟网络类似于您在数据中心中运行的传统网络，并具有使用 AWS的可扩展基础设施的优势。

最佳实践

有关开发自定义 AWS Config 规则的更多示例和最佳实践，请参阅上的官方AWS Config 规则存储库 GitHub。

操作说明

部署解决方案

Task	描述	所需技能
创建 Lambda 函数。	1. 登录 AWS Management Console，然后打开AWS Lambda 控制台。 2. 在 Functions (LAM 函数) 页面上，选择 Create function (创建函数)。	常规 AWS

Task	描述	所需技能
	3. 选择从头开始编写。 4. 在基本信息窗格中，为函数名称输入一个名称。 5. 对于运行时系统，选择 Python 3.12。 6. 将架构设置为 x86_64。 7. 选择创建函数。 8. 选择节点选项卡。 9. 在文件资源管理器中，选择 lambda_function.py。 10.将此模式的 “其他信息” 部分中提供的示例代码粘贴到 lambda_function.py 选项卡中。自定义示例代码以识别`evaluate_change_notification_compliance` 函数中的任何自定义评估逻辑。 11选择部署。	

Task	描述	所需技能
向 Lambda 函数的执行角色添加权限。	1. 在导航窗格中，选择函数。 2. 选择您刚刚创建的函数。 3. 选择 Configuration（配置），然后选择 Permissions（权限）。 4. 选择角色名称以在 AWS Identity and Access Management (IAM) 控制台中打开该角色。 5. 在“权限策略”下，选择“添加权限”，然后选择“创建内联策略”。 6. 选择 JSON。 7. 将以下策略粘贴到策略编辑器中。这允许 Lambda 函数执行以下操作： • 获取子网标签的详细信息。 • 将合规结果发回给 AWS Config。 `{ "Version": "2012-10-17", "Statement": [ { "Action": [ "config:PutEvaluations", "ec2:DescribeSubnets"`	常规 AWS

Task	描述	所需技能

	<pre>],
	            "Resource
	": "*",
	            "Effect":
	 "Allow"
	        }
	    ]
	}</pre>
	8. 选择下一步。
	9. 为策略输入名称，然后选择 Create policy (创建策略)。	

检索 Lambda 函数亚马逊资源名称 (ARN)。	1. 打开 Lambda 控制台。 2. 在导航窗格中，选择函数。 3. 选择您刚刚创建的函数。 4. 在函数概述部分的函数 ARN 下，复制该值。	常规 AWS

Task	描述	所需技能
创建 AWS Config 自定义规则。	1. 打开 AWS Config 管理控制台。 2. 在 Rules 页面，选择 Add rule。 3. 在指定规则类型页面上，选择创建自定义 Lambda 规则，然后选择下一步。 4. 在配置规则页面上，执行以下操作： a. 输入名称和描述。 b. 对于AWS Lambda 函数 ARN，请粘贴您之前复制的 ARN。 c. 对于触发器类型，选择在配置发生更改时。 d. 在 “更改范围” 中，选择 “资源”。 e. 对于资源类型，请选择 AWS EC2 NetworkInterface。 f. 选择下一步。 5. 在查看并创建页面上，验证您的规则，然后选择保存。	常规 AWS

Task	描述	所需技能
配置通知。	1. 按照创建亚马逊 SNS 主题中的说明创建亚马逊 SNS 主题。 2. 按照订阅 Amazon SNS 主题中的说明配置接收 Amazon SNS 主题通知的终端节点。 3. 按照资源不合规时如何收到通知中的说明使用 AWS Config为您的不合规 AWS 资源配置自定义 Amazon EventBridge 规则。	常规 AWS

测试解决方案

Task	描述	所需技能
创建合规资源。	1. 按照以下说明在公有子网中创建支持的资源之一： • 创建 NAT 网关 • Getting started with Network Load Balancers • 创建应用程序负载均衡器 2. 创建资源后，AWS Config 自定义规则将评估与资源关联的弹性网络接口。它将这些网络接口标记为`COMPLIANT`。您可以按照以下步骤查看中的 AWS Config 资源： a. 打开 AWS Config 管理控制台。	常规 AWS

Task	描述	所需技能
	b. 在规则页面上，选择您的规则。 c. 在规则详情页面上，前往页面底部。 d. 在 “范围内的资源” 下，选择 “合规”。确认您看到已创建 IDs 的网络接口。 e. 有关网络接口配置的更多详细信息，请选择资源 ID。	

Task	描述	所需技能
创建不合规的资源。	1. 按照以下说明在公有子网中创建不合规的资源： • 启动 Amazon EC2 实例 • 创建亚马逊关系数据库服务 (Amazon RDS) 数据库实例 • 创建 VPC 终端节点 2. 创建资源后， AWS Config 自定义规则将评估与资源关联的弹性网络接口。它将这些网络接口标记为NON_COMPLIANT 。您可以按照以下步骤查看中的 AWS Config 资源： a. 打开 AWS Config 管理控制台。 b. 在规则页面上，选择您的规则。 c. 在规则详情页面上，前往页面底部。 d. 在 “范围内的资源” 下，选择NonCompliant。确认您看到已创建 IDs 的网络接口。 e. 有关网络接口配置的更多详细信息，请选择资源 ID。 3. 确认您已在您在 Amazon SNS 中配置的终端节点上收到通知。	常规 AWS

Task	描述	所需技能
创建不适用的资源。	1. 在私有子网中，创建任何需要弹性网络 interface 的资源。 2. 创建资源后， AWS Config 自定义规则将评估与资源关联的弹性网络接口。它将这些网络接口标记为`NOT_APPLICABLE` 。这些资源未显示在 AWS Config 控制台中。	常规 AWS

相关资源

AWS 文档

- 设置 AWS Config
- AWS Config 自定义规则
- ABAC for AWS
- 为公有子网部署基于属性的预防性访问控制

其他 AWS 资源

- 大规模自动实现配置合规性 AWS
- 使用网关负载均衡器的分布式检查架构

其他信息

以下是为演示目的而提供的示例 Lambda 函数。

```
import boto3
import json
import os

# Init clients
```

```
config_client = boto3.client('config')
ec2_client = boto3.client('ec2')

def lambda_handler(event, context):

    # Init values
    compliance_value = 'NOT_APPLICABLE'
    invoking_event = json.loads(event['invokingEvent'])
    configuration_item = invoking_event['configurationItem']

    status = configuration_item['configurationItemStatus']
    eventLeftScope = event['eventLeftScope']

    # First check if the event configuration applies. Ex. resource event is not delete
    if (status == 'OK' or status == 'ResourceDiscovered') and not eventLeftScope:
        compliance_value = evaluate_change_notification_compliance(configuration_item)


    config_client.put_evaluations(
       Evaluations=[
           {
               'ComplianceResourceType': invoking_event['configurationItem']
['resourceType'],
               'ComplianceResourceId': invoking_event['configurationItem']
['resourceId'],
               'ComplianceType': compliance_value,
               'OrderingTimestamp': invoking_event['configurationItem']
['configurationItemCaptureTime']
           },
       ],
       ResultToken=event['resultToken'])

# Function with the logs to evaluate the resource
def evaluate_change_notification_compliance(configuration_item):
    is_in_scope = is_in_scope_subnet(configuration_item['configuration']['subnetId'])

    if (configuration_item['resourceType'] != 'AWS::EC2::NetworkInterface') or not
 is_in_scope:
        return 'NOT_APPLICABLE'

    else:
        alb_condition = configuration_item['configuration']['requesterId'] in ['amazon-
elb']
```

```
        nlb_condition = configuration_item['configuration']['interfaceType'] in
 ['network_load_balancer']
        nat_gateway_condition = configuration_item['configuration']['interfaceType'] in
 ['nat_gateway']

        if alb_condition or nlb_condition or nat_gateway_condition:
            return 'COMPLIANT'
    return 'NON_COMPLIANT'

# Function to check if elastic network interface is in public subnet
def is_in_scope_subnet(eni_subnet):

    subnet_description = ec2_client.describe_subnets(
        SubnetIds=[eni_subnet]
    )

    for subnet in subnet_description['Subnets']:
        for tag in subnet['Tags']:
            if tag['Key'] == os.environ.get('TAG_KEY') and tag['Value'] ==
 os.environ.get('TAG_VALUE'):
                return True

    return False
```

为公有子网部署基于属性的预防性访问控制

由 Joel Alfredo Nunez Gonzalez (AWS) 和 Samuel Ortega Sancho (AWS) 编写

摘要

在集中式网络架构中，检查和边缘虚拟私有云 (VPCs) 集中所有入站和出站流量，例如进出互联网的流量。但是，这可能会造成瓶颈或导致达到 AWS 服务限额的限制。与更常见的集中式方法相比，将网络边缘安全与其中的工作负载一起部署可 VPCs 提供前所未有的可扩展性。这称为分布式边缘架构。

尽管在工作负载账户中部署公有子网可以带来好处，但它也会带来新的安全风险，因为它增加了攻击面。我们建议您仅在这些资源的公有子网中部署 Elastic Load Balancing (ELB) 资源，例如应用程序负载均衡器或 NAT 网关。 VPCs在专用公有子网中使用负载均衡器和 NAT 网关有助于实现对入站和出站流量的精细控制。

基于属性的访问权限控制（ABAC）是根据用户属性（如部门、工作角色和团队名称）创建精细访问权限的做法。有关更多信息，请参阅适用于 AWS 的 ABAC。ABAC 可为工作负载账户中的公共子网提供防护机制。这有助于应用程序团队在不影响基础架构安全性的情况下保持敏捷性。

此模式描述了如何通过 AWS Organizations 中的服务控制策略（SCP）和 AWS Identity and Access Management (IAM) 中的策略实施 ABAC 来帮助保护公有子网。您可以将 SCP 应用于组织的成员账户或组织单位 (OU)。这些 ABAC 策略允许用户在目标子网中部署 NAT 网关，并防止他们部署其他亚马逊弹性计算云 (Amazon EC2) 资源，例如 EC2 实例和弹性网络接口。

先决条件和限制

先决条件

- AWS Organizations 中的组织
- 对 AWS Organizations 根账户的管理访问权限
- 在组织中，用于测试 SCP 的活动成员账户或 OU

限制

- 此解决方案中的 SCP 不会阻止使用服务相关角色的 Amazon Web Services 在目标子网中部署资源。这些服务的示例包括弹性负载均衡（ELB）、Amazon Elastic Container Service (Amazon ECS) 和 Amazon Relational Database Service (Amazon RDS)。有关更多信息，请参阅 AWS Organizations 文档中的 SCP 对权限的影响。实施安全控制来检测这些异常。

架构

目标技术堆栈

- SCP 应用于 AWS Organizations 中的 Amazon Web Services account 或 OU
- 以下 IAM 角色：
  - `AutomationAdminRole` — 用于在实施 SCP 后修改子网标签和创建 VPC 资源
  - `TestAdminRole` — 用于测试 SCP 是否阻止其他 IAM 主体（包括具有管理访问权限的主体）执行为 `AutomationAdminRole` 保留的操作

目标架构

1. 您可在目标账户中创建 `AutomationAdminRole` IAM 角色。此角色有权管理网络资源。请注意此角色独有的以下权限：
   - 此角色可以创建 VPCs 和公有子网。
   - 此角色可以修改目标子网的标签分配。
   - 此角色可以管理自己的权限。
2. 在 AWS Organizations 中，您可以将 SCP 应用于目标 Amazon Web Services account 或 OU。有关示例策略，请参阅此模式中的其他信息。
3. CI/CD 管道中的用户或工具可以承担 `AutomationAdminRole` 角色，将 `SubnetType` 标签应用到目标子网。
4. 通过承担其他 IAM 角色，组织中的授权 IAM 主体可以管理目标子网中的 NAT 网关以及 Amazon Web Services account 中其他允许的网络资源，例如路由表。使用 IAM policy 以授予这些权限。有关更多信息，请参阅 Amazon VPC 的身份和访问权限管理。

自动化和扩缩

为了帮助保护公有子网，必须应用相应的 AWS 标签。应用 SCP 后，NAT 网关是授权用户可以在带有标签的子网中创建的唯一一种 Amazon EC2 资源。`SubnetType:IFA`（`IFA`面向互联网的资产。）SCP 会阻止创建其他 Amazon EC2 资源，例如实例和弹性网络接口。 我们建议您使用`AutomationAdminRole`扮演角色的 CI/CD 管道来创建 VPC 资源，以便将这些标签正确应用于公有子网。

工具

Amazon Web Services

- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Organizations 是一项账户管理服务，使您可将多个 Amazon Web Services account 整合到您所创建的组织中并进行集中管理。在 AWS Organizations 中，您可以实施服务控制策略 (SCPs)，这是一种可用于管理组织内权限的策略。
- Amazon Virtual Private Cloud (Amazon VPC) 可帮助您将 AWS 资源启动到您定义的虚拟网络中。此虚拟网络类似于您在自己的数据中心内运行的传统网络，具有使用 AWS 可扩展基础设施的优势。

操作说明

应用 SCP

Task	描述	所需技能
创建测试管理员角色。	在目标 Amazon Web Services account 中，创建一个名为 `TestAdminRole` 的 IAM 角色。将 AdministratorAccesAWS 托管 IAM 策略附加到新角色。有关说明，请参阅 IAM 文档中的创建向 IAM 用户委派权限的角色。	AWS 管理员
创建自动化管理员角色。	1. 在目标 Amazon Web Services account 中，创建一个名为 `AutomationAdminRole` 的 IAM 角色。 2. 将 AdministratorAccesAWS 托管 IAM 策略附加到新角色。	AWS 管理员

Task	描述	所需技能
	以下是您可用于测试 `000000000000` 账户角色的信任策略示例。 `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "AWS": [ "arn:aws:iam::000000000000:root" ] }, "Action": "sts:AssumeRole", "Condition": {} } ] }`	
创建并附加 SCP。	1. 使用其他信息部分中提供的示例代码，创建安全控制策略。有关说明，请参阅 AWS Organizations 文档中的创建 SCP。 2. 将 SCP 附加到目标 Amazon Web Services account 或 OU。有关说明，请参阅 AWS Organizations 文档中的附加和分离服务控制策略。	AWS 管理员

测试 SCP

Task	描述	所需技能
创建 VPC 或子网。	1. 承担目标 Amazon Web Services account 中的 `TestAdminRole` 角色。 2. 尝试在现有 VPC 中创建 VPC 或新的公有子网。有关说明，请参阅 Amazon VPC 文档中的创建 VPC、子网和其他 VPC 资源。您不应能够创建这些资源。 3. 承担 `AutomationAdminRole` 角色，然后重试上一步。现在，您应该可以创建网络资源了。	AWS 管理员
管理标签。	1. 承担目标 Amazon Web Services account 中的 `TestAdminRole` 角色。 2. 将 `SubnetType:IFA` 标签添加到可用的公有子网。你应该可以添加此标签。有关如何通过 AWS 命令行界面 (AWS CLI) 添加标签的说明，请参阅 AWS CLI 命令参考中的 create-tags。 3. 在不更改凭证的情况下，尝试修改分配给此子网的 `SubnetType:IFA` 标签。您应该无法修改此标签。 4. 承担 `AutomationAdminRole` 角色，然后重试前面的步骤。此角色应能添加和修改此标签。	AWS 管理员

Task	描述	所需技能
在目标子网中部署资源。	1. 担任 `TestAdminRole` 角色。 2. 对于带有`SubnetType:IFA` 标签的公有子网，请尝试创建 EC2 实例。有关说明，请参阅 Amazon EC2 文档中的启动实例。在此子网中，除了 NAT 网关外，您不应能够创建、修改或删除任何 Amazon EC2 资源。 3. 在同一个子网中创建 NAT 网关。有关说明，请参阅 Amazon VPC 文档中的创建 NAT 网关。您应该能够创建、修改或删除此子网中的 NAT 网关。	AWS 管理员
管理 AutomationAdminRole 角色。	1. 担任 `TestAdminRole` 角色。 2. 尝试修改 `AutomationAdminRole` 角色。有关说明，请参阅 IAM 文档中的修改角色。您应该无法修改此角色。 3. 承担 `AutomationAdminRole` 角色，然后重试上一步。现在，您应该可以修改角色了。	AWS 管理员

清理

Task	描述	所需技能
清理已部署的资源。	1. 从 Amazon Web Services account 或 OU 中分离 SCP。有关说明，请参阅 AWS Organizations 文档中的分离 SCP。 2. 删除 SCP。有关说明，请参阅删除 SCP（AWS Organizations 文档）。 3. 删除 `AutomationAdminRole` 角色和 `TestAdminRole` 角色。有关说明，请参阅 IAM 文档中的删除角色。 4. 删除您为此解决方案创建的所有网络资源，例如 VPCs 和子网。	AWS 管理员

相关资源

AWS 文档

- 安装和拆卸 SCPs
- 创建、更新和删除 SCPs
- 使用 AWS Config 为公有子网部署基于侦探属性的访问控制
- 侦测性控制
- 服务授权参考
- 标记 AWS 资源
- 什么是适用于 AWS 的 ABAC？

其他 AWS 参考

- 使用 AWS Organizations 中的服务控制策略保护用于授权的资源标签（AWS Blog 文章）

其他信息

以下服务控制策略是一个示例，您可以用来在您的组织中测试这种方法。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyVPCActions",
      "Effect": "Deny",
      "Action": [
        "ec2:CreateVPC",
        "ec2:CreateRoute",
        "ec2:CreateSubnet",
        "ec2:CreateInternetGateway",
        "ec2:DeleteVPC",
        "ec2:DeleteRoute",
        "ec2:DeleteSubnet",
        "ec2:DeleteInternetGateway"
      ],
      "Resource": [
        "arn:aws:ec2:*:*:*"
      ],
      "Condition": {
        "StringNotLike": {
          "aws:PrincipalARN": ["arn:aws:iam::*:role/AutomationAdminRole"]
        }
      }
    },
    {
      "Sid": "AllowNATGWOnIFASubnet",
      "Effect": "Deny",
      "NotAction": [
        "ec2:CreateNatGateway",
        "ec2:DeleteNatGateway"
      ],
      "Resource": [
        "arn:aws:ec2:*:*:subnet/*"
      ],
      "Condition": {
        "ForAnyValue:StringEqualsIfExists": {
```

```
          "aws:ResourceTag/SubnetType": "IFA"
        },
        "StringNotLike": {
          "aws:PrincipalARN": ["arn:aws:iam::*:role/AutomationAdminRole"]
        }
      }
    },
    {
      "Sid": "DenyChangesToAdminRole",
      "Effect": "Deny",
      "NotAction": [
        "iam:GetContextKeysForPrincipalPolicy",
        "iam:GetRole",
        "iam:GetRolePolicy",
        "iam:ListAttachedRolePolicies",
        "iam:ListInstanceProfilesForRole",
        "iam:ListRolePolicies",
        "iam:ListRoleTags"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AutomationAdminRole"
      ],
      "Condition": {
        "StringNotLike": {
          "aws:PrincipalARN": ["arn:aws:iam::*:role/AutomationAdminRole"]
        }
      }
    },
    {
      "Sid": "allowbydefault",
      "Effect": "Allow",
      "Action": "*",
      "Resource": "*"
    }
  ]
}
```

使用 Terraform 为 AWS WAF 解决方案部署安全自动化

由 Rahul Sharad Gaikwad 博士 (AWS) 和 Tamilselvan P (AWS) 编写

摘要

AWS WAF 是一种 Web 应用程序防火墙，它通过使用可自定义的规则（您可以在 Web 访问控制列表中定义和部署）来帮助保护应用程序免受常见漏洞攻击（ACLs）。配置 AWS WAF 规则可能具有挑战性，特别是对于没有专门安全团队的组织而言。为了简化此过程，Amazon Web Services (AWS) 为 AWS WAF解决方案提供了安全自动化，该解决方案可自动部署具有一组过滤基于 Web 的攻击的 AWS WAF 规则的单个 Web ACL。Terraform 部署期间，您可以指定要纳入的保护功能。部署此解决方案后，会 AWS WAF 检查向现有 Amazon CloudFront 分配或应用程序负载均衡器发出的网络请求，并阻止任何不符合规则的请求。

可以按 AWS CloudFormation 照《安全自动化AWS WAF 实施指南》中的说明使用来部署 AWS WAF 解决方案的安全自动化。这种模式为使用 HashiCorp Terraform作为其首选基础设施即代码 (IaC) 工具来配置和管理其云基础架构的组织提供了另一种部署选项。部署此解决方案时，Terraform 会自动将更改应用到云端，并部署和配置 AWS WAF 设置和保护功能。

先决条件和限制

先决条件

- 活跃 AWS 账户的.
- AWS Command Line Interface (AWS CLI) 版本 2.4.25 或更高版本，已安装并配置了必要的权限。有关更多信息，请参阅入门（AWS CLI 文档）。
- 已安装并配置了 Terraform 版本 1.1.9 或更高版本。有关更多信息，请参阅安装 Terraform（Terraform 文档）。

架构

目标架构

此模式为 AWS WAF 解决方案部署安全自动化。有关目标架构的更多信息，请参阅《安全自动化 AWS WAF 实施指南》中的架构概述。有关此部署中的 AWS Lambda 自动化、应用程序日志解析器、日志解析器、IP 列表解析器和访问处理程序的更多信息，请参阅《安全自动化实施指南》中的组件详细信息。 AWS WAF AWS WAF

Terraform 部署

当您运行 `terraform apply` 时，Terraform 会执行以下操作：

1. Terraform 根据测试.tfvars 文件中的输入创建 AWS Identity and Access Management (IAM) 角色和 Lambda 函数。
2. Terraform 根据测试 .tfvars 文件中的输入创建 AWS WAF ACL 规则和 IP 集。
3. Terraform 根据测试.tfvars 文件中的输入创建了亚马逊简单存储服务 (Amazon S3) 存储桶 AWS Glue 、 EventBridge 亚马逊规则、数据库表和亚马逊雅典娜工作组。
4. Terraform 部署 AWS CloudFormation 堆栈来配置自定义资源。
5. Terraform 根据 testing.tfvars 文件中的指定输入创建 Amazon API Gateway 资源。

自动化和扩缩

您可以使用此模式为多个 AWS WAF 规则创建规则， AWS 账户 并在 AWS 区域 整个 AWS Cloud 环境中为 AWS WAF 解决方案部署安全自动化。

工具

AWS 服务

- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行外壳中的命令进行交互。
- AWS WAF是一种 Web 应用程序防火墙，可帮助您监控转发到受保护的 Web 应用程序资源的 HTTP 和 HTTPS 请求。

其他服务

- Git 是开源分布式版本控制系统。
- HashiCorp Terraform 是一款命令行界面应用程序，可帮助您使用代码来配置和管理云基础架构和资源。

代码存储库

此模式的代码可在 “使用 Terraform GitHub AWS WAF 实现自动化” 存储库中找到。

最佳实践

- 将静态文件放在单独的 Amazon S3 存储桶中。

- 避免对变量执行硬编码。
- 限制自定义脚本的使用。
- 采用副本命名约定。

操作说明

设置本地工作站

Task	描述	所需技能
安装 Git。	按照入门（Git 网站）中的说明在本地工作站上安装 Git。	DevOps 工程师
克隆存储库。	在您的本地工作站上，输入以下命令来克隆代码存储库： `git clone https://github.com/aws-samples/aws-waf-automation-terraform-samples.git`	DevOps 工程师
更新变量。	1. 通过输入以下命令导航到克隆的目录： `cd terraform-aws-waf-automation` 2. 在任何文本编辑器中，打开testing.tfvars文件。 3. 更新 testing.tfvars 文件中变量的值。 4. 保存并关闭文件。	DevOps 工程师

使用 Terraform 预置目标架构

Task	描述	所需技能
初始化 Terraform 配置。	输入以下命令来初始化包含 Terraform 配置文件的工作目录： `terraform init`	DevOps 工程师
预览 Terraform 计划。	输入以下命令。Terraform 会评估配置文件，以确定已声明资源的目标状态。然后，它将目标状态与当前状态进行比较并创建计划： `terraform plan -var-file ="testing.tfvars"`	DevOps 工程师
验证计划。	查看计划并确认它已在您的目标 AWS 账户中配置了所需的架构。	DevOps 工程师
部署解决方案。	1. 输入以下命令以应用计划： `terraform apply -var-file="testing .tfvars"` 2. 输入 `yes`以确认。Terraform 创建、更新或销毁基础设施，以实现配置文件中声明的目标状态。有关序列的更多信息，请参阅此模式的架构部分中的 Terraform 部署。	DevOps 工程师

验证并清理

Task	描述	所需技能
验证更改。	1. 在 Terraform 控制台中，验证输出是否与预期结果相符。 2. 登录到 AWS Management Console。 3. 验证 Terraform 控制台中的输出是否已成功部署到您的控制台中。 AWS 账户	DevOps 工程师
（可选）清理 基础架构。	如果想删除此解决方案所执行的所有资源和配置更改，请执行以下操作： 1. 在 Terraform 控制台中，输入以下命令： `terraform destroy -var-file="testing.tfvars”` 2. 输入 `yes`以确认。	DevOps 工程师

故障排除

事务	解决方案
`WAFV2 IPSet: WAFOptimisticLockException` 错误	如果运行 `terraform destroy` 命令时收到此错误，则必须手动删除 IP 集。有关说明，请参阅删除 IP 集（AWS WAF 文档）。

相关资源

AWS 参考文献

- 《安全自动化 AWS WAF 实施指南》
- AWS WAF（AWS 解决方案库）的安全自动化
- 常见问题解答的安全自动化 AWS WAF

Terraform 参考

- Terraform 后端配置
- Terraform P AWS rovider-文档和用法
- Terraform AWS 提供者（存储库）GitHub

检测已过期的 CA 证书的 Amazon RDS 和 Aurora 数据库实例

由 Stephen DiCato (AWS) 和 Eugene Shifer (AWS) 创作

摘要

作为安全最佳实践，建议您对应用程序服务器和关系数据库之间传输的数据进行加密。您可以使用 SSL 或 TLS 来加密与数据库 (DB) 实例或集群的连接。这些协议有助于在应用程序和数据库之间提供机密性、完整性和真实性。数据库使用服务器证书，该证书由证书颁发机构 (CA) 颁发，用于执行服务器身份验证。SSL 或 TLS 通过验证证书的数字签名并确保其未过期来验证证书的真实性。

在中 AWS Management Console，亚马逊关系数据库服务 (Amazon RDS) 和亚马逊 Aurora 提供有关需要更新证书的数据库实例的通知。但是，要查看这些通知，您必须登录每个通知 AWS 账户 并导航到每个通知中的服务控制台 AWS 区域。如果您需要评估许多作为组织管理的证书的有效性 AWS 账户 ，则此任务会变得更加复杂AWS Organizations。

通过配置此模式中提供的基础设施即代码 (IaC)，您可以检测您 AWS 账户 或 AWS 组织中所有 Amazon RDS 和 Aurora 数据库实例的 CA 证书即将过期。该AWS CloudFormation模板提供了 AWS Config 规则、 AWS Lambda 函数和必要的权限。您可以将其作为堆栈部署到单个账户中，也可以将其作为堆栈集部署到整个 AWS 组织中。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 如果你要部署到单个 AWS 账户：
  - 确保您拥有创建 CloudFormation 堆栈的权限。
  - AWS Config 在目标账户中@@ 启用。
  - （可选）AWS Security Hub 在目标账户中启用。
- 如果您要部署到 AWS 组织中：
  - 确保您拥有创建 CloudFormation 堆栈集的权限。
  - 通过 AWS Organizations 集成@@ 启用 Security Hub。
  - AWS Config 在部署此解决方案的账户中@@ 启用。
  - 指定一个作 AWS 账户 为 Security Hub AWS Config 的委托管理员。

限制

- 如果您要部署到未启用 Security Hub 的个人账户，则可以使用 AWS Config 来评估结果。
- 如果您要部署到的组织没有 AWS Config 和 Security Hub 的委托管理员，则必须登录个人成员账户才能查看调查结果。
- 如果您使用 AWS Control Tower 管理和治理组织中的账户，请使用定制 AWS Control Tower (cfc T) 按此模式部署 IaC。使用 CloudFormation 控制台会导致配置偏离 AWS Control Tower 防护栏，并要求您重新注册组织单位 (OUs) 或托管账户。
- 有些 AWS 服务 并非全部可用 AWS 区域。有关区域可用性，请参阅服务终端节点和配额页面，然后选择该服务的链接。

架构

部署到个人中 AWS 账户

以下架构图显示了 AWS 资源在单个架构中的部署 AWS 账户。它是通过直接通过 CloudFormation 控制台使用 CloudFormation 模板来实现的。如果启用了 Security Hub，则可以在 AWS Config 或 Security Hub 中查看结果。如果未启用 Security Hub，则只能在 AWS Config 控制台中查看结果。

图中显示以下步骤：

1. 你创建了一个 CloudFormation 堆栈。这将部署一个 Lambda 函数和 AWS Config 一条规则。规则和函数都设置了在中发布资源评估 AWS Config 和日志所需的 AWS Identity and Access Management (IAM) 权限。
2. 该 AWS Config 规则在侦探评估模式下运行，每 24 小时运行一次。
3. Security Hub 会收到所有 AWS Config 调查结果。
4. 您可以在 Security Hub 中或中查看调查结果 AWS Config，具体取决于账户的配置。

部署到 AWS 组织中

下图显示了通过 AWS Organizations 和管理的多个账户对证书到期情况的评估 AWS Control Tower。您可以通过 cfCT 部署 CloudFormation 模板。评估结果集中在 Security Hub 的委派管理员帐户中。图中描述 AWS CodePipeline 的工作流程显示了 cfCT 部署期间发生的背景步骤。

图中显示以下步骤：

1. 根据 cfCT 的配置，您可以在管理账户中将 iaC 推送到 AWS CodeCommit 存储库，或者将 iaC 的压缩 (ZIP) 文件上传到亚马逊简单存储服务 (Amazon S3) 存储桶。
2. cfcT 管道解压缩文件，运行 cfn-nag (GitHub) 检查，然后将其部署为堆栈集。 CloudFormation
3. 根据 cfcT 清单文件中指定的配置，将堆栈 CloudFormation StackSets 部署到单个账户或指定账户。 OUs这将在目标账户中部署一个 Lambda 函数和 AWS Config 一条规则。规则和函数均使用在中发布资源评估 AWS Config 和日志所需的 IAM 权限进行设置。
4. 该 AWS Config 规则在侦探评估模式下运行，每 24 小时运行一次。
5. AWS Config 将所有发现的结果转发给 Security Hub。
6. Security Hub 的发现结果汇总到委派的管理员帐户中。
7. 您可以使用委派的管理员帐户在 Security Hub 中查看调查结果。

工具

AWS 服务

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在资源的整个生命周期中跨地区对其 AWS 账户 进行管理。
- AWS Config提供了您的资源 AWS 账户 及其配置方式的详细视图。它可以帮助您确定资源之间的关联方式，以及它们的配置如何随时间变化。 AWS Config 规则定义了您理想的资源配置设置，并 AWS Config 可以评估您的 AWS 资源是否符合规则中的条件。
- AWS Control Tower按照规范性最佳实践，帮助您设置和管理 AWS 多账户环境。AWS Control Tower (cfcT) 的自定义可帮助您自定义 AWS Control Tower 着陆区并与 AWS 最佳实践保持一致。使用 CloudFormation 模板和服务控制策略（SCPs）实现自定义。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Organizations是一项账户管理服务，可帮助您将多个账户整合 AWS 账户 到一个由您创建和集中管理的组织中。
- AWS Security Hub提供了您的安全状态的全面视图 AWS。它还可以帮助您根据安全行业标准和最佳实践检查您的 AWS 环境。

其他工具

- Python 是通用的计算机编程语言。

代码存储库

此模式的代码可在 GitHub 检测 CA 证书即将过期的 Amazon RDS 实例存储库中找到。

最佳实践

我们建议您遵循以下资源中的最佳实践：

- 组织单位的最佳实践 AWS Organizations（AWS 云运营和迁移博客）
- 使用 AWS Control Tower on AWS（AWS 解决方案库）建立初始基础的指南
- 创建和修改 AWS Control Tower 资源的指南（AWS Control Tower 文档）
- cfCT 部署注意事项（AWS Control Tower 文档）

操作说明

查看解决方案和代码

Task	描述	所需技能
确定您的部署策略。	查看解决方案和代码，以确定如何将其部署到您的 AWS 环境中。确定您是要部署到单个账户还是 AWS 组织中。	应用程序所有者，常规 AWS
克隆存储库。	输入以下命令，使用即将过期的 CA 证书存储库克隆检测 Amazon RDS 实例。 `git clone https://github.com/aws-samples/config-rds-ca-expiry.git`	应用程序开发人员、应用程序所有者
验证 Python 版本。	1. 导航到克隆存储库中的顶级目录。 `cd config-rds-ca-expiry`	应用程序开发人员、应用程序所有者

Task	描述	所需技能
	2. 打开 config-rds-ca-expiry.yaml。 3. 在`CertExpirationCheckLambdaFunction` 资源中，确认 Python 版本与您的目标兼容 AWS 区域。默认情况下，此函数使用 Python 3.12。有关更多信息，请参阅AWS Lambda 添加对 Python 3.12 的支持。如有必要，请更新 Python 版本。 4. 保存并关闭 config-rds-ca-expiry.yaml。	

部署解决方案

Task	描述	所需技能
部署 CloudFormation 模板。	将 CloudFormation 模板部署到您的 AWS 环境中。请执行以下操作之一： • 如果您要部署到单个堆栈 AWS 账户，请按照创建堆栈中的说明进行操作。 • 如果您要部署到不受管理的组织 AWS Control Tower，请按照创建堆栈集中的说明进行操作。 • 如果您要部署到由管理的组织 AWS Control Tower，请	应用程序开发者、AWS 管理员、常规 AWS

Task	描述	所需技能
	参阅构建自己的自定义项中的说明。	
验证部署。	在CloudFormation 控制台中，验证堆栈或堆栈集是否已成功部署。	AWS 管理员、应用程序所有者

查看调查结果

Task	描述	所需技能
查看 AWS Config 规则调查结果。	在 Security Hub 中，执行以下操作以查看单个发现结果的列表： 1. 打开 S ecurity Hub 控制台。 2. 在导航窗格中，选择 调查发现。 3. 在 “添加筛选器” 框中，添加以下筛选器： • 合规状态为 FAILED • 标题是 rds-has-expiring-ca 4. 选择应用。 在 Security Hub 中，执行以下操作以查看按以下分组的全部发现结果列表 AWS 账户： 1. 打开 S ecurity Hub 控制台。 2. 在导航窗格中，选择 Insights。	AWS 管理员、AWS 系统管理员、云管理员

Task	描述	所需技能
	3. 选择 Create insight (创建见解)。 4. 要为见解选择分组属性，请执行以下操作： a. 选择搜索框以显示筛选选项。 b. 选择 Group by (分组依据)。 c. 选择 AwsAccountId。 d. 选择应用。 5. 在 “添加筛选器” 框中，添加以下筛选器： • 标题是 `rds-has-expiring-ca` • 合规状态为 `FAILED` 6. 选择 Create insight (创建见解)。 7. 输入洞察名称，然后选择创建洞察。 在中 AWS Config，要查看调查结果列表，请按照 AWS Config 文档中查看合规信息和评估结果中的说明进行操作。	

故障排除

事务	解决方案
CloudFormation 创建或删除堆栈集失败	部署后 AWS Control Tower ，它会强制执行必要的护栏，并控制 AWS Config 聚合器和规则。这包括防止通过 CloudFormation进行任何直接

事务	解决方案
	更改。要正确部署或删除此 CloudFormation 模板（包括所有相关资源），必须使用 cfcT。
cfcT 无法删除模板 CloudFormation	如果即使在清单文件中进行了必要的更改并删除了 CloudFormation 模板文件之后，模板仍然存在，请确认清单文件中是否包含`enable_stack_set_deletion` 参数并且该值已设置为`false`。有关更多信息，请参阅 cfCT 文档中的删除堆栈集。

相关资源

- 使用 SSL/TLS 加密与数据库实例或集群的连接（Amazon RDS 文档）
- AWS Config 自定义规则（AWS Config 文档）

使用 Step Functions 通过 IAM Access Analyzer 动态生成 IAM policy

由 Thomas Scott (AWS)、Adil El Kanabi (AWS)、Koen van Blijderveen (AWS) 和 Rafal Pawlaszek (AWS) 创建

摘要

注意： AWS CodeCommit 不再向新客户开放。的现有客户 AWS CodeCommit 可以继续照常使用该服务。了解更多。

最低权限是授予执行任务所需最低权限的安全最佳实践。在已经处于活动状态的 Amazon Web Services (AWS) 账户中实现最低权限访问可能具有挑战性，因为您不想通过更改用户权限来无意中阻止他们履行工作职责。在实施 AWS Identity and Access Management (IAM) 政策变更之前，您需要了解账户用户正在执行的操作和资源。

此模式旨在帮助您应用最低权限的访问原则，且不会阻碍或降低团队工作效率。它描述了如何使用 IAM Access Analyzer 以及 AWS Step Functions 如何根据账户中当前正在执行的操作为您的角色动态生成 up-to-date IAM 策略。新策略旨在允许当前活动，但会删除任何不必要的提升权限。您可通过定义允许和拒绝规则来自定义生成的策略，此解决方案集成了您的自定义规则。

此模式包括使用 AWS Cloud Development Kit (AWS CDK) 或 HashiCorp CDK for Terraform (CDKTF) 实现解决方案的选项。然后，您可以使用持续集成和持续交付 (CI/CD) 管道将新策略与角色关联。如果您有多账户架构，则可以在任何想要为角色生成更新的 IAM 策略的账户中部署此解决方案，从而提高整个 AWS Cloud 环境的安全性。

先决条件和限制

先决条件

- 已启用 AWS CloudTrail 追踪 AWS 账户 的活跃状态。
- 以下内容的 IAM 权限：
  - 创建并部署 Step Functions 工作流程。有关更多信息，请参阅 AWS Step Functions（Step Functions 文档）的操作、资源和条件键。
  - 创建 AWS Lambda 函数。有关更多信息，请参阅 执行角色和用户权限（Lambda 文档）。
  - 创建 IAM 角色。有关更多信息，请创建向 IAM 用户委派权限的角色（IAM 文档）。
- npm 已安装。有关更多信息，请参阅下载和安装 Node.js 和 npm（npm 文档）。

- 如果您使用 AWS CDK （选项 1）部署此解决方案：
  - AWS CDK 工具包，已安装和配置。有关更多信息，请参阅安装 AWS CDK（AWS CDK 文档）。
- 如果使用 CDKTF 部署此解决方案 (选项 2)：
  - CDKTF，已安装并配置。有关更多信息，请参阅安装适用于 Terraform 的 CDK（CDKTF 文档）。
  - Terraform，已安装并配置。有关更多信息，请参阅入门（Terraform 文档）。
- AWS Command Line Interface (AWS CLI) 已在本地安装并为您配置 AWS 账户。有关更多信息，请参阅安装或更新最新版本的 AWS CLI（AWS CLI 文档）。

限制

- 此模式不会将新 IAM policy 应用于 角色。在本解决方案结束时，新的 IAM 策略存储在存储 AWS CodeCommit 库中。您可以使用 CI/CD 管道将策略应用至您账户中的角色。

架构

目标架构

1. 定期安排的亚马逊 EventBridge 事件规则会启动 Step Functions 工作流程。在设置此解决方案的过程中，您可以定义此再生计划。
2. 在 Step Functions 工作流程中，Lambda 函数会生成日期范围，以便在分析日志中的账户活动时使用。 CloudTrail
3. 下一个工作流程步骤调用 IAM Access Analyzer API 开始生成策略。
4. IAM Access Analyzer 使用您在设置期间指定的角色的 Amazon 资源名称 (ARN)，分析 CloudTrail 日志中是否存在指定日期速率内的活动。根据活动，IAM Access Analyzer 生成仅允许该角色在指定日期范围内使用操作和服务的 IAM policy。完成此步骤后，此步骤将生成任务 ID。
5. 下一个工作流程步骤每 30 秒检查一次任务 ID。检测到任务 ID 后，此步骤将使用任务 ID 调用 IAM Access Analyzer API，并检索新的 IAM policy。IAM Access Analyzer 以 JSON 文件的形式返回策略。
6. 下一个工作流程步骤将 <IAM role name>/policy.json 文件置入 Amazon Simple Storage Service (Amazon S3) 存储桶。在设置此解决方案的过程中，您可以定义此 S3 存储桶。
7. Amazon S3 事件通知启动 Lambda 函数。

8. Lambda 函数从 S3 存储桶检索策略，集成您在 all ow.json 和 deny.json 文件中定义的自定义规则，然后将更新的策略推送到。 CodeCommit在设置此解决方案的过程中，您可以定义 CodeCommit 存储库、分支和文件夹路径。

工具

AWS 服务

- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，可帮助您在代码中定义和配置 AWS Cloud 基础架构。
- AWS CDK Toolkit 是一个命令行云开发套件，可帮助您与 AWS Cloud Development Kit (AWS CDK) 应用程序进行交互。
- AWS CloudTrail帮助您审计您的治理、合规和运营风险 AWS 账户。
- AWS CodeCommit是一项版本控制服务，可帮助您私下存储和管理 Git 存储库，而无需管理自己的源代码控制系统。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行外壳中的命令进行交互。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。此模式使用 IAM Access An alyzer（IAM 的一项功能）来分析您的 CloudTrail 日志，以识别 IAM 实体（用户或角色）已使用的操作和服务，然后生成基于该活动的 IAM 策略。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Storage Service (Amazon S3)是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Step Functions是一项无服务器编排服务，可帮助您组合 AWS Lambda 功能和其他功能 AWS 服务 来构建关键业务应用程序。在这种模式中，您可以使用 Step Functi ons 中的AWS SDK 服务集成从工作流程中调用服务 API 操作。

其他工具

- CDK for Terraform (CDKTF) 可帮助您使用常见的编程语言（例如 Python 和 Typescript）定义基础设施即代码（IaC）。
- Lerna 是一个构建系统，用于管理和发布来自同一存储库的多个 JavaScript 或多个 TypeScript 软件包。

- Node.js 是一个事件驱动的 JavaScript 运行时环境，专为构建可扩展的网络应用程序而设计。
- npm 是在 Node.js 环境中运行的软件注册表，用于共享或借用软件包以及管理私有软件包的部署。

代码存储库

此模式的代码可在 GitHub 自动 IAM Access Analyzer 角色策略生成器存储库中找到。

操作说明

准备部署

Task	描述	所需技能
克隆存储库。	以下命令克隆自动 IAM 访问分析器角色策略生成器 (GitHub) 存储库。 `git clone https://github.com/aws-samples/automated-iam-access-analyzer.git`	应用程序开发人员
安装 Lerna。	使用以下命令安装 Lerna。 `npm i -g lerna`	应用程序开发人员

设置依赖项。	使用以下命令安装存储库依赖项。
	`cd automated-iam-access-analyzer/`
	`npm install && npm run bootstrap`
		应用程序开发人员

构建代码。	使用以下命令测试、构建和准备 Lambda 函数的压缩包。 `npm run test:code`	应用程序开发人员

Task	描述	所需技能
	`npm run build:code` `npm run pack:code`	
构建构造。	以下命令为两者 AWS CDK 和 CDKTF 构建基础架构综合应用程序。 `npm run build:infra`	
配置任何自定义权限。	在已克隆存储库的 repo 文件夹中，编辑 allow.json 和 deny.json 文件，以定义该角色的任何自定义权限。如果 allow.json 和 deny.json 文件包含相同权限，则应用拒绝权限。	AWS 管理员、应用程序开发人员

选项 1-使用部署解决方案 AWS CDK

Task	描述	所需技能
部署 AWS CDK 堆栈。	以下命令通过 AWS CloudFormation部署基础架构。定义以下参数： • `<NAME_OF_ROLE>` – 您要为之创建新策略的 IAM 角色的 ARN。 • `<TRAIL_ARN>` — 存储角色活动的 CloudTrail 跟踪的 ARN。 • `<CRON_EXPRESSION_TO_RUN_SOLUTION>` – 定义策略重新生成计划的 Cron	应用程序开发人员

Task	描述	所需技能

	表达式。Step Functions 工作流程按该计划运行。
	• <TRAIL_LOOKBACK> – 评估角色权限时的回顾跟踪期（以天为单位）。
	
	`cd infra/cdk`
	`cdk deploy –-parameters roleArn=<NAME_OF_ROLE> \`
	`–-parameters trailArn=<TRAIL_ARN> \`
	`--parameters schedule=<CRON_EXPRESSION_TO_RUN_SOLUTION> \`
	`[ --parameters trailLookBack=<TRAIL_LOOKBACK> ]`
	
	Note
	方括号表示可选参数。	

（可选）等待新策略生效。	如果跟踪中不包含该角色的合理数量的历史活动，请等待，直到您确信有记录活动数量足以使 IAM Access Analyzer 生成准确的策略。如果该角色在账户中长期处于活动状态，则可能无需等待。	AWS 管理员
手动查看已生成策略。	在您的 CodeCommit 存储库中，查看生成的 .json <ROLE_ARN>文件以确认允许和拒绝权限适用于该角色。	AWS 管理员

选项 2– 通过 CDKTF 部署解决方案

Task	描述	所需技能
合成 Terraform 模板。	使用以下命令合成 Terraform 模板。 `lerna exec cdktf synth --scope @aiaa/tfm`	应用程序开发人员
部署 Terraform 模板。	使用以下命令导航至包含 CDKTF 定义的基础设施目录。 `cd infra/cdktf` 以下命令在目标 AWS 账户系统中部署基础架构。定义以下参数： • `<account_ID>` – 目标账户的 ID。 • `<region>`-目标 AWS 区域。 • `<selected_role_ARN>` – 您要为之创建新策略的 IAM 角色的 ARN。 • `<trail_ARN>` — 存储角色活动的 CloudTrail 跟踪的 ARN。 • `<schedule_expression>` – 定义策略重新生成计划的 Cron 表达式。Step Functions 工作流程按该计划运行。	应用程序开发人员

Task	描述	所需技能
	• `<trail_look_back>` – 评估角色权限时的回顾跟踪期（以天为单位）。 `TF_VAR_accountId=<account_ID> \ TF_VAR_region=<region> \ TF_VAR_roleArns=<selected_role_ARN> \ TF_VAR_trailArn=<trail_ARN> \ TF_VAR_schedule=<schedule_expression> \ [ TF_VAR_trailLookBack=<trail_look_back> ] \ cdktf deploy` **Note** 方括号表示可选参数。	
（可选）等待新策略生效。	如果跟踪中不包含该角色的合理数量的历史活动，请等待，直到您确信有记录活动数量足以使 IAM Access Analyzer 生成准确的策略。如果该角色在账户中长期处于活动状态，则可能无需等待。	AWS 管理员
手动查看已生成策略。	在您的 CodeCommit 存储库中，查看生成的 .json <ROLE_ARN>文件以确认允许和拒绝权限适用于该角色。	AWS 管理员

相关资源

AWS resources

- IAM Access Analyzer 端点和配额
- 正在配置 AWS CLI
- 开始使用 AWS CDK
- 最低权限许可

其他资源

- CDK for Terraform（Terraform 网站）

使用 AWS CloudFormation 模板 GuardDuty 有条件地启用 Amazon

创建者：Ram Kandaswamy (AWS)

摘要

您可以使用 AWS CloudFormation 模板 GuardDuty 在亚马逊 Web Services (AWS) 账户上启用亚马逊。默认情况下，如果您 GuardDuty 尝试使用 CloudFormation 开启堆栈时已启用，则堆栈部署将失败。但是，您可以使用 CloudFormation 模板中的条件来检查 GuardDuty 是否已启用。CloudFormation 支持使用比较静态值的条件；它不支持在同一模板中使用其他资源属性的输出。有关更多信息，请参阅 CloudFormation 文档中的条件。

在这种模式中， GuardDuty 如果尚未启用，则使用由 AWS Lambda 函数支持的 CloudFormation 自定义资源有条件地启用该资源。如果启 GuardDuty 用，堆栈将捕获状态并将其记录在堆栈的输出部分。如果 GuardDuty 未启用，则堆栈将其启用。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 有权创建、更新和删除 CloudFormation 堆栈的 AWS Identity and Access Management (IAM) 角色
- AWS Command Line Interface (AWS CLI)，已安装并配置

限制

如果 GuardDuty 已为 AWS 账户 或手动禁用此模式 AWS 区域，则不会 GuardDuty 为该目标账户或区域启用此模式。

架构

目标技术堆栈

该模式 CloudFormation 用于基础设施即代码 (IaC)。您可以使用由 Lambda 函数支持的 CloudFormation 自定义资源来实现动态服务启用功能。

目标架构

以下高级架构图显示了 GuardDuty 通过部署 CloudFormation 模板实现启用的过程：

1. 您可以部署 CloudFormation 模板来创建 CloudFormation 堆栈。
2. 堆栈创建 IAM 角色和 Lambda 函数。
3. Lambda 函数代入 IAM 角色。
4. 如果尚未 GuardDuty 在目标上启用 AWS 账户，则 Lambda 函数将其启用。

自动化和扩缩

您可以使用该 AWS CloudFormation StackSet 功能将此解决方案扩展到多个 AWS 账户 和 AWS 区域。有关更多信息，请参阅 CloudFormation 文档 AWS CloudFormation StackSets中的使用。

工具

- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行外壳中的命令进行交互。
- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在资源的整个生命周期中跨地区对其 AWS 账户 进行管理。
- Amazon GuardDuty 是一项持续的安全监控服务，可分析和处理日志，以识别您的 AWS 环境中意外和可能未经授权的活动。
- AWS Identity and Access Management (IAM) 通过控制谁经过身份验证并有权使用 AWS 资源，从而帮助您安全地管理对资源的访问权限。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。

操作说明

创建 CloudFormation 模板并部署堆栈

Task	描述	所需技能
创建 CloudFormation 模板。	1. 在 “其他信息” 部分的CloudFormation 模板中复制代码。 2. 将代码粘贴至文本编辑器中。	AWS DevOps

Task	描述	所需技能
	3. 在您的工作站上将文件另存为 `sample.yaml` 。	

创建 CloudFormation 堆栈。	1. 在中 AWS CLI，输入以下命令。这将使用该`sample.yaml` 文件创建一个新的 CloudFormation 堆栈。有关更多信息，请参阅 CloudFormation 文档中的创建堆栈。
	`aws cloudformation create-stack \ --stack-name guardduty-cf-stack \ --template-body file://sample.yaml`
	2. 确认以下值出现在中 AWS CLI，表示堆栈已成功创建。创建堆栈所需时间量可能会有所不同。
	`"StackStatus": "CREATE_COMPLETE",`
		AWS DevOps

验证 GuardDuty 是否已为启用 AWS 账户。	1. 登录 AWS Management Console 并打开GuardDuty 控制台。 2. 确认该 GuardDuty 服务已启用。	云管理员、AWS 管理员

Task	描述	所需技能
配置其他账户或区域。	根据您的用例需要，使用该 CloudFormation StackSet 功能将此解决方案扩展到多个 AWS 账户 和 AWS 区域。有关更多信息，请参阅 CloudFormation 文档 AWS CloudFormation StackSets中的使用。	云管理员、AWS 管理员

相关资源

参考

- AWS CloudFormation 文档
- AWS Lambda 资源类型参考
- CloudFormation 资源类型:: AWS: IAM:: 角色
- CloudFormation 资源类型:: AWS:GuardDuty: 探测器
- 使用 AWS CloudFormation（博客文章）检索任何 AWS 服务属性的四种方法

教程和视频

- 使用 AWS CloudFormation（教程）简化基础架构管理
- 使用亚马逊 GuardDuty 并 AWS Security Hub 保护多个账户（re AWS : Invent 2020）
- 创作最佳实践 AWS CloudFormation（re AWS : Inv ent 2019）
- 威胁检测开启 AWS：亚马逊简介 GuardDuty（r e AWS : inForce 2019）

其他信息

CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  rLambdaLogGroup:
    Type: 'AWS::Logs::LogGroup'
    DeletionPolicy: Delete
```

```
    Properties:
      RetentionInDays: 7
      LogGroupName: /aws/lambda/resource-checker
  rLambdaCheckerLambdaRole:
    Type: 'AWS::IAM::Role'
    Properties:
      RoleName: !Sub 'resource-checker-lambda-role-${AWS::Region}'
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          - Effect: Allow
            Principal:
              Service: lambda.amazonaws.com
            Action: 'sts:AssumeRole'
      Path: /
      Policies:
        - PolicyName: !Sub 'resource-checker-lambda-policy-${AWS::Region}'
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Sid: CreateLogGroup
                Effect: Allow
                Action:
                  - 'logs:CreateLogGroup'
                  - 'logs:CreateLogStream'
                  - 'logs:PutLogEvents'
                  - 'iam:CreateServiceLinkedRole'
                  - 'cloudformation:CreateStack'
                  - 'cloudformation:DeleteStack'
                  - 'cloudformation:Desc*'
                  - 'guardduty:CreateDetector'
                  - 'guardduty:ListDetectors'
                  - 'guardduty:DeleteDetector'
                Resource: '*'
  resourceCheckerLambda:
    Type: 'AWS::Lambda::Function'
    Properties:
      Description: Checks for resource type enabled and possibly name to exist
      FunctionName: resource-checker
      Handler: index.lambda_handler
      Role: !GetAtt
        - rLambdaCheckerLambdaRole
        - Arn
      Runtime: python3.13
```

```
      MemorySize: 128
      Timeout: 180
      Code:
        ZipFile: |
          import boto3
          import os
          import json
          from botocore.exceptions import ClientError
          import cfnresponse


          guardduty=boto3.client('guardduty')
          cfn=boto3.client('cloudformation')


          def lambda_handler(event, context):
              print('Event: ', event)
              if 'RequestType' in event:
                if event['RequestType'] in ["Create","Update"]:
                    enabled=False
                    try:
                      response=guardduty.list_detectors()
                      if "DetectorIds" in response and len(response["DetectorIds"])>0:
                        enabled="AlreadyEnabled"
                      elif "DetectorIds" in response and
len(response["DetectorIds"])==0:
                        cfn_response=cfn.create_stack(
                          StackName='guardduty-cfn-stack',
                          TemplateBody='{ "AWSTemplateFormatVersion": "2010-09-09",
"Description": "A sample template",    "Resources": { "IRWorkshopGuardDutyDetector": {
 "Type": "AWS::GuardDuty::Detector",    "Properties": {   "Enable": true  }   } } }'
                          )
                        enabled="True"
                    except Exception as e:
                        print("Exception: ",e)
                    responseData = {}
                    responseData['status'] = enabled
                    cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
"CustomResourcePhysicalID" )
                elif event['RequestType'] == "Delete":
                    cfn_response=cfn.delete_stack(
                            StackName='guardduty-cfn-stack')
                    cfnresponse.send(event, context, cfnresponse.SUCCESS, {})
 CheckResourceExist:
```

```
    Type: 'Custom::LambdaCustomResource'
    Properties:
      ServiceToken: !GetAtt
        - resourceCheckerLambda
        - Arn
Outputs:
  status:
    Value: !GetAtt
      - CheckResourceExist
      - status
```

Lambda 资源的替代代码选项

提供的 CloudFormation 模板使用内联代码来引用 Lambda 资源，以便于参考和指导。或者，您可以将 Lambda 代码放入亚马逊简单存储服务 (Amazon S3) 存储桶中，并在模板中引用它。 CloudFormation 内联代码不支持数据包依赖项或库。您可以通过将 Lambda 代码放入 Amazon S3 存储桶并在模板中引用它来支持这些操作。 CloudFormation

替换以下代码行：

```
Code:
         ZipFile: |
```

代码行如下：

```
Code:
      S3Bucket: <bucket name>
      S3Key: <python file name>
      S3ObjectVersion: <version>
```

如果您未在 Amazon S3 存储桶中使用版本控制，则可以省略该`S3ObjectVersion`属性。有关更多信息，请参阅 Amazon S3 文档中的在 Amazon S3 存储桶中使用版本控制。

在 Amazon RDS for SQL Server 中启用透明数据加密

由 Ranga Cherukuri (AWS) 编写

摘要

本示例介绍了如何在 Amazon Relational Database Service (Amazon RDS) 中实现透明数据加密 (TDE)，让 SQL Server 对静态数据进行加密。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- Amazon RDS for SQL Server 数据库实例

产品版本

Amazon RDS 当前支持在以下 SQL Server 版本中使用 TDE：

- SQL Server 2016 企业版
- SQL Server 2017 企业版
- SQL Server 2019 标准版和企业版
- SQL Server 2022 标准版和企业版

有关支持的版本和版本的最新信息，请参阅 Amazon RDS 文档中的 SQL Server 中对透明数据加密的支持。

架构

技术堆栈

- Amazon RDS for SQL Server

架构

工具

- Microsoft SQL Server Management Studio (SSMS) 是用于管理 SQL Server 基础结构的集成环境。它提供了用户界面和一组工具，其中包含与 SQL Server 交互的丰富脚本编辑器。

操作说明

在 Amazon RDS 控制台创建选项组

Task	描述	所需技能
打开 Amazon RDS 控制台。	登录 AWS 管理控制台 并打开 Amazon RDS 控制台。	开发人员、数据库管理员
创建选项组。	在导航窗格中，依次选择选项组和创建组。选择 sqlserver-ee 为数据库引擎，然后选择引擎版本。	开发人员、数据库管理员
添加透明数据加密选项。	编辑您创建的选项组并添加名为 `TRANSPARENT_DATA_ENCRYPTION` 的选项。	开发人员、数据库管理员

将选项组与数据库实例相关联

Task	描述	所需技能
选择数据库实例。	Amazon RDS 控制台的导航窗格，选择数据库，然后选择要与选项组关联的数据库实例。	开发人员、数据库管理员
将数据库实例与选项组相关联。	选择修改，然后使用选项组设置，将 SQL Server 数据库实例与您之前创建的选项组相关联。	开发人员、数据库管理员

Task	描述	所需技能
应用更改。	根据需要立即应用更改或在下一个维护时段内应用更改。	开发人员、数据库管理员

获取证书名称。	使用以下查询获取默认证书名称。
	`USE [master]`
	`GO`
	`SELECT name FROM sys.certificates WHERE name LIKE 'RDSTDECertificate%'`
	`GO`
		开发人员、数据库管理员

创建数据库加密密钥

Task	描述	所需技能
使用 SSMS 连接到 Amazon RDS for SQL Server 数据库实例	有关说明，请参阅 Microsoft 文档中的使用 SSMS。	开发人员、数据库管理员

通过默认证书创建数据库加密密钥。	通过之前获得的默认证书名称创建数据库加密密钥。通过以下 T-SQL 查询创建数据库加密密钥。您可指定 AES_256 算法而不是 AES_128。
	`USE [Databasename]`
	`GO`
	`CREATE DATABASE ENCRYPTION KEY`
	`WITH ALGORITHM = AES_128`
	`ENCRYPTION BY SERVER CERTIFICATE [certificatename]`
		开发人员、数据库管理员

Task	描述	所需技能
	`GO`	

在数据库启用加密。	通过以下 T-SQL 查询启用数据库加密。
	`ALTER DATABASE [Database Name]`
	`SET ENCRYPTION ON`
	`GO`
		开发人员、数据库管理员

检查加密状态。	使用以下 T-SQL 查询检查加密状态。 `SELECT DB_NAME(database_id) AS DatabaseName, encryption_state, percent_complete FROM sys.dm_database_encryption_keys`	开发人员、数据库管理员

相关资源

- SQL Server 中的透明数据加密支持（Amazon RDS 文档）
- 使用选项组（Amazon RDS 文档）
- 修改 Amazon RDS 数据库实例（Amazon RDS 文档）
- 适用于 SQL Server 的透明数据加密（Microsoft 文档）
- 使用 SSMS（Microsoft 文档）

确保 AWS 负载均衡器使用安全侦听器协议 (HTTPS、SSL/TLS)

由 Chandini Penmetsa (AWS) 和 Purushotham G K (AWS) 编写

摘要

在亚马逊网络服务 (AWS) 云上，Elastic Load Balancing 会自动将传入的应用程序流量分配到多个目标，例如亚马逊弹性计算云 (Amazon EC2) 实例、容器、IP 地址和 AWS Lambda 函数。负载均衡器使用侦听器定义负载均衡器用来接受来自用户的流量的端口和协议。应用程序负载均衡器在应用程序层做出路由决策并使用. HTTP/HTTPS protocols. Network Load Balancers make routing decisions at the transport layer and use the Transmission Control Protocol (TCP), Transport Layer Security (TLS), User Datagram Protcol (UDP), or TCP_UDP protocols. Classic Load Balancers make routing decisions at either the transport layer, using TCP or Secure Sockets Layer (SSL) protocols, or at the application layer, using HTTP/HTTPS

您的组织可能有安全或合规要求，即负载均衡器仅接受使用安全协议（例如 HTTPS 或 SSL/TLS）的用户的流量。

此模式提供了一种安全控制，它使用 Amazon EventBridge 规则来监控应用程序负载均衡`CreateListener`器和网络负载均衡器的和 `ModifyListener` API 调用，以及传统负载均衡器的`CreateLoadBalancerListeners`和 `CreateLoadBalancer` API 调用。如果将 HTTP、TCP/UDP 或 TCP_UDP 用于负载均衡器侦听器协议，则该控件将调用 Lambda 函数。Lambda 函数将消息发布到 Amazon Simple Notification Service (Amazon SNS) 主题，以发送包含负载均衡器详细信息的通知。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 接收违规通知的电子邮件地址
- 用于存储 Lambda 代码 .zip 文件的 Amazon Simple Storage Service（Amazon S3）存储桶。

限制

- 除非对负载均衡器侦听器进行了更新，否则此安全控制不会检查现有的负载均衡器。
- 此安全控制是区域性的，必须部署至您打算监控的 AWS 区域。

架构

目标技术堆栈

- Lambda 函数
- Amazon SNS 主题
- EventBridge 规则

目标架构

自动化和扩缩

- 如果您使用的是 AWS Organizations，则可以使用 AWS Cloudformation 将此模板部署 StackSets到您想要监控的多个账户中。

工具

- AWS CloudFormation — AWS CloudFormation 是一项通过使用基础设施即代码来帮助您建模和设置 AWS 资源的服务。
- 亚马逊 EventBridge — 亚马逊 EventBridge 提供来自您自己的应用程序、软件即服务 (SaaS) 应用程序和 AWS 服务的实时数据流，并将这些数据路由到 Lambda 函数等目标。
- AWS Lambda — Lambda 支持无需预置或管理服务器即可运行代码。
- Amazon S3 — Amazon Simple Storage Service (Amazon S3) 是一项高度可扩展的对象存储服务，可用于各种存储解决方案，包括网站、移动应用程序、备份和数据湖。
- Amazon SNS — Amazon Simple Notification Service (Amazon SNS) 可协调和管理发布者和客户端之间消息的传送或发送，包括 Web 服务器和电子邮件地址。订阅用户接收所有发布至他们所订阅主题的消息，并且一个主题的所有订阅用户收到的消息都相同。

最佳实践

确保所使用的 SNS 主题不公开访问。有关更多信息，请参阅 AWS 文档。

操作说明

上传 Lambda 代码

Task	描述	所需技能
定义 S3 存储桶。	在 Amazon S3 控制台上，选择或创建一个具有不包含前导斜杠的唯一名称的 S3 存储桶。S3 存储桶名称是全局唯一的，并且命名空间由所有 Amazon Web Services account 共享。您的 S3 存储桶必须与要评估的负载均衡器位于同一区域中。	云架构师
将 Lambda 代码上传至 S3 存储桶。	将“附件”部分中提供的 Lambda 代码.zip 文件上传到定义的 S3 存储桶。	云架构师
部署 AWS CloudFormation 模板。	在 AWS CloudFormation 控制台上，在与 S3 存储桶相同的 AWS 区域中，部署 “附件” 部分中提供的模板。在下一个操作说明中，提供参数的值。	云架构师

CloudFormation 参数

Task	描述	所需技能
命名 S3 存储桶。	输入您在第一个操作说明中创建的 S3 存储桶的名称。	云架构师
提供 Amazon S3 前缀。	提供 Lambda 代码 .zip 文件在 S3 存储桶中的位置，不带前导斜杠（例如，<director	云架构师

Task	描述	所需技能
	y>/<file-name>.zip ）。	
请提供 SNS 主题 ARN。	如果您想使用现有 SNS 主题发送违规通知，请提供 SNS 主题的 Amazon 资源名称（ARN）。要创建新的 SNS 主题，请将该值保留为 None（默认值）。	云架构师
提供电子邮箱地址。	提供有效的电子邮件地址以接收 Amazon SNS 通知。	云架构师
定义日志记录级别。	定义 Lambda 函数的日志记录级别和频率。Info 指明有关应用程序进度的详细信息消息。Error 指明仍允许应用程序继续运行的错误事件。Warning 指明潜在的有害情况。	云架构师

部署 CloudFormation 模板

Task	描述	所需技能
下载 模板。	下载 “附件” 部分中提供的 CloudFormation 模板。	云架构师
创建堆栈。	在与 S3 存储桶相同的区域中，导航到 CloudFormation 服务控制台，然后部署下载的模板。有关参数的详细信息，请参阅上一篇操作说明。	云架构师
验证资源。	堆栈创建完毕后，导航至资源选项卡，然后验证资源。模板将创建以下资源：	云架构师

Task	描述	所需技能
	• EventBridge 规则 • Lambda 函数 • Lambda 执行角色 • Lambda 调用权限	

确认订阅

Task	描述	所需技能
确认订阅。	成功部署模板后，如果创建新的 SNS 主题，则会向参数中提供的电子邮件地址发送订阅电子邮件。您必须确认此电子邮件订阅才能接收违规通知。	云架构师

故障排除

事务	解决方案
堆栈创建失败。错误发生在 GetObject。S3 错误代码: PermanentRedirect. S3 错误消息：存储桶位于此区域中：us-east-1。请使用此区域重试请求。	确保 S3 存储桶区域和堆栈部署区域相同。
堆栈创建失败。创建或更新 AWS Lambda 函数时不再支持 python 3.6 的运行时参数。	将第 186 行下载的模板从 Python 版本 3.6 更新至 3.9。

相关资源

- 在 AWS CloudFormation 控制台上创建堆栈
- AWS Lambda
- 什么是经典负载均衡器？

- 什么是应用程序负载均衡器？
- 什么是网络负载均衡器？
- 使用 AWS Lambda 函数的最佳实践
- AWS CloudFormation 最佳实践

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

确保在发布时启用 Amazon EMR 静态数据加密

由 Priyanka Chaudhary (AWS) 编写

摘要

此模式提供了一种安全控制，用于监控 Amazon Web Services (AWS) 上 Amazon EMR 集群的加密。

数据加密有助于防止未经授权的用户在集群和关联的数据存储系统中读取数据。这包括在网络传输时可能被截获的数据（称为传输中数据）和保存到持久性介质的数据（称为静态数据）。Amazon Simple Storage Service (Amazon S3) 中的静态数据可通过两种方式进行加密。

- 使用 Amazon S3 托管密钥进行服务器端加密 (SSE-S3)
- 使用 AWS Key Management Service (AWS KMS) 密钥进行服务器端加密 (SSE-KMS)，并使用适用于 Amazon EMR 的策略进行设置。

此安全控制可监控 API 调用，并在上RunJobFlow启动 Amazon Events CloudWatch 事件。触发器调用 AWS Lambda，后者则运行 Python 脚本。该函数从事件 JSON 输入中检索 EMR 集群 ID，并执行以下检查来确定是否存在安全违规。

1. 检查 EMR 集群是否与 Amazon EMR 特定安全配置相关联。
2. 如果特定于 Amazon EMR 的安全配置与 EMR 集群相关联，请检查是否 Encryption-at-Rest已开启。
3. 如果 Encryption-at-Rest未开启，请发送亚马逊简单通知服务 (Amazon SNS) Simple Notification 通知，其中包括 EMR 集群名称、违规详情、AWS 区域、AWS 账户以及此通知来源的 Lambda 亚马逊资源名称 (ARN)。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 存放 Lambda 代码 .zip 文件的 S3 存储桶
- 接收违规通知的电子邮件地址
- 关闭 Amazon EMR 日志记录，以便可以检索所有 API 日志

限制

- 此检测控制是区域性的，必须部署在您要监控的 AWS 区域中。

产品版本

- Amazon EMR 发行版 4.8.0 及以上版本

架构

目标技术堆栈

- Amazon EMR
- 亚马逊 CloudWatch 活动活动
- Lambda 函数
- Amazon SNS

目标架构

自动化和扩缩

如果您使用的是 AWS Organizations，则可以使用 AWS Cloudformation StackSets 在要监控的多个账户中部署此模板。

工具

工具

- AWS CloudFormation 是一项服务，可帮助您使用基础设施即代码建模和设置 AWS 资源。
- Amazon CloudWatch Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。
- Amazon EMR 是一个托管集群平台，可简化大数据框架的运行。
- AWS Lambda 支持无需预置或管理服务器即可运行代码。
- Amazon S3 是一项高度可扩展的对象存储服务，可用于各种存储解决方案，包括网站、移动应用程序、备份和数据湖。

- Amazon SNS 可协调和管理发布者和客户端之间消息的传送或发送，包括 Web 服务器和电子邮件地址。订阅用户接收所有发布至他们所订阅主题的消息，并且一个主题的所有订阅用户收到的消息都相同。

代码

- 此项目的 EMREncryption AtRest .zip 和 EMREncryption AtRest .yml 文件作为附件提供。

操作说明

删除 S3 存储桶

Task	描述	所需技能
定义 S3 存储桶。	在 Amazon S3 控制台上，选择或创建一个具有不包含前导斜杠的唯一名称的 S3 存储桶。S3 存储桶名称是全局唯一的，并且命名空间由所有 Amazon Web Services account 共享。您的 S3 存储桶需要与正在评估的 Amazon EMR 集群位于同一区域。	云架构师

将 Lambda 代码上传至 S3 存储桶

Task	描述	所需技能
将 Lambda 代码上传至 S3 存储桶。	将“附件”部分中提供的 Lambda 代码.zip 文件上传到定义的 S3 存储桶。	云架构师

部署 AWS CloudFormation 模板

Task	描述	所需技能
部署 AWS CloudFormation 模板。	在 AWS CloudFormation 控制台上，在与 S3 存储桶相同的区域中，部署作为该模式附件提供的 AWS CloudFormation 模板。在下一个操作说明中，提供参数的值。有关部署 AWS CloudFormation 模板的更多信息，请参阅 “相关资源” 部分。	云架构师

填写 AWS CloudFormation 模板中的参数

Task	描述	所需技能
命名 S3 存储桶。	输入您在第一个操作说明中创建的 S3 存储桶的名称。	云架构师
提供 Amazon S3 密钥。	提供 Lambda 代码 .zip 文件在 S3 存储桶中的位置，不带前导斜杠（例如，<directory>/<file-name>.zip）。	云架构师
提供电子邮箱地址。	提供有效的电子邮件地址以接收 Amazon SNS 通知。	云架构师
定义日志记录级别。	定义 Lambda 函数的日志记录级别与频率。“信息”表示有关应用程序进度的详细信息消息。“错误”表示仍可能允许应用程序继续运行的错误事件。“警告”表示潜在的有害情况。	云架构师

确认订阅

Task	描述	所需技能
确认订阅。	成功部署模板后，它将向提供的电子邮件地址发送订阅电子邮件。您必须确认此电子邮件订阅才能接收违规通知。	云架构师

相关资源

- 在 AWS CloudFormation 控制台上创建堆栈
- AWS Lambda
- Amazon EMR 加密选项

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

确保 IAM 配置文件与 EC2 实例关联

由 Mansi Suratwala (AWS) 编写

摘要

此模式提供了一个 AWS CloudFormation 安全控制模板，当亚马逊弹性计算云 (Amazon) 实例发生 AWS 身份和访问管理 (IAM) 个人资料违规时，该模板可以设置自动通知。 EC2

实例配置文件是 IAM 角色的容器，您可以使用该容器在 EC2 实例启动时将角色信息传递给实例。

当 A CloudWatch WS 根据`RunInstances`、`AssociateIamInstanceProfile`和`ReplaceIamInstanceProfileAssociation`作 CloudTrail 记录亚马逊 EC2 API 调用时，Amazon Events 会启动此检查。触发器调用 AWS Lambda 函数，该函数使用亚马逊 CloudWatch 事件来检查 IAM 配置文件。

如果 IAM 配置文件不存在，Lambda 函数将启动 Amazon Simple Notification Service (Amazon SNS) 电子邮件通知，其中包含 Amazon Web Services (AWS) 账户 ID 和 AWS 区域。

如果 IAM 配置文件确实存在，Lambda 函数将检查策略文档中是否有任何通配符条目。如果通配符条目存在，则启动 Amazon SNS 违规通知，这有助于您实现增强的安全性。该通知包含 IAM 配置文件的名称、事件、 EC2 实例 ID、托管策略的名称、违规行为、账户 ID 和区域。

先决条件和限制

先决条件

- 一个有效账户
- Lambda 代码 .zip 文件的 Amazon Simple Storage Service （ Amazon S3 ）存储桶

限制

- 必须仅为`RunInstances`、`AssociateIamInstanceProfile`和`ReplaceIamInstanceProfileAssociation`操作部署 AWS CloudFormation 模板。
- 安全控制不会监控 IAM 配置文件的分离。
- 安全控制不检查附加到 EC2 实例 IAM 配置文件的 IAM 策略是否有修改。
- 安全控制不考虑需要使用 `"Resource":*` 的不受支持的资源级权限。

架构

目标技术堆栈

- Amazon EC2
- AWS CloudTrail
- Amazon CloudWatch
- AWS Lambda
- Amazon S3
- Amazon SNS

目标架构

自动化和扩缩

您可以针对不同的 AWS 区域和账户多次使用 AWS CloudFormation 模板。您只需为每个账户或区域启动一次模板。

工具

工具

- 亚马逊 EC2 — Amazon 在 AWS 云中 EC2 提供可扩展的计算容量（虚拟服务器）。
- AWS CloudTrail — AWS CloudTrail 可帮助您对 AWS 账户进行治理、合规以及运营和风险审计。用户、角色或 AWS 服务采取的操作在中记录为事件 CloudTrail。
- Amazon CloudWatch Events — Amazon CloudWatch Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。
- AWS Lambda — AWS Lambda 是一项计算服务，您可用来运行代码，无需预置或管理服务器。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。
- Amazon S3 — Amazon S3 提供高度可扩展的对象存储，可用于各种存储解决方案，包括网站、移动应用程序、备份和数据湖。
- Amazon SNS — 通过 Amazon SNS，应用程序和设备可以从云中发送和接收通知。

代码

- 该项目的 .zip 文件作为附件提供。

操作说明

删除 S3 存储桶

Task	描述	所需技能
定义 S3 存储桶。	要托管 Lambda 代码 .zip 文件，请选择或创建一个具有不包含前导斜杠的唯一名称的 S3 存储桶。S3 存储桶名称是全局唯一的，并且命名空间由所有 Amazon Web Services account 共享。您的 S3 存储桶必须与正在评估的 EC2 实例位于同一区域。	云架构师

将 Lambda 代码上传至 S3 存储桶

Task	描述	所需技能
将 Lambda 代码上传至 S3 存储桶。	将附件部分中提供的 Lambda 代码上传到 S3 存储桶。S3 存储桶必须与正在评估的 EC2 实例位于同一区域。	云架构师

部署 AWS CloudFormation 模板

Task	描述	所需技能
部署 AWS CloudFormation 模板。	部署作为该模式附件提供的 AWS CloudFormation 模板。在下一个操作说明中，提供参数的值。	云架构师

填写 AWS CloudFormation 模板中的参数

Task	描述	所需技能
命名 S3 存储桶。	输入您在第一个操作说明中创建的 S3 存储桶的名称。	云架构师
提供 S3 密钥。	提供 Lambda 代码 .zip 文件在 S3 存储桶中的位置，不带前导斜杠（例如，`<directory>/<file-name>.zip`）。	云架构师
提供电子邮箱地址。	提供有效的电子邮件地址以接收 Amazon SNS 通知。	云架构师
定义日志记录级别。	定义 Lambda 函数的日志记录级别和频率。`Info` 指明有关应用程序进度的详细信息消息。`Error` 指明仍允许应用程序继续运行的错误事件。`Warning` 指明潜在的有害情况。	云架构师

确认订阅

Task	描述	所需技能
确认订阅。	成功部署模板后，它将向提供的电子邮件地址发送订阅电子邮件。您必须确认此电子邮件订阅才能接收违规通知。	云架构师

相关资源

- 创建 S3 存储桶
- 将文件上传到 S3 存储桶

- 使用实例配置文件
- 使用 AWS 创建在 AWS API 调用时触发 CloudWatch 的事件规则 CloudTrail

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

确保 Amazon Redshift 集群在创建时已加密

由 Mansi Suratwala (AWS) 编写

摘要

此模式提供了一个 AWS CloudFormation 模板，当创建未加密的新 Amazon Redshift 集群时，该模板会自动通知您。

AWS CloudFormation 模板创建了一个亚马逊活动 CloudWatch 事件和一个 AWS Lambda 函数。该事件监视任何正在创建或通过 AWS 从快照中恢复的 Amazon Redshift 集群。 CloudTrail如果创建集群时未使用 AWS Key Management Service (AWS KMS) 或云硬件安全模型 (HSM) 加密，则 CloudWatch 会启动 Lambda 函数，向您发送亚马逊简单通知服务 (Amazon SNS) Simple Notification 通知，告知您违规行为。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 包含集群子网组和关联安全组的虚拟私有云（VPC）。

限制

- 只能为`CreateCluster`和`RestoreFromClusterSnapshot`操作部署 AWS CloudFormation 模板。

架构

目标技术堆栈

- Amazon Redshift
- AWS CloudTrail
- Amazon CloudWatch
- AWS Lambda
- Amazon Simple Storage Service (Amazon S3)
- Amazon SNS

目标架构

自动化和扩缩

您可以针对不同的 AWS 区域和账户多次使用 AWS CloudFormation 模板。您只需在每个区域或账户中运行一次。

工具

工具

- Amazon Redshift — Amazon Redshift 是一种完全托管的 PB 级云中数据仓库服务。Amazon Redshift 与数据湖集成，让您可以使用数据获得对您的业务和客户的新见解。
- AWS CloudTrail — AWS CloudTrail 是一项 AWS 服务，可帮助您对 AWS 账户实施治理、合规以及运营和风险审计。用户、角色或 AWS 服务采取的操作在中记录为事件 CloudTrail。
- Amazon CloudWatch Events — Amazon CloudWatch Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。
- AWS Lambda — AWS Lambda 支持无需预置或管理服务器即可运行代码。只有在需要时 AWS Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。
- Amazon S3 — Amazon S3 是一项高度可扩展的对象存储服务，可用于各种存储解决方案，包括网站、移动应用程序、备份和数据湖。
- Amazon SNS — Amazon SNS 是一项 Web 服务，可协调和管理发布者和客户端之间消息的传送或发送，包括 Web 服务器和电子邮件地址。

代码

- 该项目的 .zip 文件作为附件提供。

操作说明

删除 S3 存储桶

Task	描述	所需技能
定义 S3 存储桶。	在 Amazon S3 控制台中，选择或创建 S3 存储桶。此 S3 存	云架构师

Task	描述	所需技能
	储桶将托管 Lambda 代码 .zip 文件。您的 S3 存储桶需要与正在评估的 Amazon Redshift 集群位于同一区域。S3 存储桶名称不得包含前导斜杠。	

将 Lambda 代码上传至 S3 存储桶

Task	描述	所需技能
将 Lambda 代码上传至 S3 存储桶。	将“附件”部分中提供的 Lambda 代码上传到 S3 存储桶。S3 存储桶需要与正在评估的 Amazon Redshift 集群位于同一区域。	云架构师

部署 AWS CloudFormation 模板

Task	描述	所需技能
部署 AWS CloudFormation 模板。	部署作为该模式附件提供的 AWS CloudFormation 模板。在下一个操作说明中，提供参数的值。	云架构师

完成 AWS CloudFormation 模板中的参数

Task	描述	所需技能
命名 S3 存储桶。	输入您在第一个操作说明中创建的 S3 存储桶的名称。	云架构师

Task	描述	所需技能
提供 S3 密钥。	提供 Lambda 代码 .zip 文件在 S3 存储桶中的位置，不带前导斜杠（例如，<directory>/<file-name>.zip）。	云架构师
提供电子邮箱地址。	提供有效的电子邮件地址以接收 Amazon SNS 通知。	云架构师
定义日志记录级别。	定义 Lambda 函数的日志记录级别和频率。Info 指明有关应用程序进度的详细信息消息。Error 指明仍允许应用程序继续运行的错误事件。Warning 指明潜在的有害情况。	云架构师

确认订阅

Task	描述	所需技能
确认订阅。	成功部署模板后，它将向提供的电子邮件地址发送订阅电子邮件。您必须确认此电子邮件订阅才能接收违规通知。	云架构师

相关资源

- 创建 S3 存储桶
- 将文件上传到 S3 存储桶
- 使用 AWS 创建在 AWS API 调用时触发 CloudWatch 的事件规则 CloudTrail
- 创建 Amazon Redshift 集群

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用导出 AWS IAM 身份中心身份及其分配的报告 PowerShell

由 Jorge Pava (AWS)、Chad Miles (AWS)、Frank Allotta (AWS) 以及 Manideep Reddy Gillela (AWS) 编写

摘要

当您使用 AWS IAM Identity Center（AWS Single Sign-On 的后续版本）集中管理对所有 Amazon Web Services (AWS) 账户和云应用程序的单点登录（SSO）访问时，通过 AWS 管理控制台报告和审核这些分配可能既乏味又耗时。如果您要报告数十个或数百个 Amazon Web Services account 中的用户或组的权限，则尤其如此。

对于许多人来说，查看此信息的理想工具是电子表格应用程序，例如 Microsoft Excel。这可以帮助您筛选、搜索和观察由 AWS Organizations 管理的整个组织的数据。

此模式描述了如何使用 AWS 工具在 IAM 身份中心生成 SSO 身份配置报告。 PowerShell 该报告的格式为 CSV 文件，包括身份名称（主体）、身份类型（用户或群组）、该身份可以访问的账户以及权限集。生成此报告后，您可在首选应用程序中将其打开，以便根据需要搜索、筛选和审核数据。下图显示了电子表格应用程序中的示例数据。

> **Important**
>
> 由于此报告包含敏感信息，因此我们强烈建议您将其安全存储并仅在 need-to-know基础上共享。

先决条件和限制

先决条件

- 已配置并启用 IAM Identity Center 和 AWS Organizations。
- PowerShell，已安装并配置。有关更多信息，请参阅安装 PowerShell（微软文档）。
- 已安装并配置 PowerShell的 AWS 工具。出于性能考虑，我们强烈建议您安装名`AWS.Tools`为的 AWS 工具的 PowerShell模块化版本。每个 Amazon Web Service 都由其自己的小模块提供支持。在 PowerShell shell 中，输入以下命令来安装此模式所需的模块：`AWS.Tools.InstallerOrganizations`、`SSOAdmin`、和`IdentityStore`。

```
Install-Module AWS.Tools.Installer
Install-AWSToolsModule -Name Organizations, SSOAdmin, IdentityStore
```

有关更多信息，请参阅在 Windows 上安装 AWS.Tools 或在 Linux 或 macOS 上安装 AWS.Tools（文档的 AWS 工具）。 PowerShell 如果您在安装模块时收到错误，请参阅此模式的故障排除部分。

- AWS 命令行界面（AWS CLI）或 AWS 开发工具包必须事先通过以下任一操作配置有效凭证：
  - 使用 AWS CLI `aws configure`。有关更多信息，请参阅快速配置（AWS CLI 文档）。
  - 配置 AWS CLI 或 AWS Cloud Development Kit (AWS CDK)，以通过 AWS Identity and Access Management (IAM) 角色获得临时访问。有关更多信息，请参阅获取用于 CLI 访问的 IAM 角色凭证（IAM Identity Center 文档）。
- AWS CLI 的命名配置文件，其中保存了 IAM 主体的证书，该主体具有以下特征：
  - 有权访问 AWS Organizations 管理账户或 IAM Identity Center 的委派管理员账户
  - `AWSSSOReadOnly` 和 `AWSSSODirectoryReadOnly` AWS 托管策略是否已应用于此

  有关更多信息，请参阅使用命名配置文件(AWS CLI 文档)和 AWS 托管策略（IAM 文档）。

限制

- 目标 Amazon Web Services account 必须在 AWS Organizations 中作为一个组织进行管理。

产品版本

- 对于所有操作系统，建议您使用 7.0 或更高PowerShell 版本。

架构

目标架构

1. 用户在 PowerShell 命令行中运行脚本。
2. 该脚本采用 AWS CLI 命名配置文件。这授予对 IAM Identity Center
3. 该脚本从 IAM Identity Center 检索 SSO 身份配置。
4. 该脚本会在本地工作站上保存脚本的同一目录中生成一个 CSV 文件。

工具

Amazon Web Services

- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- AWS IAM Identity Center 可帮助您集中管理对所有 Amazon Web Services account 和云应用程序的单点登录（SSO）访问权限。
- AWS 工具 PowerShell是一组 PowerShell 模块，可帮助您通过 PowerShell 命令行编写对 AWS 资源的操作的脚本。

其他工具

- PowerShell是一款在 Windows、Linux 和 macOS 上运行的微软自动化和配置管理程序。

操作说明

生成报告。

Task	描述	所需技能
准备脚本	1. 复制此模式的“其他信息”部分中的 PowerShell 脚本。 2. 在 `Param` 部分中，针对您的 AWS 环境，定义以下变量的值： • `OutputFile` - 报告的名称。 • `ProfileName` — 您要用来生成报告的 AWS CLI 命名配置文件。 • `Region` — 部署了 IAM Identity Center 的 AWS 区域。有关区域及其代码的完整列表，请参阅区域端点。	云管理员

Task	描述	所需技能
	3. 使用文件名 SSO-Report.ps1 保存该文件。	

运行 脚本。	建议您使用以下命令在 PowerShell shell 中运行您的自定义脚本。
	`.\SSO-Report.ps1`
	或者，您可以通过输入以下命令从其他 Shell 运行脚本。
	`pwsh .\SSO-Report.ps1`
	该脚本在脚本文件所在目录中生成一个 CSV 文件。
		云管理员

分析报告数据。	输出的 CSV 文件包含标题AccountNamePermissionSet、主文件和类型。在首选电子表格应用程序打开此文件。您可创建数据表来筛选和排序输出。	云管理员

故障排除

事务	解决方案
The term 'Get-<parameter>' is not recognized as the name of a cmdlet, function, script file, or operable program. 错误	未安装适用于 PowerShell 或其模块的 AWS 工具。在 PowerShell shell 中，输入以下命令来安装 AWS 工具 PowerShell 以及该模式所需的模块：AWS.Tools.Installer Organizations 、SSOAdmin、和IdentityStore 。 `Install-Module AWS.Tools.Installer`

事务	解决方案
	`Install-AWSToolsModule -Name Organizations, SSOAdmin, IdentityStore`
`No credentials specified or obtained from persisted/shell defaults` 错误	在操作说明???部分的准备脚本中，确认您已正确输入 `ProfileName` 和 `Region` 变量。确保指定配置文件中的设置和凭证具有足够的权限来管理 IAM Identity Center。
安装 AWS.Tools 模块时出错 `Authenticode Issuer …`	向 `Install-AWSToolsModule` 命令添加 `-SkipPublisherCheck` 参数。

`Get-ORGAccountList : Assembly AWSSDK.SSO could not be found or loaded.` 错误	当指定指定的 AWS CLI 配置文件、AWS CLI 配置为使用 IAM Identity Center 对用户进行身份验证并且 AWS CLI 配置为自动检索刷新的身份验证令牌时，可能会发生此错误。要纠正这个错误，可以执行下列操作：
	1. 输入以下命令以确认 `SSO` 和 `SSOOIDC` 模块已安装。
	`Install-AWSToolsModule SSO, SSOOIDC`
	2. 将以下行插入到 `param()` 块下方的脚本中。
	`Import-Module AWS.Tools.SSO`
	`Import-Module AWS.Tools.SSOOIDC`

相关资源

- 配置设置存储在何处？（AWS CLI 文档）
- 配置 AWS CLI 以使用 AWS IAM Identity Center（AWS CLI 文档）
- 使用命名配置文件（AWS CLI 文档）

其他信息

在以下脚本中，确定是否需要更新以下参数值：

- 如果您在 AWS CLI 中使用命名个人资料访问配置了 IAM Identity Center 的账户，请更新 `$ProfileName` 值。
- 如果 IAM 身份中心部署在与您的 AWS CLI 或 AWS 开发工具包配置的默认区域不同的 AWS 区域，请更新 `$Region` 值以使用部署 IAM 身份中心的区域。
- 如果这两种情况都不适用，则不需要更新脚本。

```
param (
    # The name of the output CSV file
    [String] $OutputFile  = "SSO-Assignments.csv",
    # The AWS CLI named profile
    [String] $ProfileName = "",
    # The AWS Region in which IAM Identity Center is configured
    [String] $Region      = ""
)
$Start = Get-Date; $OrgParams = @{}
If ($Region){ $OrgParams.Region = $Region}
if ($ProfileName){$OrgParams.ProfileName = $ProfileName}
$SSOParams   = $OrgParams.Clone(); $IdsParams = $OrgParams.Clone()
$AccountList = Get-ORGAccountList @OrgParams | Select-Object Id, Name
$SSOinstance = Get-SSOADMNInstanceList @OrgParams
$SSOParams['InstanceArn']       = $SSOinstance.InstanceArn
$IdsParams['IdentityStoreId']   = $SSOinstance.IdentityStoreId
$PSsets       = @{}; $Principals   = @{}
$Assignments  = @(); $AccountCount = 1; Write-Host ""
foreach ($Account in $AccountList) {
    $Duration = New-Timespan -Start $Start -End (Get-Date) | ForEach-Object
 {[Timespan]::New($_.Days, $_.Hours, $_.Minutes, $_.Seconds)}
    Write-Host "`r$Duration - Account $AccountCount of $($AccountList.Count)
 (Assignments:$($Assignments.Count))        " -NoNewline
    $AccountCount++
    foreach ($PS in Get-SSOADMNPermissionSetsProvisionedToAccountList -AccountId
 $Account.Id @SSOParams) {
        if (-not $PSsets[$PS]) {$PSsets[$PS] = (Get-SSOADMNPermissionSet @SSOParams -
PermissionSetArn $PS).Name;$APICalls++}
        $AssignmentsResponse = Get-SSOADMNAccountAssignmentList @SSOParams -
PermissionSetArn $PS -AccountId $Account.Id
```

```
        if ($AssignmentsResponse.NextToken) {$AccountAssignments =
 $AssignmentsResponse.AccountAssignments}
        else {$AccountAssignments = $AssignmentsResponse}
        While ($AssignmentsResponse.NextToken) {
            $AssignmentsResponse = Get-SSOADMNAccountAssignmentList @SSOParams -
PermissionSetArn $PS -AccountId $Account.Id -NextToken $AssignmentsResponse.NextToken
            $AccountAssignments += $AssignmentsResponse.AccountAssignments}
        foreach ($Assignment in $AccountAssignments) {
            if (-not $Principals[$Assignment.PrincipalId]) {
                $AssignmentType = $Assignment.PrincipalType.Value
                $Expression     = "Get-IDS"+$AssignmentType+" @IdsParams -"+
$AssignmentType+"Id "+$Assignment.PrincipalId
                $Principal      = Invoke-Expression $Expression
                if ($Assignment.PrincipalType.Value -eq "GROUP")
 { $Principals[$Assignment.PrincipalId] = $Principal.DisplayName }
                else { $Principals[$Assignment.PrincipalId] = $Principal.UserName }
            }
            $Assignments += [PSCustomObject]@{
                AccountName     = $Account.Name
                PermissionSet   = $PSsets[$PS]
                Principal       = $Principals[$Assignment.PrincipalId]
                Type            = $Assignment.PrincipalType.Value}
        }
    }
}
$Duration = New-Timespan -Start $Start -End (Get-Date) | ForEach-Object
 {[Timespan]::New($_.Days, $_.Hours, $_.Minutes, $_.Seconds)}
Write-Host "`r$($AccountList.Count) accounts done in $Duration. Outputting result to
 $OutputFile"
$Assignments | Sort-Object Account | Export-CSV -Path $OutputFile -Force
```

监控和修复 AWS KMS 密钥的计划删除

由 Mikesh Khanal (AWS) 和 Ramya Pulipaka (AWS) 创建

摘要

在 Amazon Web Services (AWS) 云上，删除 AWS Key Management Service (AWS KMS) 密钥可能会导致数据丢失。这将删除密钥材料以及与 AWS KMS 密钥关联的所有元数据，并且不可撤销。删除 AWS KMS 密钥后，您不能再解密用 AWS KMS 密钥加密的数据，这意味着该数据将无法恢复。

此模式设置监控，并在应用程序或用户计划删除 AWS KMS 密钥时发出通知。如果您收到此通知，则可能要取消删除该 AWS KMS 密钥并重新考虑删除它的决定。该模式使用 AWS Systems Manager 自动化运行手册AWSConfigRemediation-CancelKeyDeletion来简化取消删除 AWS KMS 密钥的操作。

> **Note**
>
> 该模式的 CloudFormation 模板必须部署在您想要监控 AWS KMS 密钥删除情况的所有 AWS 区域。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 了解以下 Amazon Web Services：
  - Amazon EventBridge
  - AWS KMS
  - Amazon Simple Notification Service (Amazon SNS)
  - AWS Systems Manager

限制

- 要对解决方案进行任何定制，都需要了解 AWS CloudFormation 模板和该模式中使用的 AWS 服务。
- 目前，该方案使用默认的事件总线，可以根据需求进行定制。有关自定义事件总线的更多信息，请参阅 AWS 文档。

架构

目标技术堆栈

- Amazon EventBridge
- AWS KMS
- Amazon SNS
- AWS Systems Manager
- 使用以下方法实现自动化：
  - AWS 命令行界面（AWS CLI）或 AWS SDK
  - AWS CloudFormation 堆栈

目标架构

1. 已计划删除 AWS KMS 密钥。
2. 预定删除事件由规则进行评估。 EventBridge
3. 该 EventBridge 规则涉及亚马逊 SNS 话题。
4. 该 EventBridge 规则启动 Systems Manager 自动化和运行手册。
5. 运行手册取消删除。

自动化和扩缩

CloudFormation 堆栈部署了所有必要的资源和服务，以使该解决方案发挥作用。该模式可以在单个账户中独立运行，也可以使用 AWS CloudFormation StackSets 为多个独立账户或组织运行。

```
aws cloudformation create-stack --stack-name  <stack-name>\
    --template-body file://<Full-Path-of-file> \
    --parameters ParameterKey=,ParameterValue= \
    --capabilities CAPABILITY_NAMED_IAM
```

工具

工具

- AWS CloudFormation — AWS CloudFormation 是一项服务，可帮助您建模和设置 Amazon Web Services 资源，这样您就可以花更少的时间管理这些资源，而将更多的时间集中在在 AWS 上运行的应用程序上。您可以使用 CloudFormation 模板在 AWS 区域的 AWS 账户中创建堆栈。该模板描述了您需要的所有 AWS 资源，并 CloudFormation 为您预置和配置这些资源。
- AWS CLI – AWS 命令行界面（AWS CLI）是一种开源工具，它使您能够使用命令行 shell 中的命令与 Amazon Web Services 交互。
- Amaz on EventBridge — Amazon EventBridge 是一项无服务器事件总线服务，可将您的应用程序与来自各种来源的数据连接起来。 EventBridge 提供来自您自己的应用程序和 AWS 服务的实时数据流，并将这些数据路由到目标，例如 AWS Lambda。 EventBridge 简化了构建事件驱动架构的过程。
- AWS KMS – AWS Key Management Service (AWS KMS) 是一项托管服务，用于创建和控制 AWS KMS 密钥，即用于加密数据的加密密钥。
- AWS SDKs — AWS 工具包括， SDKs 因此您可以使用自己选择的编程语言在 AWS 上开发和管理应用程序。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 是一项托管服务，提供从发布者到订阅用户（也称为创建者和使用者）的消息传输。发布者通过将消息发送至主题与订阅用户进行异步交流，主题是一个逻辑接入点和通信渠道。
- AWS Systems Manager – AWS Systems Manager 是一项 Amazon Web Services，可用于查看和控制 AWS 上的基础设施。使用 Systems Manager 控制台，您可以在 AWS 资源之间自动执行操作任务。Systems Manager 通过扫描托管实例并报告其检测到的任何策略违规行为（或采取纠正措施）来帮助您维护安全性和合规性。

代码

- 随函附上该项目的`alerting_ct_logs.yaml` CloudFormation 模板。

操作说明

准备 Amazon Web Services account

Task	描述	所需技能
安装和配置 AWS CLI。	安装 AWS CLI 版本 2。然后，配置身份的安全凭证设置、默认输出格式以及 AWS CLI 用	开发人员、安全工程师

Task	描述	所需技能
	于与 AWS 交互的默认 AWS 区域。 该身份必须具有执行任务所需权限。	

部署 AWS CloudFormation 模板

Task	描述	所需技能
下载 CloudFormation 模板。	将附件下载到计算机上的本地路径，然后解压缩 `alerting_ct_logs.yaml` 模板文件。	开发人员、安全工程师
部署模板。	在已配置 Amazon Web Services account 配置文件的终端窗口中，运行以下命令。 `aws cloudformation create-stack --stack-name <stack_name> \ --capabilities <Value> \ --template-body file://<Full_Path> \ --parameters ParameterKey=DestinationEmailAddress,ParameterValue=<Value> \ ParameterKey=SNSTopicName,ParameterValue=<Value> \ ParameterKey=EnableRemediation ,ParameterValue=<Value> \ ParameterKey=AutomationAssumeRole,`	开发人员、安全工程师

Task	描述	所需技能
	`ParameterValue=<Value>` 在下一步中，输入模板参数的值。	
完成模板参数。	输入所需参数值。 • `DestinationEmailAddress` – 在计划删除 AWS KMS 密钥时接收提醒的电子邮件地址。 • `SNSTopicName` – Amazon SNS 主题的名称。 • `EnableRemediation` – 使用 Systems Manager 运行手册取消计划的密钥删除。允许的值包括 `true`和 `false`。 • `AutomationAssumeRole` – 该角色的 Amazon 资源名称（ARN）允许 Systems Manager Automation 代表您执行操作。有关更多信息，请参阅 AWSConfigRemediation-CancelKeyDeletion文档中的“必需 IAM 权限” 部分。 • `Capabilities` — CloudFormation 要让 AWS 创建堆栈，您必须明确确认您的堆栈模板包含某些功能。	开发人员、安全工程师

确认订阅

Task	描述	所需技能
确认订阅。	检查您的电子邮件收件箱，在收到的 Amazon SNS 电子邮件中选择确认订阅。Web 浏览器窗口将打开，并显示订阅确认信息和您的订阅 ID。	开发人员、安全工程师

相关资源

参考

- 创建 Amazon Web Services 规则
- 创建亚马逊 CloudWatch 警报以检测待删除的 AWS KMS 密钥的使用情况

教程和视频

- 如何开始使用亚马逊 EventBridge
- 深入了解亚马逊 EventBridge（AWS 在线技术讲座）

AWS 研讨会

- 使用 EventBridge 规则

其他信息

以下代码提供了一些示例，用于扩展解决方案以监控任何 Amazon Web Services 中的任何更改并通知您。这些示例包括预定义模式和自定义模式。有关更多信息，请参阅中的事件和事件模式 EventBridge。

```
EventPattern:
        source:
        - aws.kms
        detail-type:
```

```
        - AWS API Call via CloudTrail
        detail:
          eventSource:
          - kms.amazonaws.com
          eventName:
          - ScheduleKeyDeletion
```

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 Security Hub 识别 AWS Organizations 其中的公有 Amazon S3 存储桶

由 Mourad Cherfaoui (AWS)、Arun Chandapillai (AWS) 和 Parag Nagwekar (AWS) 编写

摘要

此模式向您展示如何构建一种机制来识别账户中的公共亚马逊简单存储服务 (Amazon S3) 存储桶。AWS Organizations 该机制的工作原理是使用AWS 基础安全最佳实践 (FSBP) 标准中的控件 AWS Security Hub 来监控 Amazon S3 存储桶。您可以使用亚马逊 EventBridge 来处理 Security Hub 的调查结果，然后将这些发现发布到亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 主题中。组织中的利益相关者可订阅该主题，并立即收到有关调查结果的电子邮件通知。

默认情况下，新的 Amazon S3 存储桶及其对象不允许公开访问。在必须根据组织要求修改默认 Amazon S3 配置时，您可以使用此模式。例如，在这种情况下，您可能有一个托管面向公众的网站的 Amazon S3 存储桶，或者互联网上的每个人都必须能够从您的 Amazon S3 存储桶中读取的文件。

Security Hub 通常作为中央服务部署，用于整合所有安全发现，包括与安全标准和合规要求相关的安全发现。您还可以使用其他方法 AWS 服务 来检测公有 Amazon S3 存储桶，但是这种模式使用的是现有 Security Hub 部署，配置最少。

先决条件和限制

先决条件

- 使用专用 Sec urity Hub 管理员 AWS帐户的多账户设置
- Security Hub 和 AWS Config AWS 区域 ，在你要监控的中启用

> **Note**
>
> 如果要监控来自单个聚合区域的多个区域，则必须在 Security Hub 中启用跨区域聚合。

- 访问和更新 Security Hub 管理员账户的用户权限、对组织中所有 Amazon S3 存储桶的读取权限以及关闭公共访问权限的权限（如果需要）

架构

下图显示了使用 Security Hub 识别公有 Amazon S3 存储桶的架构。

图表显示了以下工作流：

1. Security Hub 使用 FSBP 安全标准中的 S3.2 和 S3.3 控件监控所有 AWS Organizations 账户（包括管理员账户）中 Amazon S3 存储桶的配置，并检测是否将存储桶配置为公共存储桶。
2. Security Hub 管理员账户可以访问所有成员账户的调查结果（包括 S3.2 和 S3.3 的结果）。
3. Security Hub 会自动将所有新发现和现有发现的所有更新 EventBridge 作为 Sec urity Hub 调查结果-导入的事件发送到。这包含来自管理员和成员账户的调查结果的事件。
4. EventBridge 规则会根据来自 S3.2 和 S3.3 的结果进行筛选，这些发现`ComplianceStatus`的结果为`FAILED`，工作流程状态为`NEW`，工作流状态为和 a `RecordState` 为。`ACTIVE`
5. 规则使用事件模式来识别事件，并在匹配后将其发送到 Amazon SNS 主题。
6. Amazon SNS 主题将事件发送给其订阅者（例如通过电子邮件）。
7. 指定接收电子邮件通知的安全分析师会审查相关的 Amazon S3 存储桶。
8. 如果存储桶已获准公开访问，则安全分析师将 Security Hub 中相应调查发现的工作流状态设置为 `SUPPRESSED`。否则，分析师会将状态设置为 `NOTIFIED`。这样可以消除未来对 Amazon S3 存储桶的通知，并减少通知噪音。
9. 如果工作流状态设置为 `NOTIFIED`，安全分析师会与存储桶拥有者一起审查调查发现，以确定公开访问是否合理，是否符合隐私和数据保护要求。调查的结果是：取消了对存储桶的公共访问权限，或者批准了公共访问权限。在后一种情况下，安全分析师将工作流状态设置为 `SUPPRESSED`。

> **Note**
>
> 架构图既适用于单区域聚合部署，也适用于跨区域聚合部署。在图中的账户 A、B 和 C 中，如启用跨区域聚合，Security Hub 可以与管理员账户属于同一个区域，也可以属于不同的区域。

工具

- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。 EventBridge 提供来自您自己的应用程序、软件即服务 (SaaS) 应用程序的实时数据流，以及 AWS 服务。 EventBridge 如果数据符合用户定义的规则，则将该数据路由到目标，例如 Amazon SNS 主题和 AWS Lambda 函数。

- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。订阅用户接收所有发布至他们所订阅主题的消息，并且一个主题的所有订阅用户收到的消息都相同。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- AWS Security Hub提供了您的安全状态的全面视图 AWS。Security Hub 还可以帮助您根据安全行业标准和最佳实践检查您的 AWS 环境。Security Hub 从各种 AWS 账户服务和支持的第三方合作伙伴产品中收集安全数据，然后帮助分析安全趋势并确定优先级最高的安全问题。

操作说明

配置 Security Hub 账户

Task	描述	所需技能
在 AWS Organizations 账户中启用 Security Hub。	要在要监控 Amazon S3 存储桶的组织账户中启用 Security Hub，请参阅 Sec urity Hub 文档中的 “指定 Security Hub 管理员账户（控制台）” 和 “管理属于组织的成员账户” 中的指南。	AWS 管理员
（可选）启用跨区域聚合。	如果您想监控单个区域中多个区域的 Amazon S3 存储桶，请设置跨区域聚合。	AWS 管理员
支持 FSBP 安全标准的 S3.2 和 S3.3 控件。	您必须按 FSBP 安全标准启用 S3.2 和 S3.3 控件。 1. 要启用 S3.2 控件，请按照 Security Hub 文档中的 [S3.2] S3 存储桶应禁止公共读取权限中的说明进行操作。 2. 要启用 S3.3 控件，请按照 Security Hub 文档中的 [3]	AWS 管理员

Task	描述	所需技能
	S3 存储桶应禁止公共写入访问中的说明进行操作。	

设置环境

Task	描述	所需技能
配置 Amazon SNS 主题和电子邮件订阅。	1. 登录 AWS Management Console 并打开 Amazon SNS 控制台。 2. 在导航窗格中，选择主题，然后选择创建主题。 3. 对于类型，选择标准。 4. 对于名称，输入您的主题的名称（例如 public-s3-buckets）。 5. 选择创建主题。 6. 在主题的订阅选项卡上，选择创建订阅。 7. 对于协议，请选择电子邮件。 8. 对于端点，请输入接收通知的电子邮件地址。您可以使用 AWS 管理员、IT 专业人员或 Infosec 专业人员的电子邮件地址。 9. 选择创建订阅。若要创建其他电子邮件订阅，请根据需要重复步骤 6—8。	AWS 管理员
配置 EventBridge 规则。	1. 打开 EventBridge 管理控制台。	AWS 管理员

Task	描述	所需技能
	2. 在 “入门” 部分中，选择 “EventBridge 规则”，然后选择 “创建规则”。 3. 在定义规则详细信息页面，在名称中输入规则的名称（例如，public -s3-buckets）。选择下一步。 4. 在事件模式部分，选择编辑模式。 5. 复制以下代码，将其粘贴至事件模式代码编辑器中，然后选择下一步。 `{ "source": ["aws.securityhub"], "detail-type": ["Security Hub Findings - Imported"], "detail": { "findings": { "Compliance": { "Status": ["FAILED"] }, "RecordState": ["ACTIVE"], "Workflow": { "Status": ["NEW"] }, "ProductFields": { "ControlId": ["S3.2", "S3.3"] } }`	

Task	描述	所需技能
	<pre> } }</pre> 6. 在选择目标页面，在选择目标中，选择 SNS 主题作为目标，然后选择您之前创建的主题。 7. 选择下一步，再次选择下一步，然后选择创建规则。	

故障排除

事务	解决方案
我有一个启用了公开访问权限的 Amazon S3 存储桶，但我没有收到有关它的电子邮件通知。	这可能是因为存储桶是在另一个区域中创建的，并且 Security Hub 管理员账户中未启用跨区域聚合。要解决此问题，请启用跨区域聚合或在您的 Amazon S3 存储桶当前所在的区域实施此模式的解决方案。

相关资源

- 什么是 AWS Security Hub？（Security Hub 文档）
- AWS 基础安全最佳实践 (FSBP) 标准（Security Hub 文档）
- AWS Security Hub 多账户启用脚本（AWS 实验室）
- Amazon S3 安全最佳实践（Amazon S3 文档）

其他信息

监控公共 Amazon S3 存储桶的工作流程

以下工作流程说明了如何监控组织中的公有 Amazon S3 存储桶。该工作流程假设您已完成配置 Amazon SNS 主题中的步骤以及有关此模式的电子邮件订阅故事。

1. 当 Amazon S3 存储桶配置为公有访问权限时，您会收到一封电子邮件通知。
   - 如果存储桶已获准公开访问，请在 Security Hub 管理员账户中将相应调查发现的工作流状态设置为 SUPPRESSED。这可以防止 Security Hub 就此存储桶发出进一步通知，并可以消除重复的警报。
   - 如果存储桶未获得公开访问批准，请将 Security Hub 管理员账户中相应调查发现的工作流状态设置为 NOTIFIED。这可以防止 Security Hub 从 Security Hub 发出关于此存储桶的进一步通知，并且可以消除噪音。
2. 如果存储桶可能包含敏感数据，则立即关闭公共访问权限，直到审查完成。如果您关闭了公共访问权限，则 Security Hub 会将工作流状态更改为 RESOLVED。然后，通过电子邮件发送存储桶停止的通知。
3. 找到将存储桶配置为公共存储桶的用户（例如，通过使用 AWS CloudTrail），然后开始审核。审查的结果是：删除对存储桶的公共访问权限或批准公共访问权限。如果公开访问获得批准，则将相应调查发现的工作流状态设置为 SUPPRESSED。

在 Microsoft Sentinel 中提取和分析 AWS 安全日志

由伊万·吉拉迪 (AWS) 和塞巴斯蒂安·温泽尔 (AWS) 创作

摘要

此模式描述了如何自动将 AWS 安全日志（例如日志、亚马逊日志数据、亚马逊 VPC 流 AWS CloudTrail CloudWatch 日志数据和亚马逊 GuardDuty 调查结果）提取到 Microsoft Sentinel 中。如果你的组织使用 Microsoft Sentinel 作为安全信息和事件管理 (SIEM) 系统，这可以帮助你集中监控和分析日志，以检测与安全相关的事件。一旦日志可用，它们将在不到 5 分钟的时间内自动传输到亚马逊简单存储服务 (Amazon S3) 存储桶。这可以帮助您快速检测 AWS 环境中的安全事件。

Microsoft Sentinel 以表格格式提取 CloudTrail 日志，其中包括记录事件的原始时间戳。采集的日志结构通过在 Microsoft Sentinel 中使用 Kusto 查询语言实现查询功能。

该模式部署了监控和警报解决方案，可在不到 1 分钟的时间内检测到摄取失败。它还包括一个可由外部 SIEM 监控的通知系统。您可以使用 AWS CloudFormation 在日志账户中部署所需的资源。

目标受众

建议有使用 AWS Control Tower、、、 AWS Organizations CloudFormation、 AWS Identity and Access Management (IAM) 和 AWS Key Management Service (AWS KMS) 经验的用户使用此模式。

先决条件和限制

先决条件

以下是部署此解决方案的先决条件：

- 在 AWS 账户 着陆区（Landing zone）中作为组织进行管理 AWS Organizations 且属于 AWS Control Tower 着陆区的活跃状态。组织应包括一个用于登录的专用帐户。有关说明，请参阅 AWS Organizations 文档中的创建和配置组织。
- 记录整个组织的事件并将日志存储在日志账户的 Amazon S3 存储桶中的 CloudTrail 跟踪。有关说明，请参阅为组织创建跟踪。
- 在日志账户中，代入具有以下权限的现有 IAM 角色的权限：
  - 部署在提供的 CloudFormation 模板中定义的资源。
  - 部署提供的 CloudFormation 模板。
  - 如果日志使用客户托管 AWS KMS 密钥加密，请修改密钥策略。

- AWS Command Line Interface (AWS CLI)，已安装并配置。
- 订阅使用微软 Sentinel 的微软 Azure 账户。
- 启用并设置 Microsoft Sentinel。有关说明，请参阅启用 Microsoft Sentinel 以及微软 Sentinel 文档中的初始功能和内容。
- 满足设置 Microsoft Sentinel S3 连接器的先决条件。

限制

- 此解决方案将安全日志从日志账户中的 Amazon S3 存储桶转发给 Microsoft Sentinel。未明确提供有关如何将日志发送到 Amazon S3 的说明。
- 此模式提供了在 landing zone AWS Control Tower 中进行部署的说明。但是， AWS Control Tower 不需要使用。
- 此解决方案与服务控制策略 (SCPs) 限制 Amazon S3 日志存储桶的环境兼容，例如不允许更改日志存档中 AWS Control Tower 创建的 Amazon S3 存储桶的存储桶策略。
- 此模式提供了转发 CloudTrail 日志的说明，但您可以调整此解决方案以发送 Microsoft Sentinel 支持的其他日志，例如来自 CloudWatch 日志的日志、Amazon VPC 流日志和。 GuardDuty
- 这些说明使用 AWS CLI 来部署 CloudFormation 模板，但您也可以使用 AWS Management Console。有关说明，请参阅使用 AWS CloudFormation 控制台。如果您使用控制台部署堆栈，请将堆栈部署在与日志存储桶 AWS 区域 相同的位置中。
- 此解决方案部署亚马逊简单队列服务 (Amazon SQS) Simple Queue Service 队列来传送亚马逊 S3 通知。该队列包含在 Amazon S3 存储桶中上传的对象的路径的消息，而不是实际数据。队列使用 SSE-SQS 加密来帮助保护消息的内容。如果要使用 SSE-KMS 对 SQS 队列进行加密，则可以使用客户托管的 KMS 密钥。有关更多信息，请参阅 Amazon SQS 中的静态加密。

架构

本节简要概述了示例代码所建立的架构。下图显示了在日志账户中部署的资源，这些资源用于将日志从现有 Amazon S3 存储桶提取到 Microsoft Sentinel 中。

架构图显示了以下资源交互：

1. 在日志账户中，微软 Sentinel 通过 OpenID Connect (OIDC) 担任 IAM 角色，访问特定 Amazon S3 存储桶和亚马逊 SQS 队列中的日志。

2. 亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 和 Amazon S3 用于加密 AWS KMS 。
3. 每当收到新日志时，Amazon S3 都会向亚马逊 SQS 队列发送通知消息。
4. 微软 Sentinel 会检查亚马逊 SQS 中是否有新消息。亚马逊 SQS 队列使用 SSE-SQS 加密。消息保留期设置为 14 天。
5. 微软 Sentinel 从亚马逊 SQS 队列中提取消息。这些消息包含上传的 Amazon S3 对象的路径。微软 Sentinel 将这些对象从亚马逊 S3 存储桶提取到微软 Azure 账户中。
6. CloudWatch 警报会监控 Amazon SQS 队列。如果在 5 分钟内未收到消息并将其从 Amazon SQS 队列中删除，则它会启动 Amazon SNS 通知并发送一封电子邮件。

AWS Control Tower 帮助您设置基础组织单位 (OU) 结构并将 CloudTrail 日志集中到日志帐户中。它还实现了保护日志存储桶的强制性 SCPs 。

我们已经在 landing zone AWS Control Tower 中提供了目标架构，但这并不是严格要求的。在此图表中，管理账户中的资源反映了为整个组织记录事件的 AWS Control Tower 部署和 CloudTrail 跟踪。

这种模式侧重于在日志账户中部署资源。如果存储在您的 AWS Control Tower 着陆区的 Amazon S3 中的日志使用客户托管的 KMS 密钥进行加密，则必须更新密钥策略以允许 Microsoft Sentinel 解密日志。在 lan AWS Control Tower ding zone 中，您可以通过管理账户管理密钥策略，管理账户是密钥的创建地。

工具

AWS 服务

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在资源的整个生命周期中跨地区对其 AWS 账户 进行管理。
- Amazon CloudWatch 可帮助您实时监控您的 AWS 资源和运行的应用程序 AWS 的指标。
- AWS Control Tower按照最佳实践，帮助您设置和管理 AWS 多账户环境。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥以帮助保护您的数据。
- AWS Organizations是一项账户管理服务，可帮助您将多个账户整合 AWS 账户 到一个由您创建和集中管理的组织中。
- Amazon Simple Queue Service (Amazon SQS) 提供了一个安全、持久且可用的托管队列，它可帮助您集成和分离分布式软件系统与组件。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

其他工具

- Microsoft Sentinel 是一个云原生 SIEM 系统，可提供安全编排、自动化和响应 (SOAR)。

代码存储库

此模式的代码可在 Microsoft Sent GitHub inel 存储库中的收录和分析 AWS 安全日志中找到。

最佳实践

- 遵循最低权限原则（IAM 文档）。
- 遵循AWS Control Tower 管理员最佳实践（AWS Control Tower 文档）。
- 遵循AWS CloudFormation 最佳实践（CloudFormation 文档）。
- 使用代码分析工具（例如 cfn_nag ）来扫描生成的模板。 CloudFormation cfn_nag 工具通过搜索模式来识别 CloudFormation 模板中的潜在安全问题。

操作说明

将微软 Sentinel 连接到亚马逊 S3

Task	描述	所需技能
准备好微软 Sentinel S3 连接器。	1. 在 Microsoft Sentinel 中，选择数据连接器。 2. 从数据连接器库中，选择 Amazon Web Services S3。 **Note** 如果你看不到连接器，请从 Microsoft Sentinel 的内容中心安装亚马逊 Web Services 解决方案。	DevOps 工程师，通用 AWS

Task	描述	所需技能
	3. 在连接器的详细信息窗格中，选择打开连接器页面。 4. 在 “配置” 部分中，复制外部 ID。稍后你需要这个身份证。	

部署堆 CloudFormation 栈

Task	描述	所需技能
克隆存储库。	在 bash Shell 中输入以下命令：这将克隆 Microsoft Sentinel 存储库中的收录和分析 AWS 安全日志。 `git clone https://github.com/aws-samples/ingest-and-analyze-aws-security-logs-in-microsoft-sentinel.git`	DevOps 工程师，通用 AWS
在日志账户中扮演 IAM 角色。	在日志账户中，扮演有权部署 CloudFormation 堆栈的 IAM 角色。有关在中担任 IAM 角色的更多信息 AWS CLI，请参阅中的使用 IAM 角色 AWS CLI。	DevOps 工程师，通用 AWS
部署 堆栈。	要部署 CloudFormation 堆栈，请输入以下命令，其中： • `<Bucket name>`是记录日志 Amazon S3 存储桶的名称。	DevOps 工程师，通用 AWS

Task	描述	所需技能
	• <Sentinel external ID>是微软 Sentinel 中亚马逊 S3 连接器的外部 ID。 • <Email address>是您想要接收通知的有效电子邮件地址。 • <Customer managed key ARN>是客户托管的 KMS 密钥的亚马逊资源名称 (ARN)。仅当使用客户托管的 KMS 密钥对日志进行加密时，才提供此参数。 • <Suffix>是一个可选参数，用于避免资源名称冲突。 • <ARN for the OIDC provider> 是 OIDC 提供商的 ARN（如果已存在）。如果您未提供此参数，则 CloudFormation 创建 OIDC 提供商。 ⚠ Important 如果使用微软 Code Defender 对 AWS 组织进行监控，则微软的 OIDC 提供商已经部署完毕。您必须提供此参数和现有提供商的 ARN。	

Task	描述	所需技能
	`aws cloudformation deploy --stack-name cloudtrail-sentinel-integration \ --no-fail-on-empty-changeset \ --template-file template.yml \ --capabilities CAPABILITY_IAM CAPABILITY_NAMED_IAM CAPABILITY_AUTO_EXPAND \ --parameter-overrides \ ControlTowerS3BucketName="<Bucket name>" \ AzureWorkspaceID="<Sentinel external ID>" \ EmailAddress="<Email address>" \ KMSKeyArn="<Customer managed key ARN>" \ Suffix="<Suffix to avoid name conflicts>" \ OIDCProviderArn="<ARN for the OIDC provider>"`	
复制输出。	从 CloudFormation 堆栈的输出中，复制`SentinelRoleArn` 和的值`SentinelSQS` 。你稍后使用这些值在 Microsoft Sentinel 中完成配置。	DevOps 工程师，通用 AWS

Task	描述	所需技能
修改密钥策略。	如果您没有使用客户托管的 KMS 密钥来加密 Amazon S3 存储桶中的日志，则可以跳过此步骤。 如果使用客户管理的 KMS 密钥对日志进行加密，请修改密钥策略以授予 Microsoft Sentinel 解密日志的权限。以下是示例密钥策略。如果 KMS 密钥位于另一个 AWS 账户密钥中，则此示例策略允许跨账户访问。 `{ "Version": "2012-10-17", "Id": "key-policy", "Statement": [ ... { "Sid": "Grant access to decrypt", "Effect": "Allow", "Principal": { "AWS": "<SentinelRoleArn>" }, "Action": "kms:Decrypt", "Resource": "<KeyArn>" } ] }`	DevOps 工程师，通用 AWS

在 Microsoft Sentinel 中配置连接器

Task	描述	所需技能
在 Microsoft Sentinel 中完成配置。	1. 在 Microsoft Sentinel 中，选择数据连接器。 2. 从数据连接器库中，选择 Amazon Web Services S3。 3. 在连接器的详细信息窗格中，选择打开连接器页面。 4. 在“配置”部分中，执行以下操作： a. 在要添加的角色中，输入您复制的`SentinelRoleArn` 值。 b. 在 SQS 网址中，输入您复制的`SentinelSQS` 值。 c. 在目标表列表中，选择`AWSCloudTrail` 。 5. 选择添加连接。	DevOps 工程师
向亚马逊 SQS 发送亚马逊 S3 事件通知。	按照使用 Amazon S3 控制台启用和配置事件通知中的说明将 Amazon S3 日志存储桶配置为向 Amazon SQS 队列发送事件通知。如果 CloudTrail 已为整个组织进行了配置，则此存储桶中的日志将带有前缀`<OrgID>/AWSLogs/<OrgID>/` ，其中`<OrgID>`是组织 ID。有关更多信息，请参阅查看有关您的组织的详细信息。	DevOps 工程师，通用 AWS

Task	描述	所需技能
确认已提取日志。	1. 等到在 Microsoft Sentinel 中提取日志。这个过程可能需要几分钟。 2. 在 Microsoft Sentinel 中，打开 Amazon S3 数据连接器页面，然后执行以下操作： • 确认 Amazon S3 数据连接器的状态为`Connected`。 • 在“已接收数据”图表中检查数据量。 有关检查数据连接器活动的更多信息，请参阅 Microsoft 文档中的数据连接器。	DevOps 工程师

验证解决方案

Task	描述	所需技能
比较 CloudWatch 和 Sentinel 日志。	在的默认配置中 AWS Control Tower， CloudTrail 日志将发送到 Amazon CloudWatch 并存储在 AWS Control Tower 管理账户中。有关更多信息，请参阅中的日志和监控 AWS Control Tower。使用以下步骤确认日志已自动提取到 Microsoft Sentinel 中： 1. 打开 CloudWatch 管理控制台。 2. 在导航窗格中，选择 Logs（日志），然后选	DevOps 工程师，通用 AWS

Task	描述	所需技能
	择 Logs Insights（日志洞察）。 3. 在“选择日志组”中，选择存储 CloudTrail 日志的日志组，例如`aws-controltower/CloudTrailLogs`。 4. 在查询编辑器框中，输入`fields eventID`。 5. 选择运行查询。 6. 选择“导出结果”，然后选择“将表格复制到剪贴板 (CSV)”。 7. 将结果粘贴到文本编辑器中。 8. 更改输出的格式，使其可以在 Microsoft Sentinel 查询中使用。以下是使用 Kusto 查询语言的示例： `AWSCloudTrail | where AwsEventId in ( 'aa08b5fe-3bfb-391a-a14e-5fcebe14dab2', '9decd805-269c-451c-b75b-762f5dce59f9' )` 9. 在 Microsoft Sentinel 中，打开 Amazon S3 数据连接器页面。在“已接收数据”图表旁边，选择“转到日志分析”。	

Task	描述	所需技能
	10.在查询编辑器框中，输入查询，然后选择运行。 11.在 Microsoft Sentinel 和中 CloudWatch，验证条目数量是否相同。如果需要，调整时间范围。	

相关资源

AWS 文档和资源

- AWS CLI 命令参考（AWS CLI 文档）
- （可选）配置 AWS KMS keys（AWS Control Tower 文档）
- 亚马逊 SQS 中的静态加密（亚马逊 SQS 文档）
- 如何防止邮件列表收件人从我的 Amazon SNS 主题电子邮件中取消订阅列表中的所有人？（AWS 知识中心）

微软文档

- 将 Microsoft Sentinel 连接到亚马逊 Web Services 以摄取 AWS 服务日志数据
- 微软 Sentinel 中的 Kusto 查询语言

使用以下 AWS IAM Identity Center 命令将权限集作为代码进行管理 AWS CodePipeline

由 Andre Cavalcante (AWS) 和 Claison Amorim (AWS) 创建

摘要

AWS IAM Identity Center 帮助您集中管理对所有 AWS 账户 和应用程序的单点登录 (SSO) 访问权限。您可以在 IAM Identity Center 中创建和管理用户身份，也可以连接现有身份源，例如 Microsoft Active Directory 域或外部身份提供者 (IdP)。IAM Identity Center 提供了统一的管理体验，可使用权限集定义、自定义和分配对 AWS 环境的精细访问权限。权限集适用于您的 IAM Identity Center 身份存储或外部 IdP 中的联合用户和群组。

此模式可帮助您在多账户环境中以代码形式管理 IAM Identity Center 权限集，该环境在中 AWS Organizations作为组织进行管理。使用此模式，您可以实现以下目标：

- 创建、删除和更新权限集
- 创建、更新或删除对目标 AWS 账户、组织单位 (OUs) 或组织根目录的权限集分配。

为了以代码形式管理 IAM Identity Center 权限和分配，此解决方案部署了使用和的持续集成和持续交付 (CI/CD) 管道。 AWS CodeBuild AWS CodePipeline您可以在存储在远程存储库中的 JSON 模板中管理权限集和分配。当 Amazon EventBridge 规则检测到存储库的更改或检测到目标 OU 中账户的修改时，它会启动一个 AWS Lambda 函数。Lambda 函数启动 CI/CD 管道，更新 IAM Identity Center 中的权限集和分配。

先决条件和限制

先决条件

- 在中作为组织管理的多账户环境。 AWS Organizations有关更多信息，请参阅创建组织。
- 已启用并配置身份源的 IAM Identity Center。有关更多信息，请参阅 IAM Identity Center 文档中的快速入门。
- 注册为以下任务的委托管理员的成员账户 AWS 服务：
  - IAM 身份中心 — 有关说明，请参阅 IAM 身份中心文档中的注册成员账户。
  - AWS Organizations — 有关说明，请参阅委派管理员了解相关信息 AWS Organizations。此账户必须具有列出和描述账户的权限，以及 OUs。

> **Note**
>
> 对于这两项服务，您必须使用与委派管理员相同的帐户。

- 在 IAM Identity Center 委派的管理员账户和组织的管理账户中部署 AWS CloudFormation 堆栈的权限。有关更多信息，请参阅 CloudFormation 文档中的控制访问权限。
- IAM Identity Center 中的亚马逊简单存储服务 (Amazon S3) Service 存储桶委派了管理员账户。您将构件代码上传到此存储桶中。有关说明，请参阅 Amazon S3 文档中的创建存储桶。
- 组织管理账户的账户 ID。有关说明，请参阅查找您的 AWS 账户 身份证。
- 源代码主机中的存储库，例如 GitHub。

限制

- 此模式不能用于管理或分配单账户环境的权限集，也不能用于未在 AWS Organizations中作为组织管理的账户管理或分配权限集。
- 权限集名称 IDs、分配和 IAM Identity Center 主体类型，部署后 IDs 无法修改。
- 此模式可帮助您创建和管理自定义权限。您不能使用此模式来管理或分配预定义权限。
- 此模式不能用于管理组织管理账户的权限集。

架构

目标架构

图表显示了以下工作流：

1. 某位用户进行了以下更改之一：
   - 向远程存储库提交一项或多项更改，例如 GitHub
   - 修改 OU 中的账户 AWS Organizations
2. 如果用户向主分支提交了对远程存储库的更改，则管道将启动。

   如果用户修改了 OU 中的账户，则该`MoveAccount` EventBridge 规则会检测到更改并在组织的管理账户中启动 Lambda 函数。
3. 启动的 Lambda 函数在中启动 CI/CD 管道。 CodePipeline

4. CodePipeline 启动`TemplateValidation` CodeBuild 项目。该`TemplateValidation` CodeBuild 项目使用远程存储库中的 Python 脚本来验证权限集模板。 CodeBuild 验证以下内容：
   - 权限集名称是唯一的。
   - 赋值语句 IDs (`Sid`) 是唯一的。
   - `CustomPolicy` 参数中的策略定义和有效。（此验证使用 AWS Identity and Access Management Access Analyzer。）
   - 托管策略的 Amazon 资源名称 (ARNs) 有效。
5. `Deploy` CodeBuild 项目中的`PermissionSet`操作组用于 适用于 Python (Boto3) 的 AWS SDK 删除、创建或更新 IAM Identity Center 中的权限集。只有带有 `SSOPipeline:true` 标签的权限集会受到影响。通过此管道管理的所有权限集都有此标签。
6. `Deploy` CodeBuild 项目中的`Assignments`操作组使用 Terraform 在 IAM Identity Center 中删除、创建或更新分配。Terraform 后端状态文件存储在同一个账户的 Amazon S3 存储桶中。
7. CodeBuild 更新 IAM 身份中心中的权限集和分配。

自动化和扩缩

由于多账户环境中的所有新账户都将移至中的特定组织单位 AWS Organizations，因此此解决方案会自动运行并向您在分配模板中指定的所有账户授予所需的权限集。无需进行其他自动化或扩展操作。

在大型环境中，向 IAM Identity Center 发出的 API 请求数量可能会导致此解决方案的运行速度变慢。Terraform 和 Boto3 会自动管理节流，以最大限度地减少任何性能下降。

工具

AWS 服务

- AWS CloudFormation帮助您设置 AWS 资源，快速一致地配置资源，并在和的整个 AWS 账户 生命周期中对其进行管理 AWS 区域。
- AWS CodeBuild是一项完全托管的生成服务，可帮助您编译源代码、运行单元测试和生成可随时部署的工件。
- AWS CodePipeline帮助您快速建模和配置软件发布的不同阶段，并自动执行持续发布软件更改所需的步骤。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如， AWS Lambda 函数、使用 API 目的地的 HTTP 调用端点或其他 AWS 账户目的地的事件总线。

- AWS IAM Identity Center帮助您集中管理对所有应用程序 AWS 账户 和云应用程序的单点登录 (SSO) 访问权限。
- AWS Organizations是一项账户管理服务，可帮助您将多个账户整合 AWS 账户 到一个由您创建和集中管理的组织中。
- 适用于 Python (Boto3) 的 AWS SDK是一个软件开发套件，可帮助您将 Python 应用程序、库或脚本与集成 AWS 服务。
- Amazon Simple Storage Service (Amazon S3)是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。

代码存储库

此模式的代码可在 aws-iam-identity-center-pipelin e 存储库中找到。存储库中的 templates 文件夹包含权限集和分配的示例模板。它还包括用于在目标客户中部署 CI/CD 管道和 AWS 资源的 AWS CloudFormation 模板。

最佳实践

- 在开始修改权限集和分配模板之前，我们建议您为您的组织规划权限集。考虑权限应是什么、权限集应适用于哪些账户或 OUs 权限集，以及哪些 IAM Identity Center 委托人（用户或群组）应受到权限集的影响。权限集名称 IDs、关联和 IAM Identity Center 委托人类型，部署后 IDs 无法修改。
- 遵循最低权限原则，并授予执行任务所需最低权限。有关更多信息，请参阅 AWS Identity and Access Management (IAM) 文档中的授予最低权限和安全最佳实践。

操作说明

规划权限集和分配

Task	描述	所需技能
克隆存储库。	在 bash Shell 中输入以下命令：这将从中克隆 aws-iam-identity-center-pipelin e 存储库。 GitHub `git clone https://github.com/aws-samp`	DevOps 工程师

Task	描述	所需技能
	`les/aws-iam-identity-center-pipeline.git`	

Task	描述	所需技能
定义权限集。	1. 在克隆的存储库中，导航到 `templates/permissionsets` 文件夹，然后打开其中一个可用模板。 2. 在 `Name` 参数中，输入权限集的名称。此值必须是唯一的，并且在部署后无法更改。 3. 在 `Description` 参数中，简要描述权限集，例如其用例。 4. 在`SessionDuration` 参数中，指定用户可以登录的时间长度 AWS 账户。使用 ISO-8601 持续时间格式（维基百科），例如 PT4H 为 4 小时。如果未定义值，则 IAM Identity Center 中的默认值为 1 小时。 5. 在`RelayState` 参数中，指定最适合用户角色的提供对控制台的快速访问的 URL。 6. 自定义权限集中的策略。以下所有参数均为可选参数，可在部署后进行修改。您必须至少使用其中一个参数才能在权限集中定义策略： • 在`ManagedPolicies` 参数中 ARNs ，输入要分配的所有AWS托管策略。	DevOps 工程师

Task	描述	所需技能

	• 在 CustomerManagedPolicies 参数中，输入您要分配的任何客户管理型策略的名称。请勿使用 ARN。
	• 在PermissionBoundary 参数中，执行以下操作以分配权限边界：
	  • 如果您使用 AWS 托管策略作为权限边界，请在PolicyType AWS、输入和输入策略的 ARN。Policy
	  • 如果您使用客户托管策略作为权限边界，请在PolicyType Customer、输入和输入策略的名称。Policy请勿使用 ARN。
	• 在 CustomPolicy 参数中，定义要分配的任何自定义 JSON 格式策略。有关 JSON 策略文档的结构和内容的更多信息，请参阅JSON 策略概览。
	7. 保存并关闭权限集模板。我们建议您使用与权限集名称相匹配的名称保存文件。
	8. 重复此过程以创建组织所需任意数量的权限集，并删除任何不需要的示例模板。	

Task	描述	所需技能
定义分配。	1. 在克隆的存储库中，导航到 `templates/assignments` 文件夹，然后打开 `iam-identitycenter-assigments.json` 。此文件描述了您希望如何将权限集分配给 AWS 账户 或 OUs。 2. 在 `SID` 参数中，输入分配的标识符。此值必须是唯一的，并且在部署后无法修改。 3. 在 `Target` 参数中，定义要在其中应用权限集的账户或组织。有效值为账户 IDs OUs、或`root`。`root`将权限集分配给组织中的所有成员帐户，不包括管理帐户。在双引号中输入值，并用逗号分隔多个值。账户 IDs 并 OUs 应遵循模式：`{{account_name}}:{{account_id}}` 或`{{ou_name}}:{{ou_id}}` 。如果要递归地为嵌套分配权限 OUs，请使用末尾带有通配符的 OU 模式。示例：`{{ou_name}}:{{ou_id}}:*` 4. 在 `PrincipalType` 参数中，输入将受权限集影响的 IAM Identity Center 主体的类型。有效值为 `USER`或	DevOps 工程师

Task	描述	所需技能
	GROUP。部署后无法修改此值。 5. 在 `PrincipalID` 参数中，输入 IAM Identity Center 身份存储中将受权限集影响的用户或组的名称。部署后无法修改此值。 6. 在 `PermissionSetName` 参数中，输入要分配的权限集的名称。 7. 重复步骤 2-6 在此文件中创建所需数量的分配。通常情况下，每个权限集都有一个分配。删除所有不需要的示例作业。 8. 保存并关闭 `iam-identitycenter-assigments.json` 文件。	

部署权限集和分配

Task	描述	所需技能
在 IAM Identity Center 委托管理员账户中部署资源。	1. 在 IAM Identity Center 委托管理员账户中，打开AWS CloudFormation 控制台。 2. 部署 `iam-identitycenter-pipeline.yaml` 模板。为堆栈指定一个清晰且具有描述性的名称，并按照指示更新参数。有关说明，请参阅	DevOps 工程师

Task	描述	所需技能
	CloudFormation 文档中的创建堆栈。	
在 AWS Organizations 管理账户中部署资源。	1. 登录组织的管理账户。 2. 打开 AWS CloudFormation 管理控制台。 3. 在导航栏中，选择当前显示的名称 AWS 区域。选择 `us-east-1` 区域。该区域是必需的，这样`MoveAccount` EventBridge 规则才能检测到与组织变更相关 AWS CloudTrail 的事件。 4. 部署 `iam-identitycenter-organization` 模板。为堆栈指定一个清晰且具有描述性的名称，并按照指示更新参数。有关说明，请参阅 CloudFormation 文档中的创建堆栈。	DevOps 工程师
完成远程存储库的设置。	将 AWS CodeConnections 连接状态从更改`PENDING`为`AVAILABLE` 。此连接是在您部署 CloudFormation 堆栈时创建的。有关说明，请参阅 CodeConnections 文档中的更新待处理连接。	DevOps 工程师

Task	描述	所需技能
将文件上传到远程存储库。	将您从`aws-samples` 存储库下载并在之前的步骤中编辑的所有文件上传到远程存储库。对`main`分支的更改会启动管道，管道会创建或更新权限集和分配。	DevOps 工程师

更新权限集和分配

Task	描述	所需技能
更新权限集和分配。	当 `MoveAccount` Amazon EventBridge 规则检测到组织中的账户被修改时，CI/CD pipeline automatically starts and updates the permission sets. For example, if you add an account to an OU specified in the assignments JSON file, then the CI/CD管道会将权限集应用于新账户。 如果要修改已部署的权限集和分配，请更新 JSON 文件，然后将其提交到远程存储库。 使用 CI/CD 管道管理先前部署的权限集和关联时，请注意以下事项： • 如果更改了权限集的名称，CI/CD 管道会删除原始权限集并创建一个新权限集。	DevOps 工程师

Task	描述	所需技能
	• 此管道仅管理具有 `SSOPipeline:true` 标签的权限集。 • 您可以在存储库的同一文件夹中拥有多个权限集和分配模板。 • 如果您删除一个模板，则管道会删除该分配或权限集。 • 如果您删除整个分配 JSON 块，管道将从 IAM Identity Center 中删除该分配。 • 您无法删除分配给的权限集 AWS 账户。首先，您必须取消分配权限集。	

故障排除

事务	解决方案
访问被拒绝错误	确认您拥有部署 CloudFormation 模板和模板中定义的资源所需的权限。有关更多信息，请参阅 CloudFormation 文档中的控制访问权限。
验证阶段的管道错误	如果权限集或分配模板中存在任何错误，则会出现此错误。 1. 在中 CodeBuild，查看版本详细信息。 2. 在构建日志中，查找验证错误，该错误提供了有关导致构建失败的原因的更多信息。 3. 更新权限集或分配模板，然后将其提交到存储库。 4. CI/CD 管道会重新启动项目。 CodeBuild 监控状态以确认验证错误已解决。

相关资源

- 权限集（IAM Identity Center 文档）

使用 AWS Secrets Manager 管理凭证

由 Durga Prasad Cheepuri (AWS) 创建

摘要

此模式将引导您使用 AWS Secrets Manager 动态获取 Java Spring 应用程序的数据库凭证。

过去，当您创建自定义应用程序以从数据库中检索信息时，通常必须在应用程序中直接嵌入访问数据库所需凭证（密钥）。当需要轮换凭证时，您必须投入时间更新应用程序以使用新的凭证，然后分配更新后的应用程序。如果您有多个应用程序共享凭证，而您错过更新其中一个，则该应用程序将会失效。为应对此类风险，许多用户选择不定期轮换凭证，然而，这种行为实际上会带来新的风险。

Secrets Manager 允许您将代码中的硬编码凭证（包括密码）替换为 API 调用，以编程方式检索密钥。这有助于确保检查者不会泄露密钥，因为代码中根本不包含密钥。此外，您还可以配置 Secrets Manager 根据您指定的计划自动轮换密钥。这样，您就可以将长期密钥替换为短期密钥，从而显著降低泄露风险。有关更多信息，请参阅 AWS Secrets Manager 文档。

先决条件和限制

先决条件

- 可访问 Secrets Manager 的 Amazon Web Services 账户
- Java Spring 应用程序

架构

源技术堆栈

- Java Spring 应用程序，其中包含访问数据库的代码，其数据库凭证由 application.properties 文件管理。

目标技术堆栈

- Java Spring 应用程序，其中包含访问数据库的代码，其数据库凭证在 Secrets Manager 中管理。application.properties 文件负责保存 Secrets Manager 的密钥。

Secrets Manager 与应用程序集成

工具

- Secrets Manager - AWS Secrets Manager 是一项 Amazon Web Service，可让您更轻松地管理密钥。密钥可以是数据库凭证、密码、第三方 API 密钥，甚至是任意文本。你可以使用 Secrets Manager 控制台、Secrets Manager 命令行界面 (CLI) 或 Secrets Manager API 和，集中存储和控制对这些密钥的访问。 SDKs

操作说明

在 Secrets Manager 中存储密钥

Task	描述	所需技能
将数据库凭证作为密钥存储在 Secrets Manager 中。	按照 Secrets Manager 文档中创建密钥中的步骤，将 Amazon Relational Database Service (Amazon RDS) 或其他数据库凭证作为密钥存储在 Secrets Manager 中。	系统管理员
为 Spring 应用程序设置访问 Secrets Manager 的权限。	根据 Java Spring 应用程序使用 Secrets Manager 的方式设置相应权限。要控制对密钥的访问权限，请根据 Secrets Manager 文档中对 Secrets Manager 使用基于身份的策略 (IAM Policy) 和 ABAC 以及对 Secrets Manager 使用基于资源的策略部分提供的信息创建策略。按照 Secrets Manager 文档中检索密钥值部分所述的步骤进行操作。	系统管理员

更新 Spring 应用程序

Task	描述	所需技能
添加 JAR 依赖项以使用 Secrets Manager。	有关详细信息，请参阅其他信息部分。	Java 开发人员
将密钥的详细信息添加到 Spring 应用程序。	使用密钥名称、端点和 AWS 区域更新 application.properties 文件。有关示例，请参阅其他信息部分。	Java 开发人员
在 Java 中更新数据库凭证检索代码。	在应用程序中，更新获取数据库凭证的 Java 代码，以从 Secrets Manager 获取这些详细信息。有关示例代码，请参阅其他信息部分。	Java 开发人员

相关资源

- AWS Secrets Manager 文档
- 对 Secrets Manager 使用基于身份的策略 (IAM Policy) 和 ABAC
- 对 Secrets Manager 使用基于资源的策略
- 示例代码

其他信息

添加使用 Secrets Manager 所需 JAR 依赖项

Maven:

```
<groupId>com.amazonaws</groupId>
    <artifactId>aws-java-sdk-secretsmanager</artifactId>
    <version>1.11. 355 </version>
```

Gradle:

```
compile group: 'com.amazonaws', name: 'aws-java-sdk-secretsmanager', version:
 '1.11.355'
```

使用密钥的详细信息更新 application.properties 文件

```
spring.aws.secretsmanager.secretName=postgres-local
spring.aws.secretsmanager.endpoint=secretsmanager.us-east-1.amazonaws.com
spring.aws.secretsmanager.region=us-east-1
```

在 Java 中更新数据库凭证检索代码

```
String  secretName  =  env.getProperty("spring.aws.secretsmanager.secretName");
String  endpoints  =  env.getProperty("spring.aws.secretsmanager.endpoint");
String  AWS Region  =  env.getProperty("spring.aws.secretsmanager.region");
AwsClientBuilder.EndpointConfiguration  config  =  new
  AwsClientBuilder.EndpointConfiguration(endpoints, AWS Region);
AWSSecretsManagerClientBuilder  clientBuilder  =
  AWSSecretsManagerClientBuilder.standard();
clientBuilder.setEndpointConfiguration(config);
AWSSecretsManager  client  =  clientBuilder.build();

ObjectMapper  objectMapper  =  new  ObjectMapper();

JsonNode  secretsJson  =  null;

ByteBuffer  binarySecretData;

GetSecretValueRequest  getSecretValueRequest  =  new
  GetSecretValueRequest().withSecretId(secretName);

GetSecretValueResult  getSecretValueResponse  =  null;

try  {
     getSecretValueResponse  =  client.getSecretValue(getSecretValueRequest);
    }

catch  (ResourceNotFoundException  e)  {
     log.error("The requested secret "  +  secretName  +  " was not found");
    }

catch  (InvalidRequestException  e)  {
```

```
     log.error("The request was invalid due to: "  +  e.getMessage());
     }

catch  (InvalidParameterException  e)  {
     log.error("The request had invalid params: "  +  e.getMessage());
     }
if  (getSecretValueResponse  ==  null)  {
     return  null;
     }  // Decrypted secret using the associated KMS key // Depending on whether the
 secret was a string or binary, one of these fields will be populated


String secret = getSecretValueResponse.getSecretString();

if (secret != null) {
     try {
                secretsJson  =  objectMapper.readTree(secret);
           }

     catch  (IOException  e)  {
                log.error("Exception while retrieving secret values: "  +
  e.getMessage());
           }
}

else  {
     log.error("The Secret String returned is null");

     return null;

     }
     String  host  =  secretsJson.get("host").textValue();
     String  port  =  secretsJson.get("port").textValue();
     String  dbname  =  secretsJson.get("dbname").textValue();
     String  username  =  secretsJson.get("username").textValue();
     String  password  =  secretsJson.get("password").textValue();
}
```

监控 ElastiCache 集群中的安全组

由 Susanne Kangnoh (AWS) 和 Archit Mathur (AWS) 创建

摘要

Amazon ElastiCache 是一项 Amazon Web Services (AWS) 服务，它为在云中分配内存数据存储或缓存环境提供了高性能、可扩展且经济实惠的缓存解决方案。它从高吞吐量和低延迟的内存数据存储中检索数据。此功能使其成为缓存、会话存储、游戏、地理空间服务、实时分析和队列等实时用例的热门选择。 ElastiCache 提供 Redis 和 Memcached 数据存储，两者都提供亚毫秒级的响应时间。

安全组通过控制入站和出站流量，充当 ElastiCache 实例的虚拟防火墙。安全组在实例级别运行，而不是子网级别。对于每个安全组，您可以添加一套规则以控制到实例的入站数据流，以及另外一套单独规则以控制出站数据流。您可以指定允许规则，但不能指定拒绝规则。

此模式提供了一种安全控制，用于监控 API 调用，并在 Amazon Events 中生成CreateReplicationGroup有关CreateCacheCluster、ModifyCacheCluster、和ModifyReplicationGroup操作 CloudWatch 的事件。此事件调用一个 AWS Lambda 函数，该函数运行 Python 脚本。该函数从事件 JSON 输入中获取复制组 ID，并执行以下检查以确定是否存在安全违规：

- 检查集群的安全组是否与 Lambda 函数中配置的安全组匹配。
- 如果集群的安全组不匹配，该函数将使用 Amazon Simple Notification Service (Amazon SNS) 通知向您提供的电子邮件地址发送违规消息。

先决条件和限制

先决条件

- 一个活跃的 AWS 账户。
- 用于上传提供的 Lambda 代码的 Amazon Simple Storage Service （ Amazon S3 ）存储桶。
- 您希望接收违规通知的电子邮件地址。
- ElastiCache 已启用日志记录，用于访问所有 API 日志。

限制

- 这种侦探控制是区域性的，必须部署在你 AWS 区域 要监控的每个地方。

- 该控件支持在虚拟私有云（VPC）中运行的复制组。

架构

工作流程架构

自动化和扩缩

- 如果您正在使用 AWS Organizations，则可以使用将此模板部署AWS CloudFormation StackSets到要监控的多个账户中。

工具

AWS 服务

- 借 ElastiCache助 Amazon，您可以轻松地在中设置、管理和扩展分布式内存缓存环境。 AWS Cloud它提供了高性能、可调整大小且经济实惠的内存缓存，同时消除了与部署和管理分布式缓存环境相关的复杂性。 ElastiCache 可与 Redis 和 Memcached 引擎一起使用。
- AWS CloudFormation帮助您对资源进行建模和设置，快速一致地配置 AWS 资源，并在资源的整个生命周期中对其进行管理。您可以使用模板来描述资源及其依赖关系，然后将它们作为堆栈一起启动和配置，而不必单独管理资源。您可以跨多个 AWS 账户 和 AWS 区域管理和配置堆栈。
- Amazon CloudWatch Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。CloudWatch 通过发送消息以响应环境、激活功能、进行更改和捕获状态信息，事件会在操作变化发生时意识到这些变化，并在必要时采取纠正措施。
- AWS Lambda是一项计算服务，支持在不预置或管理服务器的情况下运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。您只需为消耗的计算时间付费 - 代码未运行时不产生费用。
- Amazon Simple Notification Service (Amazon SNS) 协调和管理发布者和客户端之间的消息发送，包括 Web 服务器和电子邮件地址。订阅用户接收所有发布至他们所订阅主题的消息，并且一个主题的所有订阅用户收到的消息都相同。

代码

此模式包括一个包含两个文件的附件：

- `ElastiCacheAllowedSecurityGroup.zip` 是一个包含安全控制区域（Lambda 代码）的压缩文件。
- `ElastiCacheAllowedSecurityGroup.yml`是部署安全控制的 CloudFormation 模板。

有关如何使用这些文件的信息，请参阅操作说明部分。

操作说明

部署安全控件

Task	描述	所需技能
将代码上传到 S3 存储桶。	创建新的 S3 存储桶或使用现有 S3 存储桶上传附加的 `ElastiCacheAllowedSecurityGroup.zip` 文件（Lambda 代码）。此存储桶必须与您要评估 AWS 区域 的资源位于同一存储桶中。	云架构师
部署 CloudFormation 模板。	使用与 S3 存储桶 AWS 区域 相同的方法打开 CloudFormation 控制台，然后部署附件中提供的`ElastiCacheAllowedSecurityControl.yml` 文件。在下一个操作说明中，提供模板参数的值。	云架构师

填写 CloudFormation 模板中的参数

Task	描述	所需技能
提供 S3 存储桶名称。	输入您在第一篇操作说明中创建或选择的 S3 存储桶的名称。此 S3 存储桶包含 Lambda	云架构师

Task	描述	所需技能
	代码的.zip 文件，并且必须与要评估 CloudFormation 的模板和资源 AWS 区域 相同。	
提供 S3 密钥。	提供 Lambda 代码 .zip 文件在 S3 存储桶中的位置，不带前导斜杠（例如，`ElasticCacheAllowedSecurityGroup.zip` 或 `controls/ElasticCacheAllowedSecurityGroup.zip`）。	云架构师
提供电子邮箱地址。	提供要接收违规通知的活动电子邮件地址。	云架构师
指定日志记录级别。	指定日志级别和详细程度。`Info` 指定有关应用程序进度的详细信息消息，应仅用于调试。`Error` 指定仍允许应用程序继续运行的错误事件。`Warning` 表示潜在的有害情况。	云架构师

确认订阅

Task	描述	所需技能
确认电子邮件订阅。	成功部署 CloudFormation 模板后，它会向您提供的电子邮件地址发送订阅电子邮件。要接收通知，您必须确认此电子邮件订阅。	云架构师

相关资源

- 在@@ AWS CloudFormation 控制台上创建堆栈（AWS CloudFormation 文档）
- 亚马逊 VPCs 和 ElastiCache 安全（亚马逊 ElastiCache 文档）

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

在启动时监控 Amazon EMR 集群的传输中加密

由 Susanne Kangnoh (AWS) 创建

摘要

此模式提供了一种安全控制，可在启动时监控 Amazon EMR 集群，并在未启用传输中加密时发送警报。

Amazon EMR 是一项 Web 服务，可让您轻松运行大数据框架（如 Apache Hadoop）来处理和分析数据。Amazon EMR 通过并行运行映射和减少步骤，使您能够以经济高效的方式处理大量数据。

数据加密可防止未经授权的用户访问或读取静态数据或传输中数据。静态数据是指存储在介质中的数据，例如每个节点上的本地文件系统、Hadoop Distributed File System (HDFS) 或通过 Amazon Simple Storage Service (Amazon S3) 的 EMR 文件系统 (EMRFS)。传输中数据是指在网络中传输并在作业之间传输的数据。传输加密支持 Apache Spark、Apache TEZ、Apache Hadoop、Apache 和 Presto 的开源加密功能。 HBase您可以通过 AWS 命令行界面 (AWS CLI) Line Interface、控制台或 AW SDKs S 创建安全配置并指定数据加密设置来启用加密。您可以通过以下两种方式为传输中加密提供加密构件：

- 通过将证书的压缩文件上传到 Amazon S3。
- 通过引用提供加密构件的自定义 Java 类。

此模式中包含的安全控制会监控 API 调用，并针对该RunJobFlow操作生成 Amazon Events CloudWatch 事件。该事件调用运行 Python 脚本的 AWS Lambda 函数。该函数从事件 JSON 输入中获取 EMR 集群 ID，并执行以下检查以确定是否存在安全违规：

- 检查 EMR 集群是否具有特定于 Amazon EMR 的安全配置。
- 如果集群确实具有安全配置，请检查是否启用了传输中加密。
- 如果集群没有安全配置，则使用 Amazon Simple Notification Service（Amazon SNS）向您提供的电子邮件地址发送警报。该通知指定 EMR 集群名称、违规详细信息、AWS 区域和账户信息，以及通知来源的 AWS Lambda ARN（Amazon 资源名称）。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 一个 S3 存储桶，用于上传随此模式提供的 Lambda 代码。
- 您希望接收违规通知的电子邮件地址。
- 已启用 Amazon EMR 日志记录，用于访问所有 API 日志。

限制

- 此检测控制是区域性的，必须部署在要监控的每个 AWS 区域中。

产品版本

- Amazon EMR 发行版 4.8.0 或更高版本。

架构

工作流程架构

自动化和扩缩

- 如果您使用的是 AWS Organizations，则可以使用 AWS Cloudformation 将模板部署 StackSets到要监控的多个账户中。

工具

Amazon Web Services

- Amazon EMR – Amazon EMR 是一个托管集群平台，可简化在 AWS 上运行大数据框架（如 Apache Hadoop 和 Apache Spark）以处理和分析海量数据的操作。通过使用这些框架和相关的开源项目，您可以处理用于分析目的的数据和业务情报工作负载。此外，您还可以使用 Amazon EMR 转换大量数据并移出/移入到其他 AWS 数据存储和数据库中，例如 Amazon S3 和 Amazon DynamoDB。
- AWS Cloudformation — AWS 可 CloudFormation 帮助您建模和设置 AWS 资源，快速一致地配置这些资源，并在资源的整个生命周期中对其进行管理。您可以使用模板来描述资源及其依赖关系，然后将它们作为堆栈一起启动和配置，而不必单独管理资源。您可以跨多个 Amazon Web Services account 和 AWS 区域管理和预置堆栈。

- AWS Cloudwatch Ev CloudWatch ents — Amazon Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。 CloudWatch 通过发送消息以响应环境、激活功能、进行更改和捕获状态信息，事件会在操作变化发生时意识到这些变化，并在必要时采取纠正措施。
- AWS Lambda – AWS Lambda 是一项计算服务，支持无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。您只需为消耗的计算时间付费 - 代码未运行时不产生费用。
- AWS SNS – Amazon Simple Notification Service（Amazon SNS）协调和管理发布者和客户端（包括 Web 服务器和电子邮件地址）之间的消息发送。订阅用户接收所有发布至他们所订阅主题的消息，并且一个主题的所有订阅用户收到的消息都相同。

代码

此模式包括一个包含两个文件的附件：

- `EMRInTransitEncryption.zip` 是一个包含安全控制区域（Lambda 代码）的压缩文件。
- `EMRInTransitEncryption.yml`是部署安全控制的 CloudFormation 模板。

有关如何使用这些文件的信息，请参阅操作说明部分。

操作说明

部署安全控件

Task	描述	所需技能
将代码上传到 S3 存储桶。	创建新的 S3 存储桶或使用现有 S3 存储桶上传附加的 `EMRInTransitEncryption.zip` 文件（Lambda 代码）。此存储桶必须与 CloudFormation 模板和您要评估的资源位于同一 AWS 区域。	云架构师
部署 CloudFormation 模板。	在与 S3 存储桶相同的 AWS 区域中打开 Cloudformation	云架构师，

Task	描述	所需技能
	控制台，然后部署附件中提供的 `EMRInTransitEncryption.yml` 文件。在下一个操作说明中，提供模板参数的值。	

填写 CloudFormation 模板中的参数

Task	描述	所需技能
提供 S3 存储桶名称。	输入您在第一篇操作说明中创建或选择的 S3 存储桶的名称。此 S3 存储桶包含 Lambda 代码的.zip 文件，并且必须与模板和要评估 CloudFormation 的资源位于相同的 AWS 区域。	云架构师
提供 S3 密钥。	指定 Lambda 代码 .zip 文件在 S3 存储桶中的位置，不带前导斜杠（例如，`EMRInTransitEncryption.zip` 或 `controls/EMRInTransitEncryption.zip`）。	云架构师
提供电子邮箱地址。	指定要接收违规通知的活动电子邮件地址。	云架构师
指定日志记录级别。	指定 Lambda 日志的日志记录级别和详细程度。`Info` 指定有关应用程序进度的详细信息消息，应仅用于调试。`Error` 指定仍允许应用程序继续运行的错误事件。`Warning` 表示潜在的有害情况。	云架构师

确认订阅

Task	描述	所需技能
确认电子邮件订阅。	成功部署 CloudFormation 模板后，它会向您提供的电子邮件地址发送一封订阅电子邮件。要接收通知，您必须确认此电子邮件订阅。	云架构师

相关资源

- 在 AWS CloudFormation 控制台上创建堆栈（AWS CloudFormation 文档）
- 加密选项（Amazon EMR 文档）

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

监控 Amazon ElastiCache 集群的静态加密

由 Susanne Kangnoh (AWS) 创建

摘要

Amazon ElastiCache 是一项 Amazon Web Services (AWS) 服务，它为在云中分配内存数据存储或缓存环境提供了高性能、可扩展且经济实惠的缓存解决方案。它从高吞吐量和低延迟的内存数据存储中检索数据。此功能使其成为缓存、会话存储、游戏、地理空间服务、实时分析和队列等实时用例的热门选择。 ElastiCache 提供 Redis 和 Memcached 数据存储，两者都提供亚毫秒级的响应时间。

数据加密有助于防止未经授权的用户在 Redis 集群及其关联数据存储系统中读取敏感数据。这包括保存在持久性介质中的数据（称为静态数据），以及在客户端和高速缓存服务器之间的网络传输过程中可能被拦截的数据（称为传输中数据）。

通过将AtRestEncryptionEnabled参数设置为 true，可以在创建复制组时为 Redis 启用静态加密。ElastiCache 启用此参数后，它将在同步、备份和交换操作期间加密磁盘，并加密存储在 Amazon Simple Storage Service（Amazon S3）中的备份。不能对现有复制组启用静态加密。创建复制组时，可以通过以下两种方式启用静态加密：

- 通过选择默认选项，该选项使用服务托管的静态加密。
- 通过使用客户托管密钥并提供来自 AWS Key Management Service (AWS KMS) 的密钥 ID 或 Amazon 资源名称（ARN）。

此模式提供了一种安全控制，用于监控 API 调用并生成有关CreateReplicationGroup操作的 Amazon Events CloudWatch 事件。此事件调用运行 Python 脚本的 AWS Lambda 函数。该函数从事件 JSON 输入中获取复制组 ID，并执行以下检查以确定是否存在安全违规：

- 检查AtRestEncryptionEnabled密钥是否存在。
- 如果AtRestEncryptionEnabled存在，则检查该值以查看其是否为真。
- 如果该AtRestEncryptionEnabled值设置为 false，则使用亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 通知设置一个变量来跟踪违规行为并向您提供的电子邮件地址发送违规消息。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 用于上传提供的 Lambda 代码的 S3 存储桶。
- 您希望接收违规通知的电子邮件地址。
- ElastiCache 已启用日志记录，用于访问所有 API 日志。

限制

- 此检测控制是区域性的，必须部署在要监控的每个 AWS 区域中。
- 该控件支持在虚拟私有云（VPC）中运行的复制组。
- 该控件支持运行以下节点类型的复制组：
  - R5、R4、R3
  - M5、M4、M3
  - T3、T2

产品版本

- ElastiCache 适用于 Redis 版本 3.2.6 或更高版本

架构

工作流程架构

自动化和扩缩

- 如果您使用的是 AWS Organizations，则可以使用 AWS Cloudformation StackSets 在要监控的多个账户中部署此模板。

工具

Amazon Web Services

- 亚马逊 ElastiCache — Amazon ElastiCache 让您可以轻松地在 AWS 云中设置、管理和扩展分布式内存缓存环境。它提供了高性能、可调整大小且经济实惠的内存缓存，同时消除了与部署和管理分布式缓存环境相关的复杂性。 ElastiCache 可与 Redis 和 Memcached 引擎一起使用。

- AWS CloudFormation — AWS CloudFormation 可帮助您建模和设置 AWS 资源，快速一致地配置这些资源，并在资源的整个生命周期中对其进行管理。您可以使用模板来描述资源及其依赖关系，然后将它们作为堆栈一起启动和配置，而不必单独管理资源。您可以跨多个 Amazon Web Services account 和 AWS 区域管理和预置堆栈。
- AWS Cloudwatch Ev CloudWatch ents — Amazon Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。 CloudWatch 通过发送消息以响应环境、激活函数、进行更改和捕获状态信息，事件会在操作变化发生时意识到这些变化，并在必要时采取纠正措施。
- AWS Lambda – AWS Lambda 是一项计算服务，支持无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。您只需为消耗的计算时间付费 - 代码未运行时不产生费用。
- Amazon SNS - Amazon Simple Notification Service（Amazon SNS）协调和管理发布者和客户端（包括 Web 服务器和电子邮件地址）之间的消息发送。订阅用户接收所有发布至他们所订阅主题的消息，并且一个主题的所有订阅用户收到的消息都相同。

代码

此模式包括一个包含两个文件的附件：

- `ElasticCache-EncryptionAtRest.zip` 是一个包含安全控制区域（Lambda 代码）的压缩文件。
- `elasticache_encryption_at_rest.yml`是部署安全控制的 CloudFormation 模板。

有关如何使用这些文件的信息，请参阅操作说明部分。

操作说明

部署安全控件

Task	描述	所需技能
将代码上传到 S3 存储桶。	创建新的 S3 存储桶或使用现有 S3 存储桶上传附加的 `ElastiCache-EncryptionAtRest.zip` 文件（Lambda 代码）。此存储桶	云架构师

Task	描述	所需技能
	必须与要评估的资源位于同一 AWS 区域中。	
部署 CloudFormation 模板。	在与 S3 存储桶相同的 AWS 区域中打开 Cloudformation 控制台，然后部署附件中提供的 `elasticache_encryption_at_rest.yml` 文件。在下一个操作说明中，提供模板参数的值。	云架构师

填写 CloudFormation 模板中的参数

Task	描述	所需技能
提供 S3 存储桶名称。	输入您在第一篇操作说明中创建或选择的 S3 存储桶的名称。此 S3 存储桶包含 Lambda 代码的.zip 文件，并且必须与模板和要评估 CloudFormation 的资源位于相同的 AWS 区域。	云架构师
提供 S3 密钥。	提供 Lambda 代码 .zip 文件在 S3 存储桶中的位置，不带前导斜杠（例如，`ElasticCache-EncryptionAtRest.zip` 或 `controls/ElasticCache-EncryptionAtRest.zip` ）。	云架构师
提供电子邮箱地址。	提供要接收违规通知的活动电子邮件地址。	云架构师

Task	描述	所需技能
指定日志记录级别。	指定日志级别和详细程度。`Info` 指定有关应用程序进度的详细信息消息，应仅用于调试。`Error` 指定仍允许应用程序继续运行的错误事件。`Warning` 表示潜在的有害情况。	云架构师

确认订阅

Task	描述	所需技能
确认电子邮件订阅。	成功部署 CloudFormation 模板后，它会向您提供的电子邮件地址发送订阅电子邮件。要接收通知，您必须确认此电子邮件订阅。	云架构师

相关资源

- 在 AWS CloudFormation 控制台上创建堆栈（AWS CloudFormation 文档）
- ElastiCache 适用于 Redis 的 At-Rest 加密（亚马逊文档 ElastiCache ）

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用 AWS Config 监控 EC2 实例密钥对

由 Wassim Benhallam (AWS)、塞尔吉奥·毕尔巴鄂·洛佩兹 (AWS) 和 Vikrant Telkar (AWS) 创作

摘要

在亚马逊网络服务 (AWS EC2) 云上启动亚马逊弹性计算云 (Amazon) 实例时，最佳做法是创建或使用现有密钥对连接到该实例。密钥对由存储在实例中的公钥和提供给用户的私钥组成，允许通过 Secure Shell (SSH) 安全访问实例并避免使用密码。但是，用户有时可能会在不附加密钥对的情况下无意中启动实例。由于密钥对只能在实例启动期间分配，因此快速识别并标记未启动的任何没有密钥对的实例都不合规非常重要。当在要求使用密钥对进行实例访问的账户或环境中工作时，这尤其有用。

此模式描述了如何在 AWS Config 中创建自定义规则来监控 EC2 实例密钥对。当实例被确定为不合规时，将使用通过亚马逊事件启动的亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 通知发送警报。 EventBridge

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 为要监控的 AWS 区域启用 AWS Config，并配置为记录所有 AWS 资源

限制

- 此解决方案是特定于区域的。所有资源均应在同一 AWS 区域 中创建。

架构

目标技术堆栈

- AWS Config
- Amazon EventBridge
- AWS Lambda
- Amazon SNS

目标架构

1. AWS Config 启动规则。
2. 该规则调用 Lambda 函数来评估实例的合规性。 EC2
3. Lambda 函数将更新的合规性状态发送到 AWS Config。
4. AWS Config 向发送事件 EventBridge。
5. EventBridge 向 SNS 主题发布合规性变更通知。
6. Amazon SNS 通过电子邮件发送警报。

自动化和扩缩

该解决方案可以监控一个区域内任意数量的 EC2 实例。

工具

工具

- AWS Config - AWS Config 是一项可让您评测、审计和评价 AWS 资源配置的服务。AWS Config 可以持续监控和记录您的 AWS 资源配置，并让您能依据配置需求自动评估记录的配置。
- Ama zon EventBridge — Amazon EventBridge 是一项无服务器事件总线服务，用于将您的应用程序与来自各种来源的数据连接起来。
- AWS Lambda – AWS Lambda 是一种无服务器计算服务，支持在不预置或管理服务器的情况下运行代码、创建工作负载感知型集群扩展逻辑、维护事件集成或管理运行时。
- 亚马逊 SNS — 亚马逊简单通知服务 (Amazon SNS) Simple Notification 是一项完全托管的消息服务，用于 (A2A) application-to-application 和 (A2P) 通信。 application-to-person

代码

Lambda 函数的代码已附加。

操作说明

创建 Lambda 函数来评估亚马逊的合规性 EC2

Task	描述	所需技能
为 Lambda 创建 AWS Identity and Access Management (IAM) 角色。	在 AWS 管理控制台上，选择 IAM，然后创建角色，使用 Lambda 作为可信实体并添加 `AmazonEventBridgeFullAccess` 和 `AWSConfigRulesExecutionRole` 权限。有关更多信息，请参阅 AWS 文档。	DevOps
创建并部署 Lambda 函数。	1. 在 Lambda 控制台上，使用 Author 从头开始创建一个函数，将 Python 3.6 作为运行时系统和之前创建的 IAM 角色。记下 Amazon 资源名称（ARN）。 2. 在代码选项卡上，选择 `lambda_function.py`，然后粘贴附加到此模式的代码。 3. 要保存更改，请选择部署。	DevOps

创建自定义 AWS Config 规则

Task	描述	所需技能
添加自定义 AWS Config 规则。	在 AWS Config 控制台上，使用以下设置添加自定义规则： • ARN – 之前创建的 Lambda 函数的 ARN	DevOps

Task	描述	所需技能
	• 触发器类型 – 配置变更 • 变更范围 – 资源 • 资源类型 — Amazon EC2 实例 有关更多信息，请参阅 AWS 文档。	

配置检测到合规性变更事件时的电子邮件通知

Task	描述	所需技能
创建 SNS 主题和订阅。	在 Amazon SNS 控制台上，使用标准作为类型创建主题，然后使用电子邮件作为协议创建订阅。 收到确认电子邮件时，请选择链接以确认订阅。 有关更多信息，请参阅 AWS 文档。	DevOps
创建用于启动 Amazon SNS 通知的 EventBridge 规则。	在 EventBridge 控制台上，使用以下设置创建规则： • 服务名称 – AWS Config • 事件类型 – 配置规则合规性变更 • 消息类型-特定的消息类型， ComplianceChangeNotification • 特定规则名称 – 您之前创建的 AWS Config 规则的名称	DevOps

Task	描述	所需技能
	• Target – SNS 主题，您之前创建的主题 有关更多信息，请参阅 AWS 文档。	

验证规则和通知

Task	描述	所需技能
创建 EC2 实例。	创建两个任意类型的 EC2 实例并附加一个密钥对，然后创建一个不带密钥对的 EC2 实例。	DevOps
验证规则。	1. 在 AWS Config 控制台的 规则 页面上，选择您的规则。 2. 要查看合规和不合规 EC2 实例，请将范围内的资源更改为全部。验证两个实例是否被列为合规，一个实例是否被列为不合规。 3. 等待收到有关实例合规状态的 Amazon SNS 电子邮件通知。 EC2	DevOps

相关资源

- 创建向 Amazon Web Services 委托权限的角色
- 在 AWS Config 中创建自定义规则
- 创建 Amazon SNS 主题
- 订阅 Amazon SNS 主题
- 在 Amazon 中创建规则 EventBridge

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

监控 IAM 根用户活动

由 Mostefa Brougui (AWS) 编写

摘要

每个 Amazon Web Services (AWS) 账户都有一个根用户。作为 AWS Identity and Access Management (IAM) 的安全最佳实践，我们建议您使用根用户来完成只有根用户才能执行的任务。有关完整列表，请参阅《Amazon Web Services account 管理参考指南》中的需要根用户凭证的任务。由于根用户对您的所有 AWS 资源和账单信息具有完全访问权限，因此我们建议您不要使用此账户并监控其是否有任何活动，这可能表明根用户凭证已泄露。

使用此模式，您可以设置一个事件驱动的架构来监控 IAM 根用户。这种模式建立了一个 hub-and-spoke解决方案，该解决方案可以监控多个 AWS 账户、分支账户，并将管理和报告集中到一个账户（中心账户）中。

使用 IAM 根用户证书时，Amazon CloudWatch 和 AWS 会分别在日志和跟踪中 CloudTrail 记录活动。在分支账户中，Amazon EventBridge 规则将事件发送到中心账户中的中央事件总线。在中心账户中，EventBridge 规则将事件发送到 AWS Lambda 函数。该函数使用 Amazon Simple Notification Service (Amazon SNS) 主题来通知您根用户活动。

在此模式中，您可以使用 AWS CloudFormation 模板在分支账户中部署监控和事件处理服务。您可以使用 T HashiCorp erraform 模板在中心账户中部署事件管理和通知服务。

先决条件和限制

先决条件

1. 在您的 AWS 环境中部署 AWS 资源的权限。
2. 部署 CloudFormation 堆栈集的权限。有关更多信息，请参阅堆栈集操作的先决条件（CloudFormation 文档）。
3. Terraform 已安装并可供使用。有关更多信息，请参阅入门 – AWS（Terraform 文档）。
4. 每个分支账户中的现有跟踪。有关更多信息，请参阅 AWS 入门 CloudTrail（CloudTrail 文档）。
5. 该跟踪配置为向 CloudWatch 日志发送事件。有关更多信息，请参阅向 CloudWatch 日志发送事件（CloudTrail 文档）。
6. 您的中心和分支账户必须由 AWS Organizations 管理。

架构

下图说明了实现的构建基块。

1. 使用 IAM 根用户证书时， CloudWatch 分别在日志和跟踪中 CloudTrail 记录活动。
2. 在分支账户中， EventBridge 规则将事件发送到中心账户中的中央事件总线。
3. 在中心账户中， EventBridge 规则将事件发送到 Lambda 函数。
4. Lambda 函数使用一个 Amazon SNS 主题来通知您根用户活动。

工具

Amazon Web Services

- AWS CloudFormation 可帮助您设置 AWS 资源，快速一致地配置这些资源，并在 AWS 账户和区域的整个生命周期中对其进行管理。
- AWS CloudTrail 可帮助您审计 AWS 账户的治理、合规和运营风险。
- Amazon CloudWatch Lo gs 可帮助您集中管理来自所有系统、应用程序和 AWS 服务的日志，以便您可以监控它们并安全地将其存档。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如，AWS Lambda 函数、使用 API 目标的 HTTP 调用端点或其他 Amazon Web Services account 中的事件总线。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Simple Notification Service (Amazon SNS) 可帮助您协调和管理发布者与客户端（包括 Web 服务器和电子邮件地址）之间的消息交换。

其他工具和服务

- Terraform 是一个 CLI 应用程序，用于使用配置文件形式的代码来配置和管理云基础架构和资源。

代码存储库

此模式的源代码和模板可在GitHub 存储库中找到。此模式提供两个模板：

- 一个 Terraform 模板，其中包含您在中心账户中部署的资源
- 在分支账户中作为堆栈集实例部署的 CloudFormation 模板

该库的总体结构如下。

```
.
 |__README.md
 |__spoke-stackset.yaml
 |__hub.tf
 |__root-activity-monitor-module
     |__main.tf  # contains Terraform code to deploy resources in the Hub account
     |__iam      # contains IAM policies JSON files
         |__ lambda-assume-policy.json          # contains trust policy of the IAM role
 used by the Lambda function
         |__ lambda-policy.json                 # contains the IAM policy attached to
 the IAM role used by the Lambda function
     |__outputs  # contains Lambda function zip code
```

Epics 部分提供了部署模板的 step-by-step说明。

操作说明

将资源部署到中心账户

Task	描述	所需技能
克隆示例代码存储库。	1. 打开 AWS IAM 根用户活动监控存储库。 2. 在代码选项卡上，文件列表上方，选择代码，然后复制 HTTPS URL。 3. 在命令行界面中，将工作目录更改为要存储示例文件的位置。 4. 输入以下命令：	常规 AWS

Task	描述	所需技能
	`git clone <repoURL>`	
更新 Terraform 模板。	1. 检索您的组织 ID。有关说明，请参阅从管理账户查看组织的详细信息（AWS Organizations 文档）。 2. 在克隆的存储库中，打开 `hub.tf`。 3. 使用适合您的环境的值更新以下内容： • `OrganizationId` – 添加您的组织 ID。 • `SNSTopicName` – 添加 Amazon SNS 主题的名称。 • `SNSSubscriptions` – 添加应向其发送 Amazon SNS 通知的电子邮件。 • `Region` – 添加要在其中部署资源的 AWS 区域代码。例如，`eu-west-1`。 • `Tags` – 添加您的标签。有关更多信息，请参阅标记 AWS 资源（AWS 一般参考）。 4. 保存并关闭 `hub.tf`文件。	常规 AWS

Task	描述	所需技能

将资源部署到 AWS 中心账户。	1. 在 Terraform 命令行界面中，导航到克隆存储库的根文件夹，然后输入以下命令。
	`terraform init && terraform plan`
	2. 查看输出并确认要创建所述资源。
	3. 输入以下命令。
	`terraform apply`
	4. 出现提示时，输入 `yes` 以确认部署。
		常规 AWS

将资源部署到分支账户

Task	描述	所需技能
部署 CloudFormation 模板。	1. 登录 AWS 管理控制台，然后打开CloudFormation 控制台。 2. 从导航窗格中，选择 StackSets。 3. 在StackSets页面顶部，选择创建 StackSet。 4. 在 “权限” 下，选择 “服务管理权限”。 CloudFormation 自动配置部署到 AWS Organizations 管理的目标账户所需的权限。	常规 AWS

Task	描述	所需技能
	5. 在先决条件 - 准备模板下，选择模板已就绪。 6. 在指定模板下，选择上传模板文件。 7. 选择选择文件，然后在克隆的存储库中选择 `spoke-stackset.yaml` 。 8. 选择下一步。 9. 在指定 StackSet 详细信息页面上，输入堆栈集的名称。 10.在参数下，输入中心账户的账户 ID，然后选择下一步。 11.在配置 StackSet 选项页面的标签下，添加您的标签。 12.在执行配置下，选择非活动，然后选择下一步。 13.在设置部署选项页面上，指定要在其中部署堆栈集的组织单位和区域，然后选择下一步。 14.在 “查看” 页面上，选择 “我确认 AWS CloudFormation 可能会创建 IAM 资源”，然后选择 “提交”。 CloudFormation 开始部署您的堆栈集。 有关更多信息和说明，请参阅创建堆栈集（CloudFormation 文档）。	

（可选）测试通知

Task	描述	所需技能
使用根用户凭证。	1. 使用根用户凭证登录分支账户或中心账户。 2. 确认您指定的电子邮件账户已收到 Amazon SNS 通知。	常规 AWS

相关资源

- 安全最佳实践（IAM 文档）
- 使用 StackSets（CloudFormation 文档）
- 入门（Terraform 文档）

其他信息

Amazon GuardDuty 是一项持续的安全监控服务，可分析和处理日志，以识别您的 AWS 环境中意外和可能未经授权的活动。作为此解决方案的替代方案，如果您已启用 GuardDuty，它可以在使用根用户凭据时提醒您。 GuardDuty 结果为`Policy:IAMUser/RootCredentialUsage`，默认严重性为“低”。有关更多信息，请参阅管理 Amazon GuardDuty 调查结果。

在创建 IAM 用户时发送通知

由 Mansi Suratwala (AWS) 和 Sergiy Shevchenko (AWS) 编写

摘要

在 Amazon Web Services (AWS) 上，您可以使用此模式部署 AWS CloudFormation 模板，以便在创建 AWS 身份和访问管理 (IAM) 用户时自动接收通知。

使用 IAM，您可以安全地管理 Amazon Web Services 和资源的访问权限。您可以创建和管理亚马逊云科技用户和组，并使用权限来允许和拒绝此类用户和组访问亚马逊云科技资源。

该 CloudFormation 模板创建了一个亚马逊 CloudWatch 事件和一个 AWS Lambda 函数。该事件使用 AWS CloudTrail 来监控在 AWS 账户中创建的任何 IAM 用户。如果创建了用户，则 CloudWatch 事件会启动 Lambda 函数，该函数会向您发送亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 通知，告知您发生了新用户创建事件。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account
- 创建并部署了 AWS CloudTrail 跟踪

限制

- AWS CloudFormation 模板`CreateUser`只能用于部署。

架构

目标技术堆栈

- IAM
- AWS CloudTrail
- 亚马逊 CloudWatch 活动
- AWS Lambda
- Amazon Simple Storage Service (Amazon S3)

- Amazon SNS

目标架构

自动化和扩缩

您可以针对不同的 AWS 区域和账户多次使用 AWS CloudFormation 模板。您只需在每个区域或账户中运行一次。要自动部署到多个账户，请使用 AWS CloudFormation StackSets。该 CloudFormation 模板将能够在每个账户中部署所有必需的资源。

工具

工具

- IAM – AWS Identity and Access Management (IAM) 是一项 Web 服务，可帮助您安全地控制对 AWS 资源的访问。可以使用 IAM 来控制谁通过了身份验证(准许登录)并获得授权(拥有权限)来使用资源。
- AWS CloudFormation — AWS CloudFormation 可帮助您建模和设置 Amazon Web Services 资源，这样您就可以减少管理这些资源的时间，将更多时间集中在在 AWS 中运行的应用程序上。您可以创建一个描述所需所有 AWS 资源的模板，并 CloudFormation 负责为您预置和配置这些资源。
- AWS CloudTrail — AWS CloudTrail 可帮助您管理 AWS 账户的治理、合规以及运营和风险审计。用户、角色或 AWS 服务采取的操作在中记录为事件 CloudTrail。事件包括在 AWS 管理控制台、AWS 命令行界面和 AWS SDKs 和 APIs。
- Amazon CloudWatch Events — Amazon CloudWatch Events 提供一系列描述了 AWS 资源变化的系统事件。 near-real-time
- AWS Lambda – AWS Lambda 是一项计算服务，支持无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项面向互联网的存储服务。您可以通过 Amazon S3 随时在 Web 上的任何位置存储和检索的任意大小的数据。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS)是一项托管服务，可通过 Lambda、HTTP、电子邮件、手机推送通知和手机短信 (SMS) 的形式提供消息。

代码

该项目的 .zip 文件作为附件提供。

操作说明

为 Lambda 脚本创建 S3 存储桶

Task	描述	所需技能
定义 S3 存储桶。	打开 Amazon S3 控制台并选择或创建 S3 存储桶。此 S3 存储桶将托管 Lambda 代码 .zip 文件。S3 存储桶名称不得包含前导斜杠。	云架构师

将 Lambda 代码上传至 S3 存储桶

Task	描述	所需技能
上传 Lambda 代码。	将附件部分中提供的 Lambda 代码 .zip 文件上传到您定义的 S3 存储桶。	云架构师

部署 CloudFormation 模板

Task	描述	所需技能
部署 CloudFormation 模板。	在 CloudFormation 控制台上，部署作为该模式附件提供的 CloudFormation `createIAMuser.yaml` 模板。在下一个操作说明中，提供模板参数的值。	云架构师

填写 CloudFormation 模板中的参数

Task	描述	所需技能
提供 S3 存储桶名称。	输入您在第一个操作说明中创建或选择的 S3 存储桶的名称。	云架构师
提供 S3 密钥。	提供 Lambda 代码 .zip 文件在 S3 存储桶中的位置，不带前导斜杠（例如，`<directory>/<file-name>.zip`）。	云架构师
提供电子邮箱地址。	提供有效的电子邮件地址以接收 Amazon SNS 通知。	云架构师
定义日志记录级别。	定义 Lambda 函数的日志记录级别和频率。`Info` 指明有关应用程序进度的详细信息消息。`Error` 指明仍允许应用程序继续运行的错误事件。`Warning` 指明潜在的有害情况。	云架构师

确认订阅

Task	描述	所需技能
确认订阅。	成功部署模板后，它将向提供的电子邮件地址发送订阅电子邮件。要接收通知，您必须确认此电子邮件订阅。	云架构师

相关资源

- 创建跟踪

- 创建 S3 存储桶
- 将文件上传到 S3 存储桶
- 部署 CloudFormation 模板
- 创建 IAM 用户
- 使用 AWS 创建在 AWS API 调用时触发 CloudWatch 的事件规则 CloudTrail

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

使用服务控制策略阻止账户级别的互联网访问

由塞尔吉·舍甫琴科（AWS）、肖恩·奥沙利文（AWS）和维克多·马泽奥·惠特克（AWS）创作

摘要

Organizations 经常希望限制本应保持私密性的账户资源的互联网访问权限。在这些账户中，虚拟私有云 (VPCs) 中的资源不应以任何方式访问互联网。许多组织选择集中检查架构。对于集中式检查架构中的东西向（VPC 到 VPC）流量，您需要确保分支账户及其资源无法访问互联网。对于南北（互联网出口和本地）流量，您只想允许通过检查 VPC 访问互联网。

此模式使用服务控制策略 (SCP) 来帮助阻止互联网访问。您可以在账户或组织单位 (OU) 级别应用此 SCP。SCP 通过防止以下情况来限制互联网连接：

- 创建或连接允许直接通过 IPv6 Internet 访问 VPC 的互联网网关 IPv4
- 创建或接受可能允许通过其他 VPC 间接访问互联网的 VPC 对等连接
- 创建或更新可能允许通过互联网直接访问 VPC 资源的AWS Global Accelerator配置

先决条件和限制

先决条件

- 一个或多个作为组织在中进行 AWS 账户 管理 AWS Organizations。
- 所有功能均已在中启用 AWS Organizations。
- SCPs 已在组织中启用。
- 权限：
  - 访问组织的管理账户。
  - 创建 SCPs。有关最低权限的更多信息，请参阅创建 SCP。
  - 将 SCP 附加到目标客户或组织单位 (OUs)。有关最低权限的更多信息，请参阅附加和分离服务控制策略。

限制

- SCPs 不要影响管理账户中的用户或角色。它们仅影响组织中的成员账户。
- SCPs 仅影响由属于组织的账户管理的 AWS Identity and Access Management (IAM) 用户和角色。有关更多信息，请参阅 SCP 对权限的影响。

工具

Amazon Web Services

- AWS Organizations是一项账户管理服务，可帮助您将多个账户整合 AWS 账户 到一个由您创建和集中管理的组织中。在此模式中，您可以在中使用服务控制策略 (SCPs) AWS Organizations。
- Amazon Virtual Private Cloud（亚马逊 VPC）可帮助您将 AWS 资源启动到您定义的虚拟网络中。该虚拟网络类似于您在数据中心中运行的传统网络，并具有使用 AWS的可扩展基础设施的优势。

最佳实践

在您的组织中建立此 SCP 后，请务必经常对其进行更新，以解决可能影响互联网访问的任何新功能 AWS 服务 或新功能。

操作说明

创建并附加 SCP

Task	描述	所需技能
创建 SCP。	1. 登录 AWS Organizations 控制台。您必须登录组织的管理帐户。 2. 在左侧窗格中，选择 “策略”。 3. 在策略页面上，选择服务控制策略。 4. 在 Service control policies (服务控制策略) 页面上，选择 Create policy (创建策略)。 5. 在创建新的服务控制策略页面上，输入策略名称和可选的策略描述。 6. （可选）向您的策略添加AWS 标签。	AWS 管理员

Task	描述	所需技能
	7. 在 JSON 编辑器中，删除占位符策略。 8. 将下面的 策略粘贴到 JSON 编辑器中。 `{ "Version": "2012-10-17", "Statement": [{ "Action": ["ec2:AttachInternetGateway", "ec2:CreateInternetGateway", "ec2:CreateVpcPeeringConnection", "ec2:AcceptVpcPeeringConnection", "ec2:CreateEgressOnlyInternetGateway"], "Resource": "*", "Effect": "Deny" }, { "Action": ["globalaccelerator:Create*", "globalaccelerator:Update*"], "Resource": "*", "Effect": "Deny" }`	

Task	描述	所需技能

	``
	  ]
	}
	``
	9. 选择创建策略。	

附上 SCP。	1. 在服务控制策略页面上，选择您创建的策略。 2. 在 Targets (目标) 选项卡上，选择 Attach (附加)。 3. 选择要将策略附加到的 OU 或账户。您可能需要展开 OUs 才能找到所需的 OU 或帐户。 4. 选择附加策略。	AWS 管理员

相关资源

- AWS Organizations 文档
- 服务控制策略 (SCPs)
- 使用 AWS Gateway Load Balancer 和 AWS Transit Gateway（AWS 博客文章）的集中检查架构

使用，根据 IP 地址或地理位置限制访问 AWS WAF

由 Louis Hourcade (AWS) 创作

摘要

AWS WAF是一种 Web 应用程序防火墙，可帮助保护 Web 应用程序以及可能影响可用性、危及安全性或消耗过多资源的常见 Web 漏洞和僵尸程序的 APIs 侵害。通过中的 AWS WAF Web 访问控制列表 (Web ACLs)，您可以控制流量如何到达您的应用程序。在 Web ACL 中，您可以添加旨在允许合法流量、控制机器人流量和阻止常见攻击模式的规则或规则组。有关更多信息，请参阅AWS WAF 工作原理。

您可以将以下类型的规则关联到您的 AWS WAF 网站 ACLs：

- 托管规则组 — AWS 托管规则团队和 AWS Marketplace 卖家提供预配置的规则集。一些托管规则组旨在帮助保护特定类型的 Web 应用程序。其他提供针对已知威胁或常见漏洞的广泛保护。
- 自定义规则和自定义规则组 —您还可以创建规则和规则组，以自定义对Web应用程序的访问权限，以及 APIs. 例如，您可以根据特定的 IP 地址列表或国家/地区列表来限制流量。

通过使用此模式和关联的代码存储库，您可以使用 ACLs 使用自定义规则部署 AWS WAF Web。AWS Cloud Development Kit (AWS CDK)这些规则根据最终用户的 IP 地址或地理位置限制对 Web 应用程序资源的访问。您也可以选择附加多个托管规则组。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 部署 AWS WAF 资源的@@ 权限
- AWS CDK，已在您的账户中安装和配置
- Git，已安装

限制

- 您只能在可用 AWS 区域 的地方 AWS WAF 使用此模式。有关区域可用性，请参阅AWS 服务 按地区划分。

工具

AWS 服务

- AWS Cloud Development Kit (AWS CDK)是一个软件开发框架，可帮助您在代码中定义和配置 AWS Cloud 基础架构。
- AWS WAF是一种 Web 应用程序防火墙，可帮助您监控转发到受保护的 Web 应用程序资源的 HTTP 和 HTTPS 请求。

代码存储库

此模式的代码可在 AWS WAF存储库的 GitHub IP 和地理位置限制中找到。该代码部署了两个 AWS WAF 网络 ACLs。第一个是专为 Amazon API Gateway 资源设计的区域网络 ACL。第二个是 Amazon CloudFront 资源的全球 Web ACL。两个网站都 ACLs 包含以下自定义规则：

- `IPMatch`阻止来自不允许的 IP 地址的请求。
- `GeoMatch`屏蔽来自不允许的国家/地区的请求。

在部署期间，您可以选择将以下所有托管规则组附加到您的 Web ACLs：

- 核心规则集 (CRS)-此规则组包含通常适用于 Web 应用程序的规则。它有助于防止各种漏洞被利用，包括OWASP出版物中描述的一些高风险和常见漏洞，例如 O WASP Top 10。
- 管理员保护-此规则组包含可帮助您阻止外部访问公开的管理页面的规则。
- 已知的错误输入 — 此规则组可帮助阻止已知无效且与漏洞利用或发现相关的请求模式。
- Amazon IP 信誉列表-此规则组包含基于亚马逊内部威胁情报的规则。它可以帮助您屏蔽通常与机器人或其他威胁相关的 IP 地址。
- Linux 操作系统托管规则组 — 此规则组有助于阻止与利用 Linux 漏洞相关的请求模式，包括 Linux 特有的本地文件包含 (LFI) 攻击。
- SQL 数据库托管规则组-此规则组有助于阻止与 SQL 数据库利用相关的请求模式，例如 SQL 注入攻击。

操作说明

配置 AWS WAF 网络 ACLs

Task	描述	所需技能
克隆存储库。	输入以下命令，将带有 AWS WAF存储库的 IP 和地理定位限制克隆到本地工作站： `git clone https://github.com/aws-samples/ip-and-geolocation-restriction-with-waf-cdk.git`	Git

配置规则。	1. 在克隆的存储库中，打开 app.py 文件。
	2. 修改以下变量的值以自定义规则：
	`aws_acccount = "AWS_ACCOUNT"`
	`region = "AWS_REGION"`
	`ip_list = ["CIDR_RANGE_1", "CIDR_RANGE_2"]`
	`geo_list = ["COUNTRY_CODE_1", "COUNTRY_CODE_2"]`
	`aws_managed_rules = True`
	其中：
	• `aws_account` 是目标的 ID AWS 账户。
		通用 AWS、Python

Task	描述	所需技能
	• `region`是 API Gateway 资源的网络 ACL 的目标 AWS 区域 。 **Note** CloudFront 资源的 Web ACL 是全球性的，将部署在该`us-east-1` 区域。 • `ip_list`是允许访问的 CIDR 范围列表。 • `geo_list`是允许访问的国家/地区列表。有关有效值，请参阅AWS WAF 文档。 • `aws_managed_rules` 控制是否将托管规则组添加到 Web ACL。如果此值为`True`，则将其相加。如果此值为`False`，则将其排除在外。 3. 保存并关闭 app.py 文件。	

引导并部署代码

Task	描述	所需技能
引导您的 AWS 环境。	如果尚未完成此操作，则需要先引导您的 AWS 环境，然后	常规 AWS

Task	描述	所需技能

	才能部署 AWS CDK 应用程序。
	1. 在 AWS CDK CLI 中，输入以下命令以引导该us-east-1 区域：
	`cdk bootstrap aws://<account-id>/us-east-1`
	2. 如果您要将 API Gateway 的 Web ACL 部署到其他区域us-east-1 ，请输入以下命令来引导目标区域：
	`cdk bootstrap aws://<account-id>/<region>`	

部署 AWS CDK 应用程序。	1. 输入以下命令部署 AWS CDK 应用程序： `cdk deploy --all` 2. 等到 AWS CloudFormation 堆栈部署完成。	常规 AWS

验证部署

Task	描述	所需技能
确认 Web ACLs 已成功部署。	1. 登录 AWS Management Console，然后打开AWS WAF 控制台。	常规 AWS

Task	描述	所需技能
	2. 在导航窗格中，选择 Web ACLs。 3. 在列表中 AWS 区域，选择全局 (CloudFront)。 4. 验证新的 CloudFront Web ACL 是否已部署，并确认它具有您定义的 IP 地址和地理定位规则。此 Web ACL 的默认名称为`WebACLCloudfront-<ID>` 。 5. 在列表中 AWS 区域，选择您部署堆栈的区域。 6. 确认已为 API Gateway 资源部署了新的 Web ACL。确认它具有您定义的 IP 地址和地理定位规则。此 Web ACL 的默认名称为`WebACLApiGW-<ID>` 。	
（可选）将 Web ACLs 与您的资源关联。	将 AWS WAF 网站 ACLs 与您的 AWS 资源相关联，例如应用程序负载均衡器、API Gateway 或 CloudFront 分发。有关说明，请参阅将 Web ACL 与资源关联或取消关联。AWS有关示例，请参阅 AWS CDK 文档中的类 CfnWebACLAssociation（构造）。	常规 AWS

清理资源

Task	描述	所需技能
删除堆栈。	1. 解除 Web ACL 与任何 AWS 资源的关联。有关说明，请参阅AWS WAF 文档。 2. 在 AWS CDK CLI 中，输入以下命令以删除 AWS CDK 应用程序。 `cdk destroy --all`	常规 AWS

相关资源

- API 参考（AWS CDK 文档）
- aws-cdk-lib.aws_wafv2 模块（文档）AWS CDK
- 使用 Web ACLs（AWS WAF 文档）
- 管理自己的规则组（AWS WAF 文档）
- 规则（AWS WAF 文档）

使用 git-secrets 扫描 Git 存储库中的敏感信息及安全问题

由 Saurabh Singh (AWS) 创建

摘要

此模式描述了如何使用 AWS 实验室的开源 git-sec rets 工具扫描 Git 源代码库，并查找可能包含敏感信息（例如用户密码或 AWS 访问密钥）或存在任何其他安全问题的代码。

`git-secrets` 扫描提交、提交消息和合并，以防止密钥等敏感信息被添加至 Git 存储库。例如，如果提交、提交消息或合并历史记录中的任何提交与您配置的、禁用的正则表达式模式之一相匹配，则该提交将被拒绝。

先决条件和限制

先决条件

- 活跃的 AWS 账户
- 需安全扫描的 Git 存储库
- Git 客户端（版本 2.37.1 及更高版本）已安装

架构

目标架构

- Git
- `git-secrets`

工具

- git-secrets是一种防止您将敏感信息提交至 Git 存储库的工具。
- Git 是开源分布式版本控制系统。

最佳实践

- 务必通过包含所有修订版来扫描 Git 仓库：

```
git secrets --scan-history
```

操作说明

Connect 连接到 EC2 实例

Task	描述	所需技能
使用 SSH 连接到 EC2 实例。	使用 SSH 和密钥对文件连接到亚马逊弹性计算云 (Amazon EC2) 实例。 如果您要在本地计算机上扫描存储库，则可跳过此步骤。	常规 AWS

安装 Git

Task	描述	所需技能
安装 Git。	通过以下命令安装 Git： `yum install git -y` 如果使用的是本地计算机，则可安装适用于特定操作系统版本的 Git 客户端。有关更多信息，请参阅 Git 网站。	常规 AWS

克隆源存储库并安装 git-secrets

Task	描述	所需技能
克隆 Git 源存储库。	若要克隆待扫描 Git 存储库，请从主目录中选择 Git 克隆命令。	常规 AWS
克隆 git-secrets。	克隆 `git-secrets` Git 存储库。 `git clone https://github.com/awslabs/git-secrets.git` 将 `git-secrets` 放在 PATH 的某个地方，以便其在您运行 `git-secrets` 时被 Git 选中。	常规 AWS
安装 git-secrets。	对于 Unix 及其变体 (Linux/macOS)： 您可以使用 `Makefile` 的 `install` 目标（在 `git-secrets` 存储库中提供）安装工具。您可以使用 `PREFIX` 和 `MANPREFIX` 变量自定义安装路径。 `make install` 对于 Windows： 运行`git-secrets` 存储库中提供的 PowerShell `install.ps1` 脚本。此脚本将安装文件复制到安装目录	常规 AWS

Task	描述	所需技能

	（默认为 %USERPROFILE %/.git-secrets ），并将该目录添加至当前用户 PATH。
	`PS > ./install.ps1`
	对于 Homebrew（macOS 用户）：
	运行：
	`brew install git-secrets`	

扫描 git 代码存储库

Task	描述	所需技能
前往源存储库。	切换至要扫描的 Git 存储库目录： `cd my-git-repository`	常规 AWS
注册 AWS 规则集（Git 钩子）。	若要配置 `git-secrets` 以在每次提交时扫描 Git 存储库，请运行此命令： `git secrets --register-aws`	常规 AWS
扫描 存储库。	运行以下命令，以开始扫描存储库： `git secrets --scan`	常规 AWS

Task	描述	所需技能

查看输出文件。	如果在您的 Git 存储库中发现了漏洞，该工具将生成输出文件。例如：
	
	`example.sh:4:AWS_SECRET_ACCESS_KEY = *********`
	
	`[ERROR] Matched one or more prohibited patterns`
	
	`Possible mitigations:`
	`- Mark false positives as allowed using: git config --add secrets.allowed ...`
	`- Mark false positives as allowed by adding regular expressions to .gitallowed at repository's root directory`
	`- List your configured patterns: git config --get-all secrets.patterns`
	`- List your configured allowed patterns: git config --get-all secrets.allowed`
	`- List your configured allowed patterns in .gitallowed at repository's root directory`
	`- Use --no-verify if this is a one-time false positive`
		常规 AWS

相关资源

- git-secrets 工具

将警报从 AWS Network Firewall 发送到 Slack 通道

由 Venki Srivatsav (AWS) 和 Aromal Raj Jayarajan (AWS) 创建

摘要

此模式描述了如何使用带有分布式部署模型的 Amazon Web Services (AWS) Network Firewall 来部署防火墙以及如何将 AWS Network Firewall 生成的警报传播至可配置 Slack 通道。

支付卡行业数据安全标准 (PCI DSS) 等合规性标准要求您安装并维护防火墙来保护客户数据。在 Amazon Web Service Cloud 中，在此类合规性要求的上下文中虚拟私有云（VPC）被视为与物理网络相同。您可以使用 Network Firewall 监控 VPCs 和之间的网络流量，以保护 VPCs 受合规性标准管理的工作负载。Network Firewall 在检测到同一账户 VPCs 中其他人未经授权的访问时会阻止访问或生成警报。但是，Network Firewall 支持的警报发送目标有限。这些目标包括亚马逊简单存储服务 (Amazon S3) Service 存储桶、 CloudWatch 亚马逊日志组和亚马逊数据 Firehose 传输流。对这些通知的任何进一步操作都需要使用 Amazon Athena 或 Amazon Kinesis 进行离线分析。

此模式提供了将 Network Firewall 生成的警报传播至可配置的 Slack 通道的方法，以近实时地采取进一步行动。您还可以将该功能扩展到其他警报机制 PagerDuty，例如 Jira 和电子邮件。（此类自定义超出了此模式的范围。）

先决条件和限制

先决条件

- Slack 通道（请参阅 Slack 帮助中心的入门）
- 向通道发送消息所需权限
- 带 API 令牌的 Slack 端点 URL（选择您的应用程序并选择传入的 Webhook 以查看其 URL；有关更多信息，请参阅 Slack API 文档中的创建传入 Webhook）
- 工作负载子网中的亚马逊弹性计算云 (Amazon EC2) 测试实例
- Network Firewall 中的测试规则
- 触发测试规则的实际或模拟流量
- 用于存放待部署源文件的 S3 存储桶

限制

- 目前，此解决方案仅支持单个无类域间路由 (CIDR) 范围作为源和目标的过滤器。 IPs

架构

目标技术堆栈

- 一个 VPC
- 四个子网（两个用于防火墙，两个用于工作负载）
- 互联网网关
- 四个带规则的路由表
- 用作警报目标且配置了用于运行 Lambda 函数的存储桶策略和事件设置的S3 存储桶
- 用于发送 Slack 通知的执行角色的 Lambda 函数
- 存储 Slack URL 的 AWS Secrets Manager 密钥
- 带警报配置的网络防火墙
- Slack 通道

除 Slack 通道之外的所有组件均由此模式提供的 CloudFormation 模板和 Lambda 函数进行配置（参见“代码”部分）。

目标架构

此模式设置了集成 Slack 的去中心化网络防火墙。此架构由包含两个可用区的 VPC 组成。VPC 包括两个受保护子网和两个带网络防火墙端点的防火墙子网。通过创建防火墙策略和规则，可以监控所有进出受保护子网的流量。网络防火墙配置为将所有警报置入 S3 存储桶。此 S3 存储桶配置为收到 `put` 事件时调用 Lambda 函数。Lambda 函数从 Secrets Manager 获取配置的 Slack 网址，并将通知消息发送到 Slack 工作空间。

有关此架构的更多信息，请参阅 AWS Blog 文章 AWS Network Firewall 部署模型。

工具

Amazon Web Services

- AWS Network Fire wall 是一项有状态的托管式网络防火墙以及入侵检测和防御服务，适用于 AWS 云 VPCs 中。您可以使用 Network Firewall 在 VPC 外围筛选网络流量，并保护 AWS 上的工作负载。

- AWS Secrets Manager 是一项用于存储和检索凭证的服务。使用 Secrets Manager，您可以将代码中的硬编码凭证（包括密码）替换为对 Secrets Manager 的 API 调用，以便以编程方式检索密钥。此模式使用了 Secrets Manager 存储 Slack URL。
- Amazon Simple Storage Service (Amazon S3) 是一项对象存储服务。您可以通过 Amazon S3 随时在 Web 上的任何位置存储和检索的任意大小的数据。此模式使用 Amazon S3 来存储 Lambda 函数的 CloudFormation 模板和 Python 脚本。其亦将 S3 存储桶用作网络防火墙警报目标。
- AWS CloudFormation 可帮助您建模和设置 AWS 资源，快速一致地配置这些资源，并在资源的整个生命周期中对其进行管理。您可以使用模板来描述资源及其依赖关系，然后将它们作为堆栈一起启动和配置，而不必单独管理资源。此模式使用 AWS 自动 CloudFormation 为 Firewall Manager 部署分布式架构。

代码

此模式的代码可在 GitHub Network Fi rewall Slack 集成存储库中找到。在存储库的 `src` 文件夹中，您将发现：

- 一组 YAML 格式的 CloudFormation 文件。您将使用此类模板为此模式配置组件。
- 用于创建 Lambda 函数的 Python 源文件（`slack-lambda.py`）。
- 用于上传您的 Lambda 函数代码的 .zip 存档部署包（`slack-lambda.py.zip`）。

要使用这些文件，请按照下一节中的说明操作。

操作说明

设置 S3 存储桶

Task	描述	所需技能
创建 S3 存储桶。	1. 登录 AWS 管理控制台并在上打开 Amazon S3 控制台https://console.aws.amazon.com/s3/。 2. 选择或创建 S3 存储桶以托管代码。S3 存储桶名称是全局唯一的，并且命名空间由所有 Amazon Web Services	应用程序开发人员、应用程序所有者、云管理员

Task	描述	所需技能
	account 共享。S3 存储桶名称不得包含前导斜杠。建议使用前缀来组织此模式的代码。 有关更多信息，请参阅 Amazon S3 文档中的创建存储桶。	
上传 CloudFormation 模板和 Lambda 代码。	1. 从GitHub 存储库中下载此模式的以下文件： • `base.yml` • `igw-ingress-route.yml` • `slack-lambda.py` • `slackLambda.yml` • `decentralized-deployment.yml` • `protected-subnet-route.yml` • `slack-lambda.py.zip` 2. 将文件上传至您创建的 S3 存储桶。	应用程序开发人员、应用程序所有者、云管理员

部署 CloudFormation 模板

Task	描述	所需技能
启动 CloudFormation 模板。	在与 S3 存储桶相同的 AWS 区域中打开 AWS CloudFormation 控制台并部署模板`base.yml`。此模板创	应用程序开发人员、应用程序所有者、云管理员

Task	描述	所需技能
	建了将警报传输至 Slack 通道所需 AWS 资源和 Lambda 函数。 有关部署 CloudFormation 模板的更多信息，请参阅 CloudFormation 文档中的在 AWS CloudFormation 控制台上创建堆栈。	
填写模板中的参数。	指定堆栈名称并配置参数值。有关参数及其描述和默认值的列表，请参阅 “其他信息” 部分中的CloudFormation 参数。	应用程序开发人员、应用程序所有者、云管理员
创建堆栈。	1. 查看堆栈详细信息并根据您的环境要求更新值。 2. 选择创建堆栈，以部署模板。	应用程序开发人员、应用程序所有者、云管理员

验证解决方案

Task	描述	所需技能
测试部署。	使用 AWS CloudFormation 控制台或 AWS 命令行界面 (AWS CLI) 来验证目标技术堆栈部分中列出的资源是否已创建。 如果 CloudFormation 模板未能成功部署，请检查您为`pAvailabilityZone1` 和`pAvailabilityZone2` 参数提供的值。	应用程序开发人员、应用程序所有者、云管理员

Task	描述	所需技能
	这些应该适用于您部署解决方案的 AWS 区域。有关每个区域的可用区列表，请参阅 Amazon EC2 文档中的区域和区域。	

Task	描述	所需技能

测试功能。	1. 打开亚马逊 EC2 控制台，网址为https://console.aws.amazon.com/ec2/。
	
	2. 在其中一个受保护的子网中创建 EC2 实例。选择要用作 HTTPS 服务器的 Amazon Linux 2 AMI (HVM)。有关说明，请参阅 Amazon EC2 文档中的启动实例。
	
	Note
	亚马逊 Linux 2 的支持已接近终止。有关更多信息，请参阅亚马逊 Linux 2 FAQs。
	
	3. 使用以下用户数据在 EC2 实例上安装 Web 服务器：
	
	`#!/bin/bash`
	`yum install httpd -y`
	`systemctl start httpd`
	`systemctl stop firewalld`
	`cd /var/www/html`
	`echo "Hello!! this is a NFW alert test page, 200 OK" > index.html`
	
	4. 创建以下网络防火墙规则：
	
	无状态规则：
	
	`Source: 0.0.0.0/0`
		应用程序开发人员、应用程序所有者、云管理员

Task	描述	所需技能
	Destination 10.0.3.65 /32 (private IP of the EC2 instance) Action: Forward 有状态规则： Protocol: HTTP Source ip/port: Any / Any Destination ip/port: Any /Any 5. 获取在步骤 3 中创建的 Web 服务器的公有 IP。 6. 通过浏览器访问公有 IP。您将在浏览器中看到以下消息： Hello!! this is a NFW alert test page, 200 OK 您还将在 Slack 通道中收到通知。通知可能会延迟，但具体取决于消息的大小。出于测试目的，可以考虑提供不太窄的 CIDR 过滤器（例如，带有 /32 的 CIDR 过滤器会被认为太窄，带有 /8 的 CIDR 过滤器则会被认为太宽）。有关更多信息，请参阅其他信息中的筛选行为部分。	

相关资源

- AWS Network Firewall 的部署模型（AWS Blog 文章）
- AWS Network Firewall 策略（AWS 文档）
- Network Firewall Slack 集成（GitHub 存储库）
- 创建 Slack 工作空间（Slack 帮助中心）

其他信息

CloudFormation 参数

参数	描述	默认值或示例值
`pVpcName`	要创建的 VPC 的名称。	检查
`pVpcCidr`	要创建 VPC 的 CIDR 范围。	10.0.0.0/16
`pVpcInstanceTenancy`	EC2 实例如何在物理硬件之间分布。选项有 `default`（共享租赁）或 `dedicated`（单一租赁）。	默认值
`pAvailabilityZone1`	基础设施中的第一个可用区。	us-east-2a
`pAvailabilityZone2`	基础设施中的第二个可用区。	us-east-2b
`pNetworkFirewallSubnet1Cidr`	首个防火墙子网的 CIDR 范围（最小值 /28）。	10.0.1.0/24
`pNetworkFirewallSubnet2Cidr`	第二个防火墙子网的 CIDR 范围（最小值 /28）。	10.0.2.0/24
`pProtectedSubnet1Cidr`	第一个受保护（工作负载）子网的 CIDR 范围。	10.0.3.0/24
`pProtectedSubnet2Cidr`	第二个受保护（工作负载）子网的 CIDR 范围。	10.0.4.0/24

pS3BucketName	您上传 Lambda 源代码的现有 S3 存储桶的名称。	us-w2-yourname-lambda-functions
pS3KeyPrefix	您上传 Lambda 源代码的 S3 存储桶的前缀。	aod-test
pAWSSecretName4Slack	保存 Slack URL 的密钥名称。	SlackEnpoint-Cfn
pSlackChannelName	您创建的 Slack 通道名称。	somename-notifications
pSlackUserName	Slack 用户名。	Slack 用户
pSecretKey	这可以是任何密钥。建议保留默认值。	webhookUrl
pWebHookUrl	Slack URL 的值	https://hooks.slack.com/services/T？？？9T？？/A031885JRM7/9D4Y？？？？？？
pAlertS3Bucket	用作网络防火墙警报目标的 S3 存储桶的名称。将为您创建此存储桶。	us-w2-yourname-security-aod-alerts
pSecretTagName	密钥标签名称。	AppName
pSecretTagValue	指定标签名称的标签值。	LambdaSlackIntegration
pdestCidr	目标 CIDR 范围的筛选条件。有关更多信息，请参阅下一部分：筛选行为。	10.0.0.0/16
pdestCondition	用于指示排除或包含目标匹配项的标记。有关更多信息，请参阅下一部分。有效值为 include 和 exclude。	情况如：
psrcCidr	要发出警报的源 CIDR 范围的筛选条件。有关更多信息，请参阅下一部分。	118.2.0.0/16

psrcCondition	排除或纳入源匹配的标记。有关更多信息，请参阅下一部分。	情况如：

筛选行为

如果尚未在 AWS Lambda 中配置任何筛选条件，则所有已生成警报均将发送至您的 Slack 通道。生成的警报 IPs 的来源和目的地与您在部署 CloudFormation 模板时配置的 CIDR 范围相匹配。如果已找到匹配项，则应用此条件。如果源或目标位于配置 CIDR 范围内，并且其中至少有一个配置了 `include` 条件，则会生成警报。下表提供了 CIDR 值、条件和结果的示例。

	已配置 CIDR	警报 IP	已配置	警报
源	10.0.0.0/16	10.0.0.25	情况如：	是
目标位置	100.0.0.0/16	202.0.0.13	情况如：	

	已配置 CIDR	警报 IP	已配置	警报
源	10.0.0.0/16	10.0.0.25	exclude	否
目标位置	100.0.0.0/16	202.0.0.13	情况如：	

	已配置 CIDR	警报 IP	已配置	警报
源	10.0.0.0/16	10.0.0.25	情况如：	是
目标位置	100.0.0.0/16	100.0.0.13	情况如：	

	已配置 CIDR	警报 IP	已配置	警报
源	10.0.0.0/16	90.0.0.25	情况如：	是
目标位置	Null	202.0.0.13	情况如：	

	已配置 CIDR	警报 IP	已配置	警报
源	10.0.0.0/16	90.0.0.25	情况如：	否
目标位置	100.0.0.0/16	202.0.0.13	情况如：	

使用 AWS Private CA 和 AWS RAM 简化私有证书管理

由 Everett Hinckley (AWS) 和 Vivek Goyal (AWS) 编写

摘要

您可以使用 AWS 私有证书颁发机构 (AWS Private CA) 颁发私有证书，以验证内部资源和签署计算机代码。此模式为快速部署多级 CA 层次结构和一致的配置体验提供了 AWS CloudFormation 模板。或者，您可以使用 AWS Resource Access Manager (AWS RAM) 在您的组织或 AWS 组织中的组织单位 (OUs) 中安全地共享 CA，并在使用 AWS RAM 管理权限的同时集中管理 CA。无需在每个账户中都使用私有 CA，因此这种方法可为您节省资金。此外，您还可以使用 Amazon Simple Storage Service (Amazon S3) 存储证书吊销列表 (CRL) 与访问日志。

此实施提供以下功能和优点：

- 使用 AWS Private CA 集中并简化私有 CA 层次结构的管理。
- 将证书和密钥导出至 AWS 和本地客户托管的设备。
- 使用 AWS CloudFormation 模板实现快速部署和一致的配置体验。
- 创建私有根 CA 以及 1、2、3 或者 4 个从属 CA 层次结构。
- （可选）使用 AWS RAM 与其他组织或 OU 级别的账户共享终端实体下属 CA。
- 使用 AWS RAM，无需在每个账户中使用私有 CA，从而节省资金。
- 为 CRL 创建可选 S3 存储桶。
- 为 CRL 访问日志创建可选 S3 存储桶。

先决条件和限制

先决条件

如果您想在 AWS Organizations 结构中共享 CA，请确定或设置以下内容：

- 用于创建 CA 层次结构与共享的安全账户。
- 单独 OU 或用于测试的账户。
- 在 AWS Organizations 管理账户中启用共享。有关更多信息，请参阅 AWS RAM 文档中的在 AWS Organizations 内启用资源共享。

限制

- CAs 是区域资源。全部 CAs 存放在单个 AWS 账户和单个 AWS 区域中。
- 不支持用户生成的证书与密钥。对于此情况，我们建议您自定义此解决方案，以使用外部根 CA。
- 不支持公共 CRL 存储桶。我们建议您将 CRL 保密。如果需要互联网访问 CRL，请参阅 AWS Private CA 文档 CRLs 中启用 S3 区块公有访问 (BPA) 功能中关于使用 Amazon CloudFront 提供服务的部分。
- 这种模式实现单区域方法。如果您需要多区域证书颁发机构，则可以在第二个 AWS 区域或本地实施下属机构。这种复杂性超出了该模式的范围，因为实现取决于您的特定用例、工作负载量、依赖性和要求。

架构

目标技术堆栈

- AWS Private CA
- AWS RAM
- Amazon S3
- AWS Organizations
- AWS CloudFormation

目标架构

此模式提供了两个用于共享到 AWS Organizations 的选项：

选项 1 – 在组织级别创建共享。组织中的所有账户都可使用共享 CA 颁发私有证书，如下图所示。

选项 2 – 在组织单位 (OU) 级创建共享。仅指定 OU 中的账户才能使用共享 CA 颁发私有证书。例如，在下图中，如果共享是在沙盒 OU 级创建的，则开发人员 1 和开发人员 2 都可以使用共享 CA 颁发私有证书。

工具

Amazon Web Services

- AWS Private CA — AWS Private Certificate Authority (AWS Private CA) 是一项托管式私有证书颁发机构服务，可用于签发和撤销私有数字证书。它可以帮助您创建私有 CA 层次结构，包括根和下级 CA CAs，而无需支付运营本地 CA 的投资和维护成本。
- AWS RAM — AWS Resource Access Manager (AWS RAM) 可帮助您在 AWS 账户之间、您的组织内或 AWS 组织 OUs 中安全地共享资源。为了减少多账户环境中的运营开销，您可以创建资源并使用 AWS RAM 跨账户共享该资源。
- AWS Organizations — AWS Organizations 是一项账户管理服务，可让您将多个 Amazon Web Services account 整合到您创建并集中管理的组织中。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项对象存储服务。您可以通过 Amazon S3 随时在 Web 上的任何位置存储和检索的任意大小的数据。此模式使用 Amazon S3 存储证书吊销列表 (CRL) 和访问日志。
- AWS CloudFormation — AWS CloudFormation 可帮助您建模和设置 AWS 资源，快速一致地配置这些资源，并在资源的整个生命周期中对其进行管理。您可以使用模板来描述资源及其依赖关系，然后将它们作为堆栈一起启动和配置，而不必单独管理资源。此模式使用 AWS CloudFormation 自动部署多级 CA 层次结构。

代码

此模式的源代码可在 AWS 私 GitHub有 CA 层次结构存储库中找到。存储库包括：

- AWS CloudFormation 模板`ACMPCA-RootCASubCA.yaml`。您可使用此模板为此实现部署 CA 层次结构。
- 测试请求、导出、描述和删除证书等用例的文件。

若要使用这些文件，请按照操作说明部分中的说明操作。

操作说明

构建 CA 层次结构

Task	描述	所需技能
收集证书主题信息。	收集有关证书所有者的证书主题信息：组织名称、组织单位、国家、州、地区以及公用名。	云架构师、安全架构师、PKI 工程师

Task	描述	所需技能
收集有关 AWS Organizations 的可选信息。	如果 CA 将成为 AWS Organizations 结构的一部分，并且您想在该结构内共享 CA 层次结构，请收集管理账号、组织 ID 以及可选的 OU ID（如果您只想与特定 OU 共享 CA 层次结构）。此外，还要确定您要与之共享 CA 的 OUs AWS Organizations 账户（如果有）。	云架构师、安全架构师、PKI 工程师
设计 CA 层次结构。	确定哪个账户将容纳根和下属 CAs。确定根证书和最终实体证书之间的层次结构需要多少个从属级别。有关更多信息，请参阅 AWS Private CA 文档中的设计 CA 层次结构。	云架构师、安全架构师、PKI 工程师
确定 CA 层次结构的命名和标记约定。	确定 AWS 资源的名称：根 CA 和每个从属 CA。确定应将哪些标签分配给每个 CA。	云架构师、安全架构师、PKI 工程师
确定所需加密与签名算法。	确定了以下内容： • 您的组织对 CA 在颁发证书时使用的公钥的加密算法要求。默认值为 RSA_2048。 • 您的 CA 的证书签名密钥算法。默认为 SHA256 WITHRSA。	云架构师、安全架构师、PKI 工程师
确定 CA 层次结构证书吊销要求。	如果需要证书吊销功能，请为包含证书吊销列表 (CRL) 的 S3 存储桶创建命名约定。	云架构师、安全架构师、PKI 工程师

Task	描述	所需技能
确定 CA 层次结构日志记录要求。	如果需要访问日志记录功能，请为包含访问日志的 S3 存储桶建立命名约定。	云架构师、安全架构师、PKI 工程师
确定证书到期时间。	确定根证书（默认为 10 年）、最终实体证书（默认为 13 个月）和从属 CA 证书（默认为 3 年）的到期日期。从属 CA 证书应早于层次结构中较高级别的 CA 证书过期。有关更多信息，请参阅 AWS Private CA 文档中的管理私有 CA 生命周期。	云架构师、安全架构师、PKI 工程师

部署 CA 层次结构

Task	描述	所需技能
完成 必备任务。	完成此模式先决条件部分中的步骤。	云管理员、安全工程师、PKI 工程师
为各种角色创建 CA 角色。	1. 确定 AWS IAM 身份中心（AWS Single Sign-On 的继任者）中管理不同级别的 CA 层次结构（例如根 CAAdmin、从属CAAdmin 和）所需的 AWS IAM 身份与访问管理 (IAM) 角色或用户的类型。 CertificateConsumer 2. 确定划分职责所需政策精细度。	云管理员、安全工程师、PKI 工程师

Task	描述	所需技能
	3. 在 CA 层次结构所在账户的 IAM Identity Center 中创建所需 IAM 角色或用户。	
部署 CloudFormation 堆栈。	1. 从此模式的GitHub 存储库中下载 AWSPCA-RootCASubCA .yaml 模板。 2. 从 AWS CloudFormation 控制台或 AWS 命令行界面 (AWS CLI) 部署模板。有关更多信息，请参阅 CloudFormation 文档中的使用堆栈。 3. 填写模板中的参数，包括组织名称、OU 名称、密钥算法、签名算法和其他选项。	云管理员、安全工程师、PKI 工程师

Task	描述	所需技能
设计一个用于更新用户管理资源使用的证书的解决方案。	集成 Amazon Web Services 的资源（例如 Elastic Load Balancing）会在证书到期前自动更新证书。但是，用户管理的资源，例如在亚马逊弹性计算云 (Amazon EC2) 实例上运行的 Web 服务器，需要另一种机制。 1. 确定哪些用户管理的资源需要来自私有 CA 的最终实体证书。 2. 规划流程，以便在用户管理的资源和证书到期时收到通知。有关 示例，请参阅以下内容： • 使用 AWS Config 托管规则 • 使用亚马逊 CloudWatch 和亚马逊 EventBridge 3. 编写自定义脚本来更新用户管理资源上的证书，并将其与 Amazon Web Services 集成以自动执行更新。有关集成 Amazon Web Services 的更多信息，请参阅 ACM 文档中与 AWS Certifice Manager 集成的服务。	云管理员、安全工程师、PKI 工程师

验证和记录 CA 层次结构

Task	描述	所需技能
验证可选的 AWS RAM 共享。	如果 CA 层次结构与 AWS Organizations 中的其他账户共享，请从 AWS 管理控制台 登录其中一个账户，导航到 AWS Private CA 控制台，并确认新创建的 CA 已与该账户共享。只有层次结构中最低级别的 CA 才是可见的，因为这是生成最终实体证书的 CA。对与 CA 共享的账户进行抽样重复操作。	云管理员、安全工程师、PKI 工程师
通过证书生命周期测试验证 CA 层次结构。	在此模式的GitHub 存储库中，找到生命周期测试。从 AWS CLI 运行测试以申请证书、导出证书、描述证书以及删除证书。	云管理员、安全工程师、PKI 工程师
将证书链导入信任存储库。	要使浏览器和其他应用程序信任证书，证书的颁发者必须包含在浏览器的信任存储库中，即可信列表 CAs。将新 CA 层次结构的证书链添加到浏览器与应用程序的信任存储中。确认终端实体证书是可信证书。	云管理员、安全工程师、PKI 工程师
创建运行手册来记录 CA 层次结构。	创建一份运行手册文档来描述 CA 层次结构的架构、可以申请最终实体证书的账户结构、构建过程以及基本管理任务，例如颁发最终实体证书（除非您想允许子账户进行自助服务）、使用情况和跟踪。	云管理员、安全工程师、PKI 工程师

相关资源

- 设计 CA 层次结构（AWS Private CA 文档）
- 创建私有 CA（AWS Private CA 文档）
- 如何使用 AWS RAM 共享您的 AWS Private CA 跨账户（AWS Blog 文章）
- AWS Private CA 最佳实践（AWS Blog 文章）
- 在 AWS Organizations 中启用资源共享（AWS RAM 文档）
- 管理私有 CA 生命周期（AWS Private CA 文档）
- acm-certificate-expiration-check 适用于 AWS Config（AWS Config 文档）
- AWS Certifice Manager 现在通过亚马逊提供证书到期监控 CloudWatch（AWS 公告）
- 与 AWS Certificate Manager 集成的服务（ACM 文档）

其他信息

导出证书时，请使用加密强度高且符合组织的数据丢失防护策略的密码。

在多账户环境中关闭所有 Security Hub 成员账户的安全标准控件

创建者：Michael Fuellbier (AWS) 和 Ahmed Bakry (AWS)

摘要

> **Important**
>
> AWS Security Hub 现在支持跨账户对安全标准和控制进行集中配置。这项新功能解决了本 AWS 规范性指导模式中解决方案所涵盖的许多场景。在以这种模式部署解决方案之前，请参阅 Sec urity Hub 中的中央配置。

在 Amazon Web Services (AWS) 云中，只能在单个 AWS 账户控件中手动关闭（禁用）AWS Security Hub 标准控件，例如 CIS AWS 基金会基准测试或AWS 基础安全最佳实践。在多账户环境中，您无法通过“一键式”（即一次 API 调用）关闭多个 Security Hub 成员账户的控件。此模式演示如何一键关闭 Security Hub 管理员账户管理的所有 Security Hub 成员账户的 Security Hub 标准控件。

先决条件和限制

先决条件

- 多账户环境，由管理多个成员账户的 Security Hub 管理员账户组成
- AWS Command Line Interface (AWS CLI) 版本 2，已安装
- AWS Serverless Application Model 命令行界面 (AWS SAM CLI)，已安装

限制

- 此模式仅适用于多账户环境（即单个 Security Hub 管理员账户管理多个成员账户）。
- 如果您在很短的时间内更改了许多控件，则事件启动会导致多次 parallel 调用。这可能会导致 API 节流并引起调用失败。例如，如果您使用 Security Hub Controls CLI 以编程方式修改多个控件，就可能会发生这种情况。

架构

下图显示了关闭多个 Security Hub 成员账户的 Security Hub 标准控制 AWS Step Functions 的工作流程示例（从 Security Hub 管理员帐户中查看）。

图表包括以下工作流程：

1. Amazon EventBridge 规则按日程启动并调用状态机。您可以通过更新 AWS CloudFormation 模板中的`Schedule`参数来修改规则的时间。
2. 每当在 Security Hub 管理员帐户中打开或关闭控件时，就会启动 EventBridge 规则。
3. Step Functions 状态机将安全标准控件（即开启或关闭的控件）的状态从 Security Hub 管理员账户发布到成员账户。
4. 跨账户 AWS Identity and Access Management (IAM) 角色部署在每个成员账户中，并由状态机担任。状态机在每个成员账户中打开或关闭控件。
5. Amazon DynamoDB 表包含例外情况以及有关在特定账户中开启或关闭哪些控件的信息。此信息将覆盖从 Security Hub 管理员账户中获取的指定成员账户的配置。

> **Note**
>
> 计划 EventBridge 规则的目的是确保新添加的 Security Hub 成员账户与现有账户具有相同的控制状态。

工具

AWS 服务

- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- Amazon EventBridge 是一项无服务器事件总线服务，可帮助您将应用程序与来自各种来源的实时数据连接起来。例如，AWS Lambda 函数、使用 API 目的地的 HTTP 调用端点或其他 AWS 账户目的地的事件总线。
- AWS Command Line Interface (AWS CLI) 是一个开源工具，可帮助您 AWS 服务 通过命令行外壳中的命令进行交互。
- AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- AWS Serverless Application Model (AWS SAM) 是一个开源框架，可帮助您在中构建无服务器应用程序。 AWS Cloud

- AWS Security Hub提供了您的安全状态的全面视图 AWS。它还可以帮助您根据安全行业标准和最佳实践检查您的 AWS 环境。
- AWS Step Functions是一项无服务器编排服务，可帮助您组合 AWS Lambda 功能和其他功能 AWS 服务 来构建关键业务应用程序。

代码存储库

此模式的代码可在 GitHub AWS Security Hub 跨账户控制禁用器存储库中找到。代码存储库包含以下文件和文件夹：

- UpdateMembers/template.yaml — 此文件包含部署在 Security Hub 管理员帐户中的组件，包括 Step Functions 状态机和规则。 EventBridge
- member-iam-role/template.yaml — 此文件包含在成员账户中部署跨账户 IAM 角色的代码。
- StateMachine.json — 此文件定义了状态机的工作流程。
- GetMembers/index.py — 此文件包含GetMembers状态机的代码。脚本检索所有现有 Security Hub 成员账户中安全标准控件的状态。
- UpdateMember/index.py — 此文件包含更新每个成员账户控制状态的脚本。
- CheckResult/index.py-此文件包含用于检查工作流程调用状态（已接受或失败）的脚本。

操作说明

在 Security Hub 成员账户中部署跨账户 IAM 角色

Task	描述	所需技能
确定 Security Hub 管理员账户的 ID。	设置 Security Hub 管理员账户，然后记下该管理员账户的 ID。	云架构师
部署在成员账户中包含跨账户 IAM 角色的 CloudFormation 模板。	要在 Security Hub 管理员账户管理的所有成员账户中部署 member-iam-role/template.yaml 模板，请运行以下命令： `aws cloudformation deploy \`	AWS DevOps

Task	描述	所需技能
	--template-file member-iam-role/template.yaml \ --capabilities CAPABILITY_NAMED_IAM \ --stack-name <your-stack-name> \ --parameter-overrides SecurityHubAdminAccountId=<account-ID> 该 SecurityHubAdminAccountId 参数必须与您之前记下的 Security Hub 管理员账户 ID 相匹配。	

在 Security Hub 管理员账户中部署状态机

Task	描述	所需技能
Package 将包含状态机的 CloudFormation 模板打包 AWS SAM。	要在 Security Hub 管理员帐户中打包 UpdateMembers/template.yaml 模板，请运行以下命令： sam package \ --template-file UpdateMembers/template.yaml \ --output-template-file UpdateMembers/template-out.yaml \ --s3-bucket <amzn-s3-demo-bucket>	AWS DevOps

Task	描述	所需技能
	Note 您的亚马逊简单存储服务 (Amazon S3) Simple Service 存储桶必须与部署 CloudFormation 模板的位置 AWS 区域 相同。	

Task	描述	所需技能
在 Security Hub 管理员帐户中部署打包的 CloudFormation 模板。	要在 Security Hub 管理员帐户中部署 CloudFormation 模板，请运行以下命令： `aws cloudformation deploy \ --template-file UpdateMembers/template-out.yaml \ --capabilities CAPABILITY_IAM \ --stack-name <stack-name>` 在 member-iam-role/template.yaml 模板中，参数必须与`MemberIAMRolePath` 参数匹配并且必须匹配。`IAMRolePath` `MemberIAMRoleName` `IAMRoleName` **Note** 由于 Security Hub 是一项区域服务，因此您必须在每个服务中单独部署模板 AWS 区域。请务必先将解决方案打包到每个区域的 Amazon S3 存储桶中。	AWS DevOps

相关资源

- 指定 Security Hub 管理员帐户（Security Hub 文档）
- 处理错误、重试以及向 AWS Step Functions 状态机执行添加警报（AWS 博客文章）

使用从 AWS IAM 身份中心更新 AWS CLI 证书 PowerShell

创建者：Chad Miles (AWS) 和 Andy Bowen (AWS)

摘要

如果您想将 AWS IAM 身份中心（AWS Single Sign-On 的继任者）凭证与 AWS 命令行接口 (AWS CLI)、AWS 或 AWS SDKs Cloud Development Kit (AWS CDK) 一起使用，则通常必须将证书从 IAM 身份中心控制台复制并粘贴到命令行界面中。此过程可能需要相当长的时间，并且必须为每个需要访问权限的账户重复此过程。

一种常见的解决方案是使用 AWS CLI `aws sso configure` 命令。此命令在您的 AWS CLI 或 AWS SDK 中添加启用 IAM Identity Center 的配置文件。但是，此解决方案的缺点是，您必须为以这种方式配置的每个 AWS CLI 配置文件或账户运行 `aws sso login` 命令。

作为替代解决方案，此模式描述了如何使用名为 Profiles 的 AWS CLI 和 AWS 工具 PowerShell 来同时存储和刷新来自单个 IAM Identity Center 实例的多个账户的证书。该脚本还将 IAM Identity Center 会话数据存储在内存中，以便在无需再次登录 IAM Identity Center 即可刷新凭证。

先决条件和限制

先决条件

- PowerShell，已安装并配置。有关更多信息，请参阅安装 PowerShell（Microsoft 文档）。
- 已安装并配置 PowerShell的 AWS 工具。出于性能考虑，我们强烈建议您安装名`AWS.Tools`为的 AWS 工具的 PowerShell模块化版本。每个 Amazon Web Service 都由其自己的小模块提供支持。在 PowerShell 提示符中，输入以下命令以安装此模式所需的模块：`AWS.Tools.Installer`、`SSO`和`SSOIDC`。

```
Install-Module AWS.Tools.Installer
Install-AWSToolsModule SSO, SSOOIDC
```

  有关更多信息，请参阅在 Windows 上安装 AWS.Tools 或在 Linux 或 macOS 上安装 AWS.Tools。
- AWS CLI 或 AWS 开发工具包必须事先通过以下任一操作配置有效凭证：
  - 使用 AWS CLI `aws configure` 命令。有关更多信息，请参阅快速配置（AWS CLI 文档）。
  - 将 AWS CLI 或 AWS CDK 配置为通过 IAM 角色获得临时访问权限。有关更多信息，请参阅获取用于 CLI 访问的 IAM 角色凭证（IAM Identity Center 文档）。

限制

- 此脚本不可在管线或全自动解决方案中使用。部署此脚本时，必须手动授权来自 IAM Identity Center 的访问权限。然后，脚本会自动继续。

产品版本

- 对于所有操作系统，建议您使用 7.0 或更高PowerShell 版本。

架构

您可以使用此模式中的脚本同时刷新多个 IAM 身份中心证书，也可以创建证书文件以用于 AWS CLI、AWS 或 AWS SDKs CDK。

工具

Amazon Web Services

- AWS 命令行界面（AWS CLI）是一种开源工具，它可帮助您通过命令行 Shell 中的命令与 Amazon Web Services 交互。
- AWS IAM Identity Center 可帮助您集中管理对所有 Amazon Web Services account 和云应用程序的单点登录（SSO）访问权限。
- AWS 工具 PowerShell是一组 PowerShell 模块，可帮助您通过 PowerShell 命令行编写对 AWS 资源的操作的脚本。

其他工具

- PowerShell是一款在 Windows、Linux 和 macOS 上运行的微软自动化和配置管理程序。

最佳实践

为每个 IAM Identity Center 实例保留一份此脚本的副本。不支持将一个脚本用于多个实例。

操作说明

运行 SSO 脚本

Task	描述	所需技能
自定义 SSO 脚本。	1. 复制其他信息部分中的 SSO 脚本。 2. 在 `Param` 部分中，针对您的 AWS 环境，定义以下变量的值： • `DefaultRoleName` – 默认情况下要使用的 IAM 角色或权限集。 • `Region` – 部署了 IAM Identity Center 的 AWS 区域。有关区域及其代码的完整列表，请参阅区域端点。 • `StartUrl` – 用于访问 IAM Identity Center 登录页面的 URL。使用与脚本中示例值相同的格式。 • `EnvironmentName` – 用于引用此脚本副本的简称，当您在同一个会话中运行多个脚本副本时将会用到。 3. 在第 10 行（读作 `# Add your Account Information`）下方，编辑哈希表中的以下值以反映您的环境：	云管理员

Task	描述	所需技能
	• `Profile` – 用于存储临时凭证的 AWS CLI 配置文件名称。 • `AccountId` – 您要为其检索凭证的 Amazon Web Services account 的 ID。 • `RoleName` – 您要使用的 IAM Identity Center 角色或权限集的名称。如果您要使用在 `Param` 部分中定义的相同角色，您可以把它当作 `$DefaultRoleName` 。 哈希表中的每一行都必须以逗号结尾，最后一行除外。	

运行 SSO 脚本。	建议您使用以下命令在 PowerShell shell 中运行您的自定义脚本。
	`./Set-AwsCliSsoCredentials.ps1`
	或者，您可以通过输入以下命令从其他 Shell 运行脚本。
	`pwsh Set-AwsCliSsoCredentials.ps1`
		云管理员

故障排除

事务	解决方案
No Access 错误	您正在使用的 IAM 角色无权访问您在 RoleName 参数中定义的角色或权限集。更新您正在使用的角色的权限，或者在脚本中定义其他角色或权限集。

相关资源

- 配置设置存储在何处？（AWS CLI 文档）
- 配置 AWS CLI 以使用 AWS IAM Identity Center（AWS CLI 文档）
- 使用命名配置文件（AWS CLI 文档）

其他信息

SSO 脚本

在以下脚本中，用您自己的信息替换尖括号（**<>**）中的占位符，然后删除尖括号。

```
Set-AwsCliSsoCredentials.ps1
Param(
    $DefaultRoleName = '<AWSAdministratorAccess>',
    $Region          = '<us-west-2>',
    $StartUrl        = "<https://d-12345abcde.awsapps.com/start/>",
    $EnvironmentName = "<CompanyName>"
)
Try {$SsoAwsAccounts = (Get-Variable -name "$($EnvironmentName)SsoAwsAccounts" -Scope
 Global -ErrorAction 'SilentlyContinue').Value.Clone()}
Catch {$SsoAwsAccounts = $False}
if (-not $SsoAwsAccounts) { $SsoAwsAccounts = @(
# Add your account information in the list of hash tables below, expand as necessary,
 and do not forget the commas
    @{Profile = "<Account1>"      ; AccountId = "<012345678901 >"; RoleName =
 $DefaultRoleName },
    @{Profile = "<Account2>"      ; AccountId = "<123456789012>"; RoleName =
 "<AWSReadOnlyAccess>" }
)}
```

```
$ErrorActionPreference = "Stop"
if (-not (Test-Path ~\.aws))      { New-Item ~\.aws -type Directory }
if (-not (Test-Path ~\.aws\credentials)) { New-Item ~\.aws\credentials -type File }
$CredentialFile = Resolve-Path ~\.aws\credentials
$PsuedoCreds    = @{AccessKey =
 'AKAEXAMPLE123ACCESS';SecretKey='PsuedoS3cret4cceSSKey123PsuedoS3cretKey'} # Pseudo
 Creds, do not edit.
Try {$SSOTokenExpire = (Get-Variable -Scope Global -Name
 "$($EnvironmentName)SSOTokenExpire" -ErrorAction 'SilentlyContinue').Value} Catch
 {$SSOTokenExpire = $False}
Try {$SSOToken       = (Get-Variable -Scope Global -Name "$($EnvironmentName)SSOToken"
 -ErrorAction 'SilentlyContinue').Value }      Catch {$SSOToken       = $False}
if ( $SSOTokenExpire -lt (Get-Date) ) {
    $SSOToken = $Null
    $Client   = Register-SSOOIDCClient -ClientName cli-sso-client -ClientType public -
Region $Region @PsuedoCreds
    $Device   = $Client | Start-SSOOIDCDeviceAuthorization -StartUrl $StartUrl -Region
 $Region @PsuedoCreds
    Write-Host "A Browser window should open. Please login there and click ALLOW." -
NoNewline
    Start-Process $Device.VerificationUriComplete
    While (-Not $SSOToken){
        Try {$SSOToken = $Client | New-SSOOIDCToken -DeviceCode $Device.DeviceCode -
GrantType "urn:ietf:params:oauth:grant-type:device_code" -Region $Region @PsuedoCreds}
        Catch {If ($_.Exception.Message -notlike "*AuthorizationPendingException*")
{Write-Error $_.Exception} ; Start-Sleep 1}
    }
    $SSOTokenExpire = (Get-Date).AddSeconds($SSOToken.ExpiresIn)
    Set-Variable -Name "$($EnvironmentName)SSOToken" -Value $SSOToken -Scope Global
    Set-Variable -Name "$($EnvironmentName)SSOTokenExpire" -Value $SSOTokenExpire -
Scope Global
}
$CredsTime     = $SSOTokenExpire - (Get-Date)
$CredsTimeText = ('{0:D2}:{1:D2}:{2:D2} left on SSO Token' -f $CredsTime.Hours,
 $CredsTime.Minutes, $CredsTime.Seconds).TrimStart("0 :")
for ($i = 0; $i -lt $SsoAwsAccounts.Count; $i++) {
    if (([DateTimeOffset]::FromUnixTimeSeconds($SsoAwsAccounts[$i].CredsExpiration /
 1000)).DateTime -lt (Get-Date).ToUniversalTime()) {
        Write-host "`r
    `rRegistering Profile $($SsoAwsAccounts[$i].Profile)" -NoNewline
        $TempCreds = $SSOToken | Get-SSORoleCredential -AccountId
 $SsoAwsAccounts[$i].AccountId -RoleName $SsoAwsAccounts[$i].RoleName -Region $Region
 @PsuedoCreds
```

```
        [PSCustomObject]@{AccessKey = $TempCreds.AccessKeyId; SecretKey =
 $TempCreds.SecretAccessKey; SessionToken = $TempCreds.SessionToken
        } | Set-AWSCredential -StoreAs $SsoAwsAccounts[$i].Profile -ProfileLocation
 $CredentialFile
        $SsoAwsAccounts[$i].CredsExpiration = $TempCreds.Expiration
    }
}
Set-Variable -name "$($EnvironmentName)SsoAwsAccounts" -Value $SsoAwsAccounts.Clone() -
Scope Global
Write-Host "`r$($SsoAwsAccounts.Profile) Profiles registered, $CredsTimeText"
```

使用 AWS Config 监控 Amazon Redshift 安全配置

由 Lucas Kauffman (AWS) 和 abhishek sengar (AWS) 编写

摘要

使用 AWS Config，您可以评估您的 AWS 资源的安全配置。AWS Config 可以监控资源，如果配置设置违反了您定义的规则，AWS Config 会将该资源标记为不合规。

您可以使用 AWS Config 来评估和监控您的 Amazon Redshift 集群和数据库。有关安全建议和功能的更多信息，请参阅 Amazon Redshift 中的安全性。此模式包括适用于 AWS Config 的自定义 AWS Lambda 规则。您可以在您的账户中部署这些规则，以监控 Amazon Redshift 集群和数据库的安全配置。此模式中的规则可帮助您使用 AWS Config 来确认：

- 已为 Amazon Redshift 集群中的数据库启用了审计日志记录
- 连接到 Amazon Redshift 集群需要 SSL
- 正在使用联邦信息处理标准 (FIPS) 密码
- Amazon Redshift 集群中的数据库已加密
- 用户活动监控已启用

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 必须在您的 Amazon Web Services account 中启用 AWS Config。有关更多信息，请参阅使用控制台设置 AWS Config 或使用 AWS CLI 设置 AWS Config。
- AWS Lambda 处理程序必须使用 Python 版本 3.9 或更高版本。有关更多信息，请参阅使用 Python（AWS Lambda 文档）。

产品版本

- Python 版本 3.9 或更高版本

架构

目标技术堆栈

- AWS Config

目标架构

1. AWS Config 会定期运行自定义规则。
2. 自定义规则调用 Lambda 函数。
3. Lambda 函数会检查 Amazon Redshift 集群中是否存在不合规的配置。
4. Lambda 函数会将每个 Amazon Redshift 集群的合规性状态报告给 AWS Config。

自动化和扩缩

AWS Config 自定义规则可扩展以评测您账户中的所有 Amazon Redshift 集群。无需采取任何其他操作即可扩展此解决方案。

工具

Amazon Web Services

- AWS Config 提供了 Amazon Web Services account 中资源及其配置方式的详细视图。它可以帮助您确定资源之间的关联方式，以及它们的配置如何随时间变化。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Lambda 是一项计算服务，可帮助您运行代码，而无需预置或管理服务器。它仅在需要时运行您的代码，并且能自动扩缩，因此您只需为使用的计算时间付费。
- Amazon Redshift 是一项在 Amazon Web Services Cloud 中托管的 PB 级数据仓库服务。

代码存储库

此模式的代码可在 GitHub aws-config-rules存储库中找到。此存储库中的自定义规则是采用 Python 编程语言的 Lambda 规则。此存储库包含许多 AWS Config 的自定义规则。此模式中仅使用以下规则：

- `REDSHIFT_AUDIT_ENABLED` — 确认已在 Amazon Redshift 集群上启用了审核日志记录。如果您还想确认已启用用户活动监控，请改为部署 `REDSHIFT_USER_ACTIVITY_MONITORING_ENABLED` 规则。

- REDSHIFT_SSL_REQUIRED — 确认连接到 Amazon Redshift 集群需要 SSL。如果您还想确认是否正在使用联邦信息处理标准 (FIPS) 密码，请改为部署 REDSHIFT_FIPS_REQUIRED 规则。
- REDSHIFT_FIPS_REQUIRED — 确认需要使用 SSL 且正在使用 FIPS 密码。
- REDSHIFT_DB_ENCRYPTED — 确认 Amazon Redshift 集群中的数据库已加密。
- REDSHIFT_USER_ACTIVITY_MONITORING_ENABLED — 确认已启用审核日志记录和用户活动监控。

操作说明

准备部署规则

Task	描述	所需技能
配置 IAM policy。	1. 创建一个自定义的基于 IAM 身份的策略，允许 Lambda 执行角色读取 Amazon Redshift 集群配置。有关更多信息，请参阅管理对资源的访问（Amazon Redshift 文档）和创建 IAM policy（IAM 文档）。 `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "redshift:DescribeClusterParameterGroups", "redshift:DescribeClusterParameters", "redshift:DescribeClusters",`	AWS 管理员

Task	描述	所需技能

	<pre>"redshift
	:DescribeClusterSe
	curityGroups",
	"redshift
	:DescribeClusterSn
	apshots",
	"redshift
	:DescribeClusterSu
	bnetGroups",
	"redshift
	:DescribeEventSubs
	criptions",
	"redshift
	:DescribeLoggingSt
	atus"
	],
	"Resource":
	"*"
	}
	]
	}</pre> 2. 将AWSLambda执行策略和AWSConfigRulesExecutionRole托管策略分配为Lambda 执行角色的权限策略。有关说明，请参阅添加 IAM 身份权限（IAM 文档）。	

克隆存储库。	在 Bash Shell 中，运行以下命令。这将从中克隆aws-config-rules存储库。 GitHub
	<pre>git clone https://g
	ithub.com/awslabs/
	aws-config-rules.git</pre>
		常规 AWS

在 AWS Config 中部署规则

Task	描述	所需技能
在 AWS Config 中部署规则。	按照创建自定义 Lambda 规则（AWS Config 文档）中的说明，在您的账户中部署以下一项或多项规则： • `REDSHIFT_AUDIT_ENABLED` • `REDSHIFT_SSL_REQUIRED` • `REDSHIFT_FIPS_REQUIRED` • `REDSHIFT_DB_ENCRYPTED` • `REDSHIFT_USER_ACTIVITY_MONITORING_ENABLED`	AWS 管理员
验证规则是否有效。	部署规则后，请按照评估您的资源（AWS Config 文档）中的说明进行操作，确认 AWS Config 正在正确评估您的 Amazon Redshift 资源。	常规 AWS

相关资源

Amazon Web Services 文档

- Amazon Redshift 中的安全性（Amazon Redshift 文档）
- 管理数据库安全（Amazon Redshift 文档）
- AWS Config 自定义规则（AWS Config 文档）

AWS Prescriptive Guidance

- 验证新的 Amazon Redshift 集群是否有所需 SSL 端点
- 确保 Amazon Redshift 集群在创建时已加密

其他信息

您可在 AWS Config 中使用以下 AWS 托管规则来确认 Amazon Redshift 的以下安全配置：

- redshift-cluster-configuration-check— 使用此规则确认已为 Amazon Redshift 集群中的数据库启用审计日志并确认数据库已加密。
- redshift-require-tls-ssl— 使用此规则确认需要使用 SSL 才能连接到 Amazon Redshift 集群。

使用 Network Firewall 从服务器名称指示中捕获出站流量的 DNS 域名

由 Kirankumar Chandrashekar (AWS) 创建

摘要

此模式向您展示如何使用 AWS Network Firewall 收集服务器名称指示 (SNI) 在出站网络流量的 HTTPS 标头中提供的 DNS 域名。Network Firewall 是一项托管服务，可以轻松地为 Amazon Virtual Private Cloud (Amazon VPC) 部署关键网络保护，包括能够使用防火墙来保护出站流量，该防火墙可以阻止不符合某些安全要求的数据包。保护特定 DNS 域名的出站流量称为出口过滤，这是一种监视并可能限制从一个网络到另一个网络的出站信息流的做法。

捕获通过网络防火墙的 SNI 数据后，您可以使用亚马逊 CloudWatch 日志和 AWS Lambda 将数据发布到生成电子邮件通知的亚马逊简单通知服务 (Amazon SNS) Simple Notification Service 主题。电子邮件通知包括服务器名称和其他相关 SNI 信息。此外，您可使用此模式的输出通过防火墙规则在 SNI 中按域名允许或限制 SNI 中的出站流量。有关更多信息，请参阅 Network Firewall 文档中的在 AWS Network Firewall 中使用有状态规则组

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- AWS Command Line Interface (AWS CLI) 版本 2，在 Linux、macOS 或 Windows 上安装和配置。
- Network Firewall，在 Amazon VPC 中设置和配置，用于检查出站流量。您可以将 Network Firewall 配置为使用以下任何 VPC 配置：
  - 带互联网网关的简单单区域架构
  - 带互联网网关的多区域架构
  - 带互联网网关和 NAT 网关的架构

架构

下图显示了如何使用 Network Firewall 从出站网络流量中收集 SNI 数据，然后使用 CloudWatch 日志和 Lambda 将这些数据发布到 SNS 主题。

图表显示了以下工作流：

1. 网络防火墙从出站网络流量的 HTTPS 标头中的 SNI 数据收集域名。
2. CloudWatch 日志监控 SNI 数据，并在出站网络流量通过 Network Firewall 时调用 Lambda 函数。
3. Lambda 函数读取 CloudWatch 日志捕获的 SNI 数据，然后将该数据发布到 SNS 主题。
4. SNS 主题会向您发送包含 SNI 数据的电子邮件通知。

自动化和扩缩

- 您可以使用 AWS 通过使用基础设施即代码 CloudFormation来创建此模式。

技术堆栈

- Amazon CloudWatch 日志
- Amazon SNS
- Amazon VPC
- AWS Lambda
- AWS Network Firewall

工具

Amazon Web Services

- 亚马逊 CloudWatch 日志 — 您可以使用亚马逊 CloudWatch 日志来监控、存储和访问来自亚马逊弹性计算云 (Amazon EC2) 实例、AWS CloudTrail、Amazon Route 53 和其他来源的日志文件。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 是一项托管服务，提供从发布者到订阅用户（也称为创建者和使用者）的消息传输。
- Amazon Virtual Private Cloud (Amazon VPC) 可预置 Amazon Web Services Cloud 的逻辑隔离部分，您可以在其中启动您定义的虚拟网络中的 AWS 资源。这个虚拟网络与您在数据中心中运行的传统网络极其相似，并会为您提供使用Amazon云科技可扩展基础设施的优势。
- AWS Lambda — AWS Lambda 是一项计算服务，可帮助您运行代码，无需预置或管理服务器。
- AWS Networ k Firewall — AWS Network Firewall 是一项托管服务，可让您轻松地为所有亚马逊部署基本的网络保护 VPCs。

操作说明

为 Network Firewall 创建 CloudWatch 日志组

Task	描述	所需技能
创建 CloudWatch 日志组。	1. 登录 AWS 管理控制台并打开CloudWatch 控制台。 2. 在导航窗格中，选择日志组。 3. 选择操作，然后选择创建日志组。 4. 输入日志组的名称，然后选择创建日志组。 有关更多信息，请参阅 CloudWatch 文档中的使用日志组和日志流。	云管理员

创建 SNS 主题和订阅

Task	描述	所需技能
创建 SNS 主题。	要创建 SNS 主题，请按照 Amazon SNS 文档中的说明进行操作。	云管理员
为端点订阅 SNS 主题。	要订阅电子邮件地址作为您创建的 SNS 主题的端点，请按照 Amazon SNS 文档中的说明进行操作。对于协议，选择电子邮件/电子邮件-JSON。	云管理员

Task	描述	所需技能
	Note 您也可以根据自己的要求选择不同的终端节点。	

在 Network Firewall 中设置登录

Task	描述	所需技能
启用防火墙日志记录。	1. 登录 AWS 管理控制台，打开 Amazon VPC 控制台。 2. 在导航窗格中的 NETWORK FIREWALL 下，选择防火墙。 3. 在防火墙部分，选择要从出站流量的 SNI 中捕获服务器名称的防火墙。 4. 选择防火墙详细信息选项卡，然后在记录部分中选择编辑。 5. 对于“日志类型”，选择“警报”。 在警报的日志目标中，选择CloudWatch 日志组。 6. 对于CloudWatch 日志组，搜索并选择您之前创建的日志组，然后选择保存。 有关使用 CloudWatch 日志作为网络防火墙日志目标的更多信息，请参阅网络防火墙文档	云管理员

Task	描述	所需技能
	中的 Amazon CloudWatch 日志。	

在 Network Firewall 中设置状态规则

Task	描述	所需技能
创建有状态规则。	1. 登录 AWS 管理控制台，打开 Amazon VPC 控制台。 2. 在导航窗格中的网络防火墙下，选择网络防火墙规则组。 3. 选择创建 Network Firewall 规则组。 4. 在创建 Network Firewall 规则组页面上，对于规则组类型，选择状态规则组。注意：有关更多信息，请参阅使用 AWS Network Firewall 中的有状态规则组。 5. 在有状态规则组部分，输入规则组的名称和描述。 6. 在容量中，设置要允许有状态规则组的最大容量（最多 30,000）。注：创建规则组之后，您无法再更改此设置。有关如何计算容量的信息，请参阅在 AWS Network Firewall 中设置规则组容量。有关最大设置的信息，请参见 AWS Network Firewall 配额。	云管理员

Task	描述	所需技能
	7. 对于状态规则组选项，请选择 5 元组。 8. 在状态规则顺序部分选择默认。 9. 在规则变量部分下，请保留默认值。 10. 在添加规则部分，为协议选择 TLS。对于来源，选择任意。对于源端口，选择任何端口。对于目的地，选择任意。对于目标端口，选择任何端口。对于流量方向，请选择向前。对于操作，选择警报。选择添加规则。 11. 选择创建有状态规则组。	
将状态规则与 Network Firewall 关联。	1. 登录 AWS 管理控制台，打开 Amazon VPC 控制台。 2. 在导航窗格中的 NETWORK FIREWALL 下，选择防火墙。 3. 选择要从 SNI 捕获出站流量的服务器名称的防火墙。 4. 在有状态规则组部分，选择操作，然后选择添加非托管状态规则组。 5. 在添加非托管有状态规则组页面，选择您之前创建的有状态规则组，然后选择添加有状态规则组。	云管理员

创建 Lambda 函数以读取日志记录

Task	描述	所需技能

为 Lambda 函数创建代码。	在可以从 Network Firewall 读取出站流量的 CloudWatch 日志事件的集成开发环境 (IDE) 中，粘贴以下 Python 3 代码并<SNS-topic-ARN> 替换为你的值：
	
	`import json`
	`import gzip`
	`import base64`
	`import boto3`
	`sns_client = boto3.cli`
	`ent('sns')`
	`def lambda_handler(eve`
	`nt, context):`
	`  decoded_event =`
	` json.loads(gzip.de`
	`compress(base64.b6`
	`4decode(event['aws`
	`logs']['data'])))`
	`  body = '''`
	`  {filtermatch}`
	`  '''.format(`
	`  loggroup=`
	`decoded_event['log`
	`Group'],`
	`  logstream`
	`=decoded_event['lo`
	`gStream'],`
	`  filtermat`
	`ch=decoded_event['`
	`logEvents'][0]['me`
	`ssage'],`
	`  )`
	`  print(body)`
	`  filterMatch =`
	` json.loads(body)`
	`  data = []`
		应用程序开发人员

Task	描述	所需技能

	`    if 'http' in filterMatch['event']:`
	`        data.append(filterMatch['event']['http']['hostname'])`
	`    elif 'tls' in filterMatch['event']:`
	`        data.append(filterMatch['event']['tls']['sni'])`
	`    result = 'Domain accessed ' + 1*' ' + (data[0]) + 1*' ' 'via AWS Network Firewall ' + 1*' '  + (filterMatch['firewall_name'])`
	`    print(result)`
	`    message = {'ServerName': result}`
	`    send_to_sns = sns_client.publish(`
	`        TargetArn=<SNS-topic-ARN>,`
	`#Replace with the SNS topic ARN`
	`        Message=json.dumps({'default': json.dumps(message),`
	
	`    'sms': json.dumps(message),`
	
	`   'email': json.dumps(message)}),`
	`        Subject='Server Name passed through the Network Firewall',`
	`        MessageStructure='json'`
	`    )`	

Task	描述	所需技能
	此代码示例解析 CloudWatch 日志内容并在 HTTPS 标头中捕获 SNI 提供的服务器名称。	
创建 Lambda 函数。	若要创建 Lambda 函数，请按Lambda 文档中的说明，然后为选择 Python 3.9 作为运行时系统。	云管理员
将代码添加到 Lambda 函数。	要将您的 Python 代码添加到您之前创建的 Lambda 函数中，请按照 Lambda 文档中的说明进行操作。	云管理员

Task	描述	所需技能
将 CloudWatch 日志作为触发器添加到 Lambda 函数。	1. 登录 AWS 管理控制台，然后打开 Lambda 控制台。 2. 在导航窗格中，选择函数，然后选择您之前创建的函数。 3. 在函数概述部分中，选择添加触发器。 4. 在添加触发器页面的触发器配置部分，选择CloudWatch 日志，然后选择添加。 5. 对于日志组，请选择您之前创建的 CloudWatch 日志组。 6. 在筛选条件名称中，输入筛选条件的名称。 7. 选择 添加。 8. 在函数页面的配置选项卡上，在触发器部分，选择刚刚添加的触发器，然后选择启用。 有关更多信息，请参阅 Lambda 文档中的将 Lambda 与 CloudWatch 日志配合使用。	云管理员

Task	描述	所需技能

添加 SNS 发布权限。	向 Lambda 执行角色添加 sns:Publish 权限，这样 Lambda 就可以调用 API 将消息发布到 SNS。
	
	1. 找到您之前创建的 Lambda 函数的执行角色。
	2. 将以下策略添加至您的 AWS Identity and Access Management (IAM) 角色：
	
	{
	"Version": "2012-10-17",
	"Statement": [
	{
	"Sid": "AllowSNSPublish",
	"Effect": "Allow",
	"Action": [
	"sns:GetTopicAttributes",
	"sns:Subscribe",
	"sns:Unsubscribe",
	"sns:Publish"
	],
	"Resource": "*"
	}
	]
	}
		云管理员

测试您的 SNS 通知功能

Task	描述	所需技能
通过 Network Firewall 发送流量。	1. 发送或等待 HTTPS 流量通过网络防火墙。 2. 检查当流量通过网络防火墙时您从 AWS 收到的 SNS 通知电子邮件。该电子邮件中包含出站流量的 SNI 详细信息。例如，如果访问的域名为 EMAIL-JSON https://aws.amazon.com且订阅协议为 EMAIL-JSON，则根据上述 Lambda 代码生成的电子邮件将包含以下内容： `{ "Type": "Notification", "MessageId": "<messageID>", "TopicArn": "arn:aws:sns:us-west-2:123456789:testSNSTopic", "Subject": "Server Name passed through the Network Firewall", "Message": "{\"ServerName\": \"Domain 'aws.amazon.com' accessed via AWS Network Firewall 'AWS-Network-Firewall-Multi-AZ-firewall\"}",`	测试工程师

Task	描述	所需技能
	"Timestamp": "2022-03-22T04:10:04.217Z", "SignatureVersion": "1", "Signature": "<Signature>", "SigningCertURL": "<SigningCertUrl>", "UnsubscribeURL": "<UnsubscribeURL>" } 然后，按照亚马逊 CloudWatch 文档中的说明查看 Amazon CloudWatch 中的 Network Firewall 警报日志。警报日志显示以下输出： { "firewall_name": "AWS-Network-Firewall-Multi-AZ-firewall", "availability_zone": "us-east-2b", "event_timestamp": "<event timestamp>", "event": { "timestamp": "2021-03-22T04:10:04.214222+0000", "flow_id": <flow ID>, "event_type": "alert", "src_ip": "10.1.3.76", "src_port": 22761,	

Task	描述	所需技能
	"dest_ip": "99.86.59.73", "dest_port": 443, "proto": "TCP", "alert": { "action": "allowed", "signature_id": 2, "rev": 0, "signature": "", "category": "", "severity": 3 }, "tls": { "subject": "CN=aws.amazon.com", "issuerdn": "C=US, O=Amazon, OU=Server CA 1B, CN=Amazon", "serial": "<serial number>", "fingerprint": "<fingerprint ID>", "sni": "aws.amazon.com", "version": "TLS 1.2", "notbefore": "2020-09-30T00:00:00", "notafter": "2021-09-23T12:00:00", "ja3": {}, "ja3s": {}	

Task	描述	所需技能

	<pre> },
	        "app_proto":
	"tls"
	    }
	}</pre>	

使用 Terraform 自动 GuardDuty 为组织启用亚马逊

由 Aarthi Kannan (AWS) 编写

摘要

亚马逊 GuardDuty 持续监控您的亚马逊网络服务 (AWS) 账户，并使用威胁情报来识别您的 AWS 环境中意外和潜在的恶意活动。跨多个 AWS 区域或通过 AWS 管理控制台 GuardDuty 为多个账户或组织手动启用可能很麻烦。您可使用基础设施即代码（IaC）工具（例如 Terraform）实现流程自动化，该工具可在云中配置和管理多账户、多区域的服务和资源。

AWS 建议使用 AWS Organizations 在中设置和管理多个账户 GuardDuty。这种模式符合此建议。这种方法的一个好处是，在创建或向组织中添加新账户时， GuardDuty 将在这些账户中自动为所有受支持的地区启用这些账户，而无需手动干预。

此模式演示如何使用 HashiCorp Terraform GuardDuty 为组织中的三个或更多亚马逊网络服务 (AWS) 账户启用亚马逊。此模式所提供的示例代码执行以下操作：

- GuardDuty 适用于所有 AWS Organizations 中目标组织当前成员的 AWS 账户
- 在中启用 “自动启用” 功能 GuardDuty，该功能会自动 GuardDuty 为将来添加到目标组织的所有帐户启用
- 允许您选择要启用的区域 GuardDuty
- 使用组织的安全账户作为 GuardDuty 委派管理员
- 在日志账户中创建 Amazon Simple Storage Service (Amazon S3) 存储桶，并 GuardDuty 配置为发布该存储桶中所有账户的汇总结果
- 默认情况下，分配生命周期策略，在 365 天后将调查发现从 S3 存储桶转移至 Amazon S3 Glacier Flexible Retrieval 存储

您可手动运行此示例代码，也可以将其集成至持续集成和持续交付 (CI/CD) 管道。

目标受众

建议有使用 Terraform、 GuardDuty Python 和 AWS Organizations 经验的用户使用这种模式。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 在 AWS Organizations 中设立了组织，该组织至少包含以下三个账户：
  - 管理账户 — 这是您从中部署 Terraform 代码的账户，可为是独立的，也可以作为 CI/CD 管道的一部分。Terraform 状态也存储至此账户。
  - 安全帐户-此帐户用作 GuardDuty 委派管理员。有关更多信息，请参阅 GuardDuty 委派管理员的重要注意事项（GuardDuty 文档）。
  - 日志账户 — 此账户包含 S3 存储桶，用于 GuardDuty 发布所有成员账户的汇总结果。

  有关如何使用所需配置设置组织的更多信息，请参阅创建账户结构 (AWS Well-Architected Labs)。
- 一个 Amazon S3 存储桶和一个 Amazon DynamoDB 表，用作远程后端，将 Terraform 状态存储至管理账户中。有关使用远程后端获得 Terraform 状态的更多信息，请参见 S3 后端（Terraform 文档）。有关使用 S3 后端设置远程状态管理的代码示例，请参阅 remote-state-s3 后端（Terraform Registry）。请注意以下要求：
  - S3 存储桶和 DynamoDB 表必须在同一个区域中。
  - 创建 DynamoDB 表时，分区键必须为 `LockID`（区分大小写），并且分区键类型必须为字符串。所有其他表设置必须为其默认值。有关更多信息，请参阅关于主键和创建表格（DynamoDB 文档）。
- 一个 S3 存储桶，用于存储 GuardDuty 将在其中发布发现结果的 S3 存储桶的访问日志。有关更多信息，请参阅启用 Amazon S3 服务器访问日志记录（Amazon S3 文档）。如果您要部署至 AWS Control Tower 登录区，则可以为此目的重复使用日志存档账户中的 S3 存储桶。
- 已安装并配置了 Terraform 版本 0.14.6 或更高版本。有关更多信息，请参阅入门 – AWS（Terraform 文档）。
- Python 3.9.6 或更高版本已安装并配置。有关更多信息，请参见源版本（Python 网站）。
- 已安装适用于 Python 的 Amazon SDK (Boto3)。更多信息，请参阅安装（Boto3 文档）。
- 安装并配置了 jq。有关更多信息，请参阅下载 jq（jq 文档）。

限制

- 此模式支持 macOS 和Amazon Linux 2 操作系统。此模式尚未经测试，无法在 Windows 操作系统中使用。

> **Note**
>
> 亚马逊 Linux 2 的支持已接近终止。欲了解更多信息，请参阅亚马逊 Linux 2 FAQs。

- GuardDuty 必须尚未在任何目标区域的任何账户中启用。
- 这种模式的 IaC 解决方案不部署先决条件。
- 此模式专为遵循以下最佳实践标准的 AWS 登录区而设计：
  - 登录区是使用 AWS Control Tower 创建的。
  - 单独的 Amazon Web Services account 可用于安全和日志记录。

产品版本

- Terraform 版本 0.14.6 或更高版本。示例代码已经过 1.2.8 的测试。
- Python，版本 3.9.6 或更高版本。

架构

本节概括介绍此解决方案，以及由示例代码建立的架构。下图显示了在单个 AWS 区域 内跨组织不同账户部署的资源。

1. Terraform 在安全账户和日志账户中创建 GuardDutyTerraformOrgRoleAWS Identity and Access Management (IAM) 角色。
2. Terraform 在日志登录账户的默认 AWS 区域 中创建一个 S3 存储桶。此存储桶用作发布目的地，用于汇总组织中所有区域和所有账户的所有 GuardDuty 调查结果。Terraform 还在安全账户中创建了 AWS Key Management Service (AWS KMS) 密钥，用于加密 S3 存储桶中的调查发现，并配置以将结果从 S3 存储桶自动存档至 S3 Glacier Flexible Retrieval 存储。
3. 在管理帐户中，Terraform 将安全帐户指定为的委托管理员。 GuardDuty这意味着安全账户现在可以管理包括管理账户在内的所有成员账户的 GuardDuty 服务。个人成员账户不能自行暂停或 GuardDuty 禁用。
4. Terraform 在安全账户中为 GuardDuty 委托的管理员创建 GuardDuty 探测器。
5. 如果尚未启用，Terraform 将在中启用 S3 保护。 GuardDuty有关更多信息，请参阅亚马逊中的 Amazon S3 保护 GuardDuty（GuardDuty 文档）。
6. Terraform 将组织中所有当前活跃的成员帐户注册为成员。 GuardDuty
7. Terraform 将 GuardDuty 委派管理员配置为将所有成员账户的汇总结果发布到日志账户中的 S3 存储桶。
8. Terraform 会为您选择的每个 AWS 区域重复步骤 3 到 7。

自动化和扩缩

提供的示例代码为模块化特点，因此您可以将其集成至您的 CI/CD 管道中以实现自动部署。

工具

Amazon Web Services

- Amazon DynamoDB 是一项完全托管的 NoSQL 数据库服务，可提供快速、可预测和可扩展的性能。
- Amazon GuardDuty 是一项持续的安全监控服务，可分析和处理日志，以识别您的 AWS 环境中意外和可能未经授权的活动。
- AWS Identity and Access Management (AWS IAM) 通过控制验证和授权使用您 AWS 资源的用户，帮助您安全地管理对您 AWS 资源的访问。
- AWS Key Management Service (AWS KMS) 可帮助您创建和控制加密密钥以保护您的数据。
- AWS Organizations 是一项账户管理服务，使您可将多个 Amazon Web Services account 整合到您所创建的组织中并进行集中管理。
- Amazon Simple Storage Service (Amazon S3) 是一项基于云的对象存储服务，可帮助您存储、保护和检索任意数量的数据。
- 适用于 Python 的 Amazon SDK (Boto3) 是一款软件开发套件，可帮助您将 Python 应用程序、库或脚本与 Amazon Web Services 集成。

其他工具和服务

- HashiCorp Terraform 是一款命令行界面应用程序，可帮助您使用代码来配置和管理云基础架构和资源。
- Python 是通用的编程语言。
- jq 是命令行处理器，可帮助您处理 JSON 文件。

代码存储库

此模式的代码可在 GitHub amazon-guardduty-for-aws-organizations-with-terraform 存储库中找到。

操作说明

在组织 GuardDuty 中启用

Task	描述	所需技能
克隆存储库。	在 Bash Shell 中，运行以下命令。在 “其他信息” 部分的 “克隆存储库” 中，您可以复制包含 GitHub 存储库 URL 的完整命令。这会从中克隆 amazon-guardduty-for-aws-organizations-with-terraform 存储库。GitHub `git clone <github-repository-url>`	DevOps 工程师
编辑 Terraform 配置文件。	1. 在克隆存储库的 `root` 文件夹，通过运行以下命令复制 configuration.json.sample 文件。 `cp configuration.json.sample configuration.json` 2. 编辑新的 configuration.json 文件，并为以下每个变量定义值： • `management_acc_id` — 管理账户的账户 ID。 • `delegated_admin_acc_id` — 安全账户的账户 ID。 • `logging_acc_id` — 登录账户的账户 ID。	DevOps 工程师、通用 AWS、Terraform、Python

Task	描述	所需技能
	• `target_regions` — 您要启用的 AWS 区域列表，以逗号分隔。GuardDuty • `organization_id` — 您要在其中启用的组织的 AWS Organizations ID GuardDuty。 • `default_region` — 管理账户中存储您的 Terraform 状态的区域。这与您为 Terraform 后端部署 S3 存储桶以及 DynamoDB 表的区域相同。 • `role_to_assume_for_role_creation` — 您要分配给安全和日志账户中的新 IAM 角色的名称。您在下一个故事中创建这个新角色。Terraform 代入此角色，以在安全和日志账户中创建 `GuardDutyTerraformOrgRole` IAM 角色。 • `finding_publishing_frequency` — 将调查结果 GuardDuty 发布到 S3 存储桶的频率。 • `guardduty_findings_bucket_region` — 您想要在其中为发布的调	

Task	描述	所需技能
	查发现创建 S3 存储桶的首选区域。 • `logging_acc_s3_bucket_name` — 用于发布的调查发现的 S3 存储桶的首选名称。 • `security_acc_kms_key_alias` — 用于加密 GuardDuty 发现结果的密钥的 AWS KMS 别名。 • `s3_access_log_bucket_name` — 先前存在的 S3 存储桶的名称，您要在其中收集用于 GuardDuty 查找的 S3 存储桶的访问日志。此存储桶应与 GuardDuty 调查结果存储桶位于同一 AWS 区域。 • `tfm_state_backend_s3_bucket` — 用于存储 Terraform 远程后端状态的先前存在的 S3 存储桶的名称。 • `tfm_state_backend_dynamodb_table` — 用于锁定 Terraform 状态的先前存在的 DynamoDB 表的名称。 3. 保存并关闭配置文件。	

Task	描述	所需技能
为新的 IAM 角色生成 CloudFormation 模板。	此模式包括用于创建两个 CloudFormation 模板的 IaC 解决方案。这些模板创建了 Terraform 在设置期间中使用的两个 IAM 角色。这些模板遵循最低权限安全最佳实践。 1. 在 Bash shell 的存储库 `root` 文件夹中，导航至 `cfn-templates/` 。此文件夹包含带有存根的 CloudFormation 模板文件。 2. 运行以下命令。这会将存根替换为您在 configuration.json 文件中提供的值。 `bash scripts/replace_config_stubs.sh` 3. 确认已在该`cfn-templates/` 文件夹中创建了以下 CloudFormation 模板： • management-account-role.yaml — 此文件包含角色定义和管理账户中 IAM 角色的相关权限，该角色具有完成此模式所需的最低权限。 • role-to-assume-for-role-creation.yaml — 此文件包含安全账户和日志账户中 IAM 角色的角色定义和关联权限。Terraform 扮演这个角色是为了在这	DevOps 工程师，通用 AWS

Task	描述	所需技能
	些账户中创建GuardDuty TerraformOrgRole角色。	
创建 IAM 角色。	按照创建堆栈（CloudFormation 文档）中的说明，执行以下操作： 1. 在安全账户和日志role-to-assume-for账户中部署-role-creation.yaml 堆栈。 2. 在管理账户中部署 management-account-role.yaml 堆栈。成功创建堆栈并看到 CREATE_COMPLETE 堆栈状态后，请在输出中记下此新角色的 Amazon 资源名称（ARN）。	DevOps 工程师，通用 AWS
在管理账户中代入 IAM 角色。	作为安全最佳实践，我们建议您在继续操作之前担任新management-account-role的 IAM 角色。在 AWS 命令行界面（AWS CLI）的其他信息部分输入代入管理账户 IAM 角色命令	DevOps 工程师，通用 AWS

Task	描述	所需技能
运行安装脚本。	在存储库 `root` 文件夹中，运行以下命令以启动安装脚本。 `bash scripts/full-setup.sh` full-setup.sh脚本将执行以下操作： • 将所有配置值导出至环境变量 • 为每个 Terraform 模块生成 backend.tf 和 terraform.tfvars 代码 • 通过 AWS CLI 为 GuardDuty 组织内部启用可信访问。 • 将组织状态导入至 Terraform 状态 • 创建 S3 存储桶，以用于在日志账户中发布调查发现 • 创建 AWS KMS 密钥，以加密安全账户中的调查发现 • 在 GuardDuty 整个组织、所有选定区域中启用，如架构部分所述	DevOps 工程师，Python

（可选）在组织 GuardDuty 中禁用

Task	描述	所需技能
运行清理脚本。	如果您使用此模式 GuardDuty 为组织启用并想要禁用	DevOps 工程师、通用 AWS、Terraform、Python

Task	描述	所需技能
	GuardDuty，请在存储库`root`文件夹中运行以下命令来启动 cleanup-gd.sh 脚本。 `bash scripts/cleanup-gd.sh` 此脚本在目标组织 GuardDuty 中禁用，删除所有已部署的资源，并将组织恢复到使用 Terraform 启用之前的状态。GuardDuty **Note** 此脚本不会从本地和远程后端删除 Terraform 状态文件或锁定文件。如果需要执行此操作，则必须手动执行这些操作。此外，此脚本不会删除已导入的组织或由其管理的账户。在清理脚本中，并 GuardDuty 未禁用对的可信访问。	
移除 IAM 角色。	删除使用 role-to-assume-for-role-c reation.yaml 和.yaml 模板创建的堆栈。management-account-role CloudFormation 有关更多信息，请参阅删除堆栈（CloudFormation 文档）。	DevOps 工程师，通用 AWS

相关资源

AWS 文档

- 管理多个账户（GuardDuty 文档）
- 授予最低权限（IAM 文档）

AWS 营销

- Amazon GuardDuty
- AWS Organizations

其他资源

- Terraform
- Terraform CLI 文档

其他信息

克隆存储库

运行以下命令来克隆 GitHub 存储库。

```
git clone https://github.com/aws-samples/amazon-guardduty-for-aws-organizations-with-terraform
```

代入管理账户 IAM 角色

若要代入管理账户中的 IAM 角色，请运行以下命令。将 <IAM role ARN>替换为 IAM 角色的 ARN。

```
export ROLE_CREDENTIALS=$(aws sts assume-role --role-arn <IAM role ARN> --role-session-name AWSCLI-Session --output json)
export AWS_ACCESS_KEY_ID=$(echo $ROLE_CREDENTIALS | jq .Credentials.AccessKeyId | sed 's/"//g')
export AWS_SECRET_ACCESS_KEY=$(echo $ROLE_CREDENTIALS | jq .Credentials.SecretAccessKey | sed 's/"//g')
export AWS_SESSION_TOKEN=$(echo $ROLE_CREDENTIALS | jq .Credentials.SessionToken | sed 's/"//g')
```

使用验证 PCI DSS 4.0 的最佳操作实践 AWS Config

由 Tala Qraitem (AWS) 和 Alex Goff (AWS) 创作

摘要

支付卡行业数据安全标准 (PCI DSS) 概述了帮助保护支付数据的基本技术和操作协议。PCI DSS 的开发旨在鼓励和增强支付卡账户的数据安全。它还有助于全球采用一致的安全措施。尽管 PCI DSS 专为包含支付卡账户数据的环境而设计，但您可以使用 PCI DSS 来帮助抵御威胁并保护支付生态系统中的其他元素。

PCI DSS 版本 4.0 的发布旨在满足不断变化的需求，提供澄清或其他指导，并改进标准的结构和格式。有关变更的更多信息，请参阅 PCI DSS 版本 3.2.1 到 4.0 的更改摘要。

AWS Config 一致性包是一系列 AWS Config 规则和补救措施，可帮助您创建安全、运营或成本优化治理检查。您可以将一致性包部署为 AWS 账户 和中的单个实体 AWS 区域，也可以在中 AWS Organizations跨组织部署。

PCI DSS 版本 4.0 的一致性包增强并建立在 3.2.1 版一致性包的基础上。一致性包中的规则与标准中的规则对应。有关更多信息，请参阅 “附件” 部分中提供的映射。您可以在此一致性包的两个版本之间进行选择：一个包含全局资源类型，另一个不包括全局资源类型。

> **Important**
>
> 合规包的设计目的不是为了完全确保遵守特定的治理或合规性标准。您有责任自行评估使用情况是否符合适用的法律和监管要求。

先决条件和限制

先决条件

- 活跃起来 AWS 账户.
- 设置 AWS Config。
- 满足一致性包的先决条件。
- 部署 PCI DSS 版本 3.2.1 一致性包。
- 有权访问 AWS Config 和管理一致性包。有关策略示例，请参阅此模式的 “其他信息” 部分。

限制

- 您的每个配额 AWS 账户 都有默认配额，以前称为限制 AWS 服务。除非另有说明，否则，每个配额是区域特定的。您可以申请增加某些配额，但并非所有配额都可以增加。请务必熟悉AWS Config 服务限制，包括单一账户合规包和组织合规包的限制。
- 此一致性包的版本包含全球资源类型，仅适用于`us-east-1`在该地区部署。
- 此一致性包的版本不包括全球资源类型，仅适用于在以下区域部署：
  - `ap-east-1`
  - `ap-south-1`
  - `ap-northeast-2`
  - `ap-southeast-1`
  - `ap-southeast-2`
  - `ap-northeast-1`
  - `ca-central-1`
  - `eu-central-1`
  - `eu-west-1`
  - `eu-west-2`
  - `eu-west-3`
  - `eu-north-1`
  - `sa-east-1`
  - `us-east-2`
  - `us-west-1`
  - `us-west-2`

工具

AWS 服务

- AWS Config提供了您的资源 AWS 账户 及其配置方式的详细视图。它可以帮助您确定资源之间的关联方式，以及它们的配置如何随时间变化。
- AWS Systems Manager 可帮助您管理在 AWS Cloud中运行的应用程序和基础设施。它简化了应用程序和资源管理，缩短了检测和解决操作问题的时间，并帮助您大规模安全地管理 AWS 资源。

代码存储库

一致性包位于AWS Config 一致性包 GitHub 存储库中。此存储库包含以下与 PCI DSS 版本 4.0 相关的模板：

- PCI-dss-v4.0 的操作最佳实践-.yaml including-global-resourcetypes
- PCI-dss-v4.0 的操作最佳实践-.yaml excluding-global-resourcetypes

操作说明

部署和管理一致性包

Task	描述	所需技能
下载一致性包。	如果您要`us-east-1` 在该地区部署一致性包，请下载 PCI-dss-v4.0 -.yaml操作最佳实践模板。including-global-resourcetypes 如果您要在其他地区部署一致性包，请下载PCI-dss-v4.0 -.yaml操作最佳实践模板。excluding-global-resourcetypes	DevOps 工程师
（可选）修改一致性包。	您可以根据组织的独特需求修改一致性包模板。例如，您可以创建自定义修复操作。有关如何创建和修改模板的更多信息，请参阅 AWS Config 文档中的为自定义一致性包创建模板。	常规 AWS
部署合规包	如果您要在目标中部署 AWS 区域， AWS 账户 或者，请按照 AWS Config 文档中部署一致性包中的说明进行操作。你	常规 AWS

Task	描述	所需技能
	可以使用 AWS Management Console 或 AWS Command Line Interface (AWS CLI)。 如果您要在中跨组织部署一致性包 AWS Organizations，请按照文档中使用快速设置部署 AWS Config 一致性包中的 AWS Systems Manager 说明进行操作。	
（可选）编辑一致性包。	如果要编辑一致性包，请按照文档中编辑一致性包中的 AWS Config 说明进行操作。您可以使用 AWS Management Console 或 AWS CLI。	常规 AWS
（可选）删除一致性包。	如果要删除一致性包，请按照文档中删除一致性包中的 AWS Config 说明进行操作。您可以使用 AWS Management Console 或 AWS CLI。	常规 AWS

相关资源

AWS resources

- AWS Config（AWS Config 文档）的一致性包
- 使用快速设置部署 AWS Config 一致性包（Systems Manager 文档）
- AWS（网站）AWS 上的 PCI DSS 合规性
- PCI DSS 版本 4.0 已开启 AWS（合规性指南）

PCI DSS 资源

- PCI DSS 版本 4.0 资源中心

- PCI 安全标准委员会文档库
- 从 PCI DSS 版本 3.2.1 到 4.0 的变化摘要

其他信息

以下是允许用户访问 AWS Config 和管理合规包的示例 AWS Identity and Access Management (IAM) 策略：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "config:PutConfigRule",
                "config:PutConformancePack",
                "config:DeleteConfigRule",
                "config:DeleteRemediationConfiguration",
                "config:DeleteConformancePack",
                "config:PutRemediationConfigurations",
                "config:BatchGetAggregateResourceConfig",
                "config:BatchGetResourceConfig",
                "config:Get*",
                "config:Describe*",
                "config:Deliver*",
                "config:List*",
                "config:Select*"
            ],
            "Resource": "*"
        }
    ]
}
```

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

验证新的 Amazon Redshift 集群是否有所需 SSL 端点

由 Priyanka Chaudhary (AWS) 编写

摘要

此模式提供了一个亚马逊网络服务 (AWS) CloudFormation 模板，当没有安全套接字层 (SSL) 终端节点的新 Amazon Redshift 集群启动时，该模板会自动通知您。

Amazon Redshift 是一种完全托管的 PB 级基于云的数据仓库服务。它专为大规模数据集存储与分析而设计。它还用于执行大规模的数据库迁移。为安全起见，Amazon Redshift 支持 SSL 加密用户的 SQL Server 客户端应用程序与 Amazon Redshift 集群之间的连接。要将您的集群配置为需要使用 SSL 连接，在启动时，您在与集群关联的参数组中将 `require_SSL`参数设置为 `true`。

此模式提供的安全控制会监控 AWS CloudTrail 日志中的 Amazon Redshift API 调用，并为CreateCluster、、ModifyClusterRestoreFromClusterSnapshot、CreateClusterParameterGroup和启动亚马逊事件 CloudWatch 事件。ModifyClusterParameterGroup APIs当事件检测到其中一个事件时 APIs，它会调用运行 Python 脚本的 AWS Lambda。Python 函数针对列出的事件分析 CloudTrail 事件。 CloudWatch 当创建、修改或从现有快照还原 Amazon Redshift 集群、为该集群创建新的参数组或修改现有的参数组时，该函数会检查该集群的 `require_SSL` 参数。如果参数值为 `false`，则该函数会向用户发送 Amazon Simple Notification Service (Amazon SNS) 通知，其中包含相关信息：此通知来源的 Amazon Redshift 集群名称、AWS 区域、Amazon Web Services account 和 Lambda 的 Amazon 资源名称（ARN）。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 包含集群子网组和关联安全组的虚拟私有云（VPC）。

限制

- 这种安全控制为区域性的。您必须将其部署在要监控的每个 AWS 区域。

架构

目标架构

自动化和扩缩

- 如果您使用的是 AWS Organ izations，则可以使用 AWS Cloudformation StackSets 在要监控的多个账户中部署此模板。

工具

Amazon Web Services

- AWS CloudFormation — AWS CloudFormation 可帮助您建模和设置 AWS 资源，快速一致地配置这些资源，并在资源的整个生命周期中对其进行管理。您可以使用模板来描述资源及其依赖关系，然后将它们作为堆栈一起启动和配置，而不必单独管理资源。
- Amazon CloudWatch Events — Amazon CloudWatch Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。
- AWS Lambda – AWS Lambda 是一项计算服务，支持无需预置或管理服务器即可运行代码。
- Amazon Redshift – Amazon Redshift 是一种完全托管的 PB 级云中数据仓库服务。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项对象存储服务。您可以通过 Amazon S3 随时在 Web 上的任何位置存储和检索的任意大小的数据。
- Amazon SNS — Amazon Simple Notification Service (Amazon SNS) 可协调和管理发布者和客户端之间消息的传送或发送，包括 Web 服务器和电子邮件地址。订阅用户接收所有发布至他们所订阅主题的消息，并且一个主题的所有订阅用户收到的消息都相同。

代码

此模式包括以下附件：

- `RedshiftSSLEndpointsRequired.zip` — 用于安全控制的 Lambda 代码。
- `RedshiftSSLEndpointsRequired.yml`— 用于设置事件和 Lambda 函数的 CloudFormation 模板。

操作说明

设置 S3 存储桶

Task	描述	所需技能
定义 S3 存储桶。	在 Amazon S3 控制台，选择或创建 S3 存储桶来托管 Lambda 代码 .zip 文件。此 S3 存储桶必须与您想要监视的 Amazon Redshift 集群位于相同的 AWS 区域 中。S3 存储桶名称是全局唯一的，并且命名空间由所有 Amazon Web Services account 共享。S3 存储桶名称不得包含前导斜杠。	云架构师
上传 Lambda 代码。	将附件部分中提供的 Lambda 代码 .zip文件上传至 S3 存储桶。	云架构师

部署 CloudFormation 模板

Task	描述	所需技能
启动 AWS CloudFormation 模板。	在与您的 S3 存储桶相同的 AWS 区域中打开 AWS CloudFormation 控制台，然后部署所附的模板`RedshiftSSLEndpointsRequired.yml` 。有关部署 AWS CloudFormation 模板的更多信息，请参阅 CloudFormation 文档中的在 AWS CloudFormation 控制台上创建堆栈。	云架构师

Task	描述	所需技能
填写模板中的参数。	启动模板时，系统将会提示输入以下信息： • S3 存储桶：指定您在首个操作说明中创建或选择的存储桶。这是您上传所附的 Lambda 代码（文件）的地方。 • S3 密钥：指定 Lambda .zip 文件在您的 S3 存储桶中的位置（例如，filename.zip 或 controls/filename.zip）。切勿纳入前导斜字符。 • 通知电子邮件：提供有效的电子邮件地址以接收 Amazon SNS 通知。 • Lambda 日志级别：指定 Lambda 函数的日志记录级别和频率。使用信息记录有关进度的详细信息消息，使用错误记录仍然允许部署继续的错误事件，使用警告记录潜在的有害情况。	云架构师

确认订阅

Task	描述	所需技能
确认订阅。	CloudFormation 模板成功部署后，它会向您提供的电子邮件地址发送一封订阅电子邮件。您必须确认此电子邮件订阅，才能开始接收违规通知。	云架构师

相关资源

- 创建 S3 存储桶（Amazon S3 文档）
- 将文件上传至 S3 存储桶（Amazon S3 文档）
- 在 AWS CloudFormation 控制台上创建堆栈（AWS CloudFormation 文档）
- 使用 AWS 创建在 AWS API 调用时触发 CloudWatch 的事件规则 CloudTrail（AWS CloudTrail 文档）
- 创建 Amazon Redshift 集群（Amazon Redshift 文档）
- 为连接配置安全选项（Amazon Redshift 文档）

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

验证新 Amazon Redshift 集群是否在 VPC 中启动

由 Priyanka Chaudhary (AWS) 编写

摘要

此模式提供了一个亚马逊网络服务 (AWS) CloudFormation 模板，当在虚拟私有云 (VPC) 之外启动亚马逊 Redshift 集群时，该模板会自动通知您。

Amazon Redshift 是一种完全托管的 PB 级基于云的数据仓库产品。它专为大规模数据集存储与分析而设计。它还用于执行大规模的数据库迁移。Amazon Virtual Private Cloud（Amazon VPC）允许您预置 Amazon Web Services Cloud 的逻辑隔离部分，您可以在其中启动 AWS 资源，例如，您定义的虚拟网络中的 Amazon Redshift 集群。

此模式提供的安全控制会监控 AWS CloudTrail 日志中的 Amazon Redshift API 调用，并为和启动亚马逊事件 CloudWatch 事件。CreateClusterRestoreFromClusterSnapshot APIs当事件检测到其中一个事件时 APIs，它会调用运行 Python 脚本的 AWS Lambda。Python 函数会分析该 CloudWatch 事件。如果 Amazon Redshift 集群是从快照创建或还原，并且出现在 Amazon VPC 网络之外，则该函数会向用户发送 Amazon Simple Notification Service (Amazon SNS)通知，其中包含相关信息：此通知来源的 Amazon Redshift 集群名称、AWS 区域、Amazon Web Services account 和 Lambda 的 Amazon 资源名称（ARN）。

先决条件和限制

先决条件

- 一个有效的 Amazon Web Services account。
- 包含集群子网组和关联安全组的 VPC。

限制

- AWS CloudFormation 模板仅支持CreateCluster和RestoreFromClusterSnapshot操作（新集群）。它不会检测在 VPC 之外创建的、现有 Amazon Redshift 集群。
- 这种安全控制为区域性的。您必须将其部署在要监控的每个 AWS 区域。

架构

目标架构

自动化和扩缩

如果您使用的是 AWS Organ izations，则可以使用 AWS Cloudformation StackSets 在要监控的多个账户中部署此模板。

工具

Amazon Web Services

- AWS CloudFormation — AWS CloudFormation 可帮助您建模和设置 AWS 资源，快速一致地配置这些资源，并在资源的整个生命周期中对其进行管理。您可以使用模板来描述资源及其依赖关系，然后将它们作为堆栈一起启动和配置，而不必单独管理资源。
- AWS CloudTrail — AWS CloudTrail 可帮助您对 AWS 账户实施治理、合规以及运营和风险审计。用户、角色或 AWS 服务采取的操作在中记录为事件 CloudTrail。
- Amazon CloudWatch Events — Amazon CloudWatch Events 提供近乎实时的系统事件流，这些事件描述了 AWS 资源的变化。
- AWS Lambda – AWS Lambda 是一项计算服务，支持无需预置或管理服务器即可运行代码。只有在需要时 AWS Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。
- Amazon Redshift – Amazon Redshift 是一种完全托管的 PB 级云中数据仓库服务。Amazon Redshift 与数据湖集成，让您可以使用数据获得对您的业务和客户的新见解。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一项高度可扩展的对象存储服务，您可将其用于各种存储解决方案，包括网站、移动应用程序、备份和数据湖。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 可协调和管理发布者和客户端之间消息的传送或发送，包括 Web 服务器和电子邮件地址。

代码

此模式包括以下附件：

- `RedshiftMustBeInVPC.zip` — 用于安全控制的 Lambda 代码。
- `RedshiftMustBeInVPC.yml`— 用于设置事件和 Lambda 函数的 CloudFormation 模板。

要使用这些文件，请按照下一节中的说明操作。

操作说明

设置 S3 存储桶

Task	描述	所需技能
定义 S3 存储桶。	在 Amazon S3 控制台，选择或创建 S3 存储桶来托管 Lambda 代码 .zip 文件。此 S3 存储桶必须与您想要监视的 Amazon Redshift 集群位于相同的 AWS 区域 中。S3 存储桶名称是全局唯一的，并且命名空间由所有 Amazon Web Services account 共享。S3 存储桶名称不得包含前导斜杠。	云架构师
上传 Lambda 代码。	将附件部分中提供的 Lambda 代码(`RedshiftMustBeInVPC.zip` 文件)上传至 S3 存储桶。	云架构师

部署 CloudFormation 模板

Task	描述	所需技能
启动 CloudFormation 模板。	在您的 S3 存储桶所在的 AWS 区域中打开 AWS CloudFormation 控制台，然后部署附带的模板 (`RedshiftMustBeInVPC.yml`)。有关部署 AWS CloudFormation 模板的更多信息，请参阅 CloudFormation 文档中的在 AWS CloudFormation 控制台上创建堆栈。	云架构师

Task	描述	所需技能
填写模板中的参数。	启动模板时，系统将会提示输入以下信息： • S3 存储桶：指定您在首个操作说明中创建或选择的存储桶。这是您上传所附的 Lambda 代码（文件）的地方。 • S3 密钥：指定 Lambda .zip 文件在您的 S3 存储桶中的位置（例如，filename.zip 或 controls/filename.zip）。切勿纳入前导斜字符。 • 通知电子邮件：提供有效的电子邮件地址以接收 Amazon SNS 通知。 • Lambda 日志级别：指定 Lambda 函数的日志记录级别和频率。使用信息记录有关进度的详细信息消息，使用错误记录仍然允许部署继续的错误事件，使用警告记录潜在的有害情况。	云架构师

确认订阅

Task	描述	所需技能
确认订阅。	CloudFormation 模板成功部署后，它会向您提供的电子邮件地址发送一封订阅电子邮件。您必须确认此电子邮件订阅，才能开始接收违规通知。	云架构师

相关资源

- 创建 S3 存储桶（Amazon S3 文档）
- 将文件上传至 S3 存储桶（Amazon S3 文档）
- 在 AWS CloudFormation 控制台上创建堆栈（AWS CloudFormation 文档）
- 使用 AWS 创建在 AWS API 调用时触发 CloudWatch 的事件规则 CloudTrail（AWS CloudTrail 文档）
- 创建 Amazon Redshift 集群（Amazon Redshift 文档）

附件

要访问与此文档相关联的其他内容，请解压以下文件：attachment.zip

更多模式

- 使用会话管理器和 Amazon Instance Connect 访问堡垒主 EC2 机
- 使用 AWS Fargate PrivateLink、AWS 和网络负载均衡器在 Amazon ECS 上私下访问容器应用程序
- 使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon ECS 上私密访问容器应用程序
- 使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon EKS 上私密访问容器应用程序
- 允许 EC2 实例在 AMS 账户中写入对 S3 存储桶的访问权限
- 将一个 AWS CodeCommit 存储库 AWS 账户 与另一个账户中的 Amazon A SageMaker I Studio Classic 关联起来
- 使用亚马逊 Cognito 和 AWS Amplify 用户界面对现有 React 应用程序用户进行身份验证
- 通过 AWS Systems Manager 自动添加或更新 Windows 注册表项
- 使用 AWS CloudFormation 模板在 AWS Glue 中自动执行加密
- 使用云托管人和 AWS CDK 将适用于 Systems Manager 的 AWS 托管策略自动附加到 EC2 实例配置文件
- 自动加密现有和新 Amazon EBS 卷
- 通过使用 Cloud Custodian 来阻止对 Amazon RDS 的公有访问
- 使用本地微软 Active Direct AWS Managed Microsoft AD ory 集中解析 DNS
- 在部署 AWS 基础设施之前，实施集中式自定义 Checkov 扫描以强制执行策略
- 使用 cdk-nag 规则包查看 AWS CDK 应用程序或 CloudFormation 模板以了解最佳实践
- 在启动时检查 EC2 实例是否有必需的标签
- 配置对 Amazon DynamoDB 的跨账户访问
- 使用 Applicati EnterpriseOne on Load Balancer 为 Oracle WebLogic JD Edwards 配置 HTTPS 加密
- 为 AWS IoT 环境中的安全事件配置日志记录和监控
- 为在 Amazon EKS 上运行的应用程序配置双向 TLS 身份验证
- 在 pgAdmin 中使用 SSH 隧道进行连接
- 使用 AWS Amplify 创建 React 应用程序，并使用 Amazon Cognito 添加身份验证
- 创建网络访问分析器在多个入站互联网访问中发现的报告 AWS 账户
- 自定义 Amazon CloudWatch 提醒 AWS Network Firewall
- 使用 AWS Network Firewall 和 AWS Transit Gateway 部署防火墙
- 部署 ChatOps 解决方案来管理 SAST 扫描结果，方法是在聊天应用程序中使用 Amazon Q Developer 自定义操作和 AWS CloudFormation

- 记录您的 AWS 着陆区设计
- 在 Amazon RDS 中为 PostgreSQL 数据库实例启用加密连接
- 加密现有 Amazon RDS for PostgreSQL 数据库实例
- 在启动时强制对 Amazon RDS 数据库执行自动标记
- 启动时强制标记 Amazon EMR 集群
- 确保在启动时启用 Amazon EMR 日志记录到 Amazon S3
- 使用 Troposphere 生成包含 AWS Config 托管规则的 AWS CloudFormation 模板
- 当 AWS KMS 密钥的密钥状态发生变化时获取 Amazon SNS 通知
- 帮助强制执行 DynamoDB 标签
- 在 Amazon Data Firehose 资源未使用密钥加密时进行识别并发出警报 AWS KMS
- 从 SQL Server 迁移到 PostgreSQL 时，对 PII 数据实施 SHA1 哈希处理
- 通过在多个 AWS 区域、账户中启用 Amazon DevOps Guru 并 OUs 使用 AWS CDK 来提高运营绩效
- 将 EC2 Windows 实例提取并迁移到 AWS Managed Services 账户
- 使用 AWS DMS 在 SSL 模式下将 Amazon RDS for Oracle 迁移到 Amazon RDS for PostgreSQL
- 将 ELK 堆栈迁移至 Elastic Cloud on AWS
- 将 F5 BIG-IP 工作负载迁移至 Amazon Web Services Cloud 上的 F5 BIG-IP VE
- 监控 Amazon Aurora 以查找未加密的实例
- 通过部署角色自动售货机解决方案来配置权限最低的 IAM 角色
- 在不重启容器的情况下轮换数据库凭证
- 使用可信上下文在 AWS 上的 Db2 联合身份验证数据库中保护和简化用户访问
- 使用 AWS Firewall Manager 和 Amazon Data Firehose 将 AWS WAF 日志发送到 Splunk
- 使用亚马逊通过 VPC 在 Amazon S3 存储桶中提供静态内容 CloudFront
- 使用证书管理器和 “让我们加 end-to-end密” 为 Amazon EKS 上的应用程序设置加密
- IDs 在 IAM 策略中使用用户进行访问控制和自动化
- 验证 ELB 负载均衡器是否需 TLS 终止
- 使用 Splunk 查看 AWS Network Firewall 日志和指标
- 使用 Amazon 可视化所有 AWS 账户的 IAM 凭证报告 QuickSight

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。