aws

Guia do Desenvolvedor

Amazon Elastic Container Service

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

Amazon Elastic Container Service: Guia do Desenvolvedor

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

As marcas comerciais e imagens comerciais da Amazon não podem ser usadas no contexto de nenhum produto ou serviço que não seja da Amazon, nem de qualquer maneira que possa gerar confusão entre os clientes ou que deprecie ou desprestigie a Amazon. Todas as outras marcas comerciais que não pertencem à Amazon pertencem a seus respectivos proprietários, que podem ou não ser afiliados, patrocinados pela Amazon ou ter conexão com ela.

Table of Contents

O que é o Amazon ECS? 1
Terminologia e componentes 1
Atributos 3
Provisionamento 3
Preços 4
Serviços relacionados 4
Começar 6
Configuração 6
AWS Management Console 6
Inscrever-se para uma Conta da AWS 7
Criar um usuário com acesso administrativo 7
Criar uma nuvem privada virtual 9
Criar um grupo de segurança 10
Criar as credenciais e conectar-se à sua instância do EC2 14
Instalar a AWS CLI 14
Próximas etapas para usar o Amazon ECS 15
Criação de uma imagem de contêiner 16
Pré-requisitos 16
Criar uma imagem do Docker 18
Enviar a imagem para o Amazon Elastic Container Registry 20
Limpeza 22
Próximas etapas 22
Saiba como criar uma tarefa do Linux para o tipo de inicialização do Fargate 22
Pré-requisitos 23
Etapa 1: criar um cluster 23
Etapa 2: criar uma definição de tarefa 24
Etapa 3: criar um serviço 25
Etapa 4: visualizar o seu serviço 26
Etapa 5: limpar 26
Saiba como criar uma tarefa do Windows para o tipo de inicialização do Fargate 27
Pré-requisitos 27
Etapa 1: criar um cluster 28
Etapa 2: registrar uma definição de tarefa do Windows 29
Etapa 3: criar um serviço com sua definição de tarefa 30

Etapa 4: visualizar o serviço 31
Etapa 5: limpar 31
Saiba como criar uma tarefa do Windows para o tipo de inicialização do EC2 32
Pré-requisitos 32
Etapa 1: criar um cluster 33
Etapa 2: registrar uma definição de tarefa 35
Etapa 3: criar um serviço 36
Etapa 4: visualizar o serviço 37
Etapa 5: limpar 38
Como usar o AWS CDK 38
Etapa 1: configurar o projeto do AWS CDK 39
Etapa 2: usar o AWS CDK para definir um servidor Web em contêiner no Fargate 42
Etapa 3: testar o serviço da Web 49
Etapa 4: limpar 50
Próximas etapas 50
Criar recursos usando a CLI do AWS Copilot 51
Instalar a CLI do AWS Copilot 52
Implantação de uma amostra de aplicação do Amazon ECS usando a CLI do AWS Copilot 60
Amazon ECS com o AWS CloudFormation 63
Exemplos de modelos do AWS CloudFormation 64
Definições de tarefa 64
Clusters 76
Serviços 89
Perfis do IAM para o Amazon ECS 146
Criar recursos do Amazon ECS usando o console do AWS CloudFormation 149
Pré-requisitos 149
Etapa 1: criar um modelo de pilha 150
Etapa 2: criar uma pilha para recursos do Amazon ECS 155
Etapa 3: verificar 156
Etapa 4: Limpar os recursos 157
Criar recursos do Amazon ECS usando comandos da AWS CLI para o AWS CloudFormation 157
Pré-requisitos 158
Etapa 1: criar uma pilha 158
Etapa 2: Verificar a criação do recurso 164

Etapa 3: Limpeza 165
Práticas recomendadas 166
AWS Fargate 171
Instruções 171
Provedores de capacidade 172
Definições de tarefa 172
Versões da plataforma 172
Balanceamento de carga do serviço 173
Métricas de uso 173
Considerações de segurança sobre quando usar o tipo de inicialização do Fargate 174
Práticas recomendadas de segurança do Fargate 174
Uso do AWS KMS para criptografar o armazenamento temporário para o Fargate 174
Funcionalidade SYS_PTRACE para rastreamento de chamada de sistema do kernel com o Fargate 175
Uso do Amazon GuardDuty com monitoramento de runtime para o Fargate 175
Considerações de segurança do Fargate 176
Versões da plataforma do Fargate 177
..... 177
Migração para a versão da plataforma Linux 1.4.0 178
Log de alterações da versão da plataforma Linux 179
Defasagem da versão da plataforma Linux 182
Log de alteração da versão da plataforma Windows 185
Considerações sobre contêineres do Windows no Fargate para o Amazon ECS 185
Armazenamento temporário de tarefas do Fargate 187
Versões da plataforma de contêiner Linux do Fargate 187
Versões da plataforma de contêiner Windows do Fargate 188
Chaves gerenciadas pelo cliente para o armazenamento efêmero do AWS Fargate 189
Retirada e manutenção de tarefas 202
Visão geral do aviso de retirada de tarefa 203
Posso optar pela não retirada de tarefas? 207
Posso receber notificações de retirada de tarefas por meio de outros serviços da AWS? 207
Posso alterar a retirada de uma tarefa depois de programada? 207
Como o Amazon ECS processa tarefas que fazem parte de um serviço? 208
O Amazon ECS pode processar automaticamente as tarefas autônomas? 208
Prepare-se para a retirada de tarefas do AWS Fargate no Amazon ECS 208
Regiões do AWS Fargate 211

Contêineres do Linux no AWS Fargate ... 211
Contêineres do Windows no AWS Fargate ... 213
Arquitetar a solução para o Amazon ECS ... 216
Capacidade ... 217
Service endpoints ... 218
Redes ... 218
Acesso a recursos ... 219
Perfis do IAM ... 220
Registro em log ... 220
Tipos de inicialização ... 221
Tipo de inicialização do Fargate ... 221
Tipo de inicialização do EC2 ... 226
Tipo de inicialização externa ... 228
Usar endpoints de pilha dupla ... 229
Usar endpoints de pilha dupla (AWS CLI) ... 230
Usar endpoints de pilha dupla (AWS SDKs) ... 231
Usar endpoints de pilha dupla da API REST ... 231
Aplicações em sub-redes compartilhadas, zonas locais e zonas do Wavelength ... 232
Sub-redes compartilhadas ... 232
Zonas Locais ... 234
Zonas do Wavelength ... 234
Amazon Elastic Container Service no AWS Outposts ... 235
Considerações ... 235
Pré-requisitos ... 236
Visão geral da criação de clusters em AWS Outposts ... 236
Práticas recomendadas de rede ... 238
Conexão de aplicações à Internet ... 239
Práticas recomendadas para receber conexões de entrada para o Amazon ECS ... 243
Práticas recomendadas para conectar a serviços da AWS ... 248
Práticas recomendadas para conectar serviços ... 251
Práticas recomendadas para conectar serviços em rede entre contas da AWS e VPCs ... 256
Serviços da AWS para solução de problemas de rede ... 257
Otimização do tempo de inicialização da tarefa ... 258
Operação em grande escala ... 260
Controles de utilização de API e cotas de serviço do Amazon ECS ... 260
Lidar com problemas de controle de utilização ... 265

Práticas recomendadas de gerenciamento de capacidade e ajuste de escala automático 269
Determinação do tamanho de tarefas 269
Otimização do ajuste de escala automático de serviços 271
Capacidade e disponibilidade 277
Capacidade de clusters 282
Escolha de tamanhos de tarefas do Fargate 283
Aceleração do ajuste de escala automático de clusters 283
Acessar atributos com configurações de conta 285
Nomes de recursos da Amazon (ARNs) e IDs 286
Cronograma do formato do ARN e do ID do recurso 289
Container Insights 289
Conformidade do AWS Fargate com o Padrão Federal de Processamento de Informações (FIPS-140) 291
Autorização para atribuição de tags 292
Cronograma de autorização para atribuição de tags 293
Tempo de espera para a retirada da tarefa do AWS Fargate 294
Aumento de interfaces de rede de instância de contêiner do Linux 296
Monitoramento de runtime (integração com o Amazon GuardDuty) 296
VPC IPv6 de pilha dupla 297
Modo de driver de logs padrão 298
Visualização das configurações da conta usando o console 298
Modificar configurações da conta 299
Reverter para as configurações padrão da conta 300
Gerenciar configurações de conta usando o AWS CLI 301
Perfis do IAM para o Amazon ECS 303
Definições de tarefa 307
Estados de definição de tarefa 308
Recursos do Amazon ECS que podem bloquear uma exclusão 309
Projetar sua aplicação 310
Práticas recomendadas para imagens de contêiner 312
Práticas recomendas para o tamanho das tarefas 314
Redes de tarefas do Amazon ECS para o tipo de inicialização do EC2 315
Redes de tarefas para o tipo de inicialização do Fargate 327
Opções de armazenamento para tarefas 331
Gerenciar espaço de memória de troca do contêiner 422
Diferenças de definição de tarefa para o tipo de inicialização do Fargate 424

Diferenças de definição de tarefa para instâncias do EC2 executando o Windows 433
Criação de uma definição de tarefa usando o console 434
Validação de JSON 435
Pilhas do AWS CloudFormation 435
Procedimento 435
Atualização de uma definição de tarefa usando o console 467
Validação de JSON 468
Procedimento 468
Cancelamento de registro de uma revisão de definição de tarefa usando o console 469
Pilhas do AWS CloudFormation 435
Procedimento 470
Exclusão de registro de uma revisão de definição de tarefa usando o console 470
Recursos do Amazon ECS que podem bloquear uma exclusão 309
Procedimento 472
Casos de uso de definição de tarefa 472
Definições de tarefa para workloads de GPU 473
Definições de tarefa para workloads de transcodificação de vídeo 484
Definições de tarefa para workloads de machine learning do AWS Neuron 496
Definições de tarefa para instâncias de aprendizado profundo 505
Definições de tarefa para workloads do ARM de 64 bits 508
Envio de logs para o CloudWatch 510
Envio de logs para um serviço da AWS ou para uma AWS Partner 514
Uso de imagens de contêiner que não são da AWS 527
Reiniciar contêineres individuais em tarefas 531
Transferência de dados confidenciais para um contêiner 534
Parâmetros de definição de tarefa para o tipo de inicialização do Fargate 558
Família 558
Tipos de inicialização 559
Função da tarefa 559
Função de execução de tarefas 560
Modo de rede 560
Plataforma de runtime 560
Tamanho da tarefa 561
Definições de contêiner 565
Nome do acelerador de inferência elástica 612
Configuração do proxy 612

Volumes ... 614
Tags ... 621
Outros parâmetros de definição de tarefa ... 622
Parâmetros de definição de tarefa para o tipo de inicialização do EC2 ... 625
Família ... 625
Tipos de inicialização ... 626
Função da tarefa ... 626
Função de execução de tarefas ... 627
Modo de rede ... 627
Plataforma de runtime ... 629
Tamanho da tarefa ... 630
Definições de contêiner ... 631
Nome do acelerador de inferência elástica ... 682
Limitações de posicionamento de tarefa ... 683
Configuração do proxy ... 683
Volumes ... 686
Tags ... 693
Outros parâmetros de definição de tarefa ... 694
Modelo de definição de tarefa ... 696
Exemplos de definições de tarefa ... 707
Webserver ... 707
Driver de log do `splunk` ... 709
Driver de log do `fluentd` ... 710
Driver de log do `gelf` ... 711
Workloads em instâncias externas ... 711
Perfil do IAM para definição de imagem e tarefa do Amazon ECR ... 713
Ponto de entrada com comando ... 714
Dependência de contêiner ... 714
Volumes nas definições de tarefa ... 716
Definições de tarefa de exemplo do Windows ... 717
Clusters ... 719
Provedores de capacidade ... 720
Clusters para Fargate ... 721
Avisos de encerramento do Fargate Spot ... 723
Criar um cluster para o tipo de inicialização do Fargate ... 725
Provedores de capacidade para o tipo de inicialização do EC2 ... 727

Segurança da instância de contêiner do EC2 ... 730
Criação de um cluster do Amazon ECS para o tipo de inicialização do Amazon EC2 ... 731
Autoescalabilidade de cluster ... 737
Instâncias de contêiner do Amazon EC2 ... 774
Clusters para o tipo de inicialização externa ... 923
Sistemas operacionais e arquiteturas de sistema compatíveis ... 923
Considerações ... 924
Criar um cluster para o tipo de inicialização externa ... 927
Registro de uma instância externa para um cluster do Amazon ECS ... 930
Cancelar o registro de uma instância externa ... 936
Atualização do agente do AWS Systems Manager e o agente de contêiner do Amazon ECS ... 941
Atualizar um cluster ... 946
Excluir um cluster ... 948
Cancelamento do registro de uma instância de contêiner ... 949
Procedimento ... 950
Drenagem de instâncias de contêineres ... 950
Comportamento de drenagem para serviços ... 951
Comportamento de drenagem para tarefas autônomas ... 952
Procedimento ... 952
Instâncias de contêineres do Amazon EC2 ... 953
Ciclo de vida ... 954
Compatibilidade com o Docker ... 955
AMIs otimizadas para Amazon ECS ... 955
Mais informações ... 955
Configuração do agente de contêiner ... 956
Instalar o agente de contêiner do Amazon ECS ... 959
Parâmetros de configuração do log do agente de contêiner ... 966
Configuração de instâncias de contêiner para imagens do Docker privadas ... 967
Limpeza de tarefas e de imagens ... 972
Programação de contêineres ... 975
Opções de computação ... 977
Ciclo de vida da tarefa ... 978
Estados do ciclo de vida ... 979
Como o Amazon ECS posiciona tarefas em instâncias de contêineres ... 981
Tipo de inicialização do EC2 ... 981

Tipo de inicialização do Fargate 982
Uso de estratégias para definir o posicionamento de tarefas 983
Tarefas relacionadas a grupos 989
Defina quais instâncias de contêiner são usadas em tarefas 989
Tarefas autônomas 1000
Fluxo de trabalho de tarefas 1000
Execução de uma aplicação como uma tarefa 1001
Uso do Agendador do Amazon EventBridge para programar tarefas 1013
Interrupção de uma tarefa 1021
Serviços 1022
Opção de computação de infraestrutura 1024
Autoescalabilidade do serviço 1024
Balanceamento de carga do serviço 1025
Serviços de interconexão 1026
Controladores e estratégias de implantação de serviços 1027
Implantações de serviços 1185
Revisões de serviços 1195
Rebalanceamento de zonas de disponibilidade 1204
Uso do balanceamento de carga para distribuir o tráfego de serviço 1214
Autoescalabilidade do serviço 1229
Interconexão de serviços 1271
Proteção de tarefa na redução da escala horizontalmente 1331
Injeção de falhas com o Amazon ECS e o Fargate 1341
Atualizar parâmetros de serviço do Amazon ECS 1350
Excluir um serviço 1381
Migrar o ARN curto de um serviço 1382
Lógica de controle de utilização de serviço 1388
Parâmetros de definição de serviço 1389
Modelo de definição de serviço 1418
Marcar recursos 1426
Como os recursos são marcados 1427
Atribuição de tags a recursos durante a criação 1430
Restrições 1431
Tags gerenciadas pelo Amazon ECS 1431
Usar etiquetas para faturamento 1432
Adicionar tags aos recursos 1433

Adicionar etiquetas a uma instância de contêiner 1435
Instâncias de contêiner externas 1437
Relatórios de uso 1438
Custo e uso em nível de tarefa 1439
Monitoramento 1441
Práticas recomendadas para monitorar o Amazon ECS 1442
Ferramentas de monitoramento 1442
Ferramentas automatizadas 1442
Ferramentas manuais 1444
Monitoramento do Amazon ECS usando o CloudWatch 1445
Considerações 1446
Métricas recomendadas 1447
Visualizar métricas do Amazon ECS 1447
Métricas do Amazon ECS CloudWatch 1449
Métricas de uso do AWS Fargate 1479
Métricas de reserva de cluster do Amazon ECS 1481
Métricas de utilização de cluster do Amazon ECS 1483
Métricas de utilização do serviço do Amazon ECS 1485
Automatização de respostas a erros do Amazon ECS usando o EventBridge 1489
Eventos do Amazon ECS 1490
Processar eventos 1514
Monitorar contêineres do Amazon ECS utilizando o Container Insights com observabilidade aprimorada 1518
Como determinar a integridade das tarefas usando verificações de integridade do contêiner . 1520
Como a integridade da tarefa é determinada 1521
Verificações de integridade e desconexões de agentes 1523
Visualização da integridade do contêiner 1523
Monitoramento da integridade da instância de contêiner do Amazon ECS 1523
Problemas de integridade de instâncias de contêiner 1524
Identifique oportunidades de otimização do Amazon ECS usando dados de rastreamento de aplicações 1525
Permissões obrigatórias do IAM para integração da distribuição da AWS do OpenTelemetry com o AWS X-Ray 1525
Especificação do arquivo associado da distribuição da AWS do OpenTelemetry para integração do AWS X-Ray na sua definição de tarefa 1527
Correlação do desempenho da aplicação do Amazon ECS usando métricas de aplicações ... 1528

Exportação de métricas de aplicações para o Amazon CloudWatch 1529
Exportação de métricas de aplicações para o Amazon Managed Service for Prometheus . 1533
Registro em log das chamadas de API do Amazon ECS usando o AWS CloudTrail 1538
Eventos de gerenciamento do Amazon ECS no CloudTrail 1540
Exemplos de eventos do Amazon ECS 1540
Visualização do CloudWatch Logs Live Tail para serviços e tarefas do Amazon ECS 1542
Permissões obrigatórias 1542
Procedimento 1542
References da AWS CLI 1543
Monitorar workloads usando metadados 1544
Variáveis de ambiente 1545
Arquivo de metadados de contêiner 1546
Metadados de tarefas disponíveis para tarefas do Amazon ECS no EC2 1552
Metadados de tarefas disponíveis para tarefas no Fargate 1596
Introspecção de contêiner 1619
Identificação de comportamentos não autorizados usando o monitoramento de runtime 1622
Como o monitoramento de runtime funciona com o Amazon ECS 1623
Considerações 1624
Utilização de recursos 1625
Monitoramento de runtime para workloads do Fargate 1625
Monitoramento de runtime para workloads do EC2 1629
Solucionar problemas de monitoramento de runtime 1634
Monitoramento de contêineres do Amazon ECS com o ECS Exec 1638
Considerações 1638
Pré-requisitos 1641
Arquitetura 1641
Usar o ECS Exec 1642
Registro em log usando o ECS Exec 1644
Uso de políticas do IAM para limitar o acesso ao ECS Exec 1649
Recomendações do Compute Optimizer 1653
Recomendações de tamanho de tarefa para o Fargate 1653
Solução de problemas 1654
Resolver erros de tarefa interrompida 1657
Atualizações de mensagens de erro de tarefa interrompida 1658
Visualizar erros de tarefa interrompida 1663
Mensagens de erro de tarefas interrompidas 1665

Verificar conectividade de tarefa 1686
Visualização de solicitações de perfil do IAM 1691
Visualizar mensagens de eventos de serviço 1692
Mensagens de eventos do serviço do Amazon ECS 1693
Mensagens de evento não íntegro de serviço do Amazon ECS 1705
Mensagens de eventos do serviço de rebalanceamento do serviço de zonas de disponibilidade do Amazon ECS 1706
Solução de problemas relacionados aos balanceadores de carga de serviço no Amazon ECS 1708
Solução de problemas relacionados ao ajuste de escala automático de serviço no Amazon ECS 1711
Solucionar problemas causados por erros de CPU ou memória inválida na definição de tarefa 1711
Visualização de logs do agente de contêiner 1713
Coleta de logs de contêiner com o coletor de logs do Amazon ECS 1715
Introspecção do agente 1718
Diagnóstico do Docker no Amazon ECS 1720
Listagem de contêineres do Docker no Amazon ECS 1720
Visualização de logs do Docker no Amazon ECS 1721
Inspeção de contêineres do Docker no Amazon ECS 1722
Configuração da saída detalhada do daemon do Docker no Amazon ECS 1723
Solução de problemas relacionados a `API error (500): devmapper` do Docker no Amazon ECS 1725
Solução de problemas do ECS Exec 1726
Verificar usando o verificador do Exec 1726
Erro ao chamar `execute-command` 1727
Solução de problemas do Amazon ECS Anywhere 1727
Problemas de registro de instância externa 1727
Problemas de rede de instâncias externas 1728
Problemas na execução de tarefas 1729
Cotas de controle de utilização do AWS Fargate 1729
Controle de utilização da API `RunTask` no Fargate 1730
Ajuste de cotas tarifárias no Fargate 1731
Motivos de falha da API 1731
Segurança 1742
Gerenciamento de Identidade e Acesso 1743

Público ... 1743
Autenticação com identidades ... 1744
Gerenciar o acesso usando políticas ... 1748
Como o Amazon Elastic Container Service funciona com o IAM ... 1751
Exemplos de políticas baseadas em identidade ... 1762
Políticas gerenciadas pela AWS para o Amazon ECS ... 1775
Uso de perfis vinculados ao serviço ... 1789
Perfis do IAM para o Amazon ECS ... 1793
Permissões necessárias para usar o console do Amazon ECS ... 1858
Permissões necessárias para funções do Lambda em implantações azul/verde do Amazon ECS ... 1867
Permissões obrigatórias do IAM para o ajuste de escala automático do serviço Amazon ECS ... 1869
Recursos de tags durante a criação ... 1870
Solução de problemas ... 1875
Práticas recomendadas do IAM ... 1877
Registro e Monitoramento ... 1880
Validação de compatibilidade ... 1882
Práticas recomendadas de conformidade e de segurança ... 1883
Conformidade com o FIPS-140 do AWS Fargate ... 1885
Considerações sobre o FIPS-140 do AWS Fargate ... 1886
Use FIPS no Fargate ... 1886
Uso do CloudTrail para auditoria do FIPS-140 do Fargate ... 1887
Segurança da infraestrutura ... 1888
VPC endpoints de interface (AWS PrivateLink) ... 1889
Modelo de responsabilidade compartilhada ... 1896
Tipo de inicialização do Fargate ... 1896
Tipo de inicialização do EC2 ... 1897
Práticas recomendadas de segurança de rede ... 1898
Criptografia em trânsito ... 1898
Redes de tarefas ... 1900
AWS PrivateLink e Amazon ECS ... 1900
Configurações do agente de contêiner ... 1901
Recomendações de segurança de rede ... 1902
Práticas recomendadas de segurança para tarefas e contêineres ... 1903
Criar imagens mínimas ou usar imagens sem distribuição ... 1904

Verifique suas imagens em busca de vulnerabilidades 1905
Remover permissões especiais de suas imagens 1906
Criar um conjunto de imagens selecionadas 1907
Varrer pacotes de aplicações e bibliotecas em busca de vulnerabilidades 1905
Executar análise estática de código 1907
Executar contêineres como um usuário não raiz 1908
Usar um sistema de arquivos raiz somente para leitura 1908
Configurar tarefas com limites de CPU e memória (Amazon EC2) 1908
Usar tags imutáveis com o Amazon ECR 1909
Evitar executar contêineres como privilegiados (Amazon EC2) 1909
Remover recursos desnecessários do Linux do contêiner 1910
Usar uma chave gerenciada pelo cliente (CMK) para criptografar imagens enviadas ao Amazon ECR. 1910
Tutoriais 1911
Criar uma tarefa do Linux para o tipo de inicialização do Fargate com a AWS CLI 1914
Pré-requisitos 1914
Etapa 1: criar um cluster 1915
Etapa 2: registrar uma definição de tarefa do Linux 1916
Etapa 3: listar definições de tarefa 1918
Etapa 4: criar um serviço 1918
Etapa 5: listar serviços 1919
Etapa 6: descrever o serviço em execução 1919
Etapa 7: testar 1922
Etapa 8: limpar 1925
Criar uma tarefa do Windows para o tipo de inicialização do Fargate com a AWS CLI 1926
Pré-requisitos 1926
Etapa 1: criar um cluster 1927
Etapa 2: registrar uma definição de tarefa do Windows 1928
Etapa 3: listar definições de tarefa 1929
Etapa 4: criar um serviço 1930
Etapa 5: listar serviços 1930
Etapa 6: descrever o serviço em execução 1931
Etapa 7: limpar 1933
Criar uma tarefa para o tipo de inicialização do EC2 com a AWS CLI 1933
Pré-requisitos 1934
Criar um cluster 1935

Executar uma instância de contêiner com a AMI do Amazon ECS 1935
Listar instâncias de contêiner 1938
Descrever sua instância de contêiner 1938
Registrar uma definição de tarefa 1941
Listar definições de tarefa 1943
Criar um serviço 1944
Listar serviços 1944
Descrever o serviço 1945
Descrever a tarefa em execução 1946
Testar o servidor Web 1947
Limpar recursos 1948
Configurar o Amazon ECS para atuar como receptor de eventos do CloudWatch Events 1949
Pré-requisito: configurar um cluster de teste 1950
Etapa 1: criar a função do Lambda 1950
Etapa 2: registrar uma regra de evento 1951
Etapa 3: criar uma definição de tarefa 1951
Etapa 4: testar a regra 1953
Envio de alertas do Amazon Simple Notification Service para eventos de tarefas interrompidas 1953
Pré-requisito: configurar um cluster de teste 1954
Pré-requisito: configurar permissões para o Amazon SNS 1954
Etapa 1: criar e se inscrever em um tópico do Amazon SNS 1954
Etapa 2: registrar uma regra de evento 1955
Etapa 3: testar a regra 1956
Concatenar mensagens de log de várias linhas ou de rastreamento de pilha 1957
Permissões obrigatórias do IAM 1958
Determinar quando usar configuração de logs multilinha 1959
Opções de análise e concatenação 1961
Implantar o Fluent Bit em contêineres do Windows 1980
Pré-requisitos 1983
Etapa 1: criar os perfis de acesso do IAM 1984
Etapa 2: criar uma instância de contêiner do Windows do Amazon ECS 1985
Etapa 3: configurar o Fluent Bit 1986
Etapa 4: registrar uma definição de tarefa do Fluent Bit no Windows que roteia os logs para o CloudWatch 1988

Etapa 5: executar a definição de tarefa `ecs-windows-fluent-bit` como um serviço do Amazon ECS usando a estratégia de programação do daemon ... 1990
Etapa 6: registrar uma definição de tarefa do Windows que gere os logs ... 1991
Etapa 7: executar a definição de tarefa `windows-app-task` ... 1992
Etapa 8: verificar os logs no CloudWatch ... 1993
Etapa 9: limpar ... 1994
Usar gMSA para contêineres do Linux do EC2 ... 1995
Considerações ... 1995
Pré-requisitos ... 1997
Configuração ... 1998
CredSpec file ... 2005
Uso de gMSA em contêineres do Linux no Fargate ... 2006
Considerações ... 2006
Pré-requisitos ... 2007
Configuração ... 2008
CredSpec file ... 2011
Usar contêineres do Windows com gMSA sem domínio usando a AWS CLI ... 2012
Pré-requisitos ... 2013
Etapa 1: criar e configurar a conta gMSA nos Serviços de Domínio Active Directory (AD DS) ... 2015
Etapa 2: fazer upload de credenciais no Secrets Manager ... 2017
Etapa 3: modificar seu JSON do CredSpec para incluir informações do gMSA sem domínio ... 2018
Etapa 4: fazer upload do CredSpec no Amazon S3 ... 2019
Etapa 5: (opcional) criar um cluster do Amazon ECS ... 2020
Etapa 6: criar um perfil do IAM para instâncias de contêiner ... 2020
Etapa 7: criar um perfil de execução de tarefa personalizado ... 2021
Etapa 8: criar um perfil de tarefa para o Amazon ECS Exec ... 2022
Etapa 9: registrar uma definição de tarefa ... 2024
Etapa 10: registrar uma instância de contêiner do Windows ... 2025
Etapa 11: verificar a instância de contêiner ... 2026
Etapa 12: executar uma tarefa do Windows ... 2027
Etapa 13: verificar se o contêiner tem credenciais gMSA ... 2028
Etapa 14: limpar ... 2029
Depuração ... 2030
Saiba como usar gMSAs para contêineres do Windows do EC2 ... 2031

Considerações 2032
Pré-requisitos 2033
Configuração 2034
Usar o Image Builder para criar AMIs personalizadas otimizadas para o Amazon ECS 2040
Uso do ARN de imagem com a infraestrutura como código (IaC) 2041
Uso do ARN de imagem com o AWS CloudFormation 2044
Uso do ARN de imagem com o Terraform 2045
Usar containers de aprendizado profundo da AWS 2045
Cotas de serviço 2047
Gerenciamento das cotas de serviços no AWS Management Console 2047
Como lidar com as cotas de serviço e os limites de controle de utilização de API 2049
Elastic Load Balancing 261
Interfaces de rede elástica 263
AWS Cloud Map 265
Referência da API do Amazon ECS 2055
Histórico do documento 2056

O que é o Amazon Elastic Container Service?

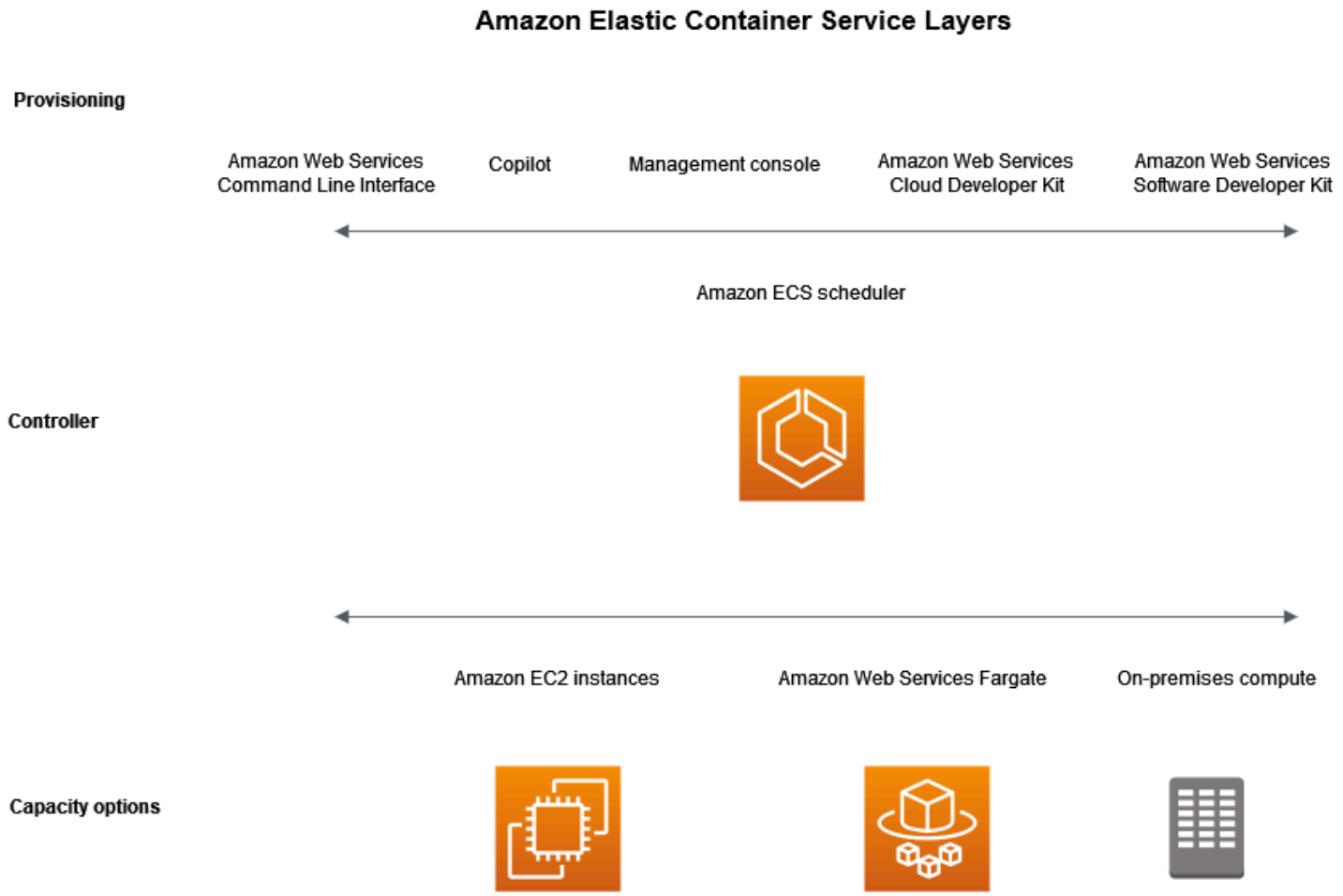
O Amazon Elastic Container Service (Amazon ECS) é um serviço totalmente gerenciado de orquestração de contêineres ajuda a implantar, gerenciar e dimensionar facilmente aplicações conteinerizadas. Como um serviço totalmente gerenciado, o Amazon ECS vem com práticas recomendadas operacionais e de configuração da AWS incorporadas. Ele é integrado tanto a ferramentas da AWS, como o Amazon Elastic Container Registry, quanto a ferramentas de terceiros, como o Docker. Essa integração torna mais fácil para as equipes se concentrarem na criação das aplicações, não no ambiente. É possível executar e escalar suas workloads de contêiner nas Regiões da AWS na nuvem e on-premises, sem a complexidade de gerenciar um ambiente de gerenciamento.

Terminologia e componentes

Há três camadas no Amazon ECS:

- Capacidade: a infraestrutura em que seus contêineres são executados
- Controlador: implanta e gerencia suas aplicações que são executadas nos contêineres
- Provisionamento: as ferramentas que podem ser usadas para interagir com o programador para implantar e gerenciar suas aplicações e contêineres

O diagrama a seguir mostra as camadas do Amazon ECS.



A capacidade é a infraestrutura onde os contêineres são executados. Veja a seguir uma visão geral das opções de capacidade:

- Instâncias do Amazon EC2 na nuvem da AWS

  Você escolhe o tipo de instância, o número de instâncias e gerencia a capacidade.

- Tecnologia sem servidor (AWS Fargate) na nuvem da AWS

  O Fargate é um mecanismo de computação sem servidor com pagamento conforme o uso. Com o Fargate, você não precisa gerenciar servidores, lidar com o planejamento de capacidade ou isolar workloads de contêineres para segurança.

- Máquinas virtuais (VM) ou servidores on-premises

  O Amazon ECS Anywhere fornece suporte para registrar uma Instância externa, como um servidor on-premises ou uma máquina virtual (VM), no cluster do Amazon ECS.

O programador do Amazon ECS é o software que gerencia suas aplicações.

Atributos

O Amazon ECS fornece os seguintes recursos de alto nível:

Definição de tarefa

: O blueprint da aplicação.

Cluster

: A infraestrutura em que a aplicação é executada.

Tarefa

: Uma aplicação, como um trabalho em lote, que executa o trabalho e depois para.

Serviço

: Um aplicação stateless de longa execução.

Configuração da conta

: Permite acesso a recursos.

Auto Scaling de cluster

: O Amazon ECS gerencia o ajuste de escala de instâncias do Amazon EC2 registradas no seu cluster.

Ajuste de escala automático do serviço

: O Amazon ECS aumenta ou diminui automaticamente o número desejado de tarefas no seu serviço.

Provisionamento

Há várias opções para provisionar o Amazon ECS:

- AWS Management Console: fornece uma interface da Web que você pode usar para acessar seus recursos do Amazon ECS.
- AWS Command Line Interface (AWS CLI): fornece comandos para um amplo conjunto de serviços da AWS, incluindo o Amazon ECS. Há suporte para o Windows, Mac e Linux. Para obter mais informações, consulte AWS Command Line Interface.

- SDKs da AWS: fornece APIs específicas de idioma e cuida de muitos dos detalhes da conexão. Elas incluem o cálculo de assinaturas, o tratamento de novas tentativas de solicitação e o tratamento de erros. Para obter mais informações, consulte AWS SDKs.
- AWS CDK: fornece um framework de desenvolvimento de software de código aberto que você pode usar para modelar e provisionar recursos de aplicações em nuvem usando linguagens de programação conhecidas. O AWS CDK provisiona seus recursos de forma segura e repetível por meio do AWS CloudFormation.

Preços

Os preços do Amazon ECS dependem da opção de capacidade que você escolher para seus contêineres.

- Preços do Amazon ECS: informações sobre preços para o Amazon ECS.
- Preços do AWS Fargate: informações sobre preços para o Fargate.

Serviços relacionados

Serviços para uso com o Amazon ECS

Você pode usar outros Serviços da AWS para obter ajuda para implantar suas tarefas e serviços no Amazon ECS.

Amazon EC2 Auto Scaling

Ajuda a garantir que você tenha o número correto de instâncias do Amazon EC2 disponíveis para processar a carga da aplicação.

Amazon CloudWatch

Monitore seus serviços e tarefas.

Amazon Elastic Container Registry

Envie por push e gerencie imagens de contêiner.

Elastic Load Balancing

Distribua automaticamente o tráfego de entrada do serviço.

Amazon GuardDuty

Detecte uso potencialmente não autorizado ou malicioso de suas instâncias de contêiner e workloads.

Saiba como criar e usar os recursos do Amazon ECS

Os guias a seguir fornecem uma introdução às ferramentas disponíveis para acessar o Amazon ECS e procedimentos introdutórios para executar contêineres. Os conceitos básico do Docker orientam você pelas etapas básicas para criar uma imagem de contêiner do Docker e carregá-la em um repositório privado do Amazon ECR. Os guias de conceitos básicos orientam você por meio da interface da linha de comando do AWS Copilot e do AWS Management Console para a conclusão das tarefas comuns para execução de contêineres no Amazon ECS e no AWS Fargate.

Conteúdo

- Configuração para usar o Amazon ECS
- Criação de uma imagem de contêiner para uso no Amazon ECS
- Saiba como criar uma tarefa do Linux no Amazon ECS para o tipo de inicialização do Fargate
- Saiba como criar uma tarefa do Windows no Amazon ECS para o tipo de inicialização do Fargate
- Saiba como criar uma tarefa do Windows no Amazon ECS para o tipo de inicialização do EC2
- Criar recursos do Amazon ECS usando o AWS CDK
- Criar de recursos do Amazon ECS usando a interface de linha de comando do AWS Copilot

Configuração para usar o Amazon ECS

Se você já tiver se cadastrado no Amazon Web Services (AWS) e usado o Amazon Elastic Compute Cloud (Amazon EC2), estará próximo de poder usar o Amazon ECS. O processo de configuração para os dois serviços é semelhante. O guia a seguir prepara você para iniciar seu primeiro cluster do Amazon ECS.

Conclua as tarefas a seguir para estar preparado para o Amazon ECS.

AWS Management Console

O AWS Management Console é uma interface baseada em navegador para gerenciar recursos do Amazon ECS. O console fornece uma visão geral visual do serviço, facilitando a exploração de recursos e funções do Amazon ECS sem a necessidade do uso de ferramentas adicionais. Muitos tutoriais e demonstrações relacionados estão disponíveis para guiar você pelo uso do console.

Para obter um tutorial que guie você pelo console, consulte Saiba como criar e usar os recursos do Amazon ECS.

Ao começar, muitos clientes preferem usar o console porque ele fornece feedback visual instantâneo quanto a se as ações que eles executam são bem-sucedidas. Clientes da AWS familiarizados com o AWS Management Console podem gerenciar facilmente recursos relacionados, como balanceadores de carga e instâncias do Amazon EC2.

Comece com o AWS Management Console.

Inscrever-se para uma Conta da AWS

Se você ainda não tem uma Conta da AWS, siga as etapas abaixo para criar uma.

Como cadastrar uma Conta da AWS

1. Abra https://portal.aws.amazon.com/billing/signup.
2. Siga as instruções online.

   Parte do procedimento de inscrição envolve receber uma chamada telefônica ou uma mensagem de texto e inserir um código de verificação pelo teclado do telefone.

   Quando você se inscreve para uma Conta da AWS, um Usuário raiz da conta da AWS é criado. O usuário-raiz tem acesso a todos os Serviços da AWS e recursos na conta. Como prática recomendada de segurança, atribua o acesso administrativo a um usuário e use somente o usuário-raiz para executar tarefas que exigem acesso de usuário-raiz.

A AWS envia um e-mail de confirmação depois que o processo de inscrição é concluído. A qualquer momento, é possível exibir as atividades da conta atual e gerenciar sua conta acessando https://aws.amazon.com/ e selecionando Minha conta.

Criar um usuário com acesso administrativo

Depois de se cadastrar em uma Conta da AWS, proteja seu Usuário raiz da conta da AWS, habilite o AWS IAM Identity Center e crie um usuário administrativo para não usar o usuário-raiz em tarefas cotidianas.

Proteger o Usuário raiz da conta da AWS

1. Faça login no AWS Management Console como o proprietário da conta ao escolher a opção Usuário-raiz e inserir o endereço de e-mail da Conta da AWS. Na próxima página, insira a senha.

Para obter ajuda ao fazer login usando o usuário-raiz, consulte Fazer login como usuário-raiz no Guia do usuário do Início de Sessão da AWS.

2. Habilite a autenticação multifator (MFA) para o usuário-raiz.

   Para obter instruções, consulte Habilitar um dispositivo MFA virtual para sua Conta da AWS de usuário-raiz (console) no Guia do usuário do IAM.

Criar um usuário com acesso administrativo

1. Habilita o Centro de Identidade do IAM.

   Para obter instruções, consulte Habilitar o AWS IAM Identity Center no Guia do usuário do AWS IAM Identity Center.

2. No Centro de Identidade do IAM, conceda o acesso administrativo a um usuário.

   Para obter um tutorial sobre como usar o Diretório do Centro de Identidade do IAM como a fonte de identidade, consulte Configurar o acesso dos usuários com o Diretório do Centro de Identidade do IAM padrão no Guia do usuário do AWS IAM Identity Center.

Iniciar sessão como o usuário com acesso administrativo

- Para fazer login com o seu usuário do Centro de Identidade do IAM, use o URL de login enviado ao seu endereço de e-mail quando o usuário do Centro de Identidade do IAM foi criado.

  Para obter ajuda para fazer login usando um usuário do Centro de Identidade do IAM, consulte Fazer login no portal de acesso da AWS, no Guia do usuário do Início de Sessão da AWS.

Atribuir acesso a usuários adicionais

1. No Centro de Identidade do IAM, crie um conjunto de permissões que siga as práticas recomendadas de aplicação de permissões com privilégio mínimo.

   Para obter instruções, consulte Criar um conjunto de permissões no Guia do usuário do AWS IAM Identity Center.

2. Atribua usuários a um grupo e, em seguida, atribua o acesso de autenticação única ao grupo.

Para obter instruções, consulte Adicionar grupos no Guia do usuário do AWS IAM Identity Center.

Criar uma nuvem privada virtual

É possível usar a Amazon Virtual Private Cloud (Amazon VPC) para iniciar recursos da AWS em uma rede virtual definida por você. É altamente recomendável ativar as instâncias de contêiner em um VPC.

Caso tenha uma VPC padrão, é possível ignorar esta seção e avançar à próxima tarefa, Criar um grupo de segurança. Para determinar se você tem uma VPC padrão, consulte Trabalhar com sua VPC padrão e sub-redes padrão no Guia do usuário da Amazon VPC. Do contrário, você pode criar uma VPC não padrão na conta usando as etapas abaixo.

Para obter informações sobre como criar uma VPC, consulte Criar uma VPC no Guia do usuário da Amazon VPC e use a tabela a seguir para determinar quais opções selecionar.

Opção	Valor
Recursos a criar	Somente VPC
Name	Forneça um nome opcional para a sua VPC.
Bloco IPv4 CIDR	Entrada manual IPv4 CIDR O tamanho do bloco CIDR deve ter um tamanho entre /16 e /28.
Bloco IPv6 CIDR	Sem bloco IPv6 CIDR
Locação	Padrão

Para saber mais sobre a Amazon VPC, consulte O Que é Amazon VPC? no Guia de usuário Amazon VPC.

Criar um grupo de segurança

Os grupos de segurança funcionam como um firewall para instâncias de contêiner associadas, controlando o tráfego de entrada e de saída no nível de instância de contêiner. É possível adicionar regras a um grupo de segurança que permite a você se conectar à instância de contêiner do endereço IP usando SSH. Você também pode adicionar regras que permitam o acesso HTTP e HTTPS de entrada e saída de qualquer lugar. Adicione eventuais regras a portas abertas exigidas pelas tarefas. As instâncias de contêiner precisam de acesso de rede externo para se comunicar com o endpoint de serviço do Amazon ECS.

Se você pretende executar instâncias de contêiner em várias regiões, será necessário criar um grupo de segurança em cada região. Para obter mais informações, consulte Regiões e Zonas de Disponibilidade no Guia do usuário do Amazon EC2.

> **Tip**
>
> O endereço IP público do computador local é necessário e pode ser obtido por meio de um serviço. Por exemplo, fornecemos o seguinte serviço: http://checkip.amazonaws.com/ ou https://checkip.amazonaws.com/. Para localizar outro serviço que forneça o endereço IP, use a frase de busca "qual é o meu endereço IP". Caso esteja se conectando por meio de um Internet Service Provider (ISP – Provedor de serviços de Internet) ou atrás de um firewall sem um endereço IP estático, você deve descobrir o intervalo de endereços IP usados por computadores cliente.

Para obter informações sobre como criar um grupo de segurança, consulte Criar um grupo de segurança para sua instância do Amazon EC2 no Guia do usuário do Amazon EC2 e use a tabela a seguir para determinar quais opções selecionar.

Opção	Valor
Região	A mesma região na qual você criou o par de chaves.
Name	Um nome que seja fácil de lembrar, como ecs-instances-default-cluster.

Opção	Valor
VPC	A VPC padrão (marcada com "[padrão]"). **Note** Se sua conta oferecer suporte ao Amazon EC2 Classic, selecione a VPC que você criou na tarefa anterior.

Para obter informações sobre as regras de saída a serem adicionadas aos seus casos de uso, consulte Regras do grupo de segurança para diferentes casos de uso no Manual do usuário do Amazon EC2.

As instâncias de contêiner do Amazon ECS não exigem que alguma porta de entrada esteja aberta. No entanto, você pode adicionar uma regra SSH para fazer login na instância de contêiner e examinar as tarefas com comandos do Docker. Você também pode adicionar regras para HTTP e HTTPS se desejar que sua instância de contêiner hospede uma tarefa que execute um servidor da Web. As instâncias de contêiner precisam de acesso de rede externa para se comunicar com o endpoint de serviço do Amazon ECS. Conclua as seguintes etapas para adicionar essas regras de security group opcionais.

Adicione as três regras de entrada a seguir ao grupo de segurança. Para obter informações sobre como criar um grupo de segurança, consulte Configurar regras de grupo de segurança no Guia do usuário do Amazon EC2.

Opção	Valor
Regra HTTP	Tipo: HTTP Origem: Qualquer lugar (`0.0.0.0/0`)

Opção	Valor	
	Essa opção adiciona automaticamente o bloco CIDR IPv4 0.0.0.0/0 como origem. Isso é aceitável por um período curto em um ambiente de teste, porém não é seguro em ambientes de produção. Em produção, autorize apenas um endereço IP específico ou um intervalo de endereços a acessar a instância.	
Regra HTTPS	Tipo: HTTPS Origem: Qualquer lugar (`0.0.0.0/0`) Isso é aceitável por um período curto em um ambiente de teste, porém não é seguro em ambientes de produção. Em produção, autorize apenas um endereço IP específico ou um intervalo de endereços a acessar a instância.	

Opção	Valor
Regra SSH	Tipo: SSH Origem: personalizada, especifique o endereço IP público do computador ou da rede em notação CIDR. Para especificar um único endereço IP em notação CIDR, adicione o prefixo de roteamento `/32`. Por exemplo, se o endereço IP for `203.0.113.25` , especifique `203.0.113.25/32` . Se sua empresa alocar endereços de um intervalo, especifiq ue o intervalo inteiro, como `203.0.113.0/24` . **Important** Por motivos de segurança, não recomendamos que você permita acesso SSH de todos os endereços IP (`0.0.0.0/0`) à sua instância, exceto para fins de teste e somente por um curto período.

Criar as credenciais e conectar-se à sua instância do EC2

Para Amazon ECS, o par de chaves só será necessário se você tiver a intenção de usar o tipo de inicialização do EC2.

A AWS usa criptografia de chave pública para proteger as informações de logon da instância. Uma instância do Linux, como uma instância de contêiner do Amazon ECS não possui senha a ser usada para acesso SSH. Você usa um par de chaves para fazer login na sua instância com segurança. Você especifica o nome do par de chaves ao ativar a instância de contêiner e, em seguida, fornece a chave privada ao fazer logon usando SSH.

Se ainda não tiver criado um par de chaves, será possível criar um usando o console do Amazon EC2. Se você planeja iniciar instâncias em várias regiões, você precisará criar um par de chaves em cada região. Para obter mais informações sobre regiões, consulte Regiões e zonas de disponibilidade no Manual do usuário do Amazon EC2.

Para criar um par de chaves

- Use o console do Amazon EC2 para criar um par de chaves. Para obter mais informações sobre como criar um par de chaves, consulte Criar um par de chaves no Manual do usuário do Amazon EC2.

Para obter informações sobre como se conectar à instância, consulte Conecte-se à sua instância do Linux no Manual do usuário do Amazon EC2.

Instalar a AWS CLI

O AWS Management Console pode ser usado para gerenciar manualmente todas as operações com o Amazon ECS. Porém, é possível instalar a AWS CLI no desktop local ou em uma área restrita de desenvolvedor para criar scripts capazes de automatizar tarefas comuns de gerenciamento no Amazon ECS.

Para usar a AWS CLI com o Amazon ECS, instale a versão mais recente da AWS CLI. Para obter informações sobre como instalar a AWS CLI ou atualizá-la para a versão mais recente, consulte Instalar ou atualizar para a versão mais recente da AWS CLI no Guia do usuário do AWS Command Line Interface.

A AWS Command Line Interface (AWS CLI) é uma ferramenta unificada que pode ser usada para gerenciar os serviços da AWS. Com esta ferramenta, isoladamente, você pode controlar vários

serviços da AWS e automatizar estes serviços por meio de scripts. Os comandos do Amazon ECS na AWS CLI são um reflexo da API do Amazon ECS.

A AWS CLI é adequada para clientes que preferem e estão acostumados a desenvolver scripts e interfaces com uma ferramenta da linha de comando e sabem exatamente que ações querem executar nos recursos do Amazon ECS. A AWS CLI também é útil para clientes que querem se familiarizar com as APIs do Amazon ECS. Os clientes podem usar a AWS CLI para executar várias operações nos recursos do Amazon ECS, inclusive operações de Criar, Ler, Atualizar e Excluir, diretamente da interface da linha de comando.

Use a AWS CLI se estiver ou quiser estar familiarizado com as APIs do Amazon ECS e com os comandos correspondentes da CLI e se quiser gravar scripts automatizados e executar ações específicas nos recursos do Amazon ECS.

AWS também fornece as ferramentas de linha de comando AWS Tools for Windows PowerShell. Para obter mais informações, consulte o Guia do usuário do AWS Tools for Windows PowerShell.

Próximas etapas para usar o Amazon ECS

Depois de instalar a AWS CLI, há muitas ferramentas diferentes que você pode utilizar à medida que continua usando o Amazon ECS. Os links a seguir explicam o que são algumas dessas ferramentas e dão exemplos de como usá-las com o Amazon ECS.

- Crie sua primeira imagem de contêiner com o Docker e envie-a para o Amazon ECR para uso em suas definições de tarefas do Amazon ECS.
- Saiba como criar uma tarefa do Linux no Amazon ECS para o tipo de inicialização do Fargate.
- Saiba como criar uma tarefa do Windows no Amazon ECS para o tipo de inicialização do Fargate.
- Saiba como criar uma tarefa do Windows no Amazon ECS para o tipo de inicialização do EC2.
- Usando sua linguagem de programação preferencial, defina infraestrutura ou arquitetura como código com Criar recursos do Amazon ECS usando o AWS CDK.
- Defina e gerencie todos os recursos da AWS no seu ambiente com implantação automatizada usando Usar o Amazon ECS com o AWS CloudFormation.
- Use o fluxo de trabalho completo Criar de recursos do Amazon ECS usando a interface de linha de comando do AWS Copilot do desenvolvedor para criar, lançar e operar aplicações de contêiner em conformidade com as práticas recomendadas da AWS para infraestrutura..

Criação de uma imagem de contêiner para uso no Amazon ECS

O Amazon ECS usa imagens do Docker nas definições de tarefa para iniciar contêineres. O Docker é uma tecnologia que fornece a você as ferramentas para criar, executar, testar e implantar aplicações em contêineres.

O Amazon ECS agenda aplicações em contêineres em instâncias de contêiner ou no AWS Fargate. As aplicações em contêineres são empacotadas como imagens de contêiner. Este exemplo cria uma imagem de contêiner para um servidor Web.

É possível criar sua primeira imagem do Docker e enviar essa imagem para o Amazon ECR, que é um registro de contêineres, para uso nas definições de tarefa do Amazon ECS. Esta demonstração supõe que você possua uma compreensão básica do que é o Docker e de como ele funciona. Para obter mais informações sobre o Docker, consulte O que é o Docker? e a Documentação do Docker.

Pré-requisitos

Antes de começar, certifique-se de que os seguintes pré-requisitos sejam atendidos.

- Certifique-se de ter concluído as etapas de configuração do Amazon ECR. Para obter mais informações, consulte Mover uma imagem ao longo do ciclo de vida no Amazon ECR no Guia do usuário do Amazon Elastic Container Registry.
- Seu usuário deve ter as permissões necessárias do IAM para acessar o serviço do Amazon ECR. Para obter mais informações, consulte Políticas gerenciadas do Amazon ECR.
- Você tem o Docker instalado. Para as etapas de instalação do Docker para o Amazon Linux 2023, consulte Instalação do Docker no AL2023. Em todos os outros sistemas operacionais, consulte a documentação do Docker em Visão geral do Docker Desktop.
- Tenha a AWS CLI instalada e configurada. Para obter mais informações, consulte Instalar ou atualizar para a versão mais recente da AWS CLI no Guia do usuário do AWS Command Line Interface.

Se você não tem ou não precisa de um ambiente de desenvolvimento local e preferir usar uma instância do Amazon EC2 para usar o Docker, disponibilizamos as seguintes etapas para a inicialização de uma instância do Amazon EC2 usando Amazon Linux 2023 e a instalação do Docker Engine e da CLI do Docker.

Instalação do Docker no AL2023

O Docker está disponível em muitos sistemas operacionais diferentes, incluindo a maioria das distribuições modernas do Linux, como o Ubuntu, e até no MacOS e no Windows. Para obter mais informações sobre como instalar o Docker no seu sistema operacional, consulte o Guia de instalação do Docker.

Não é necessário um sistema de desenvolvimento local para usar o Docker. Se você já usa o Amazon EC2, pode iniciar uma instância do Amazon Linux 2023 e instalar o Docker para começar.

Se você já tiver um Docker instalado, vá para Criar uma imagem do Docker.

Para instalar o Docker em uma instância do Amazon EC2 usando uma AMI do Amazon Linux 2023

1. Inicie uma instância com a mais recente AMI do Amazon Linux 2023. Para obter mais informações, consulte Iniciar uma instância do EC2 usando o assistente de inicialização de instâncias no Guia do usuário do Amazon EC2.
2. Conecte-se à sua instância. Para obter mais informações, consulte Conectar-se à sua instância do EC2 no Guia do usuário do Amazon EC2.
3. Atualize os pacotes instalados e o cache de pacotes em sua instância.

   ```
   sudo yum update -y
   ```

4. Instale o pacote do Docker Community Edition mais recente.

   ```
   sudo yum install docker
   ```

5. Inicie o serviço Docker.

   ```
   sudo service docker start
   ```

6. Adicione o `ec2-user` ao grupo `docker`, de modo que você possa executar comandos do Docker sem usar o `sudo`.

   ```
   sudo usermod -a -G docker ec2-user
   ```

7. Faça logout e login novamente para selecionar as novas permissões do grupo `docker`. É possível fazer isso ao fechar a janela de terminal SSH atual e se reconectar à sua instância em outra janela. Sua nova sessão SSH terá as permissões de grupo `docker` apropriadas.

8. Verifique se o `ec2-user` pode executar comandos do Docker sem `sudo`.

```
docker info
```

> **Note**
>
> Em alguns casos, pode ser necessário reinicializar sua instância para fornecer permissões para o `ec2-user` acessar o daemon do Docker. Tente reinicializar sua instância se você vir o seguinte erro:
>
> ```
> Cannot connect to the Docker daemon. Is the docker daemon running on this
>  host?
> ```

Criar uma imagem do Docker

As definições de tarefa do Amazon ECS usam imagens de contêiner para iniciar contêineres nas instâncias de contêiner dos clusters. Nesta seção, crie uma imagem do Docker de uma aplicação Web simples e teste-a no sistema ou na instância do Amazon EC2 local. Em seguida, envie a imagem a um registro de contêiner do Amazon ECR para poder usá-la em uma definição de tarefa do Amazon ECS.

Para criar uma imagem do Docker de um aplicativo web simples

1. Crie um arquivo chamado `Dockerfile`. Um Dockerfile é um manifesto que descreve a imagem básica a ser usada para a sua imagem do Docker e o que você deseja instalar e executar nela. Para obter mais informações sobre a Dockerfiles, visite Referência de Dockerfiles.

```
touch Dockerfile
```

2. Edite o `Dockerfile` que você acabou de criar e adicione o conteúdo a seguir.

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest

# Update installed packages and install Apache
RUN yum update -y && \
 yum install -y httpd

# Write hello world message
```

```
RUN echo 'Hello World!' > /var/www/html/index.html

# Configure Apache
RUN echo 'mkdir -p /var/run/httpd' >> /root/run_apache.sh && \
 echo 'mkdir -p /var/lock/httpd' >> /root/run_apache.sh && \
 echo '/usr/sbin/httpd -D FOREGROUND' >> /root/run_apache.sh && \
 chmod 755 /root/run_apache.sh

EXPOSE 80

CMD /root/run_apache.sh
```

Esse Dockerfile usa a imagem pública do Amazon Linux 2023 hospedada no Amazon ECR Public. As instruções RUN atualizam os caches de pacotes, instalam alguns pacotes de software para o servidor Web e, em seguida, gravam o conteúdo de “Hello World!” na raiz do documento dos servidores Web. A instrução EXPOSE significa que a porta 80 do contêiner é a responsável por receber, e a instrução CMD inicia o servidor Web.

3. Crie a imagem do Docker do seu Dockerfile.

> **Note**
>
> Algumas versões do Docker podem exigir o caminho completo para o seu Dockerfile no seguinte comando, em vez de o caminho relativo mostrado abaixo.
> Se você executar o comando em um sistema baseado em ARM, como o Apple Silicon, use a opção --platform “--platform linux/amd64”.

```
docker build -t hello-world .
```

4. Liste a sua imagem do contêiner.

```
docker images --filter reference=hello-world
```

Saída:

```
REPOSITORY          TAG                 IMAGE ID            CREATED
 SIZE
hello-world         latest              e9ffedc8c286        4 minutes ago
 194MB
```

5. Execute a imagem recém-criada. A opção `-p 80:80` mapeia a porta 80 exposta no contêiner para a porta 80 no sistema de host.

```
docker run -t -i -p 80:80 hello-world
```

> **Note**
>
> A saída do servidor Web Apache é exibida na janela do terminal. É possível ignorar a mensagem "`Could not reliably determine the fully qualified domain name`".

6. Abra um navegador e aponte para o servidor que está executando o Docker e hospedando seu contêiner.

   - Se você estiver usando uma instância do EC2, esse é o valor Public DNS para o servidor, que é o mesmo endereço usado para se conectar à instância com o SSH. Certifique-se de que o security group para sua instância permita o tráfego de entrada na porta 80.
   - Se você estiver executando o Docker localmente, aponte seu navegador para http://localhost/.

   Você deve ver uma página da Web com seu “Hello, World!” instrução.

7. Interrompa o contêiner do Docker digitando Ctrl+c.

Enviar a imagem para o Amazon Elastic Container Registry

O Amazon ECR é um serviço de registro de imagem gerenciado pela AWS. É possível usar a CLI do Docker para enviar, extrair e gerenciar imagens dos repositórios do Amazon ECR. Para conhecer detalhes dos produtos do Amazon ECR, estudos de caso de clientes em destaque e perguntas frequentes, consulte as Páginas de detalhes dos produtos do Amazon Elastic Container Registry.

Para marcar a imagem e enviá-la para o Amazon ECR

1. Crie um repositório do Amazon ECR para armazenar sua imagem `hello-world`. Observe `repositoryUri` na saída.

   Substitua `region` com seu Região da AWS, por exemplo, `us-east-1`.

```
aws ecr create-repository --repository-name hello-repository --region region
```

Saída:

```
{
    "repository": {
        "registryId": "aws_account_id",
        "repositoryName": "hello-repository",
        "repositoryArn": "arn:aws:ecr:region:aws_account_id:repository/hello-
repository",
        "createdAt": 1505337806.0,
        "repositoryUri": "aws_account_id.dkr.ecr.region.amazonaws.com/hello-
repository"
    }
}
```

2. Marque a imagem `hello-world` com o valor `repositoryUri` da etapa anterior.

```
docker tag hello-world aws_account_id.dkr.ecr.region.amazonaws.com/hello-repository
```

3. Execute o comando aws ecr get-login-password. Especifique o URI de registro no qual deseja fazer a autenticação. Para obter mais informações, consulte Registry Authentication (Autenticação de registro) no Guia do usuário do Amazon Elastic Container Registry.

```
aws ecr get-login-password --region region | docker login --username AWS --
password-stdin aws_account_id.dkr.ecr.region.amazonaws.com
```

Saída:

```
Login Succeeded
```

> **Important**
>
> Se você receber um erro, instale ou atualize para a versão mais recente da AWS CLI. Para obter mais informações, consulte Instalar ou atualizar para a versão mais recente da AWS CLI no Guia do usuário do AWS Command Line Interface.

4. Envie a imagem para o Amazon ECR com o valor `repositoryUri` da etapa anterior.

```
docker push aws_account_id.dkr.ecr.region.amazonaws.com/hello-repository
```

Limpeza

Para continuar a criar uma definição de tarefa do Amazon ECS e iniciar uma tarefa com a imagem do contêiner, pule para a Próximas etapas. Quando você terminar de testar a imagem do Amazon ECR, poderá excluir o repositório para não ser cobrado por armazenamento de imagens.

```
aws ecr delete-repository --repository-name hello-repository --region region --force
```

Próximas etapas

Suas definições de tarefa exigem um perfil de execução de tarefa. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.

Depois de criar e enviar a imagem do contêiner para o Amazon ECR, você pode usar essa imagem em uma definição de tarefa. Para obter mais informações, consulte um dos seguintes:

- the section called “Saiba como criar uma tarefa do Linux para o tipo de inicialização do Fargate”
- the section called “Saiba como criar uma tarefa do Windows para o tipo de inicialização do Fargate”
- Criar uma tarefa do Linux no Amazon ECS para o tipo de inicialização do Fargate com a AWS CLI

Saiba como criar uma tarefa do Linux no Amazon ECS para o tipo de inicialização do Fargate

O Amazon Elastic Container Service (Amazon ECS) é um serviço de gerenciamento de contêineres altamente escalável e rápido que facilita a execução, a interrupção e o gerenciamento de contêineres. É possível hospedar os contêineres em uma infraestrutura sem servidor gerenciada pelo Amazon ECS ao iniciar serviços ou tarefas no AWS Fargate. Para obter mais informações sobre o Fargate, consulte AWS Fargate para o Amazon ECS.

Começar a usar Amazon ECS no AWS Fargate usando o tipo de inicialização do Fargate para as tarefas nas regiões em que o Amazon ECS é compatível com o AWS Fargate.

Conclua as tarefas a seguir para começar a usar o Amazon ECS no AWS Fargate.

Pré-requisitos

Antes de começar, conclua as etapas em Configuração para usar o Amazon ECS e verifique se o usuário do IAM tem as permissões especificadas no exemplo de política do IAM de `AdministratorAccess`.

O console tenta criar automaticamente o perfil do IAM de execução da tarefa, o que é obrigatório para tarefas do Fargate. Para garantir que o console possa criar esse perfil do IAM, uma das opções a seguir deve ser verdadeira:

- O usuário tem acesso de administrador. Para obter mais informações, consulte Configuração para usar o Amazon ECS.
- O usuário tem as permissões do IAM para criar uma função do serviço. Para obter mais informações, consulte Criar uma função para delegar permissões a um serviço da AWS.
- Um usuário com acesso de administrador criou manualmente a função de execução da tarefa, de maneira que ela esteja disponível na conta a ser usada. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.

Important

O grupo de segurança selecionado ao criar um serviço com sua definição de tarefa deve ter a porta 80 aberta para tráfego de entrada. Adicione a regras de entrada a seguir ao seu grupo de segurança. Para obter informações sobre como criar um grupo de segurança, consulte Criar um grupo de segurança para sua instância do Amazon EC2 no Guia do usuário do Amazon EC2.

- Type (Tipo): HTTP
- Protocolo: TCP
- Intervalo de porta: 80
- Origem: Qualquer lugar (`0.0.0.0/0`)

Etapa 1: criar um cluster

Crie um cluster que use a VPC padrão.

Antes de começar, atribua a permissão apropriada do IAM. Para obter mais informações, consulte the section called “Exemplos de clusters do Amazon ECS”.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, selecione a Região a ser usada.
3. No painel de navegação, escolha Clusters.
4. Na página Clusters, escolha Create Cluster (Criar cluster).
5. Em Cluster configuration (Configuração do cluster), em Cluster name (Nome do cluster), insira um nome exclusivo.

   O nome pode conter até 255 letras (minúsculas e maiúsculas), números e hifens.
6. (Opcional) Para ativar o Container Insights, expanda Monitoring (Monitoramento) e, em seguida, ative Use Container Insights (Usar o Conteiner Insights).
7. (Opcional) Para ajudar a identificar seu cluster, expanda Tags (Etiquetas) e configure suas etiquetas.

   [Adicionar uma tag] Selecione Add tag (Adicionar tag) e faça o seguinte:

   - Em Key (Chave), insira o nome da chave.
   - Em Value (Valor), insira o valor da chave.

   [Remover uma tag] Escolha Remover à direita da chave e do valor da tag.
8. Escolha Criar.

Etapa 2: criar uma definição de tarefa

Uma definição de tarefa é como um guia para seu aplicativo. Sempre que você iniciar uma tarefa no Amazon ECS, especifique uma definição de tarefa. Dessa maneira, o serviço sabe qual imagem do Docker usar para os contêineres, quantos contêineres usar na tarefa e a alocação de recursos para cada contêiner.

1. No painel de navegação, selecione Definições de tarefas.
2. Escolha Create new Task Definition (Criar nova definição de tarefa), Create new revision with JSON (Criar nova revisão com JSON).
3. Copie e cole o exemplo de definição de tarefa a seguir na caixa e escolha Save (Salvar).

```
{
    "family": "sample-fargate",
    "networkMode": "awsvpc",
    "containerDefinitions": [
        {
            "name": "fargate-app",
            "image": "public.ecr.aws/docker/library/httpd:latest",
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "entryPoint": [
                "sh",
  "-c"
            ],
            "command": [
                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
            ]
        }
    ],
    "requiresCompatibilities": [
        "FARGATE"
    ],
    "cpu": "256",
    "memory": "512"
}
```

4. Escolha Criar.

Etapa 3: criar um serviço

Crie um serviço usando a definição de tarefa.

1. No painel de navegação, escolha Clusters e selecione o cluster que você criou em Etapa 1: criar um cluster.
2. Na guia Services (Serviços), escolha Create (Criar).
3. Em Deployment configuration (Configuração de implantação), especifique como a aplicação será implantada.
    a. Em Task definition (Definição de tarefa), escolha a definição de tarefa que você criou em Etapa 2: criar uma definição de tarefa.
    b. Em Service name (Nome do serviço), insira um nome para o serviço.
    c. Em Desired tasks (Tarefas desejadas), insira 1.
4. Em Redes, é possível criar um novo grupo de segurança ou escolher um grupo de segurança existente para a sua tarefa. Certifique-se de que o grupo de segurança que você usa tenha a regra de entrada listada em Pré-requisitos.
5. Escolha Criar.

Etapa 4: visualizar o seu serviço

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Clusters.
3. Escolha o cluster em que você executou o serviço.
4. Na guia Serviços, em Nome do serviço, escolha o serviço que você criou em Etapa 3: criar um serviço.
5. Escolha a guia Tarefas e, em seguida, escolha a tarefa em seu serviço.
6. Na página da tarefa, na seção Configuração, em IP púbico, escolha endereço aberto.

Etapa 5: limpar

Ao terminar de usar um cluster do Amazon ECS, é necessário limpar os recursos associados a ele para evitar cobranças por recursos que você não está usando.

Alguns recursos do Amazon ECS, como tarefas, serviços, clusters e instâncias de contêiner, são eliminados por meio do console do Amazon ECS. Outros recursos, como instâncias do Amazon EC2, balanceadores de carga do Elastic Load Balancing e grupos do Auto Scaling, devem ser excluídos

manualmente no console do Amazon EC2 ou com a exclusão da pilha do AWS CloudFormation que os criou.

1. No painel de navegação, escolha Clusters.
2. Na página Clusters, selecione o cluster que você criou para este tutorial.
3. Escolha a guia Serviços.
4. Selecione o serviço e, em seguida, escolha Excluir.
5. No prompt de confirmação, insira delete e escolha Delete (Excluir). Ou então, use a opção `Force delete` para fazer com que o Amazon ECS reduza verticalmente a escala do serviço para você antes de excluí-lo.

   Espere até que o serviço seja excluído.
6. Escolha Delete Cluster. No prompt de confirmação, insira delete ***`nome-cluster`*** (excluir nome-cluster) e escolha Delete (Excluir). A exclusão do cluster limpa os recursos associados que foram criados com o cluster, incluindo grupos do Auto Scaling, VPCs ou balanceadores de carga.

Saiba como criar uma tarefa do Windows no Amazon ECS para o tipo de inicialização do Fargate

Começar a usar Amazon ECS no AWS Fargate usando o tipo de inicialização do Fargate para as tarefas nas regiões em que o Amazon ECS é compatível com o AWS Fargate.

Conclua as tarefas a seguir para começar a usar o Amazon ECS no AWS Fargate.

Pré-requisitos

Antes de começar, conclua as etapas em Configuração para usar o Amazon ECS e verifique se o usuário do IAM tem as permissões especificadas no exemplo de política do IAM de `AdministratorAccess`.

O console tenta criar automaticamente o perfil do IAM de execução da tarefa, o que é obrigatório para tarefas do Fargate. Para garantir que o console possa criar esse perfil do IAM, uma das opções a seguir deve ser verdadeira:

- O usuário tem acesso de administrador. Para obter mais informações, consulte Configuração para usar o Amazon ECS.

- O usuário tem as permissões do IAM para criar uma função do serviço. Para obter mais informações, consulte Criar uma função para delegar permissões a um serviço da AWS.
- Um usuário com acesso de administrador criou manualmente a função de execução da tarefa, de maneira que ela esteja disponível na conta a ser usada. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.

Important

O grupo de segurança selecionado ao criar um serviço com sua definição de tarefa deve ter a porta 80 aberta para tráfego de entrada. Adicione a regras de entrada a seguir ao seu grupo de segurança. Para obter informações sobre como criar um grupo de segurança, consulte Criar um grupo de segurança para sua instância do Amazon EC2 no Guia do usuário do Amazon EC2.

- Type (Tipo): HTTP
- Protocolo: TCP
- Intervalo de porta: 80
- Origem: Qualquer lugar (`0.0.0.0/0`)

Etapa 1: criar um cluster

É possível criar um novo cluster denominado windows que use a VPC padrão.

Para criar um cluster com o AWS Management Console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, selecione a Região a ser usada.
3. No painel de navegação, escolha Clusters.
4. Na página Clusters, escolha Create Cluster (Criar cluster).
5. Em Cluster configuration (Configuração do cluster), em Cluster name (Nome do cluster), insira windows.
6. (Opcional) Para ativar o Container Insights, expanda Monitoring (Monitoramento) e, em seguida, ative Use Container Insights (Usar o Conteiner Insights).

7. (Opcional) Para ajudar a identificar seu cluster, expanda Tags (Etiquetas) e configure suas etiquetas.

   [Adicionar uma tag] Selecione Add tag (Adicionar tag) e faça o seguinte:

   - Em Key (Chave), insira o nome da chave.
   - Em Value (Valor), insira o valor da chave.

   [Remover uma tag] Escolha Remover à direita da chave e do valor da tag.
8. Escolha Criar.

Etapa 2: registrar uma definição de tarefa do Windows

Para executar contêineres do Windows no cluster do Amazon ECS, você deve registrar uma definição de tarefa. O exemplo de definição de tarefa a seguir exibe uma página da web simples na porta 8080 de uma instância de contêiner com a imagem de contêiner do `mcr.microsoft.com/windows/servercore/iis`.

Para registrar o exemplo de definição de tarefa no AWS Management Console

1. No painel de navegação, escolha Task definitions (Definições de tarefa).
2. Escolha Create new task definition (Criar nova definição de tarefa), Create new task definition with JSON (Criar nova definição de tarefa com JSON).
3. Copie e cole o exemplo de definição de tarefa a seguir na caixa e escolha Save (Salvar).

```
{
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html
 -Type file -Value '<html> <head> <title>Amazon ECS Sample App</title>
 <style>body {margin-top: 40px; background-color: #333;} </style> </head><body>
 <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p>'; C:\\ServiceMonitor.exe w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
```

```
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "networkMode": "awsvpc",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["FARGATE"]
}
```

4. Verifique suas informações e escolha Create (Criar).

Etapa 3: criar um serviço com sua definição de tarefa

Depois que tiver registrado a definição de tarefa, será possível colocar tarefas no cluster com ela. O procedimento a seguir cria um serviço com a definição de tarefa e coloca uma tarefa no cluster.

Para criar um serviço a partir de sua definição de tarefa com o console

1. No painel de navegação, escolha Clusters e selecione o cluster que você criou em Etapa 1: criar um cluster.
2. Na guia Services (Serviços), escolha Create (Criar).
3. Em Deployment configuration (Configuração de implantação), especifique como a aplicação será implantada.

    a. Em Task definition (Definição de tarefa), escolha a definição de tarefa que você criou em Etapa 2: registrar uma definição de tarefa do Windows.

    b. Em Service name (Nome do serviço), insira um nome para o serviço.

c. Em Desired tasks (Tarefas desejadas), insira 1.

4. Em Redes, é possível criar um grupo de segurança ou escolher um grupo existente. Certifique-se de que o grupo de segurança que você usa tenha a regra de entrada listada em Pré-requisitos.
5. Escolha Criar.

Etapa 4: visualizar o serviço

Depois que o serviço tiver ativado uma tarefa no cluster, será possível visualizar o serviço e abrir a página de teste do IIS em um navegador para verificar se o contêiner está em execução.

> **Note**
>
> Pode demorar até 15 minutos para a instância de contêiner baixar e extrair as camadas base de contêiner do Windows.

Para visualizar o serviço

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Clusters.
3. Escolha o cluster em que você executou o serviço.
4. Na guia Serviços, em Nome do serviço, escolha o serviço que você criou em Etapa 3: criar um serviço com sua definição de tarefa.
5. Escolha a guia Tarefas e, em seguida, escolha a tarefa em seu serviço.
6. Na página da tarefa, na seção Configuração, em IP púbico, escolha endereço aberto.

Etapa 5: limpar

Ao terminar de usar um cluster do Amazon ECS, é necessário limpar os recursos associados a ele para evitar cobranças por recursos que você não está usando.

Alguns recursos do Amazon ECS, como tarefas, serviços, clusters e instâncias de contêiner, são eliminados por meio do console do Amazon ECS. Outros recursos, como instâncias do Amazon EC2, balanceadores de carga do Elastic Load Balancing e grupos do Auto Scaling, devem ser excluídos

manualmente no console do Amazon EC2 ou com a exclusão da pilha do AWS CloudFormation que os criou.

1. No painel de navegação, escolha Clusters.
2. Na página Clusters, selecione o cluster que você criou para este tutorial.
3. Escolha a guia Serviços.
4. Selecione o serviço e, em seguida, escolha Excluir.
5. No prompt de confirmação, insira delete e escolha Delete (Excluir).

   Espere até que o serviço seja excluído.
6. Escolha Delete Cluster. No prompt de confirmação, insira delete ***nome-cluster*** (excluir nome-cluster) e escolha Delete (Excluir). A exclusão do cluster limpa os recursos associados que foram criados com o cluster, incluindo grupos do Auto Scaling, VPCs ou balanceadores de carga.

Saiba como criar uma tarefa do Windows no Amazon ECS para o tipo de inicialização do EC2

Conceitos básicos do Amazon ECS com o tipo de execução do EC2 registrando uma definição de tarefa, criando um cluster e um serviço no console.

Execute as etapas a seguir para conhecer os conceitos básicos do Amazon ECS usando o tipo de inicialização do EC2.

Pré-requisitos

Antes de começar, conclua as etapas em Configuração para usar o Amazon ECS e verifique se o usuário do IAM tem as permissões especificadas no exemplo de política do IAM de `AdministratorAccess`.

O console tenta criar automaticamente o perfil do IAM de execução da tarefa, o que é obrigatório para tarefas do Fargate. Para garantir que o console possa criar esse perfil do IAM, uma das opções a seguir deve ser verdadeira:

- O usuário tem acesso de administrador. Para obter mais informações, consulte Configuração para usar o Amazon ECS.

- O usuário tem as permissões do IAM para criar uma função do serviço. Para obter mais informações, consulte Criar uma função para delegar permissões a um serviço da AWS.
- Um usuário com acesso de administrador criou manualmente a função de execução da tarefa, de maneira que ela esteja disponível na conta a ser usada. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.

> **Important**
>
> O grupo de segurança selecionado ao criar um serviço com sua definição de tarefa deve ter a porta 80 aberta para tráfego de entrada. Adicione a regras de entrada a seguir ao seu grupo de segurança. Para obter informações sobre como criar um grupo de segurança, consulte Criar um grupo de segurança para sua instância do Amazon EC2 no Guia do usuário do Amazon EC2.
>
> - Type (Tipo): HTTP
> - Protocolo: TCP
> - Intervalo de porta: 80
> - Origem: Qualquer lugar (`0.0.0.0/0`)

Etapa 1: criar um cluster

Um cluster do Amazon ECS é um agrupamento lógico de tarefas, serviços e instâncias de contêiner.

As etapas a seguir orientam você durante a criação de um cluster com uma instância do Amazon EC2 registrada, o que nos permitirá executar uma tarefa nele. Se um determinado campo não for mencionado, deixe os valores padrão do console.

Para criar um novo cluster (console do Amazon ECS)

Antes de começar, atribua a permissão apropriada do IAM. Para obter mais informações, consulte the section called “Exemplos de clusters do Amazon ECS”.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, selecione a Região a ser usada.
3. No painel de navegação, escolha Clusters.
4. Na página Clusters, escolha Create Cluster (Criar cluster).

5. Em Cluster configuration (Configuração do cluster), em Cluster name (Nome do cluster), insira um nome exclusivo.

   O nome pode conter até 255 letras (minúsculas e maiúsculas), números e hifens.

6. (Opcional) Para alterar a VPC e as sub-redes onde suas tarefas e serviços são iniciados, em Networking (Redes), execute qualquer uma das operações a seguir:

   - Para remover uma sub-rede, em Subnets (Sub-redes), escolha X para cada sub-rede que você deseja remover.
   - Para mudar para uma VPC diferente da VPC default (padrão), em VPC, escolha uma VPC existente e, depois, Subnets (Sub-redes), e selecione cada sub-rede.

7. Para adicionar instâncias do Amazon EC2 ao seu cluster, expanda Infraestrutura e selecione Instâncias do Amazon EC2. Em seguida, configure o grupo do Auto Scaling que atua como o provedor de capacidade:

   a. Para usar um grupo do Auto Scaling existente, em Auto Scaling group (ASG) (Grupo do Auto Scaling (ASG)), selecione o grupo.
   b. Para criar um grupo do Auto Scaling, a partir de Auto Scaling group (ASG) (Grupo do Auto Scaling (ASG)), selecione Create new group (Criar novo grupo) e, em seguida, forneça os seguintes detalhes sobre o grupo:

      - Em Operating system/Architecture (Sistema operacional/arquitetura), escolha a AMI otimizada para Amazon ECS para as instâncias do grupo do Auto Scaling.
      - Em EC2 instance type (Tipo de instância do EC2), escolha o tipo de instância para suas workloads. Para obter mais informações sobre os diferentes tipos de instância, consulte Amazon EC2 Instances (Instâncias do Amazon EC2).

        A escalabilidade gerenciada funciona melhor se o grupo do Auto Scaling usa os mesmos tipos de instância ou semelhantes.
      - Em SSH key pair (Par de chaves de SSH), escolha o par que prova sua identidade quando você se conecta à instância.
      - Em Capacity (Capacidade), insira o número mínimo e o número máximo de instâncias a serem iniciadas no grupo do Auto Scaling. As instâncias do Amazon EC2 geram custos enquanto existem nos seus recursos da AWS. Para obter mais informações, consulte Preços do Amazon EC2.

8. (Opcional) Para ativar o Container Insights, expanda Monitoring (Monitoramento) e, em seguida, ative Use Container Insights (Usar o Conteiner Insights).

9. (Opcional) Para gerenciar as tags de cluster, expanda Tags e, em seguida, execute uma das seguintes operações:

   [Adicionar uma tag] Selecione Add tag (Adicionar tag) e faça o seguinte:

   - Em Key (Chave), insira o nome da chave.
   - Em Value (Valor), insira o valor da chave.

   [Remover uma tag] Escolha Remover à direita da chave e do valor da tag.

10. Escolha Criar.

Etapa 2: registrar uma definição de tarefa

Para registrar o exemplo de definição de tarefa no AWS Management Console

1. No painel de navegação, selecione Definições de tarefas.
2. Escolha Create new task definition (Criar nova definição de tarefa), Create new task definition with JSON (Criar nova definição de tarefa com JSON).
3. Copie e cole o exemplo de definição de tarefa a seguir na caixa e escolha Salvar.

```
{
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html
 -Type file -Value '<html> <head> <title>Amazon ECS Sample App</title>
 <style>body {margin-top: 40px; background-color: #333;} </style> </head><body>
 <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p>'; C:\\ServiceMonitor.exe w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
```

```
            "portMappings": [
                {
                    "hostPort": 443,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["EC2"]
}
```

4. Verifique suas informações e escolha Create (Criar).

Etapa 3: criar um serviço

Um serviço do Amazon ECS ajuda você a executar e manter simultaneamente um número especificado de instâncias de uma definição de tarefa em um cluster do Amazon ECS. Se qualquer uma das tarefas apresentar falha ou for interrompida por qualquer motivo, o programador de serviço do Amazon ECS iniciará outra instância da sua definição de tarefa para substitui-la a fim de manter o número desejado de tarefas no serviço. Para obter mais informações sobre serviços, consulte Serviços do Amazon ECS.

Para criar um serviço

1. No painel de navegação, escolha Clusters.
2. Selecione o cluster que você criou em Etapa 1: criar um cluster.
3. Na guia Services (Serviços), escolha Create (Criar).
4. Na seção Ambiente (Ambiente), faça o seguinte:
    a. Em Compute options (Opções de computação), escolha Launch type (Tipo de inicialização).
    b. Para Launch type (Tipo de execução), selecione EC2.
5. Na seção Deployment configuration (Configuração da implantação), faça o seguinte:

a. Em Family (Família), escolha a definição de tarefa que você criou em Etapa 2: registrar uma definição de tarefa.

b. Em Service name (Nome do serviço), insira um nome para o serviço.

c. Em Desired tasks (Tarefas desejadas), insira 1.

6. Revise as opções e escolha Criar.
7. Escolha View service (Visualizar serviço) para revisar o serviço.

Etapa 4: visualizar o serviço

O serviço é um aplicativo baseado na web para que você possa visualizar seus contêineres com um navegador da web.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Clusters.
3. Escolha o cluster em que você executou o serviço.
4. Na guia Serviços, em Nome do serviço, escolha o serviço que você criou em Etapa 3: criar um serviço.
5. Escolha a guia Tarefas e, em seguida, escolha a tarefa em seu serviço.
6. Na página da tarefa, na seção Configuração, em IP púbico, escolha endereço aberto. A captura de tela abaixo é a saída esperada.

Etapa 5: limpar

Ao terminar de usar um cluster do Amazon ECS, é necessário limpar os recursos associados a ele para evitar cobranças por recursos que você não está usando.

Alguns recursos do Amazon ECS, como tarefas, serviços, clusters e instâncias de contêiner, são eliminados por meio do console do Amazon ECS. Outros recursos, como instâncias do Amazon EC2, balanceadores de carga do Elastic Load Balancing e grupos do Auto Scaling, devem ser excluídos manualmente no console do Amazon EC2 ou com a exclusão da pilha do AWS CloudFormation que os criou.

1. No painel de navegação, escolha Clusters.
2. Na página Clusters, selecione o cluster que você criou para este tutorial.
3. Escolha a guia Serviços.
4. Selecione o serviço e, em seguida, escolha Excluir.
5. No prompt de confirmação, insira delete e escolha Delete (Excluir).

   Espere até que o serviço seja excluído.
6. Escolha Delete Cluster. No prompt de confirmação, insira delete ***nome-cluster*** (excluir nome-cluster) e escolha Delete (Excluir). A exclusão do cluster limpa os recursos associados que foram criados com o cluster, incluindo grupos do Auto Scaling, VPCs ou balanceadores de carga.

Criar recursos do Amazon ECS usando o AWS CDK

O AWS Cloud Development Kit (AWS CDK) é um framework de infraestrutura como código (IAC) que pode ser usado para definir a infraestrutura da nuvem AWS usando uma linguagem de programação completa. Para definir sua própria infraestrutura de nuvem, primeiro é necessário criar uma aplicação (em uma das linguagens aceitas pelo CDK) que contenha uma ou mais pilhas. Em seguida, sintetize-a em um modelo do AWS CloudFormation e implante seus recursos na Conta da AWS. Siga as etapas descritas neste tópico para implantar um servidor Web em contêiner com o Amazon Elastic Container Service (Amazon ECS) e o AWS CDK no Fargate.

A AWS Construct Library, incluída no CDK, fornece módulos que podem ser usados para modelar os recursos fornecidos por cada serviço da Serviços da AWS. Para serviços populares, a biblioteca fornece construções selecionadas com padrões inteligentes e práticas recomendadas. Um desses

módulos, especificamente o `aws-ecs-patterns`, fornece abstrações de alto nível que permitem definir o serviço em contêineres e todos os recursos de suporte necessários em algumas linhas de código.

Este tópico usa a construção `ApplicationLoadBalancedFargateService`. A construção implanta um serviço do Amazon ECS no Fargate por trás de um Application Load Balancer. O módulo `aws-ecs-patterns` também inclui construções que usam um network load balancer e são executadas no Amazon EC2.

Antes de iniciar esta tarefa, configure o ambiente de desenvolvimento do AWS CDK e instale o AWS CDK executando o comando a seguir. Para obter instruções sobre como configurar o ambiente de desenvolvimento do AWS CDK, consulte Conceitos básicos do AWS CDK: pré-requisitos.

```
npm install -g aws-cdk
```

> **Note**
>
> Essas instruções pressupõem que você está usando o AWS CDK v2.

Tópicos

- Etapa 1: configurar o projeto do AWS CDK
- Etapa 2: usar o AWS CDK para definir um servidor Web em contêiner no Fargate
- Etapa 3: testar o serviço da Web
- Etapa 4: limpar
- Próximas etapas

Etapa 1: configurar o projeto do AWS CDK

Crie um diretório para a nova aplicação do AWS CDK e inicialize o projeto.

TypeScript

```
mkdir hello-ecs
cd hello-ecs
cdk init --language typescript
```

JavaScript

```
mkdir hello-ecs
cd hello-ecs
cdk init --language javascript
```

Python

```
mkdir hello-ecs
cd hello-ecs
cdk init --language python
```

Iniciado o projeto, ative o ambiente virtual do projeto e instale as dependências de linha de base do AWS CDK.

```
source .venv/bin/activate
python -m pip install -r requirements.txt
```

Java

```
mkdir hello-ecs
cd hello-ecs
cdk init --language java
```

Importe esse projeto do Maven para seu Java IDE. Por exemplo, no Eclipse, use File (Arquivo) > Import (Importar) > Maven > Existing Maven Projects (Projetos do Maven existentes).

C#

```
mkdir hello-ecs
cd hello-ecs
cdk init --language csharp
```

Go

```
mkdir hello-ecs
cd hello-ecs
cdk init --language go
```

> **Note**
>
> O modelo de aplicação do AWS CDK usa o nome do diretório do projeto para gerar nomes para arquivos e classes de origem. Neste exemplo, o diretório se chama `hello-ecs`. Se você escolher outro nome de diretório de projeto, sua aplicação não corresponderá a estas instruções.

O AWS CDK v2 inclui construções estáveis para todos os Serviços da AWS em um único pacote chamado `aws-cdk-lib`. Esse pacote é instalado como uma dependência quando o projeto é inicializado. Quando determinadas linguagens de programação são usadas, o pacote é instalado quando o projeto é compilado pela primeira vez. Este tópico descreve como usar construção de padrões do Amazon ECS que fornece abstrações de alto nível para trabalhar com o Amazon ECS. Esse módulo depende de construções do Amazon ECS e de outras construções para provisionar os recursos necessários para sua aplicação do Amazon ECS.

Os nomes usados para importar essas bibliotecas para a aplicação do CDK podem diferir ligeiramente dependendo da linguagem de programação utilizada. Para referência, estes são os nomes usados em cada linguagem de programação aceita pelo CDK.

TypeScript

```
aws-cdk-lib/aws-ecs
aws-cdk-lib/aws-ecs-patterns
```

JavaScript

```
aws-cdk-lib/aws-ecs
aws-cdk-lib/aws-ecs-patterns
```

Python

```
aws_cdk.aws_ecs
aws_cdk.aws_ecs_patterns
```

Java

```
software.amazon.awscdk.services.ecs
```

```
software.amazon.awscdk.services.ecs.patterns
```

C#

```
Amazon.CDK.AWS.ECS
Amazon.CDK.AWS.ECS.Patterns
```

Go

```
github.com/aws/aws-cdk-go/awscdk/v2/awsecs
github.com/aws/aws-cdk-go/awscdk/v2/awsecspatterns
```

Etapa 2: usar o AWS CDK para definir um servidor Web em contêiner no Fargate

Use a imagem do contêiner `amazon-ecs-sample`. Essa imagem contém uma aplicação da Web em PHP que é executada no Ngingx.

No projeto AWS CDK criado por você, edite o arquivo que contém a definição de pilha de modo que ele se pareça com um dos exemplos a seguir.

> **Note**
>
> Uma pilha é uma unidade de implantação. Todos os recursos devem estar em uma pilha, e todos os recursos que estão em uma pilha são implantados ao mesmo tempo. Se a implantação de um recurso falhar, todos os demais recursos já implantados serão revertidos. Uma aplicação do AWS CDK pode conter várias pilhas, e os recursos de uma pilha podem se referir aos recursos de outra pilha.

TypeScript

Atualize `lib/hello-ecs-stack.ts` de modo que ele seja semelhante ao seguinte.

```
import * as cdk from 'aws-cdk-lib';
import { Construct } from 'constructs';
import * as ecs from 'aws-cdk-lib/aws-ecs';
```

```
import * as ecsp from 'aws-cdk-lib/aws-ecs-patterns';

export class HelloEcsStack extends cdk.Stack {
  constructor(scope: Construct, id: string, props?: cdk.StackProps) {
    super(scope, id, props);

    new ecsp.ApplicationLoadBalancedFargateService(this, 'MyWebServer', {
      taskImageOptions: {
        image: ecs.ContainerImage.fromRegistry('public.ecr.aws/ecs-sample-image/
amazon-ecs-sample:latest'),
      },
      publicLoadBalancer: true
    });
  }
}
```

JavaScript

Atualize `lib/hello-ecs-stack.js` de modo que ele seja semelhante ao seguinte.

```
const cdk = require('aws-cdk-lib');
const { Construct } = require('constructs');
const ecs = require('aws-cdk-lib/aws-ecs');
const ecsp = require('aws-cdk-lib/aws-ecs-patterns');

class HelloEcsStack extends cdk.Stack {
  constructor(scope = Construct, id = string, props = cdk.StackProps) {
    super(scope, id, props);

    new ecsp.ApplicationLoadBalancedFargateService(this, 'MyWebServer', {
      taskImageOptions: {
        image: ecs.ContainerImage.fromRegistry('amazon/amazon-ecs-sample'),
      },
      publicLoadBalancer: true
    });
  }
}

module.exports = { HelloEcsStack }
```

Python

Atualize `hello-ecs/hello_ecs_stack.py` de modo que ele seja semelhante ao seguinte.

```
import aws_cdk as cdk
from constructs import Construct

import aws_cdk.aws_ecs as ecs
import aws_cdk.aws_ecs_patterns as ecsp

class HelloEcsStack(cdk.Stack):

    def __init__(self, scope: Construct, construct_id: str, **kwargs) -> None:
        super().__init__(scope, construct_id, **kwargs)

        ecsp.ApplicationLoadBalancedFargateService(self, "MyWebServer",
            task_image_options=ecsp.ApplicationLoadBalancedTaskImageOptions(
                image=ecs.ContainerImage.from_registry("amazon/amazon-ecs-sample")),
            public_load_balancer=True
        )
```

Java

Atualize `src/main/java/com.myorg/HelloEcsStack.java` de modo que ele seja semelhante ao seguinte.

```
package com.myorg;

import software.constructs.Construct;
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;

import software.amazon.awscdk.services.ecs.ContainerImage;
import
 software.amazon.awscdk.services.ecs.patterns.ApplicationLoadBalancedFargateService;
import
 software.amazon.awscdk.services.ecs.patterns.ApplicationLoadBalancedTaskImageOptions;

public class HelloEcsStack extends Stack {
    public HelloEcsStack(final Construct scope, final String id) {
        this(scope, id, null);
    }

    public HelloEcsStack(final Construct scope, final String id, final StackProps
 props) {
        super(scope, id, props);
```

```
        ApplicationLoadBalancedFargateService.Builder.create(this, "MyWebServer")
         .taskImageOptions(ApplicationLoadBalancedTaskImageOptions.builder()
           .image(ContainerImage.fromRegistry("amazon/amazon-ecs-sample"))
           .build())
         .publicLoadBalancer(true)
         .build();
    }
}
```

C#

Atualize `src/HelloEcs/HelloEcsStack.cs` de modo que ele seja semelhante ao seguinte.

```
using Amazon.CDK;
using Constructs;
using Amazon.CDK.AWS.ECS;
using Amazon.CDK.AWS.ECS.Patterns;
namespace HelloEcs
{
    public class HelloEcsStack : Stack
    {
        internal HelloEcsStack(Construct scope, string id, IStackProps props =
 null) : base(scope, id, props)
        {
            new ApplicationLoadBalancedFargateService(this, "MyWebServer",
                new ApplicationLoadBalancedFargateServiceProps
                {
                    TaskImageOptions = new ApplicationLoadBalancedTaskImageOptions
                    {
                        Image = ContainerImage.FromRegistry("amazon/amazon-ecs-
sample")
                    },
                    PublicLoadBalancer = true
                });
        }
    }
}
```

Go

Atualize `hello-ecs.go` de modo que ele seja semelhante ao seguinte.

```
package main
```

```
import (
 "github.com/aws/aws-cdk-go/awscdk/v2"
 // "github.com/aws/aws-cdk-go/awscdk/v2/awssqs"
 "github.com/aws/aws-cdk-go/awscdk/v2/awsecs"
 "github.com/aws/aws-cdk-go/awscdk/v2/awsecspatterns"
 "github.com/aws/constructs-go/constructs/v10"
 "github.com/aws/jsii-runtime-go"
)

type HelloEcsStackProps struct {
 awscdk.StackProps
}

func NewHelloEcsStack(scope constructs.Construct, id string, props
 *HelloEcsStackProps) awscdk.Stack {
 var sprops awscdk.StackProps
 if props != nil {
  sprops = props.StackProps
 }
 stack := awscdk.NewStack(scope, &id, &sprops)

 // The code that defines your stack goes here

 // example resource
 // queue := awssqs.NewQueue(stack, jsii.String("HelloEcsQueue"),
 &awssqs.QueueProps{
 //  VisibilityTimeout: awscdk.Duration_Seconds(jsii.Number(300)),
 // })
 res := awsecspatterns.NewApplicationLoadBalancedFargateService(stack,
 jsii.String("MyWebServer"),
  &awsecspatterns.ApplicationLoadBalancedFargateServiceProps{
   TaskImageOptions: &awsecspatterns.ApplicationLoadBalancedTaskImageOptions{
    Image: awsecs.ContainerImage_FromRegistry(jsii.String("amazon/amazon-ecs-
sample"), &awsecs.RepositoryImageProps{}),
   },
  },
 )
 awscdk.NewCfnOutput(stack, jsii.String("LoadBalancerDNS"),
 &awscdk.CfnOutputProps{Value: res.LoadBalancer().LoadBalancerDnsName()})

 return stack
}
```

```
func main() {
 defer jsii.Close()

 app := awscdk.NewApp(nil)

 NewHelloEcsStack(app, "HelloEcsStack", &HelloEcsStackProps{
  awscdk.StackProps{
   Env: env(),
  },
 })

 app.Synth(nil)
}

// env determines the AWS environment (account+region) in which our stack is to
// be deployed. For more information see: https://docs.aws.amazon.com/cdk/latest/
guide/environments.html
func env() *awscdk.Environment {
 // If unspecified, this stack will be "environment-agnostic".
 // Account/Region-dependent features and context lookups will not work, but a
 // single synthesized template can be deployed anywhere.
 //---------------------------------------------------------------------------
 return nil

 // Uncomment if you know exactly what account and region you want to deploy
 // the stack to. This is the recommendation for production stacks.
 //---------------------------------------------------------------------------
 // return &awscdk.Environment{
 //  Account: jsii.String("123456789012"),
 //  Region:  jsii.String("us-east-1"),
 // }

 // Uncomment to specialize this stack for the AWS Account and Region that are
 // implied by the current CLI configuration. This is recommended for dev
 // stacks.
 //---------------------------------------------------------------------------
 // return &awscdk.Environment{
 //  Account: jsii.String(os.Getenv("CDK_DEFAULT_ACCOUNT")),
 //  Region:  jsii.String(os.Getenv("CDK_DEFAULT_REGION")),
 // }
}
```

O snippet curto anterior inclui o seguinte:

- O nome lógico do serviço: `MyWebServer`.
- A imagem do contêiner que foi obtida na Galeria Pública do Amazon ECR: `amazon/amazon-ecs-sample`.
- Outras informações relevantes, como o fato de que o balanceador de carga tem um endereço público e é acessível via Internet.

O AWS CDK criará todos os recursos necessários para implantar o servidor Web, incluindo os seguintes recursos. Esses recursos foram omitidos neste exemplo.

- Cluster do Amazon ECS
- Instâncias do Amazon VPC e do Amazon EC2
- Auto Scaling group (Grupo do Auto Scaling)
- Application Load Balancer
- Perfis e políticas do IAM

Alguns recursos provisionados automaticamente serão compartilhados por todos os serviços do Amazon ECS definidos na pilha.

Salve o arquivo de origem e, em seguida, execute o comando `cdk synth` no diretório principal da aplicação. O AWS CDK executa a aplicação e sintetiza um modelo do AWS CloudFormation baseado nela. Em seguida, exibe o modelo. O modelo é um arquivo YAML com aproximadamente 600 linhas. O início do arquivo é mostrado aqui. Seu modelo pode ser diferente deste exemplo.

```
Resources:
  MyWebServerLB3B5FD3AB:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      LoadBalancerAttributes:
        - Key: deletion_protection.enabled
          Value: "false"
      Scheme: internet-facing
      SecurityGroups:
        - Fn::GetAtt:
            - MyWebServerLBSecurityGroup01B285AA
            - GroupId
      Subnets:
        - Ref: EcsDefaultClusterMnL3mNNYNVpcPublicSubnet1Subnet3C273B99
```

```
        - Ref: EcsDefaultClusterMnL3mNNYNVpcPublicSubnet2Subnet95FF715A
      Type: application
    DependsOn:
      - EcsDefaultClusterMnL3mNNYNVpcPublicSubnet1DefaultRouteFF4E2178
      - EcsDefaultClusterMnL3mNNYNVpcPublicSubnet2DefaultRouteB1375520
    Metadata:
      aws:cdk:path: HelloEcsStack/MyWebServer/LB/Resource
  MyWebServerLBSecurityGroup01B285AA:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Automatically created Security Group for ELB
 HelloEcsStackMyWebServerLB06757F57
      SecurityGroupIngress:
        - CidrIp: 0.0.0.0/0
          Description: Allow from anyone on port 80
          FromPort: 80
          IpProtocol: tcp
          ToPort: 80
      VpcId:
        Ref: EcsDefaultClusterMnL3mNNYNVpc7788A521
    Metadata:
      aws:cdk:path: HelloEcsStack/MyWebServer/LB/SecurityGroup/Resource
# and so on for another few hundred lines
```

Para implantar o serviço na Conta da AWS, execute `cdk deploy` no diretório principal da aplicação. Você será avisado para aprovar as políticas do IAM que geradas pelo AWS CDK.

A implantação demora vários minutos, período durante o qual o AWS CDK cria diversos recursos. As últimas linhas da saída da implantação incluem o nome de host público do balanceador de carga e o URL do novo servidor Web. Eles são os seguintes:

```
Outputs:
HelloEcsStack.MyWebServerLoadBalancerDNSXXXXXXX = Hello-MyWeb-ZZZZZZZZZZZZZ-
ZZZZZZZZZZ.us-west-2.elb.amazonaws.com
HelloEcsStack.MyWebServerServiceURLYYYYYYYY = http://Hello-MyWeb-ZZZZZZZZZZZZZ-
ZZZZZZZZZZ.us-west-2.elb.amazonaws.com
```

Etapa 3: testar o serviço da Web

Copie o URL da saída da implantação e cole no navegador da Web. A mensagem de boas-vindas a seguir do servidor Web é exibida.

Etapa 4: limpar

Após terminar com o servidor Web, encerre o serviço usando o CDK ao executar o comando `cdk destroy` no diretório principal da aplicação. Isso evita incorrer em cobranças não intencionais no futuro.

Próximas etapas

Para saber mais sobre como desenvolver a infraestrutura da AWS usando o AWS CDK, consulte o Guia do desenvolvedor do AWS CDK.

Para obter informações sobre como criar aplicações do AWS CDK em sua linguagem preferida, consulte:

TypeScript

- Trabalhar com o AWS CDK no TypeScript

JavaScript

- Trabalhar com o AWS CDK no JavaScript

Python

- Trabalhar com o AWS CDK no Python

Java

- Trabalhar com o AWS CDK no Java

C#

Trabalhar com o AWS CDK em C#

Go

Working with the AWS CDK in Go

Para obter mais informações sobre os módulos da AWS Construct Library usados neste tópico, consulte as visões gerais da Referência da API do AWS CDK a seguir.

- aws-ecs
- aws-ecs-patterns

Criar de recursos do Amazon ECS usando a interface de linha de comando do AWS Copilot

A interface da linha de comando (CLI) do AWS Copilot simplifica a criação, o lançamento e o funcionamento de aplicações em contêineres prontas para produção no Amazon ECS em um ambiente de desenvolvimento local. A CLI do AWS Copilot se alinha aos fluxos de trabalho do desenvolvedor que oferecem suporte a práticas recomendadas de aplicações modernas: do uso da infraestrutura como código à criação de um pipeline de CI/CD provisionado em nome de um usuário. Use a CLI do AWS Copilot como parte do ciclo diário de desenvolvimento e testes como uma alternativa ao AWS Management Console.

O AWS Copilot no momento oferece suporte a sistemas Linux, macOS e Windows. Para obter mais informações sobre a versão mais recente da CLI do AWS Copilot, consulte Releases (Versões).

> **Note**
>
> O código-fonte da CLI do AWS Copilot está disponível no GitHub. Recomendamos que você envie problemas e solicitações pull para alterações que você gostaria de ter incluído. No entanto, o Amazon Web Services no momento não oferece suporte para a execução de cópias modificadas do código do AWS Copilot. Informe problemas com o AWS Copilot entrando em contato conosco no Gitter ou no GitHub. onde você pode discutir problemas, fornecer feedback e relatar bugs.

Para obter informações sobre o desenvolvimento do Copilot, consulte What's the future of Copilot?.

Tópicos

- Instalar a CLI do AWS Copilot
- Implantação de uma amostra de aplicação do Amazon ECS usando a CLI do AWS Copilot

Documentação adicional para a CLI do AWS Copilot está disponível no site do AWS Copilot

Instalar a CLI do AWS Copilot

É possível instalar a CLI do AWS Copilot ao usar o Homebrew ou ao efetuar o download do binário manualmente com as etapas apresentadas a seguir.

Uso do Homebrew

O comando a seguir é usado para instalar a CLI do AWS Copilot no sistema macOS ou Linux usando Homebrew. Antes da instalação, você deve ter o Homebrew instalado. Para obter mais informações, consulte Homebrew.

```
brew install aws/tap/copilot-cli
```

Download do binário

Como alternativa ao Homebrew, é possível instalar manualmente a CLI do AWS Copilot em seu sistema macOS, Windows ou Linux. Use o comando apresentado a seguir para o seu sistema operacional com a finalidade de efetuar download do binário. Os exemplos do macOS e do Linux também incluem comandos que aplicam permissões de execução ao binário e listam o menu de ajuda para verificar se a instalação funciona.

macOS

Para macOS:

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-darwin \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Para sistemas ARM macOS:

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-darwin-arm64 \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Linux

Para sistemas Linux x86 (64 bits):

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-linux \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Para sistemas Linux ARM:

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-linux-arm64 \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Windows

Usando Powershell, execute o comando a seguir:

```
New-Item -Path 'C:\copilot' -ItemType directory; `
  Invoke-WebRequest -OutFile 'C:\copilot\copilot.exe' https://github.com/aws/
copilot-cli/releases/latest/download/copilot-windows.exe
```

(Opcional) Verifique a CLI do AWS Copilot manualmente instalada usando assinaturas PGP

Os executáveis da CLI do AWS são assinados de forma criptografada usando assinaturas PGP. As assinaturas PGP podem ser usadas para verificar a validade do executável da CLI do AWS Copilot. Use as etapas a seguir para verificar as assinaturas por meio da ferramenta GnuPG.

1. Baixe e instale GnuPG. Para obter mais informações, consulte o Site do GnuPG.

macOS

Recomendamos o uso do Homebrew. Instale o Homebrew seguindo as instruções disponíveis no site da ferramenta. Para obter mais informações, consulte Homebrew. Depois que o Homebrew estiver instalado, use o seguinte comando no terminal do macOS.

```
brew install gnupg
```

Linux

Instale o gpg usando o gerenciador de pacotes no seu tipo de Linux.

Windows

Baixe o instalador simples do Windows no site do GnuPG e instale como administrador. Depois de instalar o GnuPG, feche e reabra o PowerShell do administrador.

Para obter mais informações, consulte GnuPG Download.

2. Verifique se o caminho do GnuPG foi adicionado ao caminho do ambiente.

macOS

```
echo $PATH
```

Se você não vir o caminho GnuPG na saída, execute o seguinte comando para adicioná-lo ao caminho.

```
PATH=$PATH:<path to GnuPG executable files>
```

Linux

```
echo $PATH
```

Se você não vir o caminho GnuPG na saída, execute o seguinte comando para adicioná-lo ao caminho.

```
export PATH=$PATH:<path to GnuPG executable files>
```

Windows

```
Write-Output $Env:PATH
```

Se você não vir o caminho GnuPG na saída, execute o seguinte comando para adicioná-lo ao caminho.

```
$Env:PATH += ";<path to GnuPG executable files>"
```

3. Crie um arquivo de texto simples local.

macOS

No terminal, insira:

```
touch <public_key_filename.txt>
```

Abra o arquivo com o TextEdit.

Linux

Crie um arquivo de texto em um editor de texto, como o gedit. Salve como `public_key_filename.txt`

Windows

Crie um arquivo de texto em um editor de texto, como o Notepad. Salve como `public_key_filename.txt`

4. Adicione o seguinte conteúdo da chave pública de PGP do Amazon ECS e salve o arquivo.

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2

mQINBFq1SasBEADliGcT1NVJ1ydfN8DqebYYe9ne3dt6jqKFmKowLmm6LLGJe7HU
jGtqhCWRDkN+qPpHqdArRgDZAtn2pXY5fEipHgar4CP8QgRnRMO2fl74lmavr4Vg
7K/KH8VHlq2uRw32/B94XLEgRbGTMdWFdKuxoPCttBQaMj3LGn6Pe+6xVWRkChQu
BoQAhjBQ+bEm0kNy0LjNgjNlnL3UMAG56t8E3LANIgGgEnpNsB1UwfWluPoGZoTx
N+6pHBJrKIL/1v/ETU4FXpYw2zvhWNahxeNRnoYj3uycHkeliCrw4kj0+skizBgO
2K7oVX8Oc3j5+ZilhL/qDLXmUCb2az5cMM1mOoF8EKX5HaNuq1KfwJxqXE6NNIcO
lFTrT7QwD5fMNld3FanLgv/ZnIrsSaqJOL6zRSq8O4LN1OWBVbndExk2Kr+5kFxn
5lBPgfPgRj5hQ+KTHMa9Y8Z7yUc64BJiN6F9Nl7FJuSsfqbdkvRLsQRbcBG9qxX3
```

```
rJAEhieJzVMEUNl+EgeCkxj5xuSkNU7zw2c3hQZqEcrADLV+hvFJktOz9Gm6xzbq
lTnWWCz4xrIWtuEBA2qE+MlDheVd78a3gIsEaSTfQq0osYXaQbvlnSWOoc1y/5Zb
zizHTJIhLtUyls9WisP2s0emeHZicVMfW61EgPrJAiupgc7kyZvFt4YwfwARAQAB
tCRBbWF6b24gRUNTIDxlY3Mtc2VjdXJpdHlAYW1hem9uLmNvbT6JAhwEEAECAAYF
AlrjL0YACgkQHivRXs0TaQrg1g/+JppwPqHnlVPmv7lessB8I5UqZeD6p6uVpHd7
Bs3pcPp8BV7BdRbs3sPLt5bV1+rkqOlw+0gZ4Q/ue/YbWtOAt4qY0OcEo0HgcnaX
lsB827QIfZIVtGWMhuh94xzm/SJkvngml6KB3YJNnWP61A9qJ37/VbVVLzvcmazA
McWB4HUMNrhd0JgBCo0gIpqCbpJEvUc02Bjn23eEJsS9kC7OUAHyQkVnx4d9UzXF
4OoISF6hmQKIBoLnRrAlj5Qvs3GhvHQ0ThYq0Grk/KMJJX2CSqt7tWJ8gk1n3H3Y
SReRXJRnv7DsDDBwFgT6r5Q2HW1TBUvaoZy5hF6maD09nHcNnvBjqADzeT8Tr/Qu
bBCLzkNSYqqkpgtwv7seoD2P4n1giRvDAOEfMZpVkUr+C252IaH1HZFEz+TvBVQM
Y8OWWxmIJW+J6evjo3N1eO19UHv71jvoF8zljbI4bsL2c+QTJmOv7nRqzDQgCWyp
Id/v2dUVVTk1j9omuLBBwNJzQCB+72LcIzJhYmaP1HC4LcKQG+/f41exuItenatK
lEJQhYtyVXcBlh6Yn/wzNg2NWOwb3vqY/F7m6u9ixAwgtIMgPCDE4aJ86zrrXYFz
N2HqkTSQh77Z8KPKmyGopsmN/reMuilPdINb249nA0dzoN+nj+tTFOYCIaLaFyjs
Z0r1QAOJAjkEEwECACMFAlq1SasCGwMHCwkIBwMCAQYVCAIJCgsEFgIDAQIeAQIX
gAAKCRC86dmkLVF4T9iFEACEnkm1dNXsWUx34R3c0vamHrPxvfkyI1FlEUen8D1h
uX9xy6jCEROHWEp0rjGK4QDPgM93sWJ+s1UAKg214QRVzft0y9/DdR+twApA0fzy
uavIthGd6+03jAAo6udYDE+cZC3P7XBbDiYEWk4XAF9I1JjB8hTZUgvXBL046JhG
eM17+crgUyQeetkiOQemLbsbXQ40Bd9V7zf7XJraFd8VrwNUwNb+9KFtgAsc9rk+
YIT/PEf+YOPysgcxI4sTWghtyCulVnuGoskgDv4v73PALU0ieUrvvQVqWMRvhVx1
0X90J7cC1KOyhlEQQ1aFTgmQjmXexVTwIBm8LvysFK6YXM41KjOrlz3+6xBIm/qe
bFyLUnf4WoiuOplAaJhK9pRY+XEnGNxdtN4D26Kd0F+PLkm3Tr3Hy3b1Ok34FlGr
KVHUq1TZD7cvMnnNKEELTUcKX+1mV3an16nmAg/my1JSUt6BNK2rJpY1s/kkSGSE
XQ4zuF2IGCpvBFhYAlt5Un5zwqkwwQR3/n2kwAoDzonJcehDw/C/cGos5D0aIU7I
K2X2aTD3+pA7Mx3IMe2hqmYqRt9X42yF1PIEVRneBRJ3HDezAgJrNh0GQWRQkhIx
gz6/cTR+ekr5TptVszS9few2GpI5bCgBKBisZIssT89aw7mAKWut0Gcm4qM9/yK6
1bkCDQRatUmrARAAxNPvVwreJ2yAiFcUpdRlVhsuOgnxvs1QgsIw3H7+Pacr9Hpe
8uftYZqdC82KeSKhpHq7c8gMTMucIINtH25x9BCc73E33EjCL9Lqov1TL7+QkgHe
T+JIhZwdD8Mx2K+LVVVu/aWkNrfMuNwyDUciSI4D5QHa8T+F8fgN4OTpwYjirzel
5yoICMr9hVcbzDNv/ozKCxjx+XKgnFc3wrnDfJfntfDAT7ecwbUTL+viQKJ646s+
psiqXRYtVvYInEhLVrJ0aV6zHFoigE/Bils6/g7ru1Q6CEHqEw++APs5CcE8VzJu
WAGSVHZgun5Y9N4quR/M9Vm+IPMhTxrAg7rOvyRN9cAXfeSMf77I+XTifigNna8x
t/MOdjXr1fjF4pThEi5u6WsuRdFwjY2azEv3vevodTi4HoJReH6dFRa6y8c+UDgl
2iHiOKIpQqLbHEfQmHcDd2fix+AaJKMnPGNku9qCFEMbgSRJpXz6BfwnY1QuKE+I
R6jA0frUNt2jhiGG/F8RceXzohaaC/Cx7LUCUFWc0n7z32C9/Dtj7I1PMOacdZzz
bjJzRKO/ZDv+UN/c9dwAkllzAyPMwGBkUaY68EBstnIliW34aWm6IiHhxioVPKSp
VJfyiXPO0EXqujtHLAeChfjcns3I12YshT1dv2PafG53fp33ZdzeUgsBo+EAEQEA
AYkCHwQYAQIACQUCWrVJqwIbDAAKCRC86dmkLVF4T+ZdD/9x/8APzgNJF3o3STrF
jvnV1ycyhWYGAeBJiu7wjsNWwzMFOv15tLjB7AqeVxZn+WKDD/mIOQ45OZvnYZuy
X7DR0JszaH9wrYTxZLVruAu+t6UL0y/XQ4L1GZ9QR6+r+7t1Mvbfy7BlHbvX/gYt
Rwe/uwdibI0CagEzyX+2D3kTOlHO5XThbXaNf8AN8zha91Jt2Q2UR2X5T6JcwtMz
FBvZnl3LSmZyE0EQehS2iUurU4uWOpGppuqVnbi0jbCvCHKgDGrqZ0smKNAQng54
F365W3g8AfY48s8XQwzmcliowYX9bT8PZiEi0J4QmQh0aXkpqZyFefuWeOL2R94S
```

```
XKzr+gRh3BAULoqF+qK+IUMxTip9KTPNvYDpiC66yBiT6gFDji5Ca9pGpJXrC3xe
TXiKQ8DBWDhBPVPrruLIaenTtZEOsPc4I85yt5U9RoPTStcOr34s3w5yEaJagt6S
Gc5r9ysjkfH6+6rbi1ujxMgROSqtqr+RyB+V9A5/OgtNZc8llK6u4UoOCde8jUUW
vqWKvjJB/Kz3u4zaeNu2ZyyHaOqOuH+TETcW+jsY9IhbEzqN5yQYGi4pVmDkY5vu
lXbJnbqPKpRXgM9BecV9AMbPgbDq/5LnHJJXg+G8YQOgp4lR/hC1TEFdIp5wM8AK
CWsENyt2o1rjgMXiZOMF8A5oBLkCDQRatUuSARAAr77kj7j2QR2SZeOSlFBvV7oS
mFeSNnz9xZssqrsm6bTwSHM6YLDwc7Sdf2esDdyzONETwqrVCg+FxgL8hmo9hS4c
rR6tmrP0mOmptr+xLLsKcaP7ogIXsyZnrEAEsvW8PnfayoiPCdc3cMCR/lTnHFGA
7EuR/XLBmi7Qg9tByVYQ5Yj5wB9V4B2yeCt3XtzPqeLKvaxl7PNelaHGJQY/xo+m
V0bndxf9IY+4oFJ4blD32WqvyxESo7vW6WBh7oqv3Zbm0yQrr8a6mDBpqLkvWwNI
3kpJR974tg5o5LfDu1BeeyHWPSGm4U/G4JB+JIG1ADy+RmoWEt4BqTCZ/knnoGvw
D5sTCxbKdmuOmhGyTssoG+3OOcGYHV7pWYPhazKHMPm201xKCjH1RfzRULzGKjD+
yMLT1I3AXFmLmZJXikAOlvE3/wgMqCXscbycbLjLD/bXIuFWo3rzoezeXjgi/DJx
jKBAyBTYO5nMcth1O9oaFd9d0HbsOUDkIMnsgGBE766Piro6MHo0T0rXl07Tp4pI
rwuSOsc6XzCzdImj0Wc6axS/HeUKRXWdXJwno5awTwXKRJMXGfhCvSvbcbc2Wx+L
IKvmB7EB4K3fmjFFE67yolmiw2qRcUBfygtH3eL5XZU28MiCpue8Y8GKJoBAUyvf
KeM1rO8Jm3iRAc5a/D0AEQEAAYkEPgQYAQIACQUCWrVLkgIbAgIpCRC86dmkLVF4
T8FdIAQZAQIABgUCWrVLkgAKCRDePL1hra+LjtHYD/9MucxdFe6bXO1dQR4tKhhQ
P0LRqy6zlBY9ILCLowNdGZdqorogUiUymgn3VhEhVtxTOoHcN7qOuM01PNsRnOeS
EYjf8Xrb1clzkD6xULwmOclTb9bBxnBc/4PFvHAbZW3QzusaZniNgkuxt6BTfloS
Of4inq71kjmGK+TlzQ6mUMQUg228NUQC+a84EPqYyAeY1sgvgB7hJBhYL0QAxhcW
6m20Rd8iEc6HyzJ3yCOCsKip/nRWAbf0OvfHfRBp0+m0ZwnJM8cPRFjOqqzFpKH9
HpDmTrC4wKP1+TL52LyEqNh4yZitXmZNV7giSRIkk0eDSko+bFy6VbMzKUMkUJK3
D3eHFAMkujmbfJmSMTJOPGn5SB1HyjCZNx6bhIIbQyEUB9gKCmUFaqXKwKpF6rj0
iQXAJxLR/shZ5Rk96VxzOphUl7T90m/PnUEEPwq8KsBhnMRgxa0RFidDP+n9fgtv
HLmrOqX9zBCVXh0mdWYLrWvmzQFWzG7AoE55fkf8nAEPsalrCdtaNUBHRXA0OQxG
AHMOdJQQvBsmqMvuAdjkDWpFu5y0My5ddU+hiUzUyQLjL5Hhd5LOUDdewlZgIw1j
xrEAUzDKetnemM8GkHxDgg8koev5frmShJuce7vSjKpCNg3EIJSgqMOPFjJuLWtZ
vjHeDNbJy6uNL65ckJy6WhGjEADS2WAW1D6Tfekkc21SsIXk/LqEpLMR/0g5OUif
wcEN1rS9IJXBwIy8MelN9qr5KcKQLmfdfBNEyyceBhyVl0MDyHOKC+7PofMtkGBq
13QieRHv5GJ8LB3fclqHV8pwTTo3Bc8z2g0TjmUYAN/ixETdReDoKavWJYSE9yoM
aaJu279ioVTrwpECse0XkiRyKToTjwOb73CGkBZZpJyqux/rmCV/fp4ALdSW8zbz
FJVORaivhoWwzjpfQKhwcU9lABXi2UvVm14v0AfeI7oiJPSU1zM4fEny4oiIBXlR
zhFNih1UjIu82X16mTm3BwbIga/s1fnQRGzyhqUIMii+mWra23EwjChaxpvjjcUH
5ilLc5Zq781aCYRygYQw+hu5nFkOH1R+Z50Ubxjd/aqUfnGIAX7kPMD3Lof4KldD
Q8ppQriUvxVo+4nPV6rpTy/PyqCLWDjkguHpJsEFsMkwajrAz0QNSAU5CJ0G2Zu4
yxvYlumHCEl7nbFrm0vIiA75Sa8KnywTDsyZsu3XcOcf3g+g1xWTpjJqy2bYXlqz
9uDOWtArWHOis6bq8l9RE6xr1RBVXS6uqgQIZFBGyq66b0dIq4D2JdsUvgEMaHbc
e7tBfeB1CMBdA64e9Rq7bFR7Tvt8gasCZYlNr3lydh+dFHIEkH53HzQe6l88HEic
+0jVnLkCDQRa55wJARAAyLya2Lx6gyoWoJN1a6740q3o8e9d4KggQOfGMTCflmeq
ivuzgN+3DZHN+9ty2KxXMtn0mhHBerZdbNJyjMNT1gAgrhPNB4HtXBXum2wS57WK
DNmade914L7FWTPAWBG2Wn448OEHTqsClICXXWy9IICgclAEyIq0Yq5mAdTEgRJS
Z8t4GpwtDL9gNQyFXaWQmDmkAsCygQMvhAlmu9xOIzQG5CxSnZFk7zcuL60k14Z3
Cmt49k4T/7ZU8goWi8tt+rU78/IL3J/fF9+1civ1OwuUidgfPCSvOUW1JojsdCQA
```

```
L+RZJcoXq7lfOFj/eNjeOSstCTDPfTCL+kThE6E5neDtbQHBYkEX1BRiTedsV4+M
ucgiTrdQFWKf89G72xdv8ut9AYYQ2BbEYU+JAYhUH8rYYui2dHKJIgjNvJscuUWb
+QEqJIRleJRhrO+/CHgMs4fZAkWF1VFhKBkcKmEjLn1f7EJJUUW84ZhKXjO/AUPX
1CHsNjziRceuJCJYox1cwsoq6jTE50GiNzcIxTn9xUc0UMKFeggNAFys1K+TDTm3
Bzo8H5ucjCUEmUm9lhkGwqTZgOlRX5eqPX+JBoSaObqhgqCa5IPinKRa6MgoFPHK
6sYKqroYwBGgZm6Js5chpNchvJMs/3WXNOEVg0J3z3vP0DMhxqWm+r+n9zlW8qsA
EQEAAYkEPgQYAQgACQUCWuecCQIbAgIpCRC86dmkLVF4T8FdIAQZAQgABgUCWuec
CQAKCRBQ3szEcQ5hr+ykD/4tOLRHFHXuKUcxgGaubUcVtsFrwBKma1cYjqaPms8u
6Sk0wfGRI32G/GhOrp0Ts/MOkbObq6VLTh8N5Yc/53MEl8zQFw9Y5AmRoW4PZXER
ujs5s7p4oR7xHMihMjCCBn1bvrR+34YPfgzTcgLiOEFHYT8UTxwnGmXOvNkMM7md
xD3CV5q6VAte8WKBo/220II3fcQlc9r/oWX4kXXkb0v9hoGwKbDJ1tzqTPrp/xFt
yohqnvImpnlz+Q9zXmbrWYL9/g8VCmW/NN2gju2G3Lu/TlFUWIT4v/5OPK6TdeNb
VKJO4+S8bTayqSG9CML1S57KSgCo5HUhQWeSNHI+fpe5oX6FALPT9JLDce8OZz1i
cZZ0MELP37mOOQun0AlmHm/hVzf0f311PtbzcqWaE51tJvgUR/nZFo6Ta3O5Ezhs
3VlEJNQ1Ijf/6DH87SxvAoRIARCuZd0qxBcDK0avpFzUtbJd24lRA3WJpkEiMqKv
RDVZkE4b6TW61f0o+LaVfK6E8oLpixegS4fiqC16mFrOdyRk+RJJfIUyz0WTDVmt
g0U1CO1ezokMSqkJ7724pyjr2xf/r9/sC6aOJwB/lKgZkJfC6NqL7TlxVA31dUga
LEOvEJTTE4gl+tYtfsCDvALCtqL0jduSkUo+RXcBItmXhA+tShW0pbS2Rtx/ixua
KohVD/0R4QxiSwQmICNtm9mw9ydIl1yjYXX5a9x4wMJracNY/LBybJPFnZnT4dYR
z4XjqysDwvvYZByaWoIe3QxjX84V6MlI2IdAT/xImu8gbaCI8tmyfpIrLnPKiR9D
VFYfGBXuAX7+HgPPSFtrHQONCALxxzlbNpS+zxt9r0MiLgcLyspWxSdmoYGZ6nQP
RO5Nm/ZVS+u2imPCRzNUZEMa+dlE6kHx0rS0dPiuJ4O7NtPeYDKkoQtNagspsDvh
cK7CSqAiKMq06UBTxqlTSRkm62eOCtcs3p3OeHu5GRZF1uzTET0ZxYkaPgdrQknx
ozjP5mC7X+45lcCfmcVt94TFNL5HwEUVJpmOgmzILCI8yoDTWzloo+i+fPFsXX4f
kynhE83mSEcr5VHFYrTY3mQXGmNJ3bCLuc/jq7ysGq69xiKmTlUeXFm+aojcRO5i
zyShIRJZ0GZfuzDYFDbMV9amA/YQGygLw//zP5ju5SW26dNxlf3MdFQE5JJ86rn9
MgZ4gcpazHEVUsbZsgkLizRp9imUiH8ymLqAXnfRGlU/LpNSefnvDFTtEIRcpOHc
bhayG0bk51Bd4mioOXnIsKy4j63nJXA27x5EVVHQ1sYRN8Ny4Fdr2tMAmj2O+X+J
qX2yy/UX5nSPU492e2CdZ1UhoU0SRFY3bxKHKB7SDbVeav+K5g==
=Gi5D
-----END PGP PUBLIC KEY BLOCK-----
```

Detalhes da chave pública PGP do Amazon ECS para referência:

```
Key ID: BCE9D9A42D51784F
Type: RSA
Size: 4096/4096
Expires: Never
User ID: Amazon ECS
Key fingerprint: F34C 3DDA E729 26B0 79BE AEC6 BCE9 D9A4 2D51 784F
```

5. Importe o arquivo com a chave pública de PGP do Amazon ECS usando o seguinte comando no terminal.

```
gpg --import <public_key_filename.txt>
```

6. Baixe as assinaturas da CLI do AWS Copilot.. As assinaturas são assinaturas PGP desanexadas de caracteres ASCII armazenadas em arquivos com a extensão `.asc`. O arquivo de assinaturas tem o mesmo nome do executável correspondente, com `.asc` adicionado.

   macOS

   No macOS, execute o comando a seguir.

   ```
   sudo curl -Lo copilot.asc https://github.com/aws/copilot-cli/releases/latest/download/copilot-darwin.asc
   ```

   Linux

   Para sistemas Linux x86 (64 bits), execute o comando a seguir.

   ```
   sudo curl -Lo copilot.asc https://github.com/aws/copilot-cli/releases/latest/download/copilot-linux.asc
   ```

   Para sistemas Linux ARM, execute o comando a seguir.

   ```
   sudo curl -Lo copilot.asc https://github.com/aws/copilot-cli/releases/latest/download/copilot-linux-arm64.asc
   ```

   Windows

   Usando Powershell, execute o comando a seguir.

   ```
   Invoke-WebRequest -OutFile 'C:\copilot\copilot.asc' https://github.com/aws/copilot-cli/releases/latest/download/copilot-windows.exe.asc
   ```

7. Verifique a assinatura com o comando a seguir.

   - Para sistemas macOS e Linux:

     ```
     gpg --verify copilot.asc /usr/local/bin/copilot
     ```

   - Apenas para sistemas Windows:

```
gpg --verify 'C:\copilot\copilot.asc' 'C:\copilot\copilot.exe'
```

Saída esperada:

```
gpg: Signature made Tue Apr  3 13:29:30 2018 PDT
gpg:                using RSA key DE3CBD61ADAF8B8E
gpg: Good signature from "Amazon ECS <ecs-security@amazon.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: F34C 3DDA E729 26B0 79BE  AEC6 BCE9 D9A4 2D51 784F
     Subkey fingerprint: EB3D F841 E2C9 212A 2BD4  2232 DE3C BD61 ADAF 8B8E
```

> **Important**
>
> O aviso na saída é esperado e não é um problema. Isso ocorre porque não existe uma cadeia de confiança entre a chave PGP pessoal (se você tiver uma) e a chave PGP do Amazon ECS. Para obter mais informações, consulte Web of trust.

8. Para instalações do Windows, execute o comando a seguir no Powershell para adicionar o diretório do AWS Copilot ao caminho.

```
$Env:PATH += ";<path to Copilot executable files>"
```

Implantação de uma amostra de aplicação do Amazon ECS usando a CLI do AWS Copilot

Após instalar a CLI do AWS Copilot, é possível seguir estas etapas para implantar uma aplicação de amostra, verificar a implantação e limpar os recursos.

Pré-requisitos

Antes de começar, certifique-se de que os seguintes pré-requisitos sejam atendidos:

- Instale e configure a AWS CLI. Para obter mais informações, consulte Interface da linha de comando da AWS.

- Execute `aws configure` para configurar um perfil padrão que a CLI do AWS Copilot usará para gerenciar as aplicações e serviços.
- Instale e execute o Docker. Para obter mais informações, consulte Conceitos básicos do Docker.

Implantação de uma aplicação do Amazon ECS de amostra usando um único comando

1. Implante uma aplicação Web de amostra que foi clonada de um repositório GitHub usando o comando apresentado a seguir. Para obter mais informações sobre o `init` do AWS Copilot e seus sinalizadores, consulte a documentação do AWS Copilot.

```
git clone https://github.com/aws-samples/aws-copilot-sample-service.git demo-app &&
 \
cd demo-app &&                                \
copilot init --app demo                       \
  --name api                                  \
  --type 'Load Balanced Web Service'          \
  --dockerfile './Dockerfile'                 \
  --port 80                                   \
  --tag  latest                               \
  --deploy
```

2. Após a conclusão da implantação, a CLI do AWS Copilot retornará um URL que você pode usar para verificar a implantação. Também é possível usar os comandos apresentados a seguir para verificar o status da aplicação.

   - Liste todas as suas aplicações do AWS Copilot.

   ```
   copilot app ls
   ```

   - Mostre as informações sobre os ambientes e serviços na aplicação.

   ```
   copilot app show
   ```

   - Mostre informações sobre os ambientes.

   ```
   copilot env ls
   ```

   - Mostre informações sobre o serviço, incluindo endpoints, capacidade e recursos relacionados.

```
copilot svc show
```

- Lista de todos os serviços em uma aplicação.

```
copilot svc ls
```

- Mostre os logs de um serviço implantado.

```
copilot svc logs
```

- Mostre o status do serviço.

```
copilot svc status
```

3. Quando terminar esta demonstração, execute o comando apresentado a seguir para limpar os recursos associados e evitar incorrer em cobranças por recursos não utilizados.

```
copilot app delete
```

Usar o Amazon ECS com o AWS CloudFormation

O Amazon ECS é integrado ao AWS CloudFormation, um serviço que pode ser usado para modelar e configurar recursos da AWS com modelos definidos por você. O AWS CloudFormation usa modelos que são um arquivo de texto em formato `YAML` ou `JSON`. Modelos são como esquemas para o recurso da AWS que você quer criar. Quando você cria e envia um modelo, o AWS CloudFormation cria uma pilha. Você gerencia os recursos definidos no seu modelo por meio da pilha. Para criar, atualizar ou excluir um recurso, crie, atualize ou exclua a pilha que foi criada com base nesse recurso. Quando se trata de atualizar suas pilhas, primeiro é necessário criar um conjunto de alterações. Os conjuntos de alterações mostram o que é afetado pela mudança antes de você fazê-la. Isso evita que você exclua bancos de dados acidentalmente alterando o nome do banco de dados, por exemplo. Para obter mais informações sobre modelos, pilhas e conjuntos de alterações, consulte Como o AWS CloudFormation funciona no Guia do usuário do AWS CloudFormation.

Ao usar o AWS CloudFormation, você pode passar menos tempo criando e gerenciando seus recursos e infraestrutura. Você pode criar um modelo para descrever todos os recursos da AWS que deseja, como clusters, definições de tarefas e, serviços específicos do Amazon ECS. Em seguida, o AWS CloudFormation se encarrega de provisionar e configurar esses recursos para você.

O AWS CloudFormation também permite reutilizar o modelo para configurar os recursos do Amazon ECS de forma consistente e reproduzível. Você descreve seus recursos uma vez e, em seguida, provisiona os mesmos recursos novamente em várias Contas da AWS e Regiões da AWS.

Os modelos do AWS CloudFormation podem ser usados com o AWS Management Console ou com a AWS Command Line Interface para criar recursos.

Para saber mais sobre o AWS CloudFormation, consulte os seguintes recursos:

- AWS CloudFormation
- Guia do usuário do AWS CloudFormation
- Guia do Usuário da Interface de Linha de Comando AWS CloudFormation

Tópicos

- Exemplos de modelos do AWS CloudFormation para o Amazon ECS
- Criar recursos do Amazon ECS usando o console do AWS CloudFormation
- Criar recursos do Amazon ECS usando comandos da AWS CLI para o AWS CloudFormation

Exemplos de modelos do AWS CloudFormation para o Amazon ECS

Para provisionar e configurar recursos para o Amazon ECS e serviços relacionados, certifique-se de estar familiarizado com os modelos do AWS CloudFormation. Para obter mais informações, consulte Modelos do AWS CloudFormation no Guia do usuário do AWS CloudFormation. Os modelos do AWS CloudFormation são arquivos de texto no formato JSON ou YAML que descrevem os recursos que você deseja provisionar em suas pilhas do AWS CloudFormation. Se não estiver familiarizado com os formatos JSON, YAML ou ambos, você poderá usar o AWS Infrastructure Composer Designer para começar a usar modelos do AWS CloudFormation. Para obter mais informações, consulte Criar modelos visualmente com o Infrastructure Composer no Guia do usuário do AWS CloudFormation.

É possível criar clusters, definições de tarefas e serviços do Amazon ECS usando o AWS CloudFormation. Os tópicos a seguir incluem modelos que demonstram como criar recursos com configurações diferentes. Você pode criar esses recursos com esses modelos usando o console do AWS CloudFormation ou a AWS CLI. Para obter mais informações sobre como criar recursos usando o console do AWS CloudFormation, consulte o Guia do usuário do AWS CloudFormation.

Os tópicos a seguir listam modelos de exemplo para definições de tarefas, clusters e serviços do Amazon ECS.

Tópicos

- Definições de tarefa
- Clusters
- Serviços
- Perfis do IAM para o Amazon ECS

Definições de tarefa

Uma definição de tarefa é um esquema para sua aplicação que descreve os parâmetros e um ou mais contêineres que formam a aplicação. A seguir estão exemplos de modelos do AWS CloudFormation para definições de tarefas do Amazon ECS. Para obter mais informações sobre definições de tarefas do Amazon ECS, consulte Definições de tarefa do Amazon ECS.

Definição de tarefas do Linux no Fargate

O modelo a seguir pode ser usado para criar uma tarefa do Linux no Fargate.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "ECS Task Definition with parameterized values",
  "Parameters": {
    "ContainerImage": {
      "Type": "String",
      "Default": "public.ecr.aws/docker/library/httpd:2.4",
      "Description": "The container image to use for the task"
    },
    "ContainerCpu": {
      "Type": "Number",
      "Default": 256,
      "Description": "The number of CPU units to reserve for the container",
      "AllowedValues": [256, 512, 1024, 2048, 4096]
    },
    "ContainerMemory": {
      "Type": "Number",
      "Default": 512,
      "Description": "The amount of memory (in MiB) to reserve for the container",
      "AllowedValues": [512, 1024, 2048, 3072, 4096, 5120, 6144, 7168, 8192]
    },
    "TaskFamily": {
      "Type": "String",
      "Default": "task-definition-cfn",
      "Description": "The name of the task definition family"
    },
    "ContainerName": {
      "Type": "String",
      "Default": "sample-fargate-app",
      "Description": "The name of the container"
    },
    "ContainerPort": {
      "Type": "Number",
      "Default": 80,
      "Description": "The port number on the container"
    },
    "HostPort": {
      "Type": "Number",
      "Default": 80,
      "Description": "The port number on the host"
    },
    "ExecutionRoleArn": {
```

```
      "Type": "String",
      "Default": "arn:aws:iam::aws_account_id:role/ecsTaskExecutionRole",
      "Description": "The ARN of the task execution role"
    },
    "LogGroup": {
      "Type": "String",
      "Default": "/ecs/fargate-task-definition",
      "Description": "The CloudWatch log group for container logs"
    },
    "NetworkMode": {
      "Type": "String",
      "Default": "awsvpc",
      "Description": "The Docker networking mode to use",
      "AllowedValues": ["awsvpc", "bridge", "host", "none"]
    },
    "OperatingSystemFamily": {
      "Type": "String",
      "Default": "LINUX",
      "Description": "The operating system for the task",
      "AllowedValues": ["LINUX", "WINDOWS_SERVER_2019_FULL",
 "WINDOWS_SERVER_2019_CORE", "WINDOWS_SERVER_2022_FULL", "WINDOWS_SERVER_2022_CORE"]
    }
  },
  "Resources": {
    "ECSTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "ContainerDefinitions": [
          {
            "Command": [
              "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</
title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html && -
foreground\""
            ],
            "EntryPoint": [
              "sh",
              "-c"
            ],
            "Essential": true,
            "Image": {"Ref": "ContainerImage"},
            "LogConfiguration": {
```

```
              "LogDriver": "awslogs",
              "Options": {
                "mode": "non-blocking",
                "max-buffer-size": "25m",
                "awslogs-create-group": "true",
                "awslogs-group": {"Ref": "LogGroup"},
                "awslogs-region": {"Ref": "AWS::Region"},
                "awslogs-stream-prefix": "ecs"
              }
            },
            "Name": {"Ref": "ContainerName"},
            "PortMappings": [
              {
                "ContainerPort": {"Ref": "ContainerPort"},
                "HostPort": {"Ref": "HostPort"},
                "Protocol": "tcp"
              }
            ]
          }
        ],
        "Cpu": {"Ref": "ContainerCpu"},
        "ExecutionRoleArn": {"Ref": "ExecutionRoleArn"},
        "Family": {"Ref": "TaskFamily"},
        "Memory": {"Ref": "ContainerMemory"},
        "NetworkMode": {"Ref": "NetworkMode"},
        "RequiresCompatibilities": [
          "FARGATE"
        ],
        "RuntimePlatform": {
          "OperatingSystemFamily": {"Ref": "OperatingSystemFamily"}
        }
      }
    }
  },
  "Outputs": {
    "TaskDefinitionArn": {
      "Description": "The ARN of the created task definition",
      "Value": {"Ref": "ECSTaskDefinition"}
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: 'ECS Task Definition to deploy a sample app'
Parameters:
  ContainerImage:
    Type: String
    Default: 'public.ecr.aws/docker/library/httpd:2.4'
    Description: The container image to use for the task
  ContainerCpu:
    Type: Number
    Default: 256
    Description: The number of CPU units to reserve for the container
    AllowedValues: [256, 512, 1024, 2048, 4096]
  ContainerMemory:
    Type: Number
    Default: 512
    Description: The amount of memory (in MiB) to reserve for the container
    AllowedValues: [512, 1024, 2048, 3072, 4096, 5120, 6144, 7168, 8192]
  TaskFamily:
    Type: String
    Default: 'task-definition-cfn'
    Description: The name of the task definition family
  ContainerName:
    Type: String
    Default: 'sample-fargate-app'
    Description: The name of the container
  ContainerPort:
    Type: Number
    Default: 80
    Description: The port number on the container
  HostPort:
    Type: Number
    Default: 80
    Description: The port number on the host
  ExecutionRoleArn:
    Type: String
    Default: 'arn:aws:iam::111122223333:role/ecsTaskExecutionRole'
    Description: The ARN of the task execution role
  LogGroup:
    Type: String
    Default: '/ecs/fargate-task-definition'
    Description: The CloudWatch log group for container logs
  NetworkMode:
```

```
    Type: String
    Default: 'awsvpc'
    Description: The Docker networking mode to use
    AllowedValues: ['awsvpc', 'bridge', 'host', 'none']
  OperatingSystemFamily:
    Type: String
    Default: 'LINUX'
    Description: The operating system for the task
    AllowedValues: ['LINUX', 'WINDOWS_SERVER_2019_FULL', 'WINDOWS_SERVER_2019_CORE',
 'WINDOWS_SERVER_2022_FULL', 'WINDOWS_SERVER_2022_CORE']
Resources:
  ECSTaskDefinition:
    Type: 'AWS::ECS::TaskDefinition'
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
              /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
              App</title> <style>body {margin-top: 40px; background-color:
              #333;} </style> </head><body> <div
              style=color:white;text-align:center> <h1>Amazon ECS Sample
              App</h1> <h2>Congratulations!</h2> <p>Your application is now
              running on a container in Amazon ECS.</p> </div></body></html>' >
              /usr/local/apache2/htdocs/index.html && httpd-foreground"
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: !Ref ContainerImage
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Ref LogGroup
              awslogs-region: !Ref AWS::Region
              awslogs-stream-prefix: ecs
          Name: !Ref ContainerName
          PortMappings:
            - ContainerPort: !Ref ContainerPort
              HostPort: !Ref HostPort
              Protocol: tcp
      Cpu: !Ref ContainerCpu
```

```
      ExecutionRoleArn: !Ref ExecutionRoleArn
      Family: !Ref TaskFamily
      Memory: !Ref ContainerMemory
      NetworkMode: !Ref NetworkMode
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: !Ref OperatingSystemFamily
Outputs:
  TaskDefinitionArn:
    Description: The ARN of the created task definition
    Value: !Ref ECSTaskDefinition
```

Definição de tarefa do Amazon EFS

É possível usar o modelo a seguir para criar uma tarefa que usa um sistema de arquivos do Amazon EFS criado por você. Para obter mais informações sobre como usar volumes do Amazon EBS com o Amazon ECS, consulte Uso de volumes do Amazon EFS com o Amazon ECS.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Create a task definition for a web server with parameterized
 values.",
  "Parameters": {
    "ExecutionRoleArn": {
      "Type": "String",
      "Default": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
      "Description": "The ARN of the task execution role"
    },
    "NetworkMode": {
      "Type": "String",
      "Default": "awsvpc",
      "Description": "The Docker networking mode to use",
      "AllowedValues": ["awsvpc", "bridge", "host", "none"]
    },
    "TaskFamily": {
      "Type": "String",
      "Default": "my-ecs-task",
      "Description": "The name of the task definition family"
    },
```

```
    "ContainerCpu": {
      "Type": "String",
      "Default": "256",
      "Description": "The number of CPU units to reserve for the container",
      "AllowedValues": ["256", "512", "1024", "2048", "4096"]
    },
    "ContainerMemory": {
      "Type": "String",
      "Default": "512",
      "Description": "The amount of memory (in MiB) to reserve for the container",
      "AllowedValues": ["512", "1024", "2048", "3072", "4096", "5120", "6144",
"7168", "8192"]
    },
    "ContainerName": {
      "Type": "String",
      "Default": "nginx",
      "Description": "The name of the container"
    },
    "ContainerImage": {
      "Type": "String",
      "Default": "public.ecr.aws/nginx/nginx:latest",
      "Description": "The container image to use for the task"
    },
    "ContainerPort": {
      "Type": "Number",
      "Default": 80,
      "Description": "The port number on the container"
    },
    "InitProcessEnabled": {
      "Type": "String",
      "Default": "true",
      "Description": "Whether to enable the init process inside the container",
      "AllowedValues": ["true", "false"]
    },
    "EfsVolumeName": {
      "Type": "String",
      "Default": "efs-volume",
      "Description": "The name of the EFS volume"
    },
    "EfsContainerPath": {
      "Type": "String",
      "Default": "/usr/share/nginx/html",
      "Description": "The path in the container where the EFS volume will be
mounted"
```

```
    },
    "LogGroup": {
      "Type": "String",
      "Default": "LogGroup",
      "Description": "The CloudWatch log group for container logs"
    },
    "LogStreamPrefix": {
      "Type": "String",
      "Default": "efs-task",
      "Description": "The prefix for the log stream"
    },
    "EfsFilesystemId": {
      "Type": "String",
      "Default": "fs-1234567890abcdef0",
      "Description": "The ID of the EFS filesystem"
    },
    "EfsRootDirectory": {
      "Type": "String",
      "Default": "/",
      "Description": "The root directory in the EFS filesystem"
    },
    "EfsTransitEncryption": {
      "Type": "String",
      "Default": "ENABLED",
      "Description": "Whether to enable transit encryption for EFS",
      "AllowedValues": ["ENABLED", "DISABLED"]
    }
  },
  "Resources": {
    "ECSTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "ExecutionRoleArn": {"Ref": "ExecutionRoleArn"},
        "NetworkMode": {"Ref": "NetworkMode"},
        "RequiresCompatibilities": ["FARGATE"],
        "Family": {"Ref": "TaskFamily"},
        "Cpu": {"Ref": "ContainerCpu"},
        "Memory": {"Ref": "ContainerMemory"},
        "ContainerDefinitions": [
          {
            "Name": {"Ref": "ContainerName"},
            "Image": {"Ref": "ContainerImage"},
            "Essential": true,
            "PortMappings": [
```

```
            {
              "ContainerPort": {"Ref": "ContainerPort"},
              "Protocol": "tcp"
            }
          ],
          "LinuxParameters": {
            "InitProcessEnabled": {"Ref": "InitProcessEnabled"}
          },
          "MountPoints": [
            {
              "SourceVolume": {"Ref": "EfsVolumeName"},
              "ContainerPath": {"Ref": "EfsContainerPath"}
            }
          ],
          "LogConfiguration": {
            "LogDriver": "awslogs",
            "Options": {
              "mode": "non-blocking",
              "max-buffer-size": "25m",
              "awslogs-group": {"Ref": "LogGroup"},
              "awslogs-region": {"Ref": "AWS::Region"},
              "awslogs-create-group": "true",
              "awslogs-stream-prefix": {"Ref": "LogStreamPrefix"}
            }
          }
        }
      ],
      "Volumes": [
        {
          "Name": {"Ref": "EfsVolumeName"},
          "EFSVolumeConfiguration": {
            "FilesystemId": {"Ref": "EfsFilesystemId"},
            "RootDirectory": {"Ref": "EfsRootDirectory"},
            "TransitEncryption": {"Ref": "EfsTransitEncryption"}
          }
        }
      ]
    }
  }
},
"Outputs": {
  "TaskDefinitionArn": {
    "Description": "The ARN of the created task definition",
    "Value": {"Ref": "ECSTaskDefinition"}
```

```
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: Create a task definition for a web server with parameterized values.
Parameters:
  ExecutionRoleArn:
    Type: String
    Default: arn:aws:iam::123456789012:role/ecsTaskExecutionRole
    Description: The ARN of the task execution role
  NetworkMode:
    Type: String
    Default: awsvpc
    Description: The Docker networking mode to use
    AllowedValues: [awsvpc, bridge, host, none]
  TaskFamily:
    Type: String
    Default: my-ecs-task
    Description: The name of the task definition family
  ContainerCpu:
    Type: String
    Default: "256"
    Description: The number of CPU units to reserve for the container
    AllowedValues: ["256", "512", "1024", "2048", "4096"]
  ContainerMemory:
    Type: String
    Default: "512"
    Description: The amount of memory (in MiB) to reserve for the container
    AllowedValues: ["512", "1024", "2048", "3072", "4096", "5120", "6144", "7168",
 "8192"]
  ContainerName:
    Type: String
    Default: nginx
    Description: The name of the container
  ContainerImage:
    Type: String
    Default: public.ecr.aws/nginx/nginx:latest
    Description: The container image to use for the task
  ContainerPort:
    Type: Number
```

```
    Default: 80
    Description: The port number on the container
  InitProcessEnabled:
    Type: String
    Default: "true"
    Description: Whether to enable the init process inside the container
    AllowedValues: ["true", "false"]
  EfsVolumeName:
    Type: String
    Default: efs-volume
    Description: The name of the EFS volume
  EfsContainerPath:
    Type: String
    Default: /usr/share/nginx/html
    Description: The path in the container where the EFS volume will be mounted
  LogGroup:
    Type: String
    Default: LogGroup
    Description: The CloudWatch log group for container logs
  LogStreamPrefix:
    Type: String
    Default: efs-task
    Description: The prefix for the log stream
  EfsFilesystemId:
    Type: String
    Default: fs-1234567890abcdef0
    Description: The ID of the EFS filesystem
  EfsRootDirectory:
    Type: String
    Default: /
    Description: The root directory in the EFS filesystem
  EfsTransitEncryption:
    Type: String
    Default: ENABLED
    Description: Whether to enable transit encryption for EFS
    AllowedValues: [ENABLED, DISABLED]
Resources:
  ECSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ExecutionRoleArn: !Ref ExecutionRoleArn
      NetworkMode: !Ref NetworkMode
      RequiresCompatibilities:
        - FARGATE
```

```
      Family: !Ref TaskFamily
      Cpu: !Ref ContainerCpu
      Memory: !Ref ContainerMemory
      ContainerDefinitions:
        - Name: !Ref ContainerName
          Image: !Ref ContainerImage
          Essential: true
          PortMappings:
            - ContainerPort: !Ref ContainerPort
              Protocol: tcp
          LinuxParameters:
            InitProcessEnabled: !Ref InitProcessEnabled
          MountPoints:
            - SourceVolume: !Ref EfsVolumeName
              ContainerPath: !Ref EfsContainerPath
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-group: !Ref LogGroup
              awslogs-region: !Ref AWS::Region
              awslogs-create-group: "true"
              awslogs-stream-prefix: !Ref LogStreamPrefix
      Volumes:
        - Name: !Ref EfsVolumeName
          EFSVolumeConfiguration:
            FilesystemId: !Ref EfsFilesystemId
            RootDirectory: !Ref EfsRootDirectory
            TransitEncryption: !Ref EfsTransitEncryption
Outputs:
  TaskDefinitionArn:
    Description: The ARN of the created task definition
    Value: !Ref ECSTaskDefinition
```

Clusters

Um cluster do Amazon ECS é um agrupamento lógico de tarefas ou serviços. É possível usar os modelos a seguir para criar clusters com configurações diferentes. Para obter mais informações sobre clusters do Amazon ECS, consulte Clusters do Amazon ECS.

Criar um cluster vazio com configurações padrão

O modelo a seguir pode ser usado para criar um cluster vazio com as configurações padrão.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": "MyEmptyCluster"
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  ECSCluster:
    Type: 'AWS::ECS::Cluster'
    Properties:
      ClusterName: MyEmptyCluster
```

Criar um cluster vazio com criptografia de armazenamento gerenciado e Container Insights aprimorado

É possível usar o modelo a seguir para criar um cluster com armazenamento gerenciado em nível de cluster e o Container Insights aprimorado habilitado. A criptografia em nível de cluster se aplica aos volumes de dados gerenciados pelo Amazon ECS, como os volumes do Amazon EBS. Para obter mais informações sobre a criptografia do Amazon EBS, consulte Criptografar dados armazenados em volumes do Amazon EBS para tarefas do Amazon ECS. Para obter mais informações sobre como usar o Container Insights com observabilidade aprimorada, consulte Monitorar contêineres do Amazon ECS utilizando o Container Insights com observabilidade aprimorada.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
   "Resources": {
        "Cluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": "EncryptedEnhancedCluster",
                "ClusterSettings": [
                    {
                        "Name": "containerInsights",
                        "Value": "enhanced"
                    }
                ],
                "Configuration": {
                    "ManagedStorageConfiguration": {
                        "KmsKeyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
                    }
                }
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  Cluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: EncryptedEnhancedCluster
      ClusterSettings:
        - Name: containerInsights
          Value: enhanced
      Configuration:
        ManagedStorageConfiguration:
          KmsKeyId: a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Criar um cluster com a AMI otimizada para Amazon ECS para AL2023

É possível usar o modelo a seguir para criar um cluster que usa um provedor de capacidade que executa instâncias do AL2023 no Amazon EC2.

> **Important**
>
> Para os IDs da AMI mais recente, consulte AMI otimizada pelo Amazon ECS, no Amazon Elastic Container Service Developer Guide.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "EC2 ECS cluster that starts out empty, with no EC2 instances
 yet. An ECS capacity provider automatically launches more EC2 instances as required
 on the fly when you request ECS to launch services or standalone tasks.",
    "Parameters": {
        "InstanceType": {
            "Type": "String",
            "Description": "EC2 instance type",
            "Default": "t2.medium",
            "AllowedValues": [
                "t1.micro",
                "t2.2xlarge",
                "t2.large",
                "t2.medium",
                "t2.micro",
                "t2.nano",
                "t2.small",
                "t2.xlarge",
                "t3.2xlarge",
                "t3.large",
                "t3.medium",
                "t3.micro",
                "t3.nano",
                "t3.small",
                "t3.xlarge"
            ]
        },
        "DesiredCapacity": {
```

```
            "Type": "Number",
            "Default": "0",
            "Description": "Number of EC2 instances to launch in your ECS cluster."
        },
        "MaxSize": {
            "Type": "Number",
            "Default": "100",
            "Description": "Maximum number of EC2 instances that can be launched in
 your ECS cluster."
        },
        "ECSAMI": {
            "Description": "The Amazon Machine Image ID used for the cluster",
            "Type": "AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>",
            "Default": "/aws/service/ecs/optimized-ami/amazon-linux-2023/
recommended/image_id"
        },
        "VpcId": {
            "Type": "AWS::EC2::VPC::Id",
            "Description": "VPC ID where the ECS cluster is launched",
            "Default": "vpc-1234567890abcdef0"
        },
        "SubnetIds": {
            "Type": "List<AWS::EC2::Subnet::Id>",
            "Description": "List of subnet IDs where the EC2 instances will be
 launched",
            "Default": "subnet-021345abcdef67890"
        }
    },
    "Resources": {
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterSettings": [
                    {
                        "Name": "containerInsights",
                        "Value": "enabled"
                    }
                ]
            }
        },
        "ECSAutoScalingGroup": {
            "Type": "AWS::AutoScaling::AutoScalingGroup",
            "DependsOn": [
                "ECSCluster",
```

```
                "EC2Role"
            ],
            "Properties": {
                "VPCZoneIdentifier": {
                    "Ref": "SubnetIds"
                },
                "LaunchTemplate": {
                    "LaunchTemplateId": {
                        "Ref": "ContainerInstances"
                    },
                    "Version": {
                        "Fn::GetAtt": [
                            "ContainerInstances",
                            "LatestVersionNumber"
                        ]
                    }
                },
                "MinSize": 0,
                "MaxSize": {
                    "Ref": "MaxSize"
                },
                "DesiredCapacity": {
                    "Ref": "DesiredCapacity"
                },
                "NewInstancesProtectedFromScaleIn": true
            },
            "UpdatePolicy": {
                "AutoScalingReplacingUpdate": {
                    "WillReplace": "true"
                }
            }
        },
        "ContainerInstances": {
            "Type": "AWS::EC2::LaunchTemplate",
            "Properties": {
                "LaunchTemplateName": "asg-launch-template-2",
                "LaunchTemplateData": {
                    "ImageId": {
                        "Ref": "ECSAMI"
                    },
                    "InstanceType": {
                        "Ref": "InstanceType"
                    },
                    "IamInstanceProfile": {
```

```
                        "Name": {
                            "Ref": "EC2InstanceProfile"
                        }
                    },
                    "SecurityGroupIds": [
                        {
                            "Ref": "ContainerHostSecurityGroup"
                        }
                    ],
                    "UserData": {
                        "Fn::Base64": {
                            "Fn::Sub": "#!/bin/bash -xe\n echo ECS_CLUSTER=
${ECSCluster} >> /etc/ecs/ecs.config\n yum install -y aws-cfn-bootstrap\n /opt/aws/
bin/cfn-init -v --stack ${AWS::StackId} --resource ContainerInstances --configsets
 full_install --region ${AWS::Region} &\n"
                        }
                    },
                    "MetadataOptions": {
                        "HttpEndpoint": "enabled",
                        "HttpTokens": "required"
                    }
                }
            }
        },
        "EC2InstanceProfile": {
            "Type": "AWS::IAM::InstanceProfile",
            "Properties": {
                "Path": "/",
                "Roles": [
                    {
                        "Ref": "EC2Role"
                    }
                ]
            }
        },
        "CapacityProvider": {
            "Type": "AWS::ECS::CapacityProvider",
            "Properties": {
                "AutoScalingGroupProvider": {
                    "AutoScalingGroupArn": {
                        "Ref": "ECSAutoScalingGroup"
                    },
                    "ManagedScaling": {
                        "InstanceWarmupPeriod": 60,
```

```
                    "MinimumScalingStepSize": 1,
                    "MaximumScalingStepSize": 100,
                    "Status": "ENABLED",
                    "TargetCapacity": 100
                },
                "ManagedTerminationProtection": "ENABLED"
            }
        }
    },
    "CapacityProviderAssociation": {
        "Type": "AWS::ECS::ClusterCapacityProviderAssociations",
        "Properties": {
            "CapacityProviders": [
                {
                    "Ref": "CapacityProvider"
                }
            ],
            "Cluster": {
                "Ref": "ECSCluster"
            },
            "DefaultCapacityProviderStrategy": [
                {
                    "Base": 0,
                    "CapacityProvider": {
                        "Ref": "CapacityProvider"
                    },
                    "Weight": 1
                }
            ]
        }
    },
    "ContainerHostSecurityGroup": {
        "Type": "AWS::EC2::SecurityGroup",
        "Properties": {
            "GroupDescription": "Access to the EC2 hosts that run containers",
            "VpcId": {
                "Ref": "VpcId"
            }
        }
    },
    "EC2Role": {
        "Type": "AWS::IAM::Role",
        "Properties": {
            "AssumeRolePolicyDocument": {
```

```
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {
                                "Service": [
                                    "ec2.amazonaws.com"
                                ]
                            },
                            "Action": [
                                "sts:AssumeRole"
                            ]
                        }
                    ]
                },
                "Path": "/",
                "ManagedPolicyArns": [
                    "arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role",
                    "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore"
                ]
            }
        },
        "ECSTaskExecutionRole": {
            "Type": "AWS::IAM::Role",
            "Properties": {
                "AssumeRolePolicyDocument": {
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {
                                "Service": [
                                    "ecs-tasks.amazonaws.com"
                                ]
                            },
                            "Action": [
                                "sts:AssumeRole"
                            ],
                            "Condition": {
                                "ArnLike": {
                                    "aws:SourceArn": {
                                        "Fn::Sub": "arn:${AWS::Partition}:ecs:
${AWS::Region}:${AWS::AccountId}:*"
                                    }
                                },
```

```
                                "StringEquals": {
                                    "aws:SourceAccount": {
                                        "Fn::Sub": "${AWS::AccountId}"
                                    }
                                }
                            }
                        }
                    ]
                },
                "Path": "/",
                "ManagedPolicyArns": [
                    "arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy"
                ]
            }
        }
    },
    "Outputs": {
        "ClusterName": {
            "Description": "The ECS cluster into which to launch resources",
            "Value": "ECSCluster"
        },
        "ECSTaskExecutionRole": {
            "Description": "The role used to start up a task",
            "Value": "ECSTaskExecutionRole"
        },
        "CapacityProvider": {
            "Description": "The cluster capacity provider that the service should
 use to request capacity when it wants to start up a task",
            "Value": "CapacityProvider"
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: EC2 ECS cluster that starts out empty, with no EC2 instances yet. An
 ECS capacity provider automatically launches more EC2 instances as required on the
 fly when you request ECS to launch services or standalone tasks.
Parameters:
  InstanceType:
    Type: String
```

```
    Description: EC2 instance type
    Default: t2.medium
    AllowedValues:
      - t1.micro
      - t2.2xlarge
      - t2.large
      - t2.medium
      - t2.micro
      - t2.nano
      - t2.small
      - t2.xlarge
      - t3.2xlarge
      - t3.large
      - t3.medium
      - t3.micro
      - t3.nano
      - t3.small
      - t3.xlarge
  DesiredCapacity:
    Type: Number
    Default: '0'
    Description: Number of EC2 instances to launch in your ECS cluster.
  MaxSize:
    Type: Number
    Default: '100'
    Description: Maximum number of EC2 instances that can be launched in your ECS
 cluster.
  ECSAMI:
    Description: The Amazon Machine Image ID used for the cluster
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ecs/optimized-ami/amazon-linux-2023/recommended/image_id
  VpcId:
    Type: AWS::EC2::VPC::Id
    Description: VPC ID where the ECS cluster is launched
    Default: vpc-1234567890abcdef0
  SubnetIds:
    Type: List<AWS::EC2::Subnet::Id>
    Description: List of subnet IDs where the EC2 instances will be launched
    Default: subnet-021345abcdef67890
Resources:
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterSettings:
```

```
        - Name: containerInsights
          Value: enabled
  ECSAutoScalingGroup:
    Type: AWS::AutoScaling::AutoScalingGroup
    DependsOn:
      - ECSCluster
      - EC2Role
    Properties:
      VPCZoneIdentifier: !Ref SubnetIds
      LaunchTemplate:
        LaunchTemplateId: !Ref ContainerInstances
        Version: !GetAtt ContainerInstances.LatestVersionNumber
      MinSize: 0
      MaxSize: !Ref MaxSize
      DesiredCapacity: !Ref DesiredCapacity
      NewInstancesProtectedFromScaleIn: true
    UpdatePolicy:
      AutoScalingReplacingUpdate:
        WillReplace: 'true'
  ContainerInstances:
    Type: AWS::EC2::LaunchTemplate
    Properties:
      LaunchTemplateName: asg-launch-template-2
      LaunchTemplateData:
        ImageId: !Ref ECSAMI
        InstanceType: !Ref InstanceType
        IamInstanceProfile:
          Name: !Ref EC2InstanceProfile
        SecurityGroupIds:
          - !Ref ContainerHostSecurityGroup
        UserData: !Base64
          Fn::Sub: |
            #!/bin/bash -xe
            echo ECS_CLUSTER=${ECSCluster} >> /etc/ecs/ecs.config
            yum install -y aws-cfn-bootstrap
            /opt/aws/bin/cfn-init -v --stack ${AWS::StackId} --resource
ContainerInstances --configsets full_install --region ${AWS::Region} &
        MetadataOptions:
          HttpEndpoint: enabled
          HttpTokens: required
  EC2InstanceProfile:
    Type: AWS::IAM::InstanceProfile
    Properties:
      Path: /
```

```
      Roles:
        - !Ref EC2Role
  CapacityProvider:
    Type: AWS::ECS::CapacityProvider
    Properties:
      AutoScalingGroupProvider:
        AutoScalingGroupArn: !Ref ECSAutoScalingGroup
        ManagedScaling:
          InstanceWarmupPeriod: 60
          MinimumScalingStepSize: 1
          MaximumScalingStepSize: 100
          Status: ENABLED
          TargetCapacity: 100
        ManagedTerminationProtection: ENABLED
  CapacityProviderAssociation:
    Type: AWS::ECS::ClusterCapacityProviderAssociations
    Properties:
      CapacityProviders:
        - !Ref CapacityProvider
      Cluster: !Ref ECSCluster
      DefaultCapacityProviderStrategy:
        - Base: 0
          CapacityProvider: !Ref CapacityProvider
          Weight: 1
  ContainerHostSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Access to the EC2 hosts that run containers
      VpcId: !Ref VpcId
  EC2Role:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - ec2.amazonaws.com
            Action:
              - sts:AssumeRole
      Path: /
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonEC2ContainerServiceforEC2Role
        - arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
```

```
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - ecs-tasks.amazonaws.com
            Action:
              - sts:AssumeRole
            Condition:
              ArnLike:
                aws:SourceArn: !Sub arn:${AWS::Partition}:ecs:${AWS::Region}:
${AWS::AccountId}:*
              StringEquals:
                aws:SourceAccount: !Sub ${AWS::AccountId}
      Path: /
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
Outputs:
  ClusterName:
    Description: The ECS cluster into which to launch resources
    Value: ECSCluster
  ECSTaskExecutionRole:
    Description: The role used to start up a task
    Value: ECSTaskExecutionRole
  CapacityProvider:
    Description: The cluster capacity provider that the service should use to
 request capacity when it wants to start up a task
    Value: CapacityProvider
```

Serviços

É possível usar um serviço do Amazon ECS para executar e manter simultaneamente um número especificado de instâncias de uma definição de tarefa em um cluster do Amazon ECS. Se uma das suas tarefas falhar ou for interrompida, o programador de serviços do Amazon ECS executará outra instância da sua definição de tarefa para substituí-la. Isso ajuda a manter o número desejado de tarefas no serviço. Os modelos a seguir podem ser usados para implantar serviços. Para obter mais informações sobre os serviços do Amazon ECS, consulte Serviços do Amazon ECS.

Implantar uma aplicação do

O modelo a seguir também cria um serviço do Amazon ECS chamado `cfn-service` que inicia e mantém a tarefa definida pela definição de tarefa. O modelo cria um cluster chamado `CFNCluster`. O cluster contém uma definição de tarefa do Linux Fargate que configura um servidor da Web. O modelo também cria os recursos de rede e log necessários. O modelo também cria um perfil de execução de tarefas do Amazon ECS. Para obter mais informações sobre a função de execução de tarefas, consulte Função do IAM de execução de tarefas do Amazon ECS.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "A template that deploys an application that is built on an
 Apache web server Docker image by creating an Amazon ECS cluster, task definition,
 and service. The template also creates networking and logging resources, and an
 Amazon ECS task execution role.",
    "Parameters": {
        "ClusterName": {
            "Type": "String",
            "Default": "CFNCluster",
            "Description": "Name of the ECS Cluster"
        },
        "TaskFamily": {
            "Type": "String",
            "Default": "task-definition-cfn",
            "Description": "Family name for the Task Definition"
        },
        "ServiceName": {
            "Type": "String",
            "Default": "cfn-service",
            "Description": "Name of the ECS Service"
        },
        "ContainerImage": {
            "Type": "String",
            "Default": "public.ecr.aws/docker/library/httpd:2.4",
            "Description": "Container image to use for the task"
        },
        "TaskCpu": {
            "Type": "Number",
            "Default": 256,
            "AllowedValues": [
                256,
```

```
                512,
                1024,
                2048,
                4096
            ],
            "Description": "CPU units for the task"
        },
        "TaskMemory": {
            "Type": "Number",
            "Default": 512,
            "AllowedValues": [
                512,
                1024,
                2048,
                4096,
                8192,
                16384
            ],
            "Description": "Memory (in MiB) for the task"
        },
        "DesiredCount": {
            "Type": "Number",
            "Default": 1,
            "Description": "Desired number of tasks to run"
        },
        "LogGroupName": {
            "Type": "String",
            "Default": "/ecs/fargate-task-definition",
            "Description": "CloudWatch Log Group name"
        },
        "VpcCidr": {
            "Type": "String",
            "Default": "10.0.0.0/16",
            "Description": "CIDR block for the VPC"
        },
        "PublicSubnet1Cidr": {
            "Type": "String",
            "Default": "10.0.0.0/24",
            "Description": "CIDR block for public subnet 1"
        },
        "PublicSubnet2Cidr": {
            "Type": "String",
            "Default": "10.0.1.0/24",
            "Description": "CIDR block for public subnet 2"
```

```
        }
    },
    "Resources": {
        "VPC": {
            "Type": "AWS::EC2::VPC",
            "Properties": {
                "CidrBlock": {
                    "Ref": "VpcCidr"
                },
                "EnableDnsSupport": true,
                "EnableDnsHostnames": true,
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-VPC"
                        }
                    }
                ]
            }
        },
        "InternetGateway": {
            "Type": "AWS::EC2::InternetGateway",
            "Properties": {
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-IGW"
                        }
                    }
                ]
            }
        },
        "InternetGatewayAttachment": {
            "Type": "AWS::EC2::VPCGatewayAttachment",
            "Properties": {
                "InternetGatewayId": {
                    "Ref": "InternetGateway"
                },
                "VpcId": {
                    "Ref": "VPC"
                }
            }
```

```
        },
        "PublicSubnet1": {
            "Type": "AWS::EC2::Subnet",
            "Properties": {
                "VpcId": {
                    "Ref": "VPC"
                },
                "AvailabilityZone": {
                    "Fn::Select": [
                        0,
                        {
                            "Fn::GetAZs": ""
                        }
                    ]
                },
                "CidrBlock": {
                    "Ref": "PublicSubnet1Cidr"
                },
                "MapPublicIpOnLaunch": true,
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-PublicSubnet1"
                        }
                    }
                ]
            }
        },
        "PublicSubnet2": {
            "Type": "AWS::EC2::Subnet",
            "Properties": {
                "VpcId": {
                    "Ref": "VPC"
                },
                "AvailabilityZone": {
                    "Fn::Select": [
                        1,
                        {
                            "Fn::GetAZs": ""
                        }
                    ]
                },
                "CidrBlock": {
```

```
                    "Ref": "PublicSubnet2Cidr"
                },
                "MapPublicIpOnLaunch": true,
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-PublicSubnet2"
                        }
                    }
                ]
            }
        },
        "PublicRouteTable": {
            "Type": "AWS::EC2::RouteTable",
            "Properties": {
                "VpcId": {
                    "Ref": "VPC"
                },
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-PublicRouteTable"
                        }
                    }
                ]
            }
        },
        "DefaultPublicRoute": {
            "Type": "AWS::EC2::Route",
            "DependsOn": "InternetGatewayAttachment",
            "Properties": {
                "RouteTableId": {
                    "Ref": "PublicRouteTable"
                },
                "DestinationCidrBlock": "0.0.0.0/0",
                "GatewayId": {
                    "Ref": "InternetGateway"
                }
            }
        },
        "PublicSubnet1RouteTableAssociation": {
            "Type": "AWS::EC2::SubnetRouteTableAssociation",
```

```
            "Properties": {
                "RouteTableId": {
                    "Ref": "PublicRouteTable"
                },
                "SubnetId": {
                    "Ref": "PublicSubnet1"
                }
            }
        },
        "PublicSubnet2RouteTableAssociation": {
            "Type": "AWS::EC2::SubnetRouteTableAssociation",
            "Properties": {
                "RouteTableId": {
                    "Ref": "PublicRouteTable"
                },
                "SubnetId": {
                    "Ref": "PublicSubnet2"
                }
            }
        },
        "ECSSecurityGroup": {
            "Type": "AWS::EC2::SecurityGroup",
            "Properties": {
                "GroupDescription": "Security group for ECS tasks",
                "VpcId": {
                    "Ref": "VPC"
                },
                "SecurityGroupIngress": [
                    {
                        "IpProtocol": "tcp",
                        "FromPort": 80,
                        "ToPort": 80,
                        "CidrIp": "0.0.0.0/0"
                    },
                    {
                        "IpProtocol": "tcp",
                        "FromPort": 443,
                        "ToPort": 443,
                        "CidrIp": "0.0.0.0/0"
                    }
                ]
            }
        },
        "ECSTaskExecutionRole": {
```

```
            "Type": "AWS::IAM::Role",
            "Properties": {
                "AssumeRolePolicyDocument": {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {
                                "Service": "ecs-tasks.amazonaws.com"
                            },
                            "Action": "sts:AssumeRole"
                        }
                    ]
                },
                "ManagedPolicyArns": [
                    "arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy"
                ]
            }
        },
        "TaskLogGroup": {
            "Type": "AWS::Logs::LogGroup",
            "DeletionPolicy": "Retain",
            "UpdateReplacePolicy": "Retain",
            "Properties": {
                "LogGroupName": {
                    "Ref": "LogGroupName"
                },
                "RetentionInDays": 30
            }
        },
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": {
                    "Ref": "ClusterName"
                }
            }
        },
        "ECSTaskDefinition": {
            "Type": "AWS::ECS::TaskDefinition",
            "Properties": {
                "ContainerDefinitions": [
                    {
```

```
                        "Command": [
                        "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
                        ],
                        "EntryPoint": [
                            "sh",
                            "-c"
                        ],
                        "Essential": true,
                        "Image": {
                            "Ref": "ContainerImage"
                        },
                        "LogConfiguration": {
                            "LogDriver": "awslogs",
                            "Options": {
                                "mode": "non-blocking",
                                "max-buffer-size": "25m",
                                "awslogs-create-group": "true",
                                "awslogs-group": {
                                    "Ref": "LogGroupName"
                                },
                                "awslogs-region": {
                                    "Ref": "AWS::Region"
                                },
                                "awslogs-stream-prefix": "ecs"
                            }
                        },
                        "Name": "sample-fargate-app",
                        "PortMappings": [
                            {
                                "ContainerPort": 80,
                                "HostPort": 80,
                                "Protocol": "tcp"
                            }
                        ]
                    }
                ],
                "Cpu": {
                    "Ref": "TaskCpu"
                },
```

```
                "ExecutionRoleArn": {
                    "Fn::GetAtt": [
                        "ECSTaskExecutionRole",
                        "Arn"
                    ]
                },
                "Family": {
                    "Ref": "TaskFamily"
                },
                "Memory": {
                    "Ref": "TaskMemory"
                },
                "NetworkMode": "awsvpc",
                "RequiresCompatibilities": [
                    "FARGATE"
                ],
                "RuntimePlatform": {
                    "OperatingSystemFamily": "LINUX"
                }
            }
        },
        "ECSService": {
            "Type": "AWS::ECS::Service",
            "DependsOn": [
                "PublicSubnet1RouteTableAssociation",
                "PublicSubnet2RouteTableAssociation"
            ],
            "Properties": {
                "ServiceName": {
                    "Ref": "ServiceName"
                },
                "Cluster": {
                    "Ref": "ECSCluster"
                },
                "DesiredCount": {
                    "Ref": "DesiredCount"
                },
                "LaunchType": "FARGATE",
                "NetworkConfiguration": {
                    "AwsvpcConfiguration": {
                        "AssignPublicIp": "ENABLED",
                        "SecurityGroups": [
                            {
                                "Ref": "ECSSecurityGroup"
```

```
                        }
                    ],
                    "Subnets": [
                        {
                            "Ref": "PublicSubnet1"
                        },
                        {
                            "Ref": "PublicSubnet2"
                        }
                    ]
                }
            },
            "TaskDefinition": {
                "Ref": "ECSTaskDefinition"
            }
        }
    }
},
"Outputs": {
    "ClusterName": {
        "Description": "The name of the ECS cluster",
        "Value": {
            "Ref": "ECSCluster"
        }
    },
    "TaskDefinitionArn": {
        "Description": "The ARN of the task definition",
        "Value": {
            "Ref": "ECSTaskDefinition"
        }
    },
    "ServiceName": {
        "Description": "The name of the ECS service",
        "Value": {
            "Ref": "ECSService"
        }
    },
    "VpcId": {
        "Description": "The ID of the VPC",
        "Value": {
            "Ref": "VPC"
        }
    },
    "PublicSubnet1": {
```

```
            "Description": "The ID of public subnet 1",
            "Value": {
                "Ref": "PublicSubnet1"
            }
        },
        "PublicSubnet2": {
            "Description": "The ID of public subnet 2",
            "Value": {
                "Ref": "PublicSubnet2"
            }
        },
        "SecurityGroup": {
            "Description": "The ID of the security group",
            "Value": {
                "Ref": "ECSSecurityGroup"
            }
        },
        "ExecutionRoleArn": {
            "Description": "The ARN of the task execution role",
            "Value": {
                "Fn::GetAtt": [
                    "ECSTaskExecutionRole",
                    "Arn"
                ]
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: A template that deploys an application that is built on an Apache
 web server Docker image by creating an Amazon ECS cluster, task definition, and
 service. The template also creates networking and logging resources, and an Amazon
 ECS task execution role.
Parameters:
  ClusterName:
    Type: String
    Default: CFNCluster
    Description: Name of the ECS Cluster
  TaskFamily:
    Type: String
```

```
    Default: task-definition-cfn
    Description: Family name for the Task Definition
  ServiceName:
    Type: String
    Default: cfn-service
    Description: Name of the ECS Service
  ContainerImage:
    Type: String
    Default: public.ecr.aws/docker/library/httpd:2.4
    Description: Container image to use for the task
  TaskCpu:
    Type: Number
    Default: 256
    AllowedValues: [256, 512, 1024, 2048, 4096]
    Description: CPU units for the task
  TaskMemory:
    Type: Number
    Default: 512
    AllowedValues: [512, 1024, 2048, 4096, 8192, 16384]
    Description: Memory (in MiB) for the task
  DesiredCount:
    Type: Number
    Default: 1
    Description: Desired number of tasks to run
  LogGroupName:
    Type: String
    Default: /ecs/fargate-task-definition
    Description: CloudWatch Log Group name
  VpcCidr:
    Type: String
    Default: 10.0.0.0/16
    Description: CIDR block for the VPC
  PublicSubnet1Cidr:
    Type: String
    Default: 10.0.0.0/24
    Description: CIDR block for public subnet 1
  PublicSubnet2Cidr:
    Type: String
    Default: 10.0.1.0/24
    Description: CIDR block for public subnet 2
Resources:
  # VPC and Networking Resources
  VPC:
    Type: AWS::EC2::VPC
```

```
  Properties:
    CidrBlock: !Ref VpcCidr
    EnableDnsSupport: true
    EnableDnsHostnames: true
    Tags:
      - Key: Name
        Value: !Sub ${AWS::StackName}-VPC
InternetGateway:
  Type: AWS::EC2::InternetGateway
  Properties:
    Tags:
      - Key: Name
        Value: !Sub ${AWS::StackName}-IGW
InternetGatewayAttachment:
  Type: AWS::EC2::VPCGatewayAttachment
  Properties:
    InternetGatewayId: !Ref InternetGateway
    VpcId: !Ref VPC
PublicSubnet1:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId: !Ref VPC
    AvailabilityZone: !Select [0, !GetAZs '']
    CidrBlock: !Ref PublicSubnet1Cidr
    MapPublicIpOnLaunch: true
    Tags:
      - Key: Name
        Value: !Sub ${AWS::StackName}-PublicSubnet1
PublicSubnet2:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId: !Ref VPC
    AvailabilityZone: !Select [1, !GetAZs '']
    CidrBlock: !Ref PublicSubnet2Cidr
    MapPublicIpOnLaunch: true
    Tags:
      - Key: Name
        Value: !Sub ${AWS::StackName}-PublicSubnet2
PublicRouteTable:
  Type: AWS::EC2::RouteTable
  Properties:
    VpcId: !Ref VPC
    Tags:
      - Key: Name
```

```
        Value: !Sub ${AWS::StackName}-PublicRouteTable
  DefaultPublicRoute:
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway
  PublicSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet1
  PublicSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet2
  # Security Group
  ECSSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Security group for ECS tasks
      VpcId: !Ref VPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 443
          ToPort: 443
          CidrIp: 0.0.0.0/0

  # IAM Roles
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
```

```
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
  # CloudWatch Logs
  TaskLogGroup:
    Type: AWS::Logs::LogGroup
    DeletionPolicy: Retain
    UpdateReplacePolicy: Retain
    Properties:
      LogGroupName: !Ref LogGroupName
      RetentionInDays: 30
  # ECS Resources
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ClusterName
  ECSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
              /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
              App</title> <style>body {margin-top: 40px; background-color:
              #333;} </style> </head><body> <div
              style=color:white;text-align:center> <h1>Amazon ECS Sample
              App</h1> <h2>Congratulations!</h2> <p>Your application is now
              running on a container in Amazon ECS.</p> </div></body></html>' >
              /usr/local/apache2/htdocs/index.html && httpd-foreground"s
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: !Ref ContainerImage
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Ref LogGroupName
              awslogs-region: !Ref 'AWS::Region'
              awslogs-stream-prefix: ecs
          Name: sample-fargate-app
```

```
          PortMappings:
            - ContainerPort: 80
              HostPort: 80
              Protocol: tcp
      Cpu: !Ref TaskCpu
      ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
      Family: !Ref TaskFamily
      Memory: !Ref TaskMemory
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: LINUX
  ECSService:
    Type: AWS::ECS::Service
    DependsOn:
      - PublicSubnet1RouteTableAssociation
      - PublicSubnet2RouteTableAssociation
    Properties:
      ServiceName: !Ref ServiceName
      Cluster: !Ref ECSCluster
      DesiredCount: !Ref DesiredCount
      LaunchType: FARGATE
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups:
            - !Ref ECSSecurityGroup
          Subnets:
            - !Ref PublicSubnet1
            - !Ref PublicSubnet2
      TaskDefinition: !Ref ECSTaskDefinition
Outputs:
  ClusterName:
    Description: The name of the ECS cluster
    Value: !Ref ECSCluster
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref ECSTaskDefinition
  ServiceName:
    Description: The name of the ECS service
    Value: !Ref ECSService
  VpcId:
    Description: The ID of the VPC
```

```
    Value: !Ref VPC
  PublicSubnet1:
    Description: The ID of public subnet 1
    Value: !Ref PublicSubnet1
  PublicSubnet2:
    Description: The ID of public subnet 2
    Value: !Ref PublicSubnet2
  SecurityGroup:
    Description: The ID of the security group
    Value: !Ref ECSSecurityGroup
  ExecutionRoleArn:
    Description: The ARN of the task execution role
    Value: !GetAtt ECSTaskExecutionRole.Arn
```

Implantar um serviço com ECS Exec habilitado

O modelo a seguir pode ser usado para implantar um serviço com o ECS Exec habilitado. O serviço é executado em um cluster com uma chave do KMS para criptografar sessões do ECS Exec e uma configuração de log para redirecionar os logs da sessão de comando de execução para um bucket do Amazon S3. Para obter mais informações, consulte Monitoramento de contêineres do Amazon ECS com o ECS Exec.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "ECS Cluster with Fargate Service and Task Definition and ECS
 Exec enabled.",
    "Parameters": {
        "ClusterName": {
            "Type": "String",
            "Default": "CFNCluster",
            "Description": "Name of the ECS Cluster"
        },
        "S3BucketName": {
            "Type": "String",
            "Default": "amzn-s3-demo-bucket",
            "Description": "S3 bucket for ECS execute command logs"
        },
        "KmsKeyId": {
            "Type": "String",
            "Default": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
```

```
            "Description": "KMS Key ID for ECS execute command encryption"
        },
        "ContainerImage": {
            "Type": "String",
            "Default": "public.ecr.aws/docker/library/httpd:2.4",
            "Description": "Container image to use for the task"
        },
        "ContainerCpu": {
            "Type": "Number",
            "Default": 256,
            "AllowedValues": [256, 512, 1024, 2048, 4096],
            "Description": "CPU units for the container (256 = 0.25 vCPU)"
        },
        "ContainerMemory": {
            "Type": "Number",
            "Default": 512,
            "AllowedValues": [512, 1024, 2048, 3072, 4096, 5120, 6144, 7168, 8192],
            "Description": "Memory for the container (in MiB)"
        },
        "DesiredCount": {
            "Type": "Number",
            "Default": 1,
            "Description": "Desired count of tasks in the service"
        },
       "SecurityGroups": {
 "Type": "List<AWS::EC2::SecurityGroup::Id>",
 "Description": "Security Group IDs for the ECS Service"
},
"Subnets": {
 "Type": "List<AWS::EC2::Subnet::Id>",
 "Description": "Subnet IDs for the ECS Service"
},
        "ServiceName": {
            "Type": "String",
            "Default": "cfn-service",
            "Description": "Name of the ECS service"
        },
        "TaskFamily": {
            "Type": "String",
            "Default": "task-definition-cfn",
            "Description": "Family name for the task definition"
        },
        "TaskExecutionRoleArn": {
            "Type": "String",
```

```
            "Description": "ARN of an existing IAM role for ECS task execution",
            "Default": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole"
        },
        "TaskRoleArn": {
            "Type": "String",
            "Description": "ARN of an existing IAM role for ECS tasks",
            "Default": "arn:aws:iam::111122223333:role/execTaskRole"
        }
    },
    "Resources": {
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": {"Ref": "ClusterName"},
                "Configuration": {
                    "ExecuteCommandConfiguration": {
                        "Logging": "OVERRIDE",
                        "LogConfiguration": {
                            "S3BucketName": {"Ref": "S3BucketName"}
                        },
                        "KmsKeyId": {"Ref": "KmsKeyId"}
                    }
                },
                "Tags": [
                    {
                        "Key": "Environment",
                        "Value": {"Ref": "AWS::StackName"}
                    }
                ]
            }
        },
        "ECSTaskDefinition": {
            "Type": "AWS::ECS::TaskDefinition",
            "Properties": {
                "ContainerDefinitions": [
                    {
                        "Command": [
                            "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS
 Sample App</title> <style>body {margin-top: 40px; background-color: #333;} </style>
 </head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
                        ],
```

```
                        "EntryPoint": [
                            "sh",
                            "-c"
                        ],
                        "Essential": true,
                        "Image": {"Ref": "ContainerImage"},
                        "LogConfiguration": {
                            "LogDriver": "awslogs",
                            "Options": {
                                "mode": "non-blocking",
                                "max-buffer-size": "25m",
                                "awslogs-create-group": "true",
                                "awslogs-group": {"Fn::Sub": "/ecs/
${AWS::StackName}"},
                                "awslogs-region": {"Ref": "AWS::Region"},
                                "awslogs-stream-prefix": "ecs"
                            }
                        },
                        "Name": "sample-fargate-app",
                        "PortMappings": [
                            {
                                "ContainerPort": 80,
                                "HostPort": 80,
                                "Protocol": "tcp"
                            }
                        ]
                    }
                ],
                "Cpu": {"Ref": "ContainerCpu"},
                "ExecutionRoleArn": {"Ref": "TaskExecutionRoleArn"},
                "TaskRoleArn": {"Ref": "TaskRoleArn"},
                "Family": {"Ref": "TaskFamily"},
                "Memory": {"Ref": "ContainerMemory"},
                "NetworkMode": "awsvpc",
                "RequiresCompatibilities": [
                    "FARGATE"
                ],
                "RuntimePlatform": {
                    "OperatingSystemFamily": "LINUX"
                },
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {"Fn::Sub": "${AWS::StackName}-TaskDefinition"}
```

```
                    }
                ]
            }
        },
        "ECSService": {
            "Type": "AWS::ECS::Service",
            "Properties": {
                "ServiceName": {"Ref": "ServiceName"},
                "Cluster": {"Ref": "ECSCluster"},
                "DesiredCount": {"Ref": "DesiredCount"},
                "EnableExecuteCommand": true,
                "LaunchType": "FARGATE",
                "NetworkConfiguration": {
                    "AwsvpcConfiguration": {
                        "AssignPublicIp": "ENABLED",
                        "SecurityGroups": {"Ref": "SecurityGroups"},
                        "Subnets": {"Ref": "Subnets"}
                    }
                },
                "TaskDefinition": {"Ref": "ECSTaskDefinition"},
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {"Fn::Sub": "${AWS::StackName}-Service"}
                    }
                ]
            }
        }
    },
    "Outputs": {
        "ClusterName": {
            "Description": "The name of the ECS cluster",
            "Value": {"Ref": "ECSCluster"}
        },
        "ServiceName": {
            "Description": "The name of the ECS service",
            "Value": {"Ref": "ServiceName"}
        },
        "TaskDefinitionArn": {
            "Description": "The ARN of the task definition",
            "Value": {"Ref": "ECSTaskDefinition"}
        },
        "ClusterArn": {
            "Description": "The ARN of the ECS cluster",
```

```
            "Value": {"Fn::GetAtt": ["ECSCluster", "Arn"]}
        },
        "StackName": {
            "Description": "The name of this stack",
            "Value": {"Ref": "AWS::StackName"}
        },
        "StackId": {
            "Description": "The unique identifier for this stack",
            "Value": {"Ref": "AWS::StackId"}
        },
        "Region": {
            "Description": "The AWS Region where the stack is deployed",
            "Value": {"Ref": "AWS::Region"}
        },
        "AccountId": {
            "Description": "The AWS Account ID",
            "Value": {"Ref": "AWS::AccountId"}
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: ECS Cluster with Fargate Service and Task Definition and ECS Exec
 enabled.
Parameters:
  ClusterName:
    Type: String
    Default: CFNCluster
    Description: Name of the ECS Cluster
  S3BucketName:
    Type: String
    Default: amzn-s3-demo-bucket
    Description: S3 bucket for ECS execute command logs
  KmsKeyId:
    Type: String
    Default: a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
    Description: KMS Key ID for ECS execute command encryption
  ContainerImage:
    Type: String
    Default: public.ecr.aws/docker/library/httpd:2.4
    Description: Container image to use for the task
```

```
  ContainerCpu:
    Type: Number
    Default: 256
    AllowedValues: [256, 512, 1024, 2048, 4096]
    Description: CPU units for the container (256 = 0.25 vCPU)
  ContainerMemory:
    Type: Number
    Default: 512
    AllowedValues: [512, 1024, 2048, 3072, 4096, 5120, 6144, 7168, 8192]
    Description: Memory for the container (in MiB)
  DesiredCount:
    Type: Number
    Default: 1
    Description: Desired count of tasks in the service
 SecurityGroups:
    Type: List<AWS::EC2::SecurityGroup::Id>
    Description: Security Group IDs for the ECS Service
  Subnets:
    Type: List<AWS::EC2::Subnet::Id>
    Description: Subnet IDs for the ECS Service
  ServiceName:
    Type: String
    Default: cfn-service
    Description: Name of the ECS service
  TaskFamily:
    Type: String
    Default: task-definition-cfn
    Description: Family name for the task definition
  TaskExecutionRoleArn:
    Type: String
    Description: ARN of an existing IAM role for ECS task execution
    Default: 'arn:aws:iam::111122223333:role/ecsTaskExecutionRole'
  TaskRoleArn:
    Type: String
    Description: ARN of an existing IAM role for ECS tasks
    Default: 'arn:aws:iam::111122223333:role/execTaskRole'
Resources:
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ClusterName
      Configuration:
        ExecuteCommandConfiguration:
          Logging: OVERRIDE
```

```
          LogConfiguration:
            S3BucketName: !Ref S3BucketName
          KmsKeyId: !Ref KmsKeyId
      Tags:
        - Key: Environment
          Value: !Ref AWS::StackName
  ECSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
             /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
              App</title> <style>body {margin-top: 40px; background-color:
              #333;} </style> </head><body> <div
              style=color:white;text-align:center> <h1>Amazon ECS Sample
              App</h1> <h2>Congratulations!</h2> <p>Your application is now
              running on a container in Amazon ECS.</p> </div></body></html>' >
              /usr/local/apache2/htdocs/index.html && httpd-foreground"
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: !Ref ContainerImage
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Sub /ecs/${AWS::StackName}
              awslogs-region: !Ref AWS::Region
              awslogs-stream-prefix: ecs
          Name: sample-fargate-app
          PortMappings:
            - ContainerPort: 80
              HostPort: 80
              Protocol: tcp
      Cpu: !Ref ContainerCpu
      ExecutionRoleArn: !Ref TaskExecutionRoleArn
      TaskRoleArn: !Ref TaskRoleArn
      Family: !Ref TaskFamily
      Memory: !Ref ContainerMemory
      NetworkMode: awsvpc
```

```
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: LINUX
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-TaskDefinition
  ECSService:
    Type: AWS::ECS::Service
    Properties:
      ServiceName: !Ref ServiceName
      Cluster: !Ref ECSCluster
      DesiredCount: !Ref DesiredCount
      EnableExecuteCommand: true
      LaunchType: FARGATE
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups: !Ref SecurityGroups
          Subnets: !Ref Subnets
      TaskDefinition: !Ref ECSTaskDefinition
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-Service
Outputs:
  ClusterName:
    Description: The name of the ECS cluster
    Value: !Ref ECSCluster
  ServiceName:
    Description: The name of the ECS service
    Value: !Ref ServiceName
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref ECSTaskDefinition
  ClusterArn:
    Description: The ARN of the ECS cluster
    Value: !GetAtt ECSCluster.Arn
  StackName:
    Description: The name of this stack
    Value: !Ref AWS::StackName
  StackId:
    Description: The unique identifier for this stack
    Value: !Ref AWS::StackId
  Region:
```

```
    Description: The AWS Region where the stack is deployed
    Value: !Ref AWS::Region
  AccountId:
    Description: The AWS Account ID
    Value: !Ref AWS::AccountId
```

Implantar serviço que usa o Amazon VPC Lattice

É possível usar o modelo a seguir para começar a criar um serviço do Amazon ECS com o VPC Lattice. Talvez seja necessário concluir as seguintes etapas adicionais para configurar o VPC Lattice:

- Atualize as regras de entrada do seu grupo de segurança do VPC Lattice para permitir o prefixo da regra de entrada `vpc-lattice` e permitir o tráfego na porta 80.
- Associe a VPC do serviço a uma rede de serviços do VPC Lattice.
- Configure uma zona hospedada privada ou pública com o Amazon Route 53.
- Configure receptores e regras de receptor em um serviço VPC Lattice.
- Revise as configurações de verificação de integridade do grupo de destino.

Para obter mais informações sobre o uso do VPC Lattice com o Amazon ECS, consulte Use o Amazon VPC Lattice para conectar, observar e proteger seus serviços do Amazon ECS.

JSON

```
{
 "AWSTemplateFormatVersion": "2010-09-09",
 "Description": "The template used to create an ECS Service with VPC Lattice.",
 "Parameters": {
  "ECSClusterName": {
   "Type": "String",
   "Default": "vpc-lattice-cluster"
  },
  "ECSServiceName": {
   "Type": "String",
   "Default": "vpc-lattice-service"
  },
  "SecurityGroupIDs": {
   "Type": "List<AWS::EC2::SecurityGroup::Id>",
   "Description": "Security Group IDs for the ECS Service"
  },
```

```
  "SubnetIDs": {
   "Type": "List<AWS::EC2::Subnet::Id>",
   "Description": "Subnet IDs for the ECS Service"
  },
  "VpcID": {
   "Type": "AWS::EC2::VPC::Id",
   "Description": "VPC ID for the resources"
  },
  "ContainerImage": {
   "Type": "String",
   "Default": "public.ecr.aws/docker/library/httpd:2.4",
   "Description": "Container image to use for the task"
  },
  "TaskCpu": {
   "Type": "Number",
   "Default": 256,
   "AllowedValues": [256, 512, 1024, 2048, 4096],
   "Description": "CPU units for the task"
  },
  "TaskMemory": {
   "Type": "Number",
   "Default": 512,
   "AllowedValues": [512, 1024, 2048, 4096, 8192, 16384],
   "Description": "Memory (in MiB) for the task"
  },
  "LogGroupName": {
   "Type": "String",
   "Default": "/ecs/vpc-lattice-task",
   "Description": "CloudWatch Log Group name"
  },
  "EnableContainerInsights": {
   "Type": "String",
   "Default": "enabled",
   "AllowedValues": ["enabled", "disabled"],
   "Description": "Enable or disable CloudWatch Container Insights for the cluster"
  }
 },
 "Resources": {
  "ECSCluster": {
   "Type": "AWS::ECS::Cluster",
   "Properties": {
    "ClusterName": {"Ref": "ECSClusterName"},
    "ClusterSettings": [
     {
```

```
      "Name": "containerInsights",
      "Value": {"Ref": "EnableContainerInsights"}
     }
    ],
    "Tags": [
     {
      "Key": "Name",
      "Value": {"Ref": "ECSClusterName"}
     }
    ]
   }
  },
  "ECSTaskExecutionRole": {
   "Type": "AWS::IAM::Role",
   "Properties": {
    "AssumeRolePolicyDocument": {
     "Version": "2012-10-17",
     "Statement": [
      {
       "Effect": "Allow",
       "Principal": {
        "Service": "ecs-tasks.amazonaws.com"
       },
       "Action": "sts:AssumeRole"
      }
     ]
    },
    "ManagedPolicyArns": [
     "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
    ]
   }
  },
  "TaskLogGroup": {
   "Type": "AWS::Logs::LogGroup",
   "DeletionPolicy": "Retain",
   "UpdateReplacePolicy": "Retain",
   "Properties": {
    "LogGroupName": {"Ref": "LogGroupName"},
    "RetentionInDays": 30
   }
  },
  "VpcLatticeTaskDefinition": {
   "Type": "AWS::ECS::TaskDefinition",
   "Properties": {
```

```
   "ContainerDefinitions": [
    {
     "Command": [
      "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</title>
<style>body {margin-top: 40px; background-color: #333;} </style> </head><body>
<div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
<h2>Congratulations!</h2> <p>Your application is now running on a container in
Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
httpd-foreground\""
     ],
     "EntryPoint": [
      "sh",
      "-c"
     ],
     "Essential": true,
     "Image": {"Ref": "ContainerImage"},
     "LogConfiguration": {
      "LogDriver": "awslogs",
      "Options": {
       "mode": "non-blocking",
       "max-buffer-size": "25m",
       "awslogs-create-group": "true",
       "awslogs-group": {"Ref": "LogGroupName"},
       "awslogs-region": {"Ref": "AWS::Region"},
       "awslogs-stream-prefix": "ecs"
      }
     },
     "Name": "vpc-lattice-container",
     "PortMappings": [
      {
       "ContainerPort": 80,
       "HostPort": 80,
       "Protocol": "tcp",
       "Name": "vpc-lattice-port"
      }
     ]
    }
   ],
   "Cpu": {"Ref": "TaskCpu"},
   "ExecutionRoleArn": {"Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]},
   "Family": "vpc-lattice-task-definition",
   "Memory": {"Ref": "TaskMemory"},
   "NetworkMode": "awsvpc",
   "RequiresCompatibilities": [
```

```
      "FARGATE"
     ],
     "RuntimePlatform": {
      "OperatingSystemFamily": "LINUX"
     }
    }
   },
   "ECSService": {
    "Type": "AWS::ECS::Service",
    "Properties": {
     "Cluster": {"Ref": "ECSCluster"},
     "TaskDefinition": {"Ref": "VpcLatticeTaskDefinition"},
     "LaunchType": "FARGATE",
     "ServiceName": {"Ref": "ECSServiceName"},
     "SchedulingStrategy": "REPLICA",
     "DesiredCount": 2,
     "AvailabilityZoneRebalancing": "ENABLED",
     "NetworkConfiguration": {
      "AwsvpcConfiguration": {
       "AssignPublicIp": "ENABLED",
       "SecurityGroups": {
        "Ref": "SecurityGroupIDs"
       },
       "Subnets": {
        "Ref": "SubnetIDs"
       }
      }
     },
     "PlatformVersion": "LATEST",
     "VpcLatticeConfigurations": [
      {
       "RoleArn": "arn:aws:iam::111122223333:role/ecsInfrastructureRole",
       "PortName": "vpc-lattice-port",
       "TargetGroupArn": {
        "Ref": "TargetGroup1"
       }
      }
     ],
     "DeploymentConfiguration": {
      "DeploymentCircuitBreaker": {
       "Enable": true,
       "Rollback": true
      },
      "MaximumPercent": 200,
```

```
     "MinimumHealthyPercent": 100
    },
    "DeploymentController": {
     "Type": "ECS"
    },
    "ServiceConnectConfiguration": {
     "Enabled": false
    },
    "Tags": [],
    "EnableECSManagedTags": true
   }
  },
  "TargetGroup1": {
   "Type": "AWS::VpcLattice::TargetGroup",
   "Properties": {
    "Type": "IP",
    "Name": "first-target-group",
    "Config": {
     "Port": 80,
     "Protocol": "HTTP",
     "VpcIdentifier": {"Ref": "VpcID"},
     "HealthCheck": {
      "Enabled": true,
      "Path": "/"
     }
    },
    "Tags": [
     {
      "Key": "ecs-application-networking/ServiceName",
      "Value": {"Ref": "ECSServiceName"}
     },
     {
      "Key": "ecs-application-networking/ClusterName",
      "Value": {"Ref": "ECSClusterName"}
     },
     {
      "Key": "ecs-application-networking/TaskDefinition",
      "Value": {"Ref": "VpcLatticeTaskDefinition"}
     },
     {
      "Key": "ecs-application-networking/VpcId",
      "Value": {"Ref": "VpcID"}
     }
    ]
```

```
    }
   }
  },
  "Outputs": {
   "ClusterName": {
    "Description": "The cluster used to create the service.",
    "Value": {
     "Ref": "ECSCluster"
    }
   },
   "ClusterArn": {
    "Description": "The ARN of the ECS cluster",
    "Value": {
     "Fn::GetAtt": ["ECSCluster", "Arn"]
    }
   },
   "ECSService": {
    "Description": "The created service.",
    "Value": {
     "Ref": "ECSService"
    }
   },
   "TaskDefinitionArn": {
    "Description": "The ARN of the task definition",
    "Value": {
     "Ref": "VpcLatticeTaskDefinition"
    }
   }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: The template used to create an ECS Service with VPC Lattice.

Parameters:
  ECSClusterName:
    Type: String
    Default: vpc-lattice-cluster
  ECSServiceName:
    Type: String
    Default: vpc-lattice-service
```

```
  SecurityGroupIDs:
    Type: List<AWS::EC2::SecurityGroup::Id>
    Description: Security Group IDs for the ECS Service
  SubnetIDs:
    Type: List<AWS::EC2::Subnet::Id>
    Description: Subnet IDs for the ECS Service
  VpcID:
    Type: AWS::EC2::VPC::Id
    Description: VPC ID for the resources
  ContainerImage:
    Type: String
    Default: public.ecr.aws/docker/library/httpd:2.4
    Description: Container image to use for the task
  TaskCpu:
    Type: Number
    Default: 256
    AllowedValues: [256, 512, 1024, 2048, 4096]
    Description: CPU units for the task
  TaskMemory:
    Type: Number
    Default: 512
    AllowedValues: [512, 1024, 2048, 4096, 8192, 16384]
    Description: Memory (in MiB) for the task
  LogGroupName:
    Type: String
    Default: /ecs/vpc-lattice-task
    Description: CloudWatch Log Group name
  EnableContainerInsights:
    Type: String
    Default: 'enhanced'
    AllowedValues: ['enabled', 'disabled', 'enhanced']
    Description: Enable or disable CloudWatch Container Insights for the cluster

Resources:
  # ECS Cluster
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ECSClusterName
      ClusterSettings:
        - Name: containerInsights
          Value: !Ref EnableContainerInsights
      Tags:
        - Key: Name
```

```
        Value: !Ref ECSClusterName

# IAM Roles
ECSTaskExecutionRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Effect: Allow
          Principal:
            Service: ecs-tasks.amazonaws.com
          Action: sts:AssumeRole
    ManagedPolicyArns:
      - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy

# CloudWatch Logs
TaskLogGroup:
  Type: AWS::Logs::LogGroup
  DeletionPolicy: Retain
  UpdateReplacePolicy: Retain
  Properties:
    LogGroupName: !Ref LogGroupName
    RetentionInDays: 30

# Task Definition
VpcLatticeTaskDefinition:
  Type: AWS::ECS::TaskDefinition
  Properties:
    ContainerDefinitions:
      - Command:
          - >-
            /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
            App</title> <style>body {margin-top: 40px; background-color:
            #333;} </style> </head><body> <div
            style=color:white;text-align:center> <h1>Amazon ECS Sample
            App</h1> <h2>Congratulations!</h2> <p>Your application is now
            running on a container in Amazon ECS.</p> </div></body></html>' >
            /usr/local/apache2/htdocs/index.html && httpd-foreground"
        EntryPoint:
          - sh
          - '-c'
        Essential: true
        Image: !Ref ContainerImage
```

```
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Ref LogGroupName
              awslogs-region: !Ref 'AWS::Region'
              awslogs-stream-prefix: ecs
          Name: vpc-lattice-container
          PortMappings:
            - ContainerPort: 80
              HostPort: 80
              Protocol: tcp
              Name: vpc-lattice-port
      Cpu: !Ref TaskCpu
      ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
      Family: vpc-lattice-task-definition
      Memory: !Ref TaskMemory
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: LINUX

  ECSService:
    Type: AWS::ECS::Service
    Properties:
      Cluster: !Ref ECSCluster
      TaskDefinition: !Ref VpcLatticeTaskDefinition
      LaunchType: FARGATE
      ServiceName: !Ref ECSServiceName
      SchedulingStrategy: REPLICA
      DesiredCount: 2
      AvailabilityZoneRebalancing: ENABLED
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups: !Ref SecurityGroupIDs
          Subnets: !Ref SubnetIDs
      PlatformVersion: LATEST
      VpcLatticeConfigurations:
        - RoleArn: arn:aws:iam::111122223333:role/ecsInfrastructureRole
          PortName: vpc-lattice-port
```

```
          TargetGroupArn: !Ref TargetGroup1
      DeploymentConfiguration:
        DeploymentCircuitBreaker:
          Enable: true
          Rollback: true
        MaximumPercent: 200
        MinimumHealthyPercent: 100
      DeploymentController:
        Type: ECS
      ServiceConnectConfiguration:
        Enabled: false
      Tags: []
      EnableECSManagedTags: true

  TargetGroup1:
    Type: AWS::VpcLattice::TargetGroup
    Properties:
      Type: IP
      Name: first-target-group
      Config:
        Port: 80
        Protocol: HTTP
        VpcIdentifier: !Ref VpcID
        HealthCheck:
          Enabled: true
          Path: /
      Tags:
        - Key: ecs-application-networking/ServiceName
          Value: !Ref ECSServiceName
        - Key: ecs-application-networking/ClusterName
          Value: !Ref ECSClusterName
        - Key: ecs-application-networking/TaskDefinition
          Value: !Ref VpcLatticeTaskDefinition
        - Key: ecs-application-networking/VpcId
          Value: !Ref VpcID

Outputs:
  ClusterName:
    Description: The cluster used to create the service.
    Value: !Ref ECSCluster
  ClusterArn:
    Description: The ARN of the ECS cluster
    Value: !GetAtt ECSCluster.Arn
  ECSService:
```

```
    Description: The created service.
    Value: !Ref ECSService
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref VpcLatticeTaskDefinition
```

Implantar serviço com uma configuração de volume

O modelo a seguir inclui uma configuração de volume na definição do serviço. O Amazon ECS oferece suporte à configuração dos seguintes volumes de dados usando uma configuração de volume na inicialização: volumes do Amazon EBS. Para obter mais informações sobre volumes do Amazon EBS, consulte Uso de volumes do Amazon EBS com o Amazon ECS.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "The template used to create an ECS Service that includes a
 volume configuration. The configuration is used to create Amazon EBS volumes for
 attachment to the tasks. One volume is attached per task.",
  "Parameters": {
    "ECSClusterName": {
      "Type": "String",
      "Default": "volume-config-cluster",
      "Description": "Name of the ECS cluster"
    },
    "SecurityGroupIDs": {
      "Type": "List<AWS::EC2::SecurityGroup::Id>",
      "Description": "Security Group IDs for the ECS Service"
    },
    "SubnetIDs": {
      "Type": "List<AWS::EC2::Subnet::Id>",
      "Description": "Subnet IDs for the ECS Service"
    },
    "InfrastructureRoleArn": {
      "Type": "String",
      "Description": "ARN of the IAM role that ECS will use to manage EBS volumes"
    },
    "ContainerImage": {
      "Type": "String",
      "Default": "public.ecr.aws/nginx/nginx:latest",
      "Description": "Container image to use for the task"
```

```
    },
    "TaskCpu": {
      "Type": "String",
      "Default": "2048",
      "Description": "CPU units for the task"
    },
    "TaskMemory": {
      "Type": "String",
      "Default": "4096",
      "Description": "Memory (in MiB) for the task"
    },
    "VolumeSize": {
      "Type": "String",
      "Default": "10",
      "Description": "Size of the EBS volume in GiB"
    },
    "VolumeType": {
      "Type": "String",
      "Default": "gp3",
      "AllowedValues": ["gp2", "gp3", "io1", "io2", "st1", "sc1", "standard"],
      "Description": "EBS volume type"
    },
    "VolumeIops": {
      "Type": "String",
      "Default": "3000",
      "Description": "IOPS for the EBS volume (required for io1, io2, and gp3)"
    },
    "VolumeThroughput": {
      "Type": "String",
      "Default": "125",
      "Description": "Throughput for the EBS volume (only for gp3)"
    },
    "FilesystemType": {
      "Type": "String",
      "Default": "xfs",
      "AllowedValues": ["xfs", "ext4"],
      "Description": "Filesystem type for the EBS volume"
    },
    "EnableContainerInsights": {
      "Type": "String",
      "Default": "enhanced",
      "AllowedValues": ["enabled", "disabled", "enhanced"],
      "Description": "Enable or disable CloudWatch Container Insights for the
  cluster"
```

```
    }
  },
  "Resources": {
    "ECSCluster": {
      "Type": "AWS::ECS::Cluster",
      "Properties": {
        "ClusterName": {"Ref": "ECSClusterName"},
        "ClusterSettings": [
          {
            "Name": "containerInsights",
            "Value": {"Ref": "EnableContainerInsights"}
          }
        ],
        "Tags": [
          {
            "Key": "Name",
            "Value": {"Ref": "ECSClusterName"}
          }
        ]
      }
    },
    "ECSTaskExecutionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
        },
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
        ]
      }
    },
    "EBSTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
```

```
        "Family": "ebs-task-attach-task-def",
        "ExecutionRoleArn": {"Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]},
        "NetworkMode": "awsvpc",
        "RequiresCompatibilities": [
          "EC2",
          "FARGATE"
        ],
        "Cpu": {"Ref": "TaskCpu"},
        "Memory": {"Ref": "TaskMemory"},
        "ContainerDefinitions": [
          {
            "Name": "nginx",
            "Image": {"Ref": "ContainerImage"},
            "Essential": true,
            "PortMappings": [
              {
                "Name": "nginx-80-tcp",
                "ContainerPort": 80,
                "HostPort": 80,
                "Protocol": "tcp",
                "AppProtocol": "http"
              }
            ],
            "MountPoints": [
              {
                "SourceVolume": "ebs-vol",
                "ContainerPath": "/foo-container-path",
                "ReadOnly": false
              }
            ]
          }
        ],
        "Volumes": [
          {
            "Name": "ebs-vol",
            "ConfiguredAtLaunch": true
          }
        ]
      }
    },
    "ECSService": {
      "Type": "AWS::ECS::Service",
      "Properties": {
        "Cluster": {"Ref": "ECSCluster"},
```

```
        "TaskDefinition": {"Ref": "EBSTaskDefinition"},
        "LaunchType": "FARGATE",
        "ServiceName": "ebs",
        "SchedulingStrategy": "REPLICA",
        "DesiredCount": 1,
        "NetworkConfiguration": {
          "AwsvpcConfiguration": {
            "AssignPublicIp": "ENABLED",
            "SecurityGroups": {"Ref": "SecurityGroupIDs"},
            "Subnets": {"Ref": "SubnetIDs"}
          }
        },
        "PlatformVersion": "LATEST",
        "DeploymentConfiguration": {
          "MaximumPercent": 200,
          "MinimumHealthyPercent": 100,
          "DeploymentCircuitBreaker": {
            "Enable": true,
            "Rollback": true
          }
        },
        "DeploymentController": {
          "Type": "ECS"
        },
        "Tags": [],
        "EnableECSManagedTags": true,
        "VolumeConfigurations": [
          {
            "Name": "ebs-vol",
            "ManagedEBSVolume": {
              "RoleArn": {"Ref": "InfrastructureRoleArn"},
              "VolumeType": {"Ref": "VolumeType"},
              "Iops": {"Ref": "VolumeIops"},
              "Throughput": {"Ref": "VolumeThroughput"},
              "SizeInGiB": {"Ref": "VolumeSize"},
              "FilesystemType": {"Ref": "FilesystemType"},
              "TagSpecifications": [
                {
                  "ResourceType": "volume",
                  "PropagateTags": "TASK_DEFINITION"
                }
              ]
            }
          }
```

```
        ]
      }
    }
  },
  "Outputs": {
    "ClusterName": {
      "Description": "The cluster used to create the service.",
      "Value": {"Ref": "ECSCluster"}
    },
    "ClusterArn": {
      "Description": "The ARN of the ECS cluster",
      "Value": {"Fn::GetAtt": ["ECSCluster", "Arn"]}
    },
    "ECSService": {
      "Description": "The created service.",
      "Value": {"Ref": "ECSService"}
    },
    "TaskDefinitionArn": {
      "Description": "The ARN of the task definition",
      "Value": {"Ref": "EBSTaskDefinition"}
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: The template used to create an ECS Service that includes a volume
 configuration. The configuration is used to create Amazon EBS volumes for
 attachment to the tasks. One volume is attached per task.
Parameters:
  ECSClusterName:
    Type: String
    Default: volume-config-cluster
    Description: Name of the ECS cluster

  SecurityGroupIDs:
    Type: List<AWS::EC2::SecurityGroup::Id>
    Description: Security Group IDs for the ECS Service

  SubnetIDs:
    Type: List<AWS::EC2::Subnet::Id>
    Description: Subnet IDs for the ECS Service
```

```
InfrastructureRoleArn:
  Type: String
  Description: ARN of the IAM role that ECS will use to manage EBS volumes

ContainerImage:
  Type: String
  Default: public.ecr.aws/nginx/nginx:latest
  Description: Container image to use for the task

TaskCpu:
  Type: String
  Default: "2048"
  Description: CPU units for the task

TaskMemory:
  Type: String
  Default: "4096"
  Description: Memory (in MiB) for the task

VolumeSize:
  Type: String
  Default: "10"
  Description: Size of the EBS volume in GiB

VolumeType:
  Type: String
  Default: gp3
  AllowedValues: [gp2, gp3, io1, io2, st1, sc1, standard]
  Description: EBS volume type

VolumeIops:
  Type: String
  Default: "3000"
  Description: IOPS for the EBS volume (required for io1, io2, and gp3)

VolumeThroughput:
  Type: String
  Default: "125"
  Description: Throughput for the EBS volume (only for gp3)

FilesystemType:
  Type: String
  Default: xfs
```

```
    AllowedValues: [xfs, ext4]
    Description: Filesystem type for the EBS volume

  EnableContainerInsights:
    Type: String
    Default: 'enhanced'
    AllowedValues: ['enabled', 'disabled', 'enhanced']
    Description: Enable or disable CloudWatch Container Insights for the cluster

Resources:
  # ECS Cluster
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ECSClusterName
      ClusterSettings:
        - Name: containerInsights
          Value: !Ref EnableContainerInsights
      Tags:
        - Key: Name
          Value: !Ref ECSClusterName

  # IAM Role for Task Execution
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy

  # Task Definition
  EBSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      Family: ebs-task-attach-task-def
      ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
      NetworkMode: awsvpc
      RequiresCompatibilities:
```

```
        - EC2
        - FARGATE
      Cpu: !Ref TaskCpu
      Memory: !Ref TaskMemory
      ContainerDefinitions:
        - Name: nginx
          Image: !Ref ContainerImage
          Essential: true
          PortMappings:
            - Name: nginx-80-tcp
              ContainerPort: 80
              HostPort: 80
              Protocol: tcp
              AppProtocol: http
          MountPoints:
            - SourceVolume: ebs-vol
              ContainerPath: /foo-container-path
              ReadOnly: false
      Volumes:
        - Name: ebs-vol
          ConfiguredAtLaunch: true

  ECSService:
    Type: AWS::ECS::Service
    Properties:
      Cluster: !Ref ECSCluster
      TaskDefinition: !Ref EBSTaskDefinition
      LaunchType: FARGATE
      ServiceName: ebs
      SchedulingStrategy: REPLICA
      DesiredCount: 1
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups: !Ref SecurityGroupIDs
          Subnets: !Ref SubnetIDs
      PlatformVersion: LATEST
      DeploymentConfiguration:
        MaximumPercent: 200
        MinimumHealthyPercent: 100
        DeploymentCircuitBreaker:
          Enable: true
          Rollback: true
      DeploymentController:
```

```
        Type: ECS
      Tags: []
      EnableECSManagedTags: true
      VolumeConfigurations:
        - Name: ebs-vol
          ManagedEBSVolume:
            RoleArn: !Ref InfrastructureRoleArn
            VolumeType: !Ref VolumeType
            Iops: !Ref VolumeIops
            Throughput: !Ref VolumeThroughput
            SizeInGiB: !Ref VolumeSize
            FilesystemType: !Ref FilesystemType
            TagSpecifications:
              - ResourceType: volume
                PropagateTags: TASK_DEFINITION

Outputs:
  ClusterName:
    Description: The cluster used to create the service.
    Value: !Ref ECSCluster
  ClusterArn:
    Description: The ARN of the ECS cluster
    Value: !GetAtt ECSCluster.Arn
  ECSService:
    Description: The created service.
    Value: !Ref ECSService
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref EBSTaskDefinition
```

Implantar serviço com provedores de capacidade

O modelo a seguir define um serviço que usa o provedor de capacidade para solicitar capacidade do AL2023 para execução. Os contêineres serão executados nas instâncias do AL2023 assim que se tornarem online.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "An example service that deploys in AWS VPC networking mode on
 EC2 capacity. Service uses a capacity provider to request EC2 instances to run
```

```
on. Service runs with networking in private subnets, but still accessible to the
internet via a load balancer hosted in public subnets.",
 "Parameters": {
     "VpcId": {
         "Type": "String",
         "Description": "The VPC that the service is running inside of"
     },
     "PublicSubnetIds": {
         "Type": "List<AWS::EC2::Subnet::Id>",
         "Description": "List of public subnet ID's to put the load balancer in"
     },
     "PrivateSubnetIds": {
         "Type": "List<AWS::EC2::Subnet::Id>",
         "Description": "List of private subnet ID's that the AWS VPC tasks are in"
     },
     "ClusterName": {
         "Type": "String",
         "Description": "The name of the ECS cluster into which to launch
capacity."
     },
     "ECSTaskExecutionRole": {
         "Type": "String",
         "Description": "The role used to start up an ECS task"
     },
     "CapacityProvider": {
         "Type": "String",
         "Description": "The cluster capacity provider that the service should use
to request capacity when it wants to start up a task"
     },
     "ServiceName": {
         "Type": "String",
         "Default": "web",
         "Description": "A name for the service"
     },
     "ImageUrl": {
         "Type": "String",
         "Default": "public.ecr.aws/docker/library/nginx:latest",
         "Description": "The url of a docker image that contains the application
process that will handle the traffic for this service"
     },
     "ContainerCpu": {
         "Type": "Number",
         "Default": 256,
         "Description": "How much CPU to give the container. 1024 is 1 CPU"
```

```
        },
        "ContainerMemory": {
            "Type": "Number",
            "Default": 512,
            "Description": "How much memory in megabytes to give the container"
        },
        "ContainerPort": {
            "Type": "Number",
            "Default": 80,
            "Description": "What port that the application expects traffic on"
        },
        "DesiredCount": {
            "Type": "Number",
            "Default": 2,
            "Description": "How many copies of the service task to run"
        }
    },
    "Resources": {
        "TaskDefinition": {
            "Type": "AWS::ECS::TaskDefinition",
            "Properties": {
                "Family": {
                    "Ref": "ServiceName"
                },
                "Cpu": {
                    "Ref": "ContainerCpu"
                },
                "Memory": {
                    "Ref": "ContainerMemory"
                },
                "NetworkMode": "awsvpc",
                "RequiresCompatibilities": [
                    "EC2"
                ],
                "ExecutionRoleArn": {
                    "Ref": "ECSTaskExecutionRole"
                },
                "ContainerDefinitions": [
                    {
                        "Name": {
                            "Ref": "ServiceName"
                        },
                        "Cpu": {
                            "Ref": "ContainerCpu"
```

```
                    },
                    "Memory": {
                        "Ref": "ContainerMemory"
                    },
                    "Image": {
                        "Ref": "ImageUrl"
                    },
                    "PortMappings": [
                        {
                            "ContainerPort": {
                                "Ref": "ContainerPort"
                            },
                            "HostPort": {
                                "Ref": "ContainerPort"
                            }
                        }
                    ],
                    "LogConfiguration": {
                        "LogDriver": "awslogs",
                        "Options": {
                            "mode": "non-blocking",
                            "max-buffer-size": "25m",
                            "awslogs-group": {
                                "Ref": "LogGroup"
                            },
                            "awslogs-region": {
                                "Ref": "AWS::Region"
                            },
                            "awslogs-stream-prefix": {
                                "Ref": "ServiceName"
                            }
                        }
                    }
                }
            ]
        }
    },
    "Service": {
        "Type": "AWS::ECS::Service",
        "DependsOn": "PublicLoadBalancerListener",
        "Properties": {
            "ServiceName": {
                "Ref": "ServiceName"
            },
```

```
                "Cluster": {
                    "Ref": "ClusterName"
                },
                "PlacementStrategies": [
                    {
                        "Field": "attribute:ecs.availability-zone",
                        "Type": "spread"
                    },
                    {
                        "Field": "cpu",
                        "Type": "binpack"
                    }
                ],
                "CapacityProviderStrategy": [
                    {
                        "Base": 0,
                        "CapacityProvider": {
                            "Ref": "CapacityProvider"
                        },
                        "Weight": 1
                    }
                ],
                "NetworkConfiguration": {
                    "AwsvpcConfiguration": {
                        "SecurityGroups": [
                            {
                                "Ref": "ServiceSecurityGroup"
                            }
                        ],
                        "Subnets": {
                            "Ref": "PrivateSubnetIds"
                        }
                    }
                },
                "DeploymentConfiguration": {
                    "MaximumPercent": 200,
                    "MinimumHealthyPercent": 75
                },
                "DesiredCount": {
                    "Ref": "DesiredCount"
                },
                "TaskDefinition": {
                    "Ref": "TaskDefinition"
                },
```

```
            "LoadBalancers": [
                {
                    "ContainerName": {
                        "Ref": "ServiceName"
                    },
                    "ContainerPort": {
                        "Ref": "ContainerPort"
                    },
                    "TargetGroupArn": {
                        "Ref": "ServiceTargetGroup"
                    }
                }
            ]
        }
    },
    "ServiceSecurityGroup": {
        "Type": "AWS::EC2::SecurityGroup",
        "Properties": {
            "GroupDescription": "Security group for service",
            "VpcId": {
                "Ref": "VpcId"
            }
        }
    },
    "ServiceTargetGroup": {
        "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
        "Properties": {
            "HealthCheckIntervalSeconds": 6,
            "HealthCheckPath": "/",
            "HealthCheckProtocol": "HTTP",
            "HealthCheckTimeoutSeconds": 5,
            "HealthyThresholdCount": 2,
            "TargetType": "ip",
            "Port": {
                "Ref": "ContainerPort"
            },
            "Protocol": "HTTP",
            "UnhealthyThresholdCount": 10,
            "VpcId": {
                "Ref": "VpcId"
            },
            "TargetGroupAttributes": [
                {
                    "Key": "deregistration_delay.timeout_seconds",
```

```
                    "Value": 0
                }
            ]
        }
    },
    "PublicLoadBalancerSG": {
        "Type": "AWS::EC2::SecurityGroup",
        "Properties": {
            "GroupDescription": "Access to the public facing load balancer",
            "VpcId": {
                "Ref": "VpcId"
            },
            "SecurityGroupIngress": [
                {
                    "CidrIp": "0.0.0.0/0",
                    "IpProtocol": -1
                }
            ]
        }
    },
    "PublicLoadBalancer": {
        "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
        "Properties": {
            "Scheme": "internet-facing",
            "LoadBalancerAttributes": [
                {
                    "Key": "idle_timeout.timeout_seconds",
                    "Value": "30"
                }
            ],
            "Subnets": {
                "Ref": "PublicSubnetIds"
            },
            "SecurityGroups": [
                {
                    "Ref": "PublicLoadBalancerSG"
                }
            ]
        }
    },
    "PublicLoadBalancerListener": {
        "Type": "AWS::ElasticLoadBalancingV2::Listener",
        "Properties": {
            "DefaultActions": [
```

```
                {
                    "Type": "forward",
                    "ForwardConfig": {
                        "TargetGroups": [
                            {
                                "TargetGroupArn": {
                                    "Ref": "ServiceTargetGroup"
                                },
                                "Weight": 100
                            }
                        ]
                    }
                }
            ],
            "LoadBalancerArn": {
                "Ref": "PublicLoadBalancer"
            },
            "Port": 80,
            "Protocol": "HTTP"
        }
    },
    "ServiceIngressfromLoadBalancer": {
        "Type": "AWS::EC2::SecurityGroupIngress",
        "Properties": {
            "Description": "Ingress from the public ALB",
            "GroupId": {
                "Ref": "ServiceSecurityGroup"
            },
            "IpProtocol": -1,
            "SourceSecurityGroupId": {
                "Ref": "PublicLoadBalancerSG"
            }
        }
    },
    "LogGroup": {
        "Type": "AWS::Logs::LogGroup"
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
```

```
Description: >-
  An example service that deploys in AWS VPC networking mode on EC2 capacity.
  Service uses a capacity provider to request EC2 instances to run on. Service
  runs with networking in private subnets, but still accessible to the internet
  via a load balancer hosted in public subnets.
Parameters:
  VpcId:
    Type: String
    Description: The VPC that the service is running inside of
  PublicSubnetIds:
    Type: 'List<AWS::EC2::Subnet::Id>'
    Description: List of public subnet ID's to put the load balancer in
  PrivateSubnetIds:
    Type: 'List<AWS::EC2::Subnet::Id>'
    Description: List of private subnet ID's that the AWS VPC tasks are in
  ClusterName:
    Type: String
    Description: The name of the ECS cluster into which to launch capacity.
  ECSTaskExecutionRole:
    Type: String
    Description: The role used to start up an ECS task
  CapacityProvider:
    Type: String
    Description: >-
      The cluster capacity provider that the service should use to request
      capacity when it wants to start up a task
  ServiceName:
    Type: String
    Default: web
    Description: A name for the service
  ImageUrl:
    Type: String
    Default: 'public.ecr.aws/docker/library/nginx:latest'
    Description: >-
      The url of a docker image that contains the application process that will
      handle the traffic for this service
  ContainerCpu:
    Type: Number
    Default: 256
    Description: How much CPU to give the container. 1024 is 1 CPU
  ContainerMemory:
    Type: Number
    Default: 512
    Description: How much memory in megabytes to give the container
```

```
  ContainerPort:
    Type: Number
    Default: 80
    Description: What port that the application expects traffic on
  DesiredCount:
    Type: Number
    Default: 2
    Description: How many copies of the service task to run
Resources:
  TaskDefinition:
    Type: 'AWS::ECS::TaskDefinition'
    Properties:
      Family: !Ref ServiceName
      Cpu: !Ref ContainerCpu
      Memory: !Ref ContainerMemory
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - EC2
      ExecutionRoleArn: !Ref ECSTaskExecutionRole
      ContainerDefinitions:
        - Name: !Ref ServiceName
          Cpu: !Ref ContainerCpu
          Memory: !Ref ContainerMemory
          Image: !Ref ImageUrl
          PortMappings:
            - ContainerPort: !Ref ContainerPort
              HostPort: !Ref ContainerPort
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-group: !Ref LogGroup
              awslogs-region: !Ref AWS::Region
              awslogs-stream-prefix: !Ref ServiceName
  Service:
    Type: AWS::ECS::Service
    DependsOn: PublicLoadBalancerListener
    Properties:
      ServiceName: !Ref ServiceName
      Cluster: !Ref ClusterName
      PlacementStrategies:
        - Field: 'attribute:ecs.availability-zone'
          Type: spread
```

```
        - Field: cpu
          Type: binpack
      CapacityProviderStrategy:
        - Base: 0
          CapacityProvider: !Ref CapacityProvider
          Weight: 1
      NetworkConfiguration:
        AwsvpcConfiguration:
          SecurityGroups:
            - !Ref ServiceSecurityGroup
          Subnets: !Ref PrivateSubnetIds
      DeploymentConfiguration:
        MaximumPercent: 200
        MinimumHealthyPercent: 75
      DesiredCount: !Ref DesiredCount
      TaskDefinition: !Ref TaskDefinition
      LoadBalancers:
        - ContainerName: !Ref ServiceName
          ContainerPort: !Ref ContainerPort
          TargetGroupArn: !Ref ServiceTargetGroup
  ServiceSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: Security group for service
      VpcId: !Ref VpcId
  ServiceTargetGroup:
    Type: 'AWS::ElasticLoadBalancingV2::TargetGroup'
    Properties:
      HealthCheckIntervalSeconds: 6
      HealthCheckPath: /
      HealthCheckProtocol: HTTP
      HealthCheckTimeoutSeconds: 5
      HealthyThresholdCount: 2
      TargetType: ip
      Port: !Ref ContainerPort
      Protocol: HTTP
      UnhealthyThresholdCount: 10
      VpcId: !Ref VpcId
      TargetGroupAttributes:
        - Key: deregistration_delay.timeout_seconds
          Value: 0
  PublicLoadBalancerSG:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
```

```
      GroupDescription: Access to the public facing load balancer
      VpcId: !Ref VpcId
      SecurityGroupIngress:
        - CidrIp: 0.0.0.0/0
          IpProtocol: -1
  PublicLoadBalancer:
    Type: 'AWS::ElasticLoadBalancingV2::LoadBalancer'
    Properties:
      Scheme: internet-facing
      LoadBalancerAttributes:
        - Key: idle_timeout.timeout_seconds
          Value: '30'
      Subnets: !Ref PublicSubnetIds
      SecurityGroups:
        - !Ref PublicLoadBalancerSG
  PublicLoadBalancerListener:
    Type: 'AWS::ElasticLoadBalancingV2::Listener'
    Properties:
      DefaultActions:
        - Type: forward
          ForwardConfig:
            TargetGroups:
              - TargetGroupArn: !Ref ServiceTargetGroup
                Weight: 100
      LoadBalancerArn: !Ref PublicLoadBalancer
      Port: 80
      Protocol: HTTP
  ServiceIngressfromLoadBalancer:
    Type: 'AWS::EC2::SecurityGroupIngress'
    Properties:
      Description: Ingress from the public ALB
      GroupId: !Ref ServiceSecurityGroup
      IpProtocol: -1
      SourceSecurityGroupId: !Ref PublicLoadBalancerSG
  LogGroup:
    Type: 'AWS::Logs::LogGroup'
```

Perfis do IAM para o Amazon ECS

É possível usar modelos do AWS CloudFormation para criar perfis do IAM para uso com o Amazon ECS. Para obter mais informações, consulte Perfis do IAM para Amazon ECS, consulte Perfis do IAM para o Amazon ECS.

Perfil para execução de tarefas do Amazon ECS

A função de execução de tarefas concede ao contêiner do Amazon ECS e aos agentes do Fargate permissão para fazer chamadas da API da AWS em seu nome. Dependendo dos requisitos da sua tarefa, o perfil pode ser necessário. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.

O modelo a seguir pode ser usado para criar um perfil de execução de tarefa simples que usa a política gerenciada `AmazonECSTaskExecutionRolePolicy`.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "CloudFormation template for ECS Task Execution Role",
  "Resources": {
    "ECSTaskExecutionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": ["ecs-tasks.amazonaws.com"]
              },
              "Action": ["sts:AssumeRole"],
              "Condition": {
                "ArnLike": {
                  "aws:SourceArn": {
                    "Fn::Sub": "arn:aws:ecs:${AWS::Region}:${AWS::AccountId}:*"
                  }
                },
                "StringEquals": {
                  "aws:SourceAccount": {
                    "Ref": "AWS::AccountId"
                  }
                }
              }
            }
          ]
        },
        "Path": "/",
```

```
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
        ]
      }
    }
  },
  "Outputs": {
    "ECSTaskExecutionRoleARN": {
      "Description": "ARN of the ECS Task Execution Role",
      "Value": {
        "Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]
      },
      "Export": {
        "Name": {
          "Fn::Sub": "${AWS::StackName}-ECSTaskExecutionRoleARN"
        }
      }
    },
    "ECSTaskExecutionRoleName": {
      "Description": "Name of the ECS Task Execution Role",
      "Value": {
        "Ref": "ECSTaskExecutionRole"
      },
      "Export": {
        "Name": {
          "Fn::Sub": "${AWS::StackName}-ECSTaskExecutionRoleName"
        }
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: 'CloudFormation template for ECS Task Execution Role'
Resources:
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Effect: Allow
```

```
            Principal:
              Service: [ecs-tasks.amazonaws.com]
            Action: ['sts:AssumeRole']
            Condition:
              ArnLike:
                aws:SourceArn: !Sub arn:aws:ecs:${AWS::Region}:${AWS::AccountId}:*
              StringEquals:
                aws:SourceAccount: !Ref AWS::AccountId
      Path: /
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
Outputs:
  ECSTaskExecutionRoleARN:
    Description: ARN of the ECS Task Execution Role
    Value: !GetAtt ECSTaskExecutionRole.Arn
    Export:
      Name: !Sub "${AWS::StackName}-ECSTaskExecutionRoleARN"
  ECSTaskExecutionRoleName:
    Description: Name of the ECS Task Execution Role
    Value: !Ref ECSTaskExecutionRole
    Export:
      Name: !Sub "${AWS::StackName}-ECSTaskExecutionRoleName"
```

Criar recursos do Amazon ECS usando o console do AWS CloudFormation

Uma forma de usar o Amazon ECS com o AWS CloudFormation é por meio do AWS Management Console. Aqui você pode criar suas pilhas do AWS CloudFormation para componentes do Amazon ECS, como definições de tarefas, clusters e serviços, e implantá-los diretamente via console. O tutorial a seguir mostra como usar o console do AWS CloudFormation para criar um serviço, uma definição de tarefa e um cluster do Amazon ECS.

Pré-requisitos

Esse tutorial pressupõe que os pré-requisitos a seguir foram concluídos.

- As etapas em Configuração para usar o Amazon ECS foram concluídas.
- O usuário do IAM tem as permissões necessárias especificadas no exemplo de política do IAM de AmazonECS_FullAccess.

Etapa 1: criar um modelo de pilha

Use as etapas a seguir para criar um modelo de pilha do AWS CloudFormation para um serviço e outros recursos relacionados do Amazon ECS.

1. Usando um editor de texto qualquer, crie um arquivo chamado `ecs-tutorial-template.yaml`.
2. No arquivo `ecs-tutorial-template.yaml`, cole o modelo a seguir e salve as alterações.

```
AWSTemplateFormatVersion: 2010-09-09
Description: A template that deploys an application that is built on an Apache
 web server Docker image by creating an Amazon ECS cluster, task definition, and
 service. The template also creates networking and logging resources, and an Amazon
 ECS task execution role.
Parameters:
  ClusterName:
    Type: String
    Default: CFNCluster
    Description: Name of the ECS Cluster
  TaskFamily:
    Type: String
    Default: task-definition-cfn
    Description: Family name for the Task Definition
  ServiceName:
    Type: String
    Default: cfn-service
    Description: Name of the ECS Service
  ContainerImage:
    Type: String
    Default: public.ecr.aws/docker/library/httpd:2.4
    Description: Container image to use for the task
  TaskCpu:
    Type: Number
    Default: 256
    AllowedValues: [256, 512, 1024, 2048, 4096]
    Description: CPU units for the task
  TaskMemory:
    Type: Number
    Default: 512
    AllowedValues: [512, 1024, 2048, 4096, 8192, 16384]
    Description: Memory (in MiB) for the task
  DesiredCount:
```

```
    Type: Number
    Default: 1
    Description: Desired number of tasks to run
  LogGroupName:
    Type: String
    Default: /ecs/fargate-task-definition
    Description: CloudWatch Log Group name
  VpcCidr:
    Type: String
    Default: 10.0.0.0/16
    Description: CIDR block for the VPC
  PublicSubnet1Cidr:
    Type: String
    Default: 10.0.0.0/24
    Description: CIDR block for public subnet 1
  PublicSubnet2Cidr:
    Type: String
    Default: 10.0.1.0/24
    Description: CIDR block for public subnet 2
Resources:
  # VPC and Networking Resources
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCidr
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-VPC
  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-IGW
  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref VPC
  PublicSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
```

```
      VpcId: !Ref VPC
      AvailabilityZone: !Select [0, !GetAZs '']
      CidrBlock: !Ref PublicSubnet1Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicSubnet1
  PublicSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [1, !GetAZs '']
      CidrBlock: !Ref PublicSubnet2Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicSubnet2
  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicRouteTable
  DefaultPublicRoute:
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway
  PublicSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet1
  PublicSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet2
  # Security Group
  ECSSecurityGroup:
    Type: AWS::EC2::SecurityGroup
```

```
    Properties:
      GroupDescription: Security group for ECS tasks
      VpcId: !Ref VPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 443
          ToPort: 443
          CidrIp: 0.0.0.0/0
  # IAM Roles
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
  # CloudWatch Logs
  TaskLogGroup:
    Type: AWS::Logs::LogGroup
    DeletionPolicy: Retain
    UpdateReplacePolicy: Retain
    Properties:
      LogGroupName: !Ref LogGroupName
      RetentionInDays: 30
  # ECS Resources
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ClusterName
  ECSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
```

```
            /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
            App</title> <style>body {margin-top: 40px; background-color:
            #333;} </style> </head><body> <div
            style=color:white;text-align:center> <h1>Amazon ECS Sample
            App</h1> <h2>Congratulations!</h2> <p>Your application is now
            running on a container in Amazon ECS.</p> </div></body></html>' >
            /usr/local/apache2/htdocs/index.html && httpd-foreground"s
        EntryPoint:
          - sh
          - '-c'
        Essential: true
        Image: !Ref ContainerImage
        LogConfiguration:
          LogDriver: awslogs
          Options:
            mode: non-blocking
            max-buffer-size: 25m
            awslogs-create-group: 'true'
            awslogs-group: !Ref LogGroupName
            awslogs-region: !Ref 'AWS::Region'
            awslogs-stream-prefix: ecs
        Name: sample-fargate-app
        PortMappings:
          - ContainerPort: 80
            HostPort: 80
            Protocol: tcp
    Cpu: !Ref TaskCpu
    ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
    Family: !Ref TaskFamily
    Memory: !Ref TaskMemory
    NetworkMode: awsvpc
    RequiresCompatibilities:
      - FARGATE
    RuntimePlatform:
      OperatingSystemFamily: LINUX
ECSService:
  Type: AWS::ECS::Service
  DependsOn:
    - PublicSubnet1RouteTableAssociation
    - PublicSubnet2RouteTableAssociation
  Properties:
    ServiceName: !Ref ServiceName
    Cluster: !Ref ECSCluster
    DesiredCount: !Ref DesiredCount
```

```
      LaunchType: FARGATE
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups:
            - !Ref ECSSecurityGroup
          Subnets:
            - !Ref PublicSubnet1
            - !Ref PublicSubnet2
      TaskDefinition: !Ref ECSTaskDefinition
Outputs:
  ClusterName:
    Description: The name of the ECS cluster
    Value: !Ref ECSCluster
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref ECSTaskDefinition
  ServiceName:
    Description: The name of the ECS service
    Value: !Ref ECSService
  VpcId:
    Description: The ID of the VPC
    Value: !Ref VPC
  PublicSubnet1:
    Description: The ID of public subnet 1
    Value: !Ref PublicSubnet1
  PublicSubnet2:
    Description: The ID of public subnet 2
    Value: !Ref PublicSubnet2
  SecurityGroup:
    Description: The ID of the security group
    Value: !Ref ECSSecurityGroup
  ExecutionRoleArn:
    Description: The ARN of the task execution role
    Value: !GetAtt ECSTaskExecutionRole.Arn
```

Etapa 2: criar uma pilha para recursos do Amazon ECS

Depois de criar um arquivo para o modelo, siga estas etapas para criar uma pilha com o modelo usando o console do AWS CloudFormation.

1. Faça login no AWS Management Console e abra o console AWS CloudFormation em https://console.aws.amazon.com/cloudformation.
2. Na página Pilhas, no canto superior direito, escolha Criar pilha e, em seguida, selecione Com novos recursos (padrão).
3. Escolha Escolher um modelo existente.
4. Escolha Carregar um arquivo de modelo e, em seguida, Escolher arquivo para selecionar o arquivo `ecs-tutorial-template`.

   Depois que o arquivo for carregado em um bucket do Amazon S3, você poderá escolher Visualizar no Infrastructure Composer para ver o modelo no Infrastructure Composer. Para obter mais informações sobre modelos do AWS CloudFormation e o Infrastructure Composer, consulte Criar modelos visualmente com o Infrastructure Composer no Guia do usuário do AWS CloudFormation.
5. Escolha Próximo.
6. Na página Especificar detalhes da pilha, em Nome da pilha, forneça o seguinte nome para a pilha: `ecs-tutorial-stack`. Mantenha todos os valores dos parâmetros em Parâmetros como padrão e escolha Avançar.
7. Na página Configurar opções da pilha, em Capacidades, marque a caixa de seleção para confirmar a criação de recursos do IAM com o AWS CloudFormation. Essa confirmação é necessária para criar a função de execução de tarefas do Amazon ECS, conforme definido no modelo. Mantenha todas as outras configurações em seus valores padrão e escolha Próximo.
8. Revise os detalhes da pilha na página Revisar e criar e escolha Enviar para iniciar a criação da pilha.

Etapa 3: verificar

Use as etapas a seguir para verificar a criação de recursos do Amazon ECS usando o modelo fornecido.

1. Faça login no AWS Management Console e abra o console AWS CloudFormation em https://console.aws.amazon.com/cloudformation.
2. Na página Pilhas, escolha ecs-tutorial-stack.
3. Escolha a guia Eventos. Se o status do evento indicar `CREATE_IN_PROGRESS`, espere até que a criação seja concluída e os status mudem para `CREATE_COMPLETE`.

4. Depois que os status dos eventos mudarem para `CREATE_COMPLETE`, escolha a guia Recursos. Você verá recursos com o ID lógico `ECSCluster`, `ECSTaskDefinition` e `ECSService` respectivamente.
5. Para verificar a criação de um cluster do Amazon ECS, escolha o ID físico associado ao `ECSCluster`. Você será redirecionado para o console do Amazon ECS, onde poderá ver o cluster criado chamado `CFNCluster`.
6. Para verificar a criação de um serviço do Amazon ECS, escolha o ID físico associado ao `ECSService`. Você será redirecionado para o console do Amazon ECS, onde poderá ver o serviço chamado `cfn-service` que foi criado no cluster `cfnCluster`.
7. Para verificar a criação de uma definição de tarefa do Amazon ECS, escolha o ID físico associado ao `ECSTaskDefinition`. Você será redirecionado para o console do Amazon ECS, onde poderá ver a revisão da definição da tarefa com o nome `task-definition-cfn`.

Etapa 4: Limpar os recursos

Para limpar recursos e evitar custos adicionais, siga estas etapas.

1. Faça login no AWS Management Console e abra o console AWS CloudFormation em https://console.aws.amazon.com/cloudformation.
2. Na página Pilhas, escolha ecs-tutorial-stack.
3. Escolha Excluir.
4. Quando a confirmação for solicitada, escolha Excluir novamente.
5. Escolha a guia Eventos. O Status de `ecs-tutorial-stack` muda para DELETE_IN_PROGRESS e, em seguida, para DELETE_COMPLETE após a exclusão ou cancelamento do registro dos recursos. A exclusão leva alguns minutos.
6. Escolha a guia Recursos. Agora você verá uma lista de ID lógico com o Status atualizado para DELETE_COMPLETE.

Criar recursos do Amazon ECS usando comandos da AWS CLI para o AWS CloudFormation

O tutorial a seguir mostra como você pode usar o AWS CLI para criar recursos do Amazon ECS com um modelo do AWS CloudFormation.

Pré-requisitos

- As etapas em Configuração para usar o Amazon ECS foram concluídas.
- O usuário do IAM tem as permissões necessárias especificadas no exemplo de política do IAM de AmazonECS_FullAccess.

Etapa 1: criar uma pilha

Para criar um novo estágio usando a AWS CLI, siga estas etapas.

1. Usando um editor de texto qualquer, crie um arquivo chamado `ecs-tutorial-template.yaml`.
2. No arquivo `ecs-tutorial-template.yaml`, cole o modelo a seguir e salve as alterações.

```
AWSTemplateFormatVersion: 2010-09-09
Description: A template that deploys an application that is built on an Apache
 web server Docker image by creating an Amazon ECS cluster, task definition, and
 service. The template also creates networking and logging resources, and an Amazon
 ECS task execution role.
Parameters:
  ClusterName:
    Type: String
    Default: CFNCluster
    Description: Name of the ECS Cluster
  TaskFamily:
    Type: String
    Default: task-definition-cfn
    Description: Family name for the Task Definition
  ServiceName:
    Type: String
    Default: cfn-service
    Description: Name of the ECS Service
  ContainerImage:
    Type: String
    Default: public.ecr.aws/docker/library/httpd:2.4
    Description: Container image to use for the task
  TaskCpu:
    Type: Number
    Default: 256
    AllowedValues: [256, 512, 1024, 2048, 4096]
    Description: CPU units for the task
```

```
  TaskMemory:
    Type: Number
    Default: 512
    AllowedValues: [512, 1024, 2048, 4096, 8192, 16384]
    Description: Memory (in MiB) for the task
  DesiredCount:
    Type: Number
    Default: 1
    Description: Desired number of tasks to run
  LogGroupName:
    Type: String
    Default: /ecs/fargate-task-definition
    Description: CloudWatch Log Group name
  VpcCidr:
    Type: String
    Default: 10.0.0.0/16
    Description: CIDR block for the VPC
  PublicSubnet1Cidr:
    Type: String
    Default: 10.0.0.0/24
    Description: CIDR block for public subnet 1
  PublicSubnet2Cidr:
    Type: String
    Default: 10.0.1.0/24
    Description: CIDR block for public subnet 2
Resources:
  # VPC and Networking Resources
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCidr
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-VPC
  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-IGW
  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
```

```
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref VPC
  PublicSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [0, !GetAZs '']
      CidrBlock: !Ref PublicSubnet1Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicSubnet1
  PublicSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [1, !GetAZs '']
      CidrBlock: !Ref PublicSubnet2Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicSubnet2
  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicRouteTable
  DefaultPublicRoute:
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway
  PublicSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet1
  PublicSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
```

```
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet2
  # Security Group
  ECSSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Security group for ECS tasks
      VpcId: !Ref VPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 443
          ToPort: 443
          CidrIp: 0.0.0.0/0
  # IAM Roles
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
  # CloudWatch Logs
  TaskLogGroup:
    Type: AWS::Logs::LogGroup
    DeletionPolicy: Retain
    UpdateReplacePolicy: Retain
    Properties:
      LogGroupName: !Ref LogGroupName
      RetentionInDays: 30
  # ECS Resources
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ClusterName
```

```
  ECSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
              /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
              App</title> <style>body {margin-top: 40px; background-color:
              #333;} </style> </head><body> <div
              style=color:white;text-align:center> <h1>Amazon ECS Sample
              App</h1> <h2>Congratulations!</h2> <p>Your application is now
              running on a container in Amazon ECS.</p> </div></body></html>' >
              /usr/local/apache2/htdocs/index.html && httpd-foreground"s
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: !Ref ContainerImage
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Ref LogGroupName
              awslogs-region: !Ref 'AWS::Region'
              awslogs-stream-prefix: ecs
          Name: sample-fargate-app
          PortMappings:
            - ContainerPort: 80
              HostPort: 80
              Protocol: tcp
      Cpu: !Ref TaskCpu
      ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
      Family: !Ref TaskFamily
      Memory: !Ref TaskMemory
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: LINUX
  ECSService:
    Type: AWS::ECS::Service
    DependsOn:
```

```
      - PublicSubnet1RouteTableAssociation
      - PublicSubnet2RouteTableAssociation
    Properties:
      ServiceName: !Ref ServiceName
      Cluster: !Ref ECSCluster
      DesiredCount: !Ref DesiredCount
      LaunchType: FARGATE
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups:
            - !Ref ECSSecurityGroup
          Subnets:
            - !Ref PublicSubnet1
            - !Ref PublicSubnet2
      TaskDefinition: !Ref ECSTaskDefinition
Outputs:
  ClusterName:
    Description: The name of the ECS cluster
    Value: !Ref ECSCluster
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref ECSTaskDefinition
  ServiceName:
    Description: The name of the ECS service
    Value: !Ref ECSService
  VpcId:
    Description: The ID of the VPC
    Value: !Ref VPC
  PublicSubnet1:
    Description: The ID of public subnet 1
    Value: !Ref PublicSubnet1
  PublicSubnet2:
    Description: The ID of public subnet 2
    Value: !Ref PublicSubnet2
  SecurityGroup:
    Description: The ID of the security group
    Value: !Ref ECSSecurityGroup
  ExecutionRoleArn:
    Description: The ARN of the task execution role
    Value: !GetAtt ECSTaskExecutionRole.Arn
```

3. Depois de criar um arquivo de modelo, use o comando a seguir para criar uma pilha. O sinalizador `--capabilities` é necessário para criar um perfil de execução de tarefa do

Amazon ECS, conforme especificado no modelo. Você também pode especificar o sinalizador `--parameters` para personalizar os parâmetros do modelo.

```
aws cloudformation create-stack \
      --stack-name ecs-stack \
      --template-body file://ecs-tutorial-template.yaml \
      --capabilities CAPABILITY_IAM
```

Etapa 2: Verificar a criação do recurso

Para garantir que os recursos sejam criados corretamente, siga estas etapas. Você também pode verificar o console do Amazon ECS.

1. Execute o comando a seguir para listar todas as definições de tarefa em uma Região da AWS.

   ```
   aws ecs list-task-definitions
   ```

   O comando retorna uma lista de nomes do recurso da Amazon (ARN) de definições de tarefas. O ARN da definição de tarefa que você criou usando o modelo será exibido no formato a seguir.

   ```
   {
       "taskDefinitionArns": [
        .....
           "arn:aws:ecs:aws-region:111122223333:task-definition/task-definition-
   cfn:1",
        .....
       ]
   }
   ```

2. Execute o comando a seguir para listar todos os clusters em uma Região da AWS.

   ```
   aws ecs list-clusters
   ```

   O comando retorna uma lista de ARNs de cluster. O ARN do cluster que você criou usando o modelo será exibido no formato a seguir.

   ```
   {
       "clusterArns": [
           .....
   ```

```
        "arn:aws:ecs:aws-region:111122223333:cluster/CFNCluster",
        .....
    ]
}
```

3. Execute o comando a seguir para listar todos os serviços em execução no cluster `CFNCluster`.

```
aws ecs list-services \
      --cluster CFNCluster
```

O comando retorna uma lista de ARNs de serviço. O ARN do serviço que você criou usando o modelo será exibido no formato a seguir.

```
{
    "serviceArns": [
        "arn:aws:ecs:aws-region:111122223333:service/CFNCluster/cfn-service"
    ]
}
```

Etapa 3: Limpeza

Para limpar os recursos que você criou, execute o comando a seguir.

```
aws cloudformation delete-stack \
      --stack-name ecs-stack
```

O comando `delete-stack` inicia a exclusão da pilha AWS CloudFormation que foi criada neste tutorial e exclui todos os recursos na pilha. Para verificar a exclusão, repita o procedimento em Etapa 2: Verificar a criação do recurso. A lista de ARNs nas saídas não incluirá mais uma definição de tarefa chamada `task-definition-cfn` ou um cluster chamado `CFNCluster`. A chamada `list-services` falhará.

Práticas recomendadas do Amazon ECS

Você pode usar qualquer uma das páginas a seguir para aprender as práticas operacionais recomendadas mais importantes para a rede do Amazon ECS.

Visão geral das práticas recomendadas	Saiba mais
Conexão de aplicações à Internet	Conexão de aplicações do Amazon ECS à Internet
Receba conexões de entrada para o Amazon ECS vindas da Internet.	Práticas recomendadas para receber conexões de entrada para o Amazon ECS vindas da Internet
Conecte o Amazon ECS a outros serviços da AWS dentro de uma VPC	Práticas recomendadas para conectar o Amazon ECS a serviços da AWS de dentro da sua VPC
Serviços de rede em Contas da AWS e VPCs	Práticas recomendadas para conectar serviços do Amazon ECS em rede entre Contas da AWS e VPCs
Corrigir problemas de rede	Serviços da AWS para solução de problemas de rede do Amazon ECS

Você pode usar qualquer uma das páginas a seguir para aprender as práticas operacionais recomendadas mais importantes para o Fargate no Amazon ECS.

Visão geral das práticas recomendadas	Saiba mais	
Segurança do Fargate	Práticas recomendadas de segurança do Fargate no Amazon ECS	
Considerações de segurança do Fargate	Considerações de segurança do Fargate para o Amazon ECS	
Contêineres do Linux no comportamento de extração de imagem de contêiner do Fargate	Comportamento dos contêineres Linux na extração de imagens de contêiner do Fargate para o Amazon ECS	
Contêineres do Windows no comportamento de extração de imagem de contêiner do Fargate	Comportamento dos contêineres Windows na extração de imagens de contêiner do Fargate para o Amazon ECS	
Retirada de tarefas do Fargate	Retirada e manutenção de tarefas para o AWS Fargate no Amazon ECS	

Você pode usar qualquer uma das páginas a seguir para aprender as práticas operacionais recomendadas mais importantes para definições de tarefas.

Visão geral das práticas recomendadas	Saiba mais	
Imagens de contêiner	Práticas recomendadas para imagens de contêiner do Amazon ECS	

Visão geral das práticas recomendadas	Saiba mais	
Tamanho da tarefa	Práticas recomendadas para tamanhos de tarefas do Amazon ECS	
Práticas recomendadas de volume	Opções de armazenamento para tarefas do Amazon ECS	

Você pode usar qualquer uma das páginas a seguir para aprender as práticas operacionais recomendadas mais importantes para clusters e capacidade.

Visão geral das práticas recomendadas	Saiba mais	
Segurança do Fargate	Práticas recomendadas de segurança do Fargate no Amazon ECS	
Considerações de segurança sobre a instância de contêiner do EC2	Considerações de segurança sobre a instância de contêiner do Amazon EC2 para o Amazon ECS	
Autoescalabilidade de cluster	Otimização do ajuste de escala automático de cluster do Amazon ECS	
Operação em grande escala	Operação do Amazon ECS em grande escala	
Gerenciamento de capacidade e ajuste de escala automático	Práticas recomendadas de gerenciamento de capacidade e ajuste de escala automático do Amazon ECS	

Você pode usar qualquer uma das páginas a seguir para aprender as práticas operacionais recomendadas mais importantes para tarefas e serviços.

Visão geral das práticas recomendadas	Saiba mais
Otimização do tempo de inicialização da tarefa	Otimização do tempo de inicialização da tarefa do Amazon ECS
Parâmetros do serviço	Práticas recomendadas para parâmetros de serviço do Amazon ECS
Otimização dos parâmetros de verificação de integridade do balanceador de carga	Otimização dos parâmetros de verificação de integridade do balanceador de carga para o Amazon ECS
Otimização de parâmetros de drenagem da conexão do balanceador de carga	Otimização de parâmetros de drenagem da conexão do balanceador de carga para o Amazon ECS
Otimização do ajuste de escala automático do serviço	Otimização do ajuste de escala automático de serviços do Amazon ECS

Você pode usar qualquer uma das páginas a seguir para aprender as práticas operacionais recomendadas mais importantes para segurança.

Visão geral das práticas recomendadas	Saiba mais
Segurança de rede	Práticas recomendadas de segurança de rede para o Amazon ECS

Visão geral das práticas recomendadas	Saiba mais	
Segurança de tarefas e contêineres	Práticas recomendadas de segurança para tarefas e contêineres do Amazon ECS	

AWS Fargate para o Amazon ECS

O AWS Fargate é uma tecnologia que pode ser usada com o Amazon ECS para executar contêineres sem a necessidade de gerenciar servidores ou clusters de instâncias do Amazon EC2. Com o AWS Fargate, não é mais necessário provisionar, configurar nem dimensionar os clusters de máquinas virtuais para executar contêineres. Isso elimina a necessidade de escolher tipos de servidor, decidir quando dimensionar clusters ou otimizar o agrupamento de clusters.

Ao executar suas tarefas e serviços com o tipo de inicialização do Fargate, você empacota sua aplicação em contêineres, especifica os requisitos de CPU e de memória, define as políticas de rede e do IAM e inicia a aplicação. Cada tarefa do Fargate tem seu próprio limite de isolamento e não compartilha o kernel subjacente, os recursos de CPU, os recursos de memória nem a interface de rede elástica com outra tarefa. Você configura suas definições de tarefas para o Fargate configurando o parâmetro de definição de tarefa `requiresCompatibilities` como `FARGATE`. Para obter mais informações, consulte Tipos de inicialização.

O Fargate oferece versões da plataforma para o Amazon Linux 2 (1.3.0), sistema operacional Bottlerocket (1.4.0) e as edições Full e Core do Microsoft Windows Server 2019. A menos que especificado de outra maneira, as informações nesta página são aplicáveis a todas as plataformas Fargate.

Este tópico descreve os diferentes componentes das tarefas e serviços do Fargate e identifica as considerações especiais para usar o Fargate com o Amazon ECS.

Para obter informações sobre as regiões que oferecem suporte a contêineres de Linux no Fargate, consulte the section called “Contêineres do Linux no AWS Fargate”.

Para obter informações sobre as regiões que oferecem suporte a contêineres de Windows no Fargate, consulte the section called “Contêineres do Windows no AWS Fargate”.

Instruções

Para obter informações sobre como começar a usar o console, consulte:

- Saiba como criar uma tarefa do Linux no Amazon ECS para o tipo de inicialização do Fargate
- Saiba como criar uma tarefa do Windows no Amazon ECS para o tipo de inicialização do Fargate

Para obter informações sobre como começar a usar a AWS CLI, consulte:

- Criar uma tarefa do Linux no Amazon ECS para o tipo de inicialização do Fargate com a AWS CLI
- Criar uma tarefa do Windows no Amazon ECS para o tipo de inicialização do Fargate com a AWS CLI

Provedores de capacidade

Os seguintes provedores de capacidade estão disponíveis:

- Fargate
- Fargate Spot: executa tarefas tolerantes a interrupções do Amazon ECS com uma taxa de desconto em comparação ao preço do AWS Fargate. O Fargate Spot executa tarefas com capacidade adicional de computação. Quando a AWS precisar da capacidade de volta, suas tarefas serão interrompidas com um aviso de dois minutos. Para obter mais informações, consulte Clusters do Amazon ECS para Fargate.

Definições de tarefa

As tarefas que usam o tipo de inicialização do Fargate não são compatíveis com todos os parâmetros de definição de tarefa do Amazon ECS disponíveis. Alguns parâmetros são totalmente incompatíveis e outros se comportam de maneira diferente nas tarefas do Fargate. Para obter mais informações, consulte CPU e memória da tarefa.

Versões da plataforma

As versões da plataforma do AWS Fargate são usadas para fazer referência a um ambiente de runtime para a infraestrutura de tarefas do Fargate. Trata-se de uma combinação da versão do kernel e do runtime do contêiner. Você seleciona uma versão da plataforma ao executar uma tarefa ou ao criar um serviço para manter várias tarefas idênticas.

Novas revisões de versões da plataforma são lançadas conforme o ambiente do runtime evolui, por exemplo, em caso de atualizações no kernel ou no sistema operacional, novos recursos, correções de erros ou atualizações de segurança. Uma versão da plataforma Fargate é atualizada por meio de uma nova revisão da versão da plataforma. Cada tarefa é executada em uma revisão de versão da plataforma durante seu ciclo de vida. Se você quiser usar a revisão mais recente da versão da plataforma, será necessário iniciar uma nova tarefa. Uma nova tarefa executada no Fargate sempre

é executada na revisão mais recente de uma versão da plataforma, garantindo que as tarefas sejam sempre iniciadas em uma infraestrutura segura e corrigida.

Se houver um problema de segurança que afete uma versão existente da plataforma, a AWS vai criar uma nova revisão corrigida da versão da plataforma e retirar as tarefas em execução na revisão vulnerável. Em alguns casos, será possível receber notificações de que suas tarefas no Fargate foram programadas para retirada. Para obter mais informações, consulte Retirada e manutenção de tarefas para o AWS Fargate no Amazon ECS .

Para obter mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.

Balanceamento de carga do serviço

O serviço do Amazon ECS no AWS Fargate pode ser configurado opcionalmente para usar o Elastic Load Balancing para distribuir o tráfego uniformemente entre as tarefas do serviço.

Os serviços do Amazon ECS no AWS Fargate oferecem suporte aos tipos balanceadores de carga Application Load Balancer e Network Load Balancer. Application Load Balancers são usados para encaminhar o tráfego HTTP/HTTPS (ou camada 7). Os Network Load Balancers são usados para encaminhar o tráfego TCP ou UDP (ou camada 4). Para obter mais informações, consulte Uso do balanceamento de carga para distribuir o tráfego de serviço do Amazon ECS.

Ao criar grupos de destino para esses serviços, você precisa escolher `ip` como o tipo de destino, e não `instance`. Isso ocorre porque as tarefas que usam o modo de rede `awsvpc` estão associadas a uma interface de rede elástica, e não a uma instância do Amazon EC2. Para obter mais informações, consulte Uso do balanceamento de carga para distribuir o tráfego de serviço do Amazon ECS.

O uso de um Network Load Balancer para encaminhar o tráfego UDP para tarefas do Amazon ECS no AWS Fargate só será compatível quando for usada a versão 1.4 da plataforma, ou posterior.

Métricas de uso

É possível usar métricas de uso do CloudWatch para fornecer visibilidade sobre o uso dos recursos da sua conta. Use essas métricas para visualizar o uso do serviço atual nos gráficos e painéis do CloudWatch.

As métricas de uso do AWS Fargate correspondem às cotas de serviço da AWS. Também é possível configurar alarmes que alertem você quando o uso se aproximar de uma cota de serviço. Para obter mais informações sobre cotas de serviço do AWS Fargate, consulte Endpoints e cotas do Amazon ECS no Referência geral da Amazon Web Services.

Para obter mais informações sobre métricas de uso do AWS Fargate, consulte Métricas de uso do AWS Fargate.

Considerações de segurança do Amazon ECS sobre quando usar o tipo de inicialização do Fargate

Recomendamos que os clientes que buscam um forte isolamento para as tarefas usem o Fargate. O Fargate executa cada tarefa em um ambiente de virtualização de hardware. Isso garante que essas workloads em contêineres não compartilhem interfaces de rede, armazenamento temporário do Fargate, CPU ou memória com outras tarefas. Para obter mais informações, consulte Security Overview of AWS Fargate.

Práticas recomendadas de segurança do Fargate no Amazon ECS

Recomendamos que você leve em consideração as práticas recomendadas a seguir ao usar o AWS Fargate. Para obter orientação adicional, consulte Visão geral de segurança do AWS Fargate.

Uso do AWS KMS para criptografar o armazenamento temporário para o Fargate

Você deve ter seu armazenamento temporário criptografado por um AWS KMS ou por suas próprias chaves gerenciadas pelo cliente. Para tarefas hospedadas no Fargate usando a versão da plataforma `1.4.0` ou posterior, cada tarefa recebe 20 GiB de armazenamento efêmero. Para obter mais informações, consulte chave gerenciada pelo cliente (CMK). É possível aumentar a quantidade total de armazenamento temporário, até um máximo de 200 GiB, com a especificação do parâmetro `ephemeralStorage` na definição da tarefa. Para tais tarefas que foram iniciadas desde 28 de maio de 2020, o armazenamento efêmero é criptografado com um algoritmo de criptografia AES-256 usando uma chave de criptografia gerenciada pelo Fargate.

Para obter mais informações, consulte Opções para o armazenamento de tarefas do Amazon ECS.

Exemplo: iniciar uma tarefa na versão da plataforma Fargate 1.4.0 com criptografia de armazenamento efêmero

O comando a seguir iniciará uma tarefa na versão da plataforma Fargate 1.4.0 Como essa tarefa é iniciada como parte do cluster, ela usa 20 GiB do armazenamento efêmero que é criptografado automaticamente.

```
aws ecs run-task --cluster clustername \
  --task-definition taskdefinition:version \
  --count 1
  --launch-type "FARGATE" \
  --platform-version 1.4.0 \
  --network-configuration
 "awsvpcConfiguration={subnets=[subnetid],securityGroups=[securitygroupid]}" \
  --region region
```

Funcionalidade SYS_PTRACE para rastreamento de chamada de sistema do kernel com o Fargate

A configuração padrão dos recursos do Linux adicionados ou removidos do seu contêiner é fornecida pelo Docker.

As tarefas executadas no Fargate são compatíveis apenas com a adição do recurso kernel do SYS_PTRACE.

O vídeo apresentado abaixo mostra como usar esse recurso por meio do projeto Falco da Sysdig.

#ContainersFromTheCouch: solução de problemas da tarefa do Fargate usando o recurso SYS_PTRACE

O código discutido no vídeo anterior pode ser encontrado no GitHub aqui.

Uso do Amazon GuardDuty com monitoramento de runtime para o Fargate

O Amazon GuardDuty é um serviço de detecção de ameaças que ajuda a proteger contas, contêineres, workloads e dados no ambiente da AWS. Usando modelos de machine learning (ML) e recursos de detecção de anomalias e ameaças, o GuardDuty monitora continuamente diferentes fontes de log e atividades de runtime para identificar e priorizar possíveis riscos de segurança e atividades maliciosas no seu ambiente.

O monitoramento de runtime no GuardDuty protege as workloads em execução no Fargate monitorando continuamente as atividades de log e rede da AWS para identificar comportamentos maliciosos ou não autorizados. O monitoramento de runtime usa um agente de segurança do GuardDuty leve e totalmente gerenciado que analisa o comportamento no host, como acesso a arquivos, execução de processos e conexões de rede. Isso inclui problemas como escalação de privilégios, uso de credenciais expostas, comunicação com endereços IP ou domínios maliciosos e

a presença de malware nas instâncias e workloads de contêiner do Amazon EC2. Para obter mais informações, consulte GuardDuty Runtime Monitoring no Guia do usuário do GuardDuty.

Considerações de segurança do Fargate para o Amazon ECS

Cada tarefa tem uma capacidade de infraestrutura dedicada porque o Fargate executa cada workload em um ambiente virtual isolado. As workloads executadas no Fargate não compartilham interfaces de rede, armazenamento temporário, CPU ou memória com outras tarefas. É possível executar vários contêineres em uma tarefa, incluindo contêineres de aplicações e contêineres auxiliares, ou simplesmente arquivos associados. Um arquivo associado é um contêiner que é executado junto com um contêiner de aplicação em uma tarefa do Amazon ECS. Enquanto o contêiner da aplicação executa o código principal da aplicação, os processos executados em arquivos associados podem aumentar a aplicação. Os arquivos associados ajudam você a separar as funções da aplicação em contêineres dedicados, facilitando a atualização de partes da sua aplicação.

Os contêineres que fazem parte da mesma tarefa compartilham recursos para o tipo de inicialização do Fargate, pois esses contêineres sempre serão executados no mesmo host e compartilharão recursos de computação. Esses contêineres também compartilham o armazenamento temporário fornecido pela Fargate. Os contêineres de Linux em uma tarefa compartilham namespaces de rede, incluindo o endereço IP e as portas de rede. Dentro de uma tarefa, os contêineres que pertencem à tarefa podem se intercomunicar por meio do host local.

O ambiente de runtime no Fargate impede que você use determinados recursos do controlador com suporte nas instâncias do EC2. Considere o seguinte ao arquitetar workloads para execução no Fargate:

- Sem contêineres ou acesso privilegiados: recursos como contêineres ou acesso privilegiados estão atualmente indisponíveis no Fargate. Isso afetará casos de uso, como executar o Docker no Docker.
- Acesso limitado aos recursos do Linux: o ambiente no qual os contêineres são executados no Fargate está bloqueado. Recursos adicionais do Linux, como CAP_SYS_ADMIN e CAP_NET_ADMIN, são restringidos para evitar um aumento de privilégios. O Fargate oferece suporte à adição do recurso CAP_SYS_PTRACE do Linux às tarefas para permitir que as ferramentas de observabilidade e segurança implantadas na tarefa monitorem a aplicação em contêineres.
- Sem acesso ao host subjacente: nem os clientes, nem os operadores da AWS podem se conectar a um host que execute workloads do cliente. É possível usar o ECS Exec para executar comandos

em ou obter um shell para um contêiner em execução no Fargate. É possível usar o ECS exec para ajudar a coletar informações de diagnóstico para depuração. O Fargate também impede que os contêineres acessem os recursos do host subjacente, como o sistema de arquivos, dispositivos, a rede e o runtime do contêiner.

- Rede: é possível usar grupos de segurança e ACLs de rede para controlar o tráfego de entrada e saída. As tarefas do Fargate recebem um endereço IP da sub-rede configurada em sua VPC.

Versões da plataforma do Fargate para o Amazon ECS

As versões da plataforma do AWS Fargate são usadas para fazer referência a um ambiente de runtime para a infraestrutura de tarefas do Fargate. Trata-se de uma combinação da versão do kernel e do runtime do contêiner. Você seleciona uma versão da plataforma ao executar uma tarefa ou ao criar um serviço para manter várias tarefas idênticas.

Novas revisões de versões da plataforma são lançadas conforme o ambiente do runtime evolui, por exemplo, em caso de atualizações no kernel ou no sistema operacional, novos recursos, correções de erros ou atualizações de segurança. Uma versão da plataforma Fargate é atualizada por meio de uma nova revisão da versão da plataforma. Cada tarefa é executada em uma revisão de versão da plataforma durante seu ciclo de vida. Se você quiser usar a revisão mais recente da versão da plataforma, será necessário iniciar uma nova tarefa. Uma nova tarefa executada no Fargate sempre é executada na revisão mais recente de uma versão da plataforma, garantindo que as tarefas sejam sempre iniciadas em uma infraestrutura segura e corrigida.

Se houver um problema de segurança que afete uma versão existente da plataforma, a AWS vai criar uma nova revisão corrigida da versão da plataforma e retirar as tarefas em execução na revisão vulnerável. Em alguns casos, será possível receber notificações de que suas tarefas no Fargate foram programadas para retirada. Para obter mais informações, consulte Retirada e manutenção de tarefas para o AWS Fargate no Amazon ECS .

Você especifica a versão da plataforma ao executar uma tarefa ou implementar um serviço.

Considere o seguinte ao especificar uma versão de plataforma:

- É possível especificar um número de versão específico, por exemplo, `1.4.0` ou `LATEST`.

  A versão da plataforma Linux MAIS RECENTE é a `1.4.0`.

  A versão da plataforma Windows MAIS RECENTE é a `1.0.0`.

- Se você quiser atualizar a versão da plataforma para um serviço, crie uma implantação. Por exemplo, suponha que você tenha um serviço que executa tarefas na versão `1.3.0` da plataforma Linux. Para alterar o serviço para executar tarefas na plataforma Linux versão `1.4.0`, você pode atualizar seu serviço e especificar uma nova versão da plataforma. Suas tarefas serão reimplantadas com a versão mais recente da plataforma e a revisão mais recente da versão da plataforma. Para obter mais informações sobre implantações, consulte Serviços do Amazon ECS.
- Caso seu serviço seja expandido sem atualizar a versão da plataforma, essas tarefas receberão a versão especificada na implantação atual dele. Por exemplo, suponha que você tenha um serviço que executa tarefas na versão `1.3.0` da plataforma Linux. Se você aumentar a contagem desejada do serviço, o programador de serviços iniciará as novas tarefas usando a versão mais recente da plataforma, a revisão da versão `1.3.0` da plataforma.
- As novas tarefas sempre são executadas na revisão mais recente de uma versão da plataforma. Isso garante que as tarefas estejam sempre em uma infraestrutura protegida e corrigida.
- Os números de versão da plataforma para contêineres Linux e contêineres Windows no Fargate são independentes. Por exemplo, o comportamento, os recursos e o software usados na versão da plataforma `1.0.0` para contêineres de Windows no Fargate não são comparáveis aos da versão da plataforma `1.0.0` para contêineres de Linux no Fargate.
- O seguinte é válido para versões da plataforma Windows para o Fargate.

  As imagens de contêiner do Microsoft Windows Server devem ser criadas com base em uma versão específica do Windows Server. Você deve selecionar a mesma versão do Windows Server na `platformFamily` ao executar uma tarefa ou criar um serviço que corresponda à imagem de contêiner do Windows Server. Além disso, é possível fornecer uma `operatingSystemFamily` correspondente na definição da tarefa para evitar que as tarefas sejam executadas na versão errada do Windows. Para obter mais informações, consulte Correspondência de versão do host do contêiner com versões de imagens do contêiner no site Microsoft Learn.

Migração para a versão da plataforma Linux 1.4.0 no Amazon ECS

Considere o seguinte ao migrar suas tarefas do Amazon ECS no Fargate da versão de plataforma `1.0.0`, `1.1.0`, `1.2.0` ou `1.3.0` para a versão de plataforma `1.4.0`. É prática recomendada confirmar se sua tarefa funciona corretamente na versão `1.4.0` da plataforma antes de migrar suas tarefas.

- O comportamento de tráfego de rede de e para tarefas foi atualizado. A partir da versão 1.4.0 da plataforma, todas as tarefas do Amazon ECS no Fargate recebem uma única interface de rede

elástica (conhecida como ENI de tarefa), e todo o tráfego de rede flui por essa ENI dentro da VPC. O tráfego é visível para você por meio dos logs de fluxo da VPC. Para obter mais informações, consulte Opções de redes de tarefas do Amazon ECS para o tipo de inicialização do Fargate.

- Se você usa endpoints da VPC de interface, considere o seguinte.
  - Para imagens de contêiner hospedadas com o Amazon ECR, os endpoints a seguir são necessários. Para obter mais informações, consulte Endpoints da VPC de interface do Amazon ECR (AWS PrivateLink) no Guia do usuário do Amazon Elastic Container Registry.
    - Endpoint da VPC do Amazon ECR com.amazonaws.***region***.ecr.dkr
    - Endpoint da VPC do Amazon ECR com.amazonaws.***region***.ecr.api
    - Endpoint do gateway do Amazon S3
  - Quando a definição da tarefa fizer referência a segredos do Secrets Manager para recuperar dados sigilosos para os contêineres, você deverá criar os endpoints da VPC de interface para o Secrets Manager. Para obter mais informações, consulte Usar o Secrets Manager com Endpoints da VPC no Guia do usuário do AWS Secrets Manager.
  - Quando a definição de tarefa fizer referência a parâmetros do Systems Manager Parameter Store para recuperar dados sigilosos para os contêineres, crie os endpoints da VPC de interface para o Systems Manager. Para obter mais informações, consulte Melhorar a segurança das instâncias do EC2 usando endpoints da VPC para o Systems Manager no Guia do usuário do AWS Systems Manager.
  - O grupo de segurança da interface de rede elástica (ENI) associada à sua tarefa precisa das regras do grupo de segurança para permitir o tráfego entre a tarefa e os endpoints da VPC.

Log de alterações da versão da plataforma Linux para o Fargate

Veja a seguir as versões da plataforma Linux disponíveis. Para obter informações sobre a substituição de versões anteriores da plataforma, consulte Descontinuação da versão da plataforma Linux do AWS Fargate.

1.4.0

Veja a seguir o changelog da versão `1.4.0` da plataforma.

- A partir de 5 de novembro de 2020, qualquer nova tarefa do Amazon ECS lançada no Fargate usando a versão `1.4.0` da plataforma poderá usar os seguintes recursos:
  - Ao usar o Secrets Manager para armazenar dados sigilosos, você pode injetar uma chave JSON específica ou uma versão específica de um segredo como uma variável de ambiente ou

em uma configuração de log. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.

- Especifique variáveis de ambiente em massa usando o parâmetro `environmentFiles` de definição de contêiner. Para obter mais informações, consulte Transferência de uma variável de ambiente individual para um contêiner do Amazon ECS.
- Tarefas executadas em uma VPC e em uma sub-rede habilitadas para IPv6 receberão um endereço IPv4 privado e um endereço IPv6. Para obter mais informações, consulte Opções de rede de tarefas do Amazon ECS para o tipo de execução do Fargate.
- O endpoint de metadados da tarefa versão 4 fornece metadados adicionais sobre a tarefa e o contêiner, incluindo o tipo de inicialização da tarefa, o nome do recurso da Amazon (ARN) do contêiner e o driver de log e as opções de driver de log usadas. Ao consultar o endpoint `/stats`, você também recebe dados estatísticos de taxa de rede para seus contêineres. Para obter mais informações, consulte Task metadata endpoint version 4.

- A partir de 30 de julho de 2020, qualquer nova tarefa do Amazon ECS iniciada no Fargate usando a versão `1.4.0` da plataforma poderá encaminhar o tráfego UDP usando um Network Load Balancer para tarefas do Amazon ECS no Fargate. Para obter mais informações, consulte Uso do balanceamento de carga para distribuir o tráfego de serviço do Amazon ECS.
- A partir de 28 de maio de 2020, qualquer nova tarefa do Amazon ECS iniciada no Fargate usando a versão `1.4.0` da plataforma terá seu armazenamento temporário criptografado com um algoritmo de criptografia AES-256 usando uma chave de criptografia gerenciada pela AWS. Para obter mais informações, consulte Armazenamento efêmero de tarefas do Fargate para o Amazon ECS e Opções de armazenamento para tarefas do Amazon ECS.
- Adicionado suporte para usar volumes do sistema de arquivos do Amazon EFS para o armazenamento de tarefas persistentes. Para obter mais informações, consulte Uso de volumes do Amazon EFS com o Amazon ECS.
- O armazenamento temporário de tarefas foi aumentado para no mínimo 20 GB para cada tarefa. Para obter mais informações, consulte Armazenamento efêmero de tarefas do Fargate para o Amazon ECS.
- O comportamento de tráfego de rede de e para tarefas foi atualizado. A partir da versão 1.4.0 da plataforma, todas as tarefas do Fargate recebem uma única interface de rede elástica (conhecida como ENI de tarefa), e todo o tráfego de rede flui por essa ENI dentro da VPC e será visível por meio dos logs de fluxo da VPC. Para obter mais informações sobre redes para o tipo de inicialização do Amazon EC2, consulte Opções de rede de tarefas do Amazon ECS para o tipo de inicialização do EC2. Para obter mais informações sobre redes para o tipo de execução do

Fargate, consulte Opções de redes de tarefas do Amazon ECS para o tipo de inicialização do Fargate.

- As ENIs de tarefas adicionam suporte para quadros jumbo. As interfaces de rede são configuradas com uma unidade de transmissão máxima (MTU), que se refere ao tamanho da maior carga útil que cabe em um único quadro. Quanto maior a MTU, maior será a carga útil do aplicativo que pode caber em um único quadro, o que reduz a sobrecarga por quadro e aumenta a eficiência. O suporte a quadros jumbo reduzirá a sobrecarga quando o caminho de rede entre a tarefa e o destino oferecer suporte a quadros jumbo, assim como todo o tráfego que permanece dentro da VPC.
- O CloudWatch Container Insights inclui métricas de performance de rede para tarefas do Fargate. Para obter mais informações, consulte Monitorar contêineres do Amazon ECS utilizando o Container Insights com observabilidade aprimorada.
- Adicionado suporte para o endpoint de metadados de tarefas versão 4, que fornece informações adicionais para suas tarefas do Fargate, incluindo dados estatísticos da rede da tarefa e em qual zona de disponibilidade a tarefa está sendo executada. Para obter mais informações, consulte Endpoint de metadados de tarefas do Amazon ECS versão 4 e Endpoint de metadados de tarefas do Amazon ECS versão 4 para tarefas no Fargate.
- Suporte adicionado para o parâmetro Linux `SYS_PTRACE` nas definições de contêiner. Para obter mais informações, consulte Parâmetros do Linux.
- O agente de contêiner do Fargate substitui o uso do agente de contêiner do Amazon ECS do para todas as tarefas do Fargate. Normalmente, essa alteração não afeta a forma como suas tarefas são executadas.
- O runtime do contêiner agora está usando containerd em vez do docker. Provavelmente, essa alteração não afeta a forma como suas tarefas são executadas. Você perceberá que algumas mensagens de erro provenientes do runtime do contêiner deixarão de mencionar o Docker e mostrarão erros mais gerais. Para obter mais informações, consulte Mensagens de erro de tarefas interrompidas do Amazon ECS.
- Com base no Amazon Linux 2.

1.3.0

Veja a seguir o changelog da versão `1.3.0` da plataforma.

- A partir de 30 de setembro de 2019, qualquer nova tarefa do Fargate iniciada é compatível com o driver de log `awsfirelens`. Configure o FireLens para Amazon ECS para usar parâmetros

de definição de tarefa para encaminhar logs para um serviço da AWS ou um destino da Rede de Parceiros da AWS (APN) para armazenamento e análise de log. Para obter mais informações, consulte Envio de logs do Amazon ECS para um serviço da AWS ou para uma AWS Partner.

- Adicionada a reciclagem de tarefas para as tarefas do Fargate, que é o processo de atualização das tarefas que fazem parte de um serviço do Amazon ECS. Para obter mais informações, consulte Retirada e manutenção de tarefas para AWS Fargate no Amazon ECS.
- A partir de 27 de março de 2019, todas as novas tarefas do Fargate iniciadas podem usar parâmetros adicionais de definição de tarefa que você usa para definir uma configuração de proxy, dependências para startup e desligamento de contêiner, além de um valor de tempo limite de início e interrupção por contêiner. Para ter mais informações, consulte Configuração do proxy, Dependência de contêiner e Tempos limite de contêiner.
- A partir de 2 de abril de 2019, todas as novas tarefas do Fargate iniciadas oferecem suporte à injeção de dados sigilosos nos seus contêineres. Isso é feito armazenando seus dados sigilosos em segredos do AWS Secrets Manager ou em parâmetros do AWS Systems Manager Parameter Store e fazendo referência a eles na definição do contêiner. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.
- A partir de 1.º de maio de 2019, todas as novas tarefas do Fargate iniciadas oferecem suporte à referência a dados sigilosos na configuração de log de um contêiner usando o parâmetro `secretOptions` de definição do contêiner. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.
- A partir de 1.º de maio de 2019, todas as novas tarefas do Fargate que iniciadas oferecem suporte ao driver de log `splunk`, além do driver de log `awslogs`. Para obter mais informações, consulte Armazenamento e registro.
- A partir de 9 de julho de 2019, todas as novas tarefas do Fargate iniciadas oferecem suporte ao CloudWatch Container Insights. Para obter mais informações, consulte Monitorar contêineres do Amazon ECS utilizando o Container Insights com observabilidade aprimorada.
- A partir de 3 de dezembro de 2019, há suporte para o provedor de capacidade Fargate Spot. Para obter mais informações, consulte Clusters do Amazon ECS para Fargate.
- Com base no Amazon Linux 2.

Descontinuação da versão da plataforma Linux do AWS Fargate

Essa página lista as versões da plataforma Linux que o AWS Fargate descontinuou ou que tiveram sua descontinuação programada. Essas versões de plataforma permanecem disponíveis até a data publicada da substituição.

Uma data de atualização forçada é fornecida para cada versão da plataforma com substituição programada. Na data de atualização forçada, qualquer serviço que usar a versão LATEST da plataforma e que for direcionado para uma versão da plataforma com substituição programada será atualizado por meio da opção de forçar nova implantação. Quando o serviço é atualizado usando a opção de forçar nova implantação, todas as tarefas executadas em uma versão da plataforma com substituição programada são interrompidas e novas tarefas são iniciadas usando a versão da plataforma para a qual a etiqueta LATEST aponta, naquele momento. Tarefas ou serviços autônomos com um conjunto explícito de versões de plataforma não são afetados pela data de atualização forçada.

Recomendamos que as tarefas autônomas de serviços sejam atualizadas usar a versão mais recente da plataforma. Para obter mais informações sobre como migrar para a versão mais recente da plataforma, consulte Migração para a versão da plataforma Linux 1.4.0 no Amazon ECS.

Quando uma versão da plataforma atingir a data da substituição, não estará mais disponível para novas tarefas ou serviços. Todas as tarefas ou serviços autônomos que usarem explicitamente uma versão obsoleta da plataforma continuarão a usar esta versão até que as tarefas sejam interrompidas. Após a data de substituição, qualquer versão obsoleta da plataforma não receberá mais atualizações de segurança ou correções de bugs.

Versão da plataforma	Data da atualização forçada	Data da substituição
1.0.0	26 de outubro de 2020	14 de dezembro de 2020
1.1.0	26 de outubro de 2020	14 de dezembro de 2020
1.2.0	26 de outubro de 2020	14 de dezembro de 2020

Para obter informações sobre as versões atuais da plataforma, consulte Versões da plataforma do Fargate para o Amazon ECS.

Log de alterações das versões defasadas do Linux para o Fargate

1.2.0

Veja a seguir o changelog da versão `1.2.0` da plataforma.

> **Note**
>
> A versão `1.2.0` da plataforma não está mais disponível. Para obter informações sobre a substituição de versões anteriores da plataforma, consulte Descontinuação da versão da plataforma Linux do AWS Fargate.

- Adicionado suporte para autenticação de registro privado usando AWS Secrets Manager. Para obter mais informações, consulte Uso de imagens de contêiner que não são da AWS no Amazon ECS.

1.1.0

Veja a seguir o changelog da versão `1.1.0` da plataforma.

> **Note**
>
> A versão `1.1.0` da plataforma não está mais disponível. Para obter informações sobre a substituição de versões anteriores da plataforma, consulte Descontinuação da versão da plataforma Linux do AWS Fargate.

- Adicionado suporte para o endpoint de metadados de tarefa do Amazon ECS. Para obter mais informações, consulte Metadados de tarefas do Amazon ECS disponíveis para tarefas no Fargate.
- Inclusão de suporte às verificações de integridade do Docker nas definições de contêiner. Para obter mais informações, consulte Verificação de integridade.
- Adicionado suporte para a descoberta de serviços do Amazon ECS. Para obter mais informações, consulte Uso da descoberta de serviços para conectar serviços do Amazon ECS com nomes DNS.

1.0.0

Veja a seguir o changelog da versão `1.0.0` da plataforma.

> **Note**
>
> A versão `1.0.0` da plataforma não está mais disponível. Para obter informações sobre a substituição de versões anteriores da plataforma, consulte Descontinuação da versão da plataforma Linux do AWS Fargate.

- Com base no Amazon Linux 2017.09.
- Versão inicial.

Log de alterações da versão da plataforma Windows para o Fargate

Veja a seguir as versões disponíveis da plataforma para contêineres do Windows.

1.0.0

Veja a seguir o changelog da versão `1.0.0` da plataforma.

- Versão inicial para suporte nos seguintes sistemas operacionais Microsoft Windows Server:
  - Windows Server 2019 Full
  - Windows Server 2019 Core
  - Windows Server 2022 Full
  - Windows Server 2022 Core

Considerações sobre contêineres do Windows no Fargate para o Amazon ECS

A seguir estão as diferenças e considerações que você deve saber ao executar contêineres do Windows no AWS Fargate.

Caso precise executar tarefas em contêineres do Linux e do Windows, é necessário criar definições de tarefas separadas em cada sistema operacional.

A AWS administra o gerenciamento de licenças do sistema operacional, de modo que você não precise de nenhuma licença adicional do Microsoft Windows Server.

Os contêineres do Windows no AWS Fargate oferecem suporte aos seguintes sistemas operacionais:

- Windows Server 2019 Full
- Windows Server 2019 Core
- Windows Server 2022 Full
- Windows Server 2022 Core

Os contêineres do Windows no AWS Fargate oferecem suporte ao driver awslogs. Para obter mais informações, consulte the section called “Envio de logs para o CloudWatch ”.

Não há suporte para os seguintes recursos em contêineres do Windows no Fargate:

- Amazon FSx
- Entroncamento ENI
- gMSAs para contêineres do Windows
- Integração de proxy e serviço App Mesh para tarefas
- Integração do roteador de log Firelens para tarefas
- Volumes do EFS
- Volumes do EBS
- Os seguintes parâmetros de definição de tarefa:
  - `maxSwap`
  - `swappiness`
  - `environmentFiles`
- O provedor de capacidade Fargate Spot
- Volumes de imagem

  A opção `volume` do Dockerfile é ignorada. Em vez disso, use montagens bind na sua definição de tarefa. Para obter mais informações, consulte Uso de montagens vinculadas com o Amazon ECS.
- Os parâmetros de CPU e memória ao nível de tarefa são ignorados para contêineres do Windows. É recomendável especificar recursos em nível de contêiner para contêineres do Windows.
- memória para tarefa
- mermoryReservation em contêineres
- Políticas de reinicialização em contêineres

Armazenamento efêmero de tarefas do Fargate para o Amazon ECS

Quando provisionada, cada tarefa do Amazon ECS hospedada em contêineres Linux no AWS Fargate recebe o armazenamento temporário a seguir, para montagens bind. Isso pode ser montado e compartilhado entre os contêineres usando os parâmetros `volumes`, `mountPoints` e `volumesFrom` na definição da tarefa. Não há suporte para isso nos contêineres do Windows no AWS Fargate.

Versões da plataforma de contêiner Linux do Fargate

Versão 1.4.0 ou posterior

Por padrão, as tarefas do Amazon ECS hospedadas no Fargate usando a versão `1.4.0` ou posterior da plataforma recebem, no mínimo, 20 GiB de armazenamento temporário. A quantidade total de armazenamento temporário pode ser aumentada, até um máximo de 200 GiB. Para fazer isso, especifique o parâmetro `ephemeralStorage` na definição de tarefa.

A imagem de contêiner extraída, compactada e descompactada da tarefa é armazenada no armazenamento temporário. Para determinar a quantidade total de armazenamento temporário que sua tarefa precisa usar, subtraia a quantidade de armazenamento usada pela imagem do contêiner da quantidade total de armazenamento temporário em que a tarefa está alocada..

Para tarefas que usam a versão `1.4.0` ou posterior da plataforma iniciadas em 28 de maio de 2020 ou depois, o armazenamento temporário é criptografado com um algoritmo de criptografia AES-256. Esse algoritmo usa uma chave de criptografia pertencente à AWS ou você pode criar sua própria chave gerenciada pelo cliente. Para obter mais informações, consulte Customer managed keys for AWS Fargate ephemeral storage.

Para tarefas que usam uma versão `1.4.0` ou posterior da plataforma, lançadas a partir de 18 de novembro de 2022, o uso de armazenamento temporário é informado por meio do endpoint de metadados da tarefa. As aplicações nas suas tarefas podem consultar a versão 4 do endpoint de metadados da tarefa para obter o tamanho reservado de armazenamento temporário e a quantidade usada.

Além disso, o tamanho reservado do armazenamento temporário e a quantidade usada serão enviados ao Amazon CloudWatch Container Insights se você ativar o Container Insights.

> **Note**
>
> O Fargate reserva espaço no disco. Esse espaço é usado apenas pelo Fargate. Você não é cobrado por isso. Ele não é mostrado nessas métricas. Porém, você pode ver esse armazenamento adicional em outras ferramentas, como o `df`.

Versão 1.3.0 ou anterior

Para as tarefas do Amazon ECS no Fargate que usam a versão `1.3.0` ou anterior da plataforma, cada tarefa recebe o armazenamento temporário a seguir.

- 10 GB de armazenamento de camadas do Docker

  > **Note**
  >
  > Essa quantidade inclui artefatos de imagem de contêiner compactados e não compactados.

- Mais 4 GB para montagens de volume. Isso pode ser montado e compartilhado entre os contêineres usando os parâmetros `volumes`, `mountPoints` e `volumesFrom` na definição da tarefa.

Versões da plataforma de contêiner Windows do Fargate

Versão 1.0.0 ou posterior

Por padrão, as tarefas do Amazon ECS hospedadas no Fargate usando a versão `1.0.0` ou posterior da plataforma recebem, no mínimo, 20 GiB de armazenamento temporário. A quantidade total de armazenamento temporário pode ser aumentada, até um máximo de 200 GiB. Para fazer isso, especifique o parâmetro `ephemeralStorage` na definição de tarefa.

A imagem de contêiner extraída, compactada e descompactada da tarefa é armazenada no armazenamento temporário. Para determinar a quantidade total de armazenamento temporário que sua tarefa precisa usar, subtraia a quantidade de armazenamento usada pela imagem do contêiner da quantidade total de armazenamento temporário em que a tarefa está alocada..

Para obter mais informações, consulte Uso de montagens vinculadas com o Amazon ECS.

Chaves gerenciadas pelo cliente para o armazenamento efêmero do AWS Fargate para o Amazon ECS

O AWS Fargate é compatível com chaves gerenciadas pelo cliente para criptografar dados para tarefas do Amazon ECS mantidas em armazenamento efêmero para ajudar clientes que estão sujeitos a regulamentações a seguir suas políticas internas de segurança. Os clientes ainda obtêm o benefício da tecnologia sem servidor do Fargate, ao mesmo tempo em que oferecem maior visibilidade da criptografia de armazenamento autogerenciada aos auditores de conformidade. Embora o Fargate tenha, por padrão, criptografia de armazenamento efêmero gerenciada pelo Fargate, os clientes também podem usar suas próprias chaves autogerenciadas ao criptografar dados confidenciais, como informações financeiras ou médicas.

Você pode importar suas próprias chaves para o AWS KMS ou criar as chaves no AWS KMS. Essas chaves autogerenciadas são armazenadas no AWS KMS e realizam as ações padrão do ciclo de vida do AWS KMS, como alternar, desativar e excluir. Você pode auditar o acesso e o uso das chaves nos logs do CloudTrail.

Por padrão, a chave KMS é compatível com 50.000 concessões por chave. O Fargate usa uma única concessão do AWS KMS por tarefa de chave gerenciada pelo cliente, sendo compatível com até 50.000 tarefas simultâneas para uma chave. Se quiser aumentar esse número, você pode solicitar um aumento de limite, que é aprovado caso a caso.

O Fargate não cobra nenhum adicional pelo uso de chaves gerenciadas pelo cliente. Você só paga o preço padrão pelo uso das chaves do AWS KMS para solicitações de armazenamento e de API.

Tópicos

- Criar uma chave de criptografia para o armazenamento efêmero do Fargate para o Amazon ECS
- Gerenciar chaves AWS KMS para o armazenamento efêmero do Fargate para o Amazon ECS

Criar uma chave de criptografia para o armazenamento efêmero do Fargate para o Amazon ECS

Crie uma chave gerenciada do cliente para criptografar dados armazenados no armazenamento temporário do Fargate.

> **Note**
>
> A criptografia de armazenamento efêmero do Fargate com chaves gerenciadas pelo cliente não está disponível para clusters de tarefas do Windows.
> A criptografia do armazenamento efêmero do Fargate com chaves gerenciadas pelo cliente não está disponível nas `platformVersions` anteriores à versão `1.4.0`.
> O Fargate reserva espaço em um armazenamento efêmero que só é usado pelo Fargate, e você não é cobrado por esse espaço. A alocação pode ser diferente nas tarefas de chave não gerenciadas pelo cliente, mas o espaço total permanece o mesmo. Você pode ver essa mudança em ferramentas como `df`.
> Chaves de várias regiões não são compatíveis com armazenamento efêmero do Fargate.
> Aliases de chaves do KMS não são compatíveis com armazenamento efêmero do Fargate.

Para criar um a chave gerenciada pelo cliente (CMK) para criptografar armazenamento efêmero para o Fargate no AWS KMS, siga estas etapas.

1. Navegue até https://console.aws.amazon.com/kms.
2. Siga as instruções de Creating Keys no AWS Key Management Service Developer Guide.
3. Ao criar sua chave AWS KMS, certifique-se de fornecer as permissões operacionais relevantes do AWS KMS ao serviço Fargate nas políticas de chave. As operações de API a seguir devem ser permitidas na política para usar a chave gerenciada pelo cliente com os recursos de cluster do Amazon ECS.

   - `kms:GenerateDataKeyWithoutPlainText`: chame `GenerateDataKeyWithoutPlainText` para gerar uma chave de dados criptografada a partir da chave AWS KMS fornecida.
   - `kms:CreateGrant`: adiciona uma concessão a uma chave gerenciada pelo cliente. Concede controle de acesso a uma chave AWS KMS especificada, o que permite acesso às operações de concessão que o Fargate do Amazon ECS requer. Para obter mais informações sobre Utilizar concessões, consulte o Guia do desenvolvedor do AWS Key Management Service. Isso permite que o Fargate do Amazon ECS faça o seguinte:
     - Chame `Decrypt` para que o AWS KMS obtenha a chave de criptografia para descriptografar os dados do armazenamento efêmero.
     - Configure uma entidade principal aposentada para permitir que o serviço para `RetireGrant`.

- `kms:DescribeKey`: fornece os detalhes da chave gerenciada pelo cliente para permitir que o Amazon ECS valide a chave se ela for simétrica e estiver habilitada.

O exemplo a seguir mostra uma política de chave AWS KMS que você aplicaria à chave de destino para criptografia. Para usar as instruções do exemplo de política, substitua os *espaços reservados para entrada do usuário* por suas próprias informações. Como sempre, configure apenas as permissões necessárias, mas você precisará fornecer o AWS KMS com permissões para pelo menos um usuário a fim de evitar erros.

```
{
      "Sid": "Allow generate data key access for Fargate tasks.",
      "Effect": "Allow",
      "Principal": { "Service":"fargate.amazonaws.com" },
      "Action": [
        "kms:GenerateDataKeyWithoutPlaintext"
      ],
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:ecs:clusterAccount": [
            "customerAccountId"
          ],
          "kms:EncryptionContext:aws:ecs:clusterName": [
             "clusterName"
          ]
        }
      },
      "Resource": "*"
    },
    {
      "Sid": "Allow grant creation permission for Fargate tasks.",
      "Effect": "Allow",
      "Principal": { "Service":"fargate.amazonaws.com" },
      "Action": [
        "kms:CreateGrant"
      ],
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:ecs:clusterAccount": [
            "customerAccountId"
          ],
          "kms:EncryptionContext:aws:ecs:clusterName": [
```

```
                "clusterName"
            ]
          },
          "ForAllValues:StringEquals": {
            "kms:GrantOperations": [
                "Decrypt"
            ]
          }
        },
        "Resource": "*"
    },
    {
        "Sid": "Allow describe key permission for cluster operator - CreateCluster
 and UpdateCluster.",
        "Effect": "Allow",
        "Principal": { "AWS":"arn:aws:iam::customerAccountId:role/customer-chosen-
role" },
        "Action": [
          "kms:DescribeKey"
        ],
        "Resource": "*"
    }
```

As tarefas do Fargate usam as chaves de contexto de criptografia `aws:ecs:clusterAccount` e `aws:ecs:clusterName` para operações criptográficas com a chave. Os clientes devem adicionar essas permissões para restringir o acesso a uma conta e/ou cluster específicos. Use o nome do cluster e não o ARN ao especificar o cluster.

Para obter mais informações, consulte Contexto de criptografia no Guia do desenvolvedor AWS KMS.

Ao criar ou atualizar um cluster, você tem a opção usar a chave de condição `fargateEphemeralStorageKmsKeyId`. Essa chave de condição permite que os clientes tenham um controle mais granular das políticas do IAM. As atualizações da configuração `fargateEphemeralStorageKmsKeyId` só se aplicam a novas implantações do serviço.

O exemplo a seguir mostra como deixar que os clientes só concedam permissões a um conjunto específico de chaves AWS KMS aprovadas.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:CreateCluster",
        "ecs:UpdateCluster"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "ecs:fargate-ephemeral-storage-kms-key": "arn:aws:kms:us-
west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
        }
      }
    }
  ]
}
```

O próximo exemplo mostra como negar tentativas de remover chaves AWS KMS que já estão associadas a um cluster.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": [
        "ecs:CreateCluster",
        "ecs:UpdateCluster"
    ],
    "Resource": "*",
    "Condition": {
      "Null": {
        "ecs:fargate-ephemeral-storage-kms-key": "true"
```

```
      }
    }
  }
}
```

Os clientes podem ver se as tarefas não gerenciadas ou as tarefas de serviço estão criptografadas com a chave usando os comandos AWS CLI, `describe-tasks`, `describe-cluster` ou `describe-services`.

Para obter mais informações, consulte Condition keys for AWS KMS in the AWS KMS Developer Guide.

AWS Management Console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Escolha Clusters no painel de navegação esquerdo, depois selecione Criar cluster no canto superior direito ou escolha um cluster existente. Para um cluster existente, escolha Atualizar cluster no canto superior direito.
3. Na seção Criptografia do fluxo de trabalho, você terá a opção de selecionar sua chave AWS KMS em Armazenamento gerenciado e Armazenamento efêmero do Fargate. Você também pode optar por criar uma chave AWS KMS aqui.
4. Escolha Criar quando terminar de criar o novo cluster ou Atualizar, se estiver atualizando um cluster existente.

AWS CLI

O exemplo a seguir mostra como criar um cluster e configurar o armazenamento efêmero do Fargate usando a AWS CLI (substitua os valores em *vermelho* pelos seus):

```
aws ecs create-cluster --cluster clusterName \
--configuration '{"managedStorageConfiguration":
{"fargateEphemeralStorageKmsKeyId":"arn:aws:kms:us-
west-2:012345678901:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"}}'
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-west-2:012345678901:cluster/clusterName",
        "clusterName": "clusterName",
        "configuration": {
```

```
            "managedStorageConfiguration": {
                "fargateEphemeralStorageKmsKeyId": "arn:aws:kms:us-
west-2:012345678901:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
            }
        },
        "status": "ACTIVE",
        "registeredContainerInstancesCount": 0,
        "runningTasksCount": 0,
        "pendingTasksCount": 0,
        "activeServicesCount": 0,
        "statistics": [],
        "tags": [],
        "settings": [],
        "capacityProviders": [],
        "defaultCapacityProviderStrategy": []
    },
    "clusterCount": 5
}
```

AWS CloudFormation

O exemplo a seguir mostra como criar um cluster e configurar o armazenamento efêmero do Fargate usando a AWS CloudFormation o (substitua os valores em *vermelho* pelos):

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  MyCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: "clusterName"
      Configuration:
        ManagedStorageConfiguration:
          FargateEphemeralStorageKmsKeyId: "arn:aws:kms:us-
west-2:012345678901:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

Gerenciar chaves AWS KMS para o armazenamento efêmero do Fargate para o Amazon ECS

Depois de criar ou importar sua chave AWS KMS para criptografar o armazenamento efêmero do Fargate, você o gerencia como faria com qualquer outra chave AWS KMS.

Alternância automática de chaves AWS KMS

Você pode ativar a troca automática das chaves ou fazer isso manualmente. A troca automática da chave faz isso para você uma vez por ano, gerando novo material criptográfico para a chave. O AWS KMS também salva todas as versões anteriores do material criptográfico, para que você possa descriptografar dados que usaram as versões anteriores da chave. O AWS KMS não excluirá nenhum material que tenha sido trocado até você excluir a chave.

A troca automática é opcional e pode ser habilitada ou desabilitada a qualquer momento.

Desativando ou revogando chaves AWS KMS

Desabilitar uma chave gerenciada pelo cliente no AWS KMS não terá nenhum impacto na execução das tarefas e elas continuarão funcionando durante todo o ciclo de vida. Uma tarefa que usar a chave desabilitada ou revogada falhará porque não poderá acessar a chave. Você deve definir um alarme do CloudWatch ou similar para garantir que uma chave desabilitada nunca seja necessária para descriptografar dados já criptografados.

Excluir chaves AWS KMS

Excluir chaves deve ser sempre o último recurso e só deve ser usado se você tiver certeza de que a chave excluída nunca mais será necessária. Novas tarefas que tentarem usar a chave excluída falharão porque não conseguirão acessá-la. O AWS KMS recomenda desabilitar a chave em vez de excluí-la. Se você achar necessário excluir uma chave, sugerimos desativá-la primeiro e definir um alarme do CloudWatch para ter certeza de que ela não é necessária. Se resolver excluir uma chave, o AWS KMS dá a você pelo menos sete dias para mudar de ideia.

Auditar acesso a chaves AWS KMS

Você pode usar os logs do CloudTrail para auditar o acesso à sua chave AWS KMS. Você pode verificar as operações `CreateGrant`, `GenerateDataKeyWithoutPlaintext` e `Decrypt` do AWS KMS. Essas operações também mostram a `aws:ecs:clusterAccount` e o `aws:ecs:clusterName` como parte do `EncryptionContext` registrado em log do CloudTrail.

O exemplo a seguir mostra eventos do CloudTrail para `GenerateDataKeyWithoutPlaintext`, `GenerateDataKeyWithoutPlaintext (DryRun)`, `CreateGrant`, `CreateGrant (DryRun)` e `RetireGrant` (*substitua os valores em vermelho* pelos seus).

GenerateDataKeyWithoutPlaintext

```
{
    "eventVersion": "1.08",
```

```
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "ec2-frontend-api.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:13Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyWithoutPlaintext",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "ec2-frontend-api.amazonaws.com",
    "userAgent": "ec2-frontend-api.amazonaws.com",
    "requestParameters": {
        "numberOfBytes": 64,
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "encryptionContext": {
            "aws:ecs:clusterAccount": "account-id",
            "aws:ebs:id": "vol-xxxxxxx",
            "aws:ecs:clusterName": "cluster-name"
        }
    },
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": true,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```


GenerateDataKeyWithoutPlaintext (DryRun)


```
{
    "eventVersion": "1.08",
```

```
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "fargate.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:11Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyWithoutPlaintext",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "fargate.amazonaws.com",
    "userAgent": "fargate.amazonaws.com",
    "errorCode": "DryRunOperationException",
    "errorMessage": "The request would have succeeded, but the DryRun option is
 set.",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "dryRun": true,
        "numberOfBytes": 64,
        "encryptionContext": {
            "aws:ecs:clusterAccount": "account-id",
            "aws:ecs:clusterName": "cluster-name"
        }
    },
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": true,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

CreateGrant


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "ec2-frontend-api.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:13Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "ec2-frontend-api.amazonaws.com",
    "userAgent": "ec2-frontend-api.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "granteePrincipal": "fargate.us-west-2.amazonaws.com",
        "operations": [
            "Decrypt"
        ],
        "constraints": {
            "encryptionContextSubset": {
                "aws:ecs:clusterAccount": "account-id",
                "aws:ebs:id": "vol-xxxx",
                "aws:ecs:clusterName": "cluster-name"
            }
        },
        "retiringPrincipal": "ec2.us-west-2.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855",
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
    },
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": false,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
```

```
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

CreateGrant (DryRun)

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "fargate.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:11Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "fargate.amazonaws.com",
    "userAgent": "fargate.amazonaws.com",
    "errorCode": "DryRunOperationException",
    "errorMessage": "The request would have succeeded, but the DryRun option is
 set.",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "granteePrincipal": "fargate.us-west-2.amazonaws.com",
        "dryRun": true,
        "operations": [
            "Decrypt"
        ],
        "constraints": {
            "encryptionContextSubset": {
                "aws:ecs:clusterAccount": "account-id",
                "aws:ecs:clusterName": "cluster-name"
            }
        }
    },
```

```
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": false,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

RetireGrant

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2024-04-20T18:37:38Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "RetireGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": null,
    "responseElements": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
    },
    "additionalEventData": {
        "grantId":
 "e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855"
    },
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
```

```
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": false,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```


Retirada e manutenção de tarefas para o AWS Fargate no Amazon ECS

A AWS é responsável por manter a infraestrutura subjacente do AWS Fargate. A AWS determina quando uma revisão da versão da plataforma precisa ser substituída por uma nova revisão da infraestrutura. Isso é conhecido como retirada de tarefas. A AWS envia uma notificação de retirada da tarefa quando uma revisão da versão da plataforma é retirada. Atualizamos rotineiramente nossas versões da plataforma com suporte para introduzir uma revisão contendo atualizações do software de runtime do Fargate e dependências subjacentes, como o sistema operacional e o runtime do contêiner. Quando uma revisão mais recente é disponibilizada, retiramos a revisão mais antiga para garantir que todas as workloads do cliente sejam executadas na revisão mais atualizada da versão da plataforma do Fargate. Quando uma revisão é retirada, todas as tarefas em execução nessa revisão são interrompidas.

As tarefas do Amazon ECS podem ser categorizadas como tarefas de serviço ou tarefas autônomas. As tarefas de serviço são implantadas como parte de um serviço e controladas pela programação do Amazon ECS. Para obter mais informações, consulte Serviços do Amazon ECS. As tarefas autônomas são tarefas iniciadas pela API `RunTask` do Amazon ECS, diretamente ou por um agendador externo, como tarefas programadas (que são iniciadas pelo Amazon EventBridge), AWS Batch ou AWS Step Functions. Você não precisa realizar nenhuma ação em resposta à retirada de

tarefas de suas tarefas de serviço porque o programador do Amazon ECS substitui automaticamente as tarefas.

Para tarefas autônomas, talvez seja necessário realizar um tratamento adicional em resposta à retirada da tarefa. Para obter mais informações, consulte O Amazon ECS pode processar automaticamente as tarefas autônomas?.

Para tarefas de serviço, você não precisa realizar nenhuma ação para a retirada da tarefa, a menos que queira substituí-las antes de a AWS fazê-lo. Quando o agendador do Amazon ECS interrompe as tarefas, ele usa `maximumPercent` e executa uma nova tarefa na tentativa de manter a contagem desejada do serviço. Siga as práticas recomendadas para minimizar o impacto da inativação de tarefas. O valor padrão `maximumPercent` para um serviço que usa o agendador de serviço REPLICA é 200%. Portanto, quando o AWS Fargate inicia a retirada de tarefas, o Amazon ECS primeiro agenda uma nova tarefa e espera que ela seja executada, antes de retirar uma tarefa antiga. Quando o valor `maximumPercent` é definido como 100%, o Amazon ECS interrompe a tarefa primeiro e, em seguida, a substitui.

Para a retirada autônoma da tarefa, a AWS interrompe a tarefa na data de retirada da tarefa ou após ela. O Amazon ECS não inicia uma tarefa de substituição quando uma tarefa é interrompida. Se precisar que essas tarefas continuem em execução, é necessário interromper a execução delas e iniciar uma tarefa substituta antes da hora indicada na notificação. Portanto, recomendamos que os clientes monitorem o estado das tarefas autônomas e, se necessário, implementem a lógica para substituir as tarefas interrompidas.

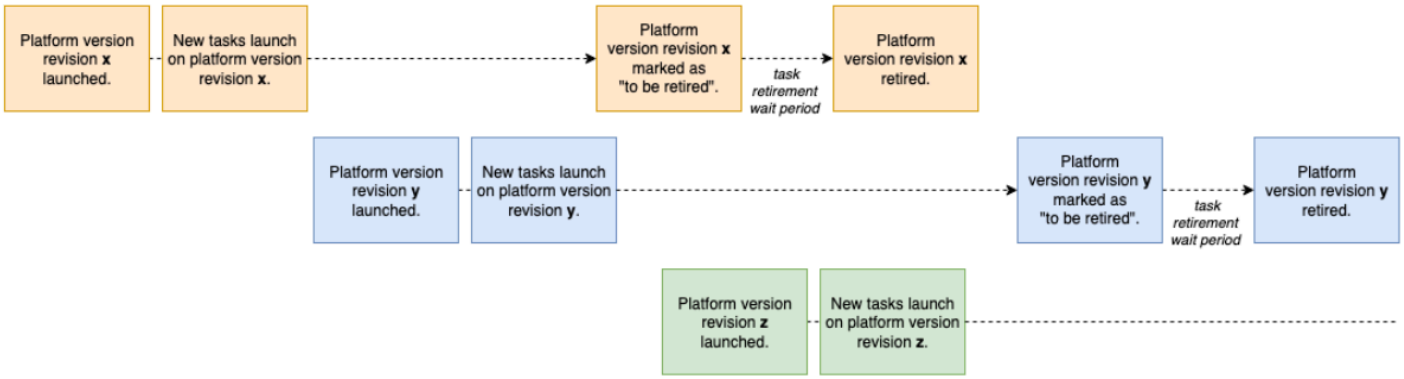
Quando uma tarefa é interrompida em qualquer um dos cenários, é possível executar `describe-tasks`. O `stoppedReason` na resposta é `ECS is performing maintenance on the underlying infrastructure hosting the task`.

A manutenção de tarefas se aplica quando uma revisão da nova versão da plataforma precisa ser substituída por uma nova. Se houver um problema com um host subjacente do Fargate, o Amazon ECS substitui o host sem um aviso de retirada da tarefa.

Visão geral do aviso de retirada de tarefa

Quando a AWS marca uma revisão da versão da plataforma como precisando ser retirada, identificamos todas as tarefas que estão sendo executadas nessa revisão da versão da plataforma em todas as regiões. Em seguida, enviamos uma notificação por conta por região, destacando as tarefas ou serviços afetados e a data em que as retiradas começarão a ocorrer.

A ilustração a seguir mostra o ciclo de vida de uma revisão da versão da plataforma Fargate, desde o lançamento de uma nova revisão até a retirada da revisão da plataforma.



As informações a seguir fornecem detalhes.

- Depois que uma nova revisão da versão da plataforma é lançada, todas as novas tarefas são agendadas nessa revisão.
- As tarefas existentes que foram agendadas e executadas permanecem na revisão em que foram originalmente colocadas durante a tarefa e não são migradas para a nova revisão.
- Novas tarefas, por exemplo, como parte de uma atualização de um serviço ou da retirada de tarefas do Fargate, são inseridas na revisão mais recente da versão da plataforma disponível no momento do lançamento.

As notificações de retirada de tarefa são enviadas por meio do Painel do AWS Health e para o endereço de e-mail registrado e incluem as seguintes informações:

- A data de retirada da tarefa: a tarefa é interrompida nessa data ou após essa data.
- Para tarefas autônomas, os IDs das tarefas.
- Para tarefas de serviço, o ID do cluster em que o serviço é executado e os IDs do serviço.
- As próximas etapas que você precisa seguir.

Normalmente, enviamos uma notificação de serviço e tarefas autônomas em cada Região da AWS. No entanto, em alguns casos, você pode receber mais de um evento para cada tipo de tarefa, por exemplo, quando há muitas tarefas a serem retiradas que ultrapassarão os limites nos mecanismos de notificação.

É possível identificar tarefas programadas para retirada das maneiras a seguir:

- A AWS Health Dashboard

As notificações AWS Health podem ser enviadas por meio do Amazon EventBridge para que um armazenamento de arquivamento, como o Amazon Simple Storage Service, execute ações automatizadas, como executar uma função do AWS Lambda, ou outros sistemas de notificação, como o Amazon Simple Notification Service. Para obter mais informações, consulte Monitoramento de eventos do AWS Health com o Amazon EventBridge. Para obter um exemplo de configuração do envio de notificações para o Amazon Chime, Slack, ou Microsoft Teams, consulte o repositório AWS Health Aware no GitHub.

Veja a seguir um exemplo de evento do EventBridge.


```
{
    "version": "0",
    "id": "3c268027-f43c-0171-7425-1d799EXAMPLE",
    "detail-type": "AWS Health Event",
    "source": "aws.health",
    "account": "123456789012",
    "time": "2023-08-16T23:18:51Z",
    "region": "us-east-1",
    "resources": [
        "cluster|service",
        "cluster|service"
    ],
    "detail": {
        "eventArn": "arn:aws:health:us-east-1::event/ECS/
AWS_ECS_TASK_PATCHING_RETIREMENT/AWS_ECS_TASK_PATCHING_RETIREMENT_test1",
        "service": "ECS",
        "eventScopeCode": "ACCOUNT_SPECIFIC",
        "communicationId":
 "7988399e2e6fb0b905ddc88e0e2de1fd17e4c9fa60349577446d95a18EXAMPLE",
        "lastUpdatedTime": "Wed, 16 Aug 2023 23:18:52 GMT",
        "eventRegion": "us-east-1",
        "eventTypeCode": "AWS_ECS_TASK_PATCHING_RETIREMENT",
        "eventTypeCategory": "scheduledChange",
        "startTime": "Wed, 16 Aug 2023 23:18:51 GMT",
        "endTime": "Fri, 18 Aug 2023 23:18:51 GMT",
        "eventDescription": [
            {
                "language": "en_US",
                "latestDescription": "\\nA software update has been deployed to
 Fargate which includes CVE patches or other critical patches. No action is required
```

```
 on your part. All new tasks launched automatically uses the latest software
 version. For existing tasks, your tasks need to be restarted in order for these
 updates to apply. Your tasks running as part of the following ECS Services will
 be automatically updated beginning Wed, 16 Aug 2023 23:18:51 GMT.\\n\\nAfter Wed,
 16 Aug 2023 23:18:51 GMT, the ECS scheduler will gradually replace these tasks,
 respecting the deployment settings for your service. Typically, services should
 see little to no interruption during the update and no action is required. When AWS
 stops tasks, AWS uses the minimum healthy percent (1) and launches a new task in
 an attempt to maintain the desired count for the service. By default, the minimum
 healthy percent of a service is 100 percent, so a new task is started first before
 a task is stopped. Service tasks are routinely replaced in the same way when
 you scale the service or deploy configuration changes or deploy task definition
 revisions. If you would like to control the timing of this restart you can update
 the service before Wed, 16 Aug 2023 23:18:51 GMT, by running the update-service
 command from the ECS command-line interface specifying force-new-deployment for
 services using Rolling update deployment type. For example:\\n\\n$ aws ecs update-
service -service service_name \\\n--cluster cluster_name -force-new-deployment\
\n\\nFor services using Blue/Green deployment type with AWS CodeDeploy:\\nPlease
 refer to create-deployment document (2) and create new deployment using same task
 definition revision.\\n\\nFor further details on ECS deployment types, please
 refer to ECS Deployment Developer Guide (1).\\nFor further details on Fargate's
 update process, please refer to the AWS Fargate User Guide (3).\\nIf you have
 any questions or concerns, please contact AWS Support (4).\\n\\n(1) https://
docs.aws.amazon.com/AmazonECS/latest/developerguide/deployment-types.html\\n(2)
 https://docs.aws.amazon.com/cli/latest/reference/deploy/create-deployment.html\\n(3)
 https://docs.aws.amazon.com/AmazonECS/latest/userguide/task-maintenance.html\\n(4)
 https://aws.amazon.com/support\\n\\nA list of your affected resources(s) can be
 found in the 'Affected resources' tab in the 'Cluster/ Service' format in the AWS
 Health Dashboard. \\n\\n"
            }
        ],
       "affectedEntities": [
                {
                    "entityValue": "arn:aws:ecs:eu-west-1:111222333444:task/
examplecluster/00805ce1d81940b5a37398e5a2c23333"
                },
                {
                    "entityValue": "arn:aws:ecs:eu-west-1:111222333444:task/
examplecluster/00805ce1d81940b5a37398e5a2c25555"
                }
            }
        ]
    }
```

```
}
```

- E-mail

  Um email é enviado para o email registrado para obter o ID Conta da AWS.

Para obter informações sobre como se preparar para a retirada de tarefas, consulte Prepare-se para a retirada de tarefas do AWS Fargate no Amazon ECS .

Posso optar pela não retirada de tarefas?

Não. Como parte do modelo de responsabilidade compartilhada da AWS, a AWS é responsável por gerenciar e manter a infraestrutura subjacente do AWS Fargate. Isso inclui a realização de atualizações periódicas da plataforma para garantir a segurança e a estabilidade. Essas atualizações são aplicadas automaticamente pela AWS e não são algo que os clientes possam optar por não aceitar. Esse é um dos principais benefícios de usar um AWS Fargate em comparação com a execução de suas workloads em instâncias do EC2, a responsabilidade pela manutenção da plataforma subjacente é assumida pela AWS. Esse modelo permite que você se concentre em suas aplicações em vez da manutenção da infraestrutura. Ao aplicar automaticamente essas atualizações da plataforma, a AWS é capaz de manter o ambiente do Fargate atualizado e seguro, sem nenhuma ação exigida de você como cliente. Isso ajuda a fornecer um ambiente em contêineres confiável e seguro para executar suas workloads no Fargate.

Posso receber notificações de retirada de tarefas por meio de outros serviços da AWS?

A AWS envia uma notificação de retirada da tarefa para o AWS Health Dashboard e para o principal contato de e-mail na Conta da AWS. O AWS Health Dashboard fornece várias integrações com outros serviços da AWS, incluindo o EventBridge. Você pode usar o EventBridge para automatizar a visibilidade dos avisos (por exemplo, encaminhar a mensagem para uma ferramenta de ChatOps). Para obter mais informações, consulte Solution overview: Capturing task retirement notifications.

Posso alterar a retirada de uma tarefa depois de programada?

Não. O cronograma é baseado no tempo de espera de retirada da tarefa, que tem um padrão de sete dias. Se precisar de mais tempo, você pode configurar o período de espera para 14 dias. Para obter mais informações, consulte Etapa 2: capturar as notificações de retiradas de tarefas para alertar as equipes e tomar medidas. A alteração nessa configuração se aplica às retiradas que

serão programadas no futuro. As retiradas programadas atualmente não são afetadas. Em caso de dúvidas, entre em contato com o Suporte.

Como o Amazon ECS processa tarefas que fazem parte de um serviço?

Para tarefas de serviço, você não precisa realizar nenhuma ação em resposta à retirada de tarefas, a menos que queira substituí-las antes de a AWS fazê-lo. Quando o agendador do Amazon ECS interrompe as tarefas, ele usa a porcentagem mínima de integridade e executa uma nova tarefa na tentativa de manter a contagem desejada do serviço. Para minimizar o impacto da retirada de tarefas do Fargate, você deve seguir as práticas recomendadas do Amazon ECS ao implantar workloads. Por exemplo, ao implantar uma aplicação sem estado como um serviço do Amazon ECS, como um servidor Web ou de API, os clientes devem implantar várias réplicas de tarefas e defini-las como minimumHealthyPercent a 100%. Por padrão, a porcentagem mínima de integridade de um serviço é 100%. Portanto, quando o Fargate inicia a retirada de tarefas, o Amazon ECS primeiro agenda uma nova tarefa e espera que ela seja executada, antes de retirar uma tarefa antiga. As tarefas de serviço são substituídas rotineiramente como parte da retirada de tarefas da mesma forma quando você escala o serviço, implanta alterações de configuração ou implanta revisões de definição de tarefas. Para se preparar para o processo de retirada de tarefas, consulte Prepare-se para a retirada de tarefas do AWS Fargate no Amazon ECS .

O Amazon ECS pode processar automaticamente as tarefas autônomas?

Não. A AWS não consegue criar uma tarefa substituta para tarefas autônomas iniciadas por `RunTask`, tarefas programadas (por exemplo, por meio do Agendador do EventBridge), AWS Batch ou AWS Step Functions. O Amazon ECS gerencia somente tarefas que fazem parte de um serviço.

Prepare-se para a retirada de tarefas do AWS Fargate no Amazon ECS

Para se preparar para a retirada da tarefa, execute as seguintes operações:

1. Defina o tempo de espera de retirada da tarefa.
2. Capture notificações de retirada de tarefas para notificar os membros da equipe.
3. Você não pode controlar o momento exato da retirada de uma tarefa, no entanto, você pode controlar a retirada de tarefas atualizando o serviço com a opção de implantação forçada.

Etapa 1: configurar o tempo de espera da tarefa

É possível configurar a hora em que o Fargate inicia a retirada da tarefa. Para workloads que exijam a aplicação imediata das atualizações, escolha a configuração imediata (`0`). Quando precisar de mais controle, por exemplo, quando uma tarefa só puder ser interrompida durante uma determinada janela, configure a opção de 7 dias (`7`) ou 14 dias (`14`).

Recomendamos que você escolha um período de espera mais curto para receber as revisões das versões da plataforma mais recentes mais cedo.

Configure o período de espera executando `put-account-setting-default` ou `put-account-setting` como usuário-raiz ou um usuário administrativo. Use a opção `fargateTaskRetirementWaitPeriod` para o `name` e a opção `value` definida como um dos valores a seguir:

- `0`: A AWS envia a notificação e imediatamente começa a retirar as tarefas afetadas.
- `7`: A AWS envia a notificação e aguarda 7 dias corridos antes de começar a retirar as tarefas afetadas.
- `14`: A AWS envia a notificação e aguarda 14 dias corridos antes de começar a retirar as tarefas afetadas.

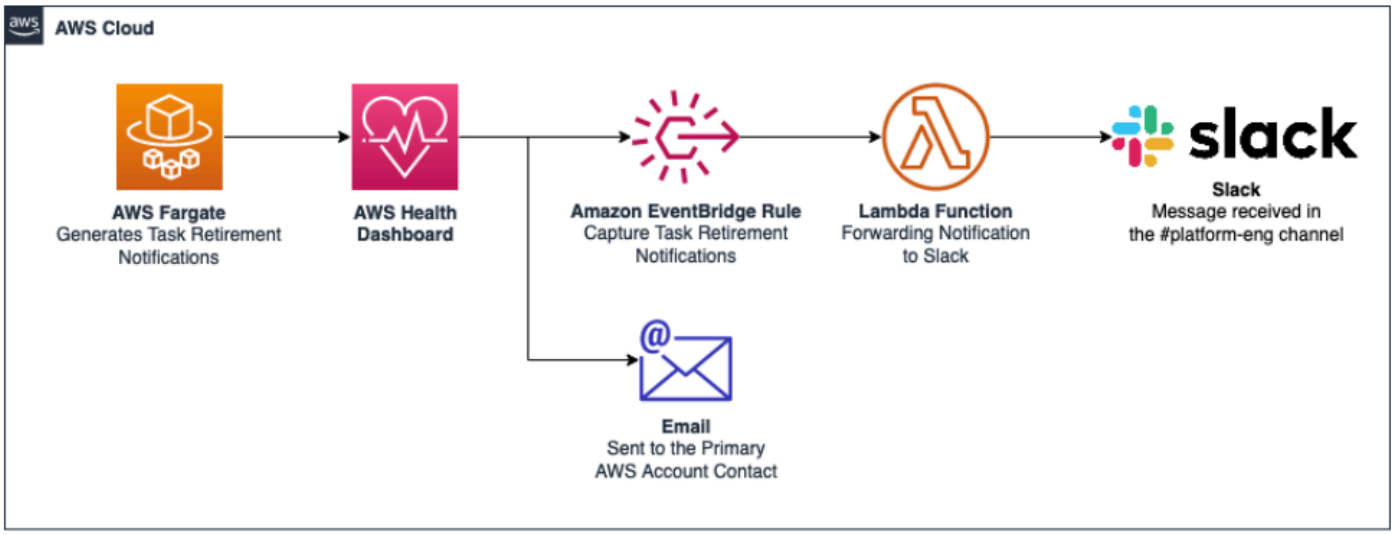
O padrão são 7 dias.

Para obter mais informações, consulte put-account-setting-default e put-account-setting na Referência de API do Amazon Elastic Container Service.

Etapa 2: capturar as notificações de retiradas de tarefas para alertar as equipes e tomar medidas

Quando houver um retirada de tarefa próxima, a AWS envia uma notificação de retirada de tarefa para o Painel do AWS Health e para o contato de e-mail principal na Conta da AWS. O Painel do AWS Health fornece várias integrações com outros serviços da AWS, incluindo o Amazon EventBridge. É possível usar o EventBridge para criar automações a partir de uma notificação de retirada de tarefa, como aumentar a visibilidade da próxima retirada, ao encaminhar a mensagem para uma ferramenta ChatOps. AWS Health O Aware é um recurso que mostra o poder do Painel do AWS Health e como as notificações podem ser distribuídas por toda a organização. Você pode encaminhar uma notificação de retirada de tarefa para um aplicativo de bate-papo, como o Slack.

A ilustração a seguir mostra a visão geral da solução.



As informações a seguir fornecem detalhes.

- O Fargate envia a notificação de retirada da tarefa para o Painel do AWS Health.
- O Painel do AWS Health envia e-mail para o contato de e-mail principal na Conta da AWS e notifica o EventBridge.
- O EventBridge tem uma regra que captura a notificação de retirada.

  A regra de busca de eventos com o tipo de detalhe do evento: `"AWS Health Event" and the Event Detail Type Code: "AWS_ECS_TASK_PATCHING_RETIREMENT"`
- A regra aciona uma função do Lambda que encaminha as informações para o Slack usando um webhook de entrada do Slack. Para obter mais informações, consulte Webhooks de entrada.

Para ver um exemplo de código, consulte Capturing AWS Fargate Task Retirement Notifications no Github.

Etapa 3: controlar a retirada de tarefas

Você não pode controlar o momento exato da retirada de uma tarefa, no entanto, você pode definir um tempo de espera. Se quiser controlar a retirada de tarefas de acordo com sua própria programação, é possível capturar o aviso de retirada da tarefa para primeiro entender a data de retirada da tarefa. Você pode então reimplantar seu serviço para iniciar retiradas de tarefas e, da mesma forma, substituir quaisquer tarefas autônomas. Para serviços que usam implantação

contínua, você atualiza o serviço usando `update-service` com a opção `force-deployment` antes do horário de início da retirada.

O exemplo `update-service` a seguir usa a opção `force-deployment`.

```
aws ecs update-service –-service service_name \
    --cluster cluster_name \
     --force-new-deployment
```

Em serviços que usam a implantação azul/verde, você precisa criar uma implantação no AWS CodeDeploy. Para obter informações sobre como criar a implantação, consulte create-deployment na Referência da AWS Command Line Interface.

Regiões com suporte para Amazon ECS no AWS Fargate

Você pode usar as tabelas a seguir para verificar o suporte regional para contêineres do Linux no AWS Fargate e contêineres do Windows no AWS Fargate.

Contêineres do Linux no AWS Fargate

Os contêineres do Linux do Amazon ECS no AWS Fargate são compatíveis com as seguintes Regiões da AWS. Os IDs das zonas de disponibilidade compatíveis são anotados quando aplicável.

Nome da região	Região
Leste dos EUA (Ohio)	us-east-2
Leste dos EUA (Norte da Virgínia)	us-east-1
Oeste dos EUA (Norte da Califórnia)	us-west-1 (somente `usw1-az1` e `usw1-az3`)
Oeste dos EUA (Oregon)	us-west-2
Canadá (Central)	ca-central-1
Oeste do Canadá (Calgary)	ca-west-1
México (Central)	mx-central-1
África (Cidade do Cabo)	af-south-1

Nome da região	Região
Ásia-Pacífico (Hong Kong)	ap-east-1
Ásia-Pacífico (Mumbai)	ap-south-1
Ásia-Pacífico (Tóquio)	ap-northeast-1 (somente `apne1-az1` , `apne1-az2` e `apne1-az4`)
Ásia-Pacífico (Seul)	ap-northeast-2
Ásia-Pacífico (Osaka)	ap-northeast-3
Ásia-Pacífico (Hyderabad)	ap-south-2
Ásia-Pacífico (Singapura)	ap-southeast-1
Ásia-Pacífico (Sydney)	ap-southeast-2
Ásia-Pacífico (Tailândia)	ap-southeast-7
Ásia-Pacífico (Jacarta)	ap-southeast-3
Ásia-Pacífico (Melbourne)	ap-southeast-4
Ásia-Pacífico (Malásia)	ap-southeast-5
Canadá (Central)	ca-central-1
Oeste do Canadá (Calgary)	ca-west-1
China (Pequim)	cn-north-1 (somente `cnn1-az1` e `cnn1-az2`)
China (Ningxia)	cn-northwest-1
Europa (Frankfurt)	eu-central-1
Europa (Zurique)	eu-central-2
Europa (Irlanda)	eu-west-1
Europa (Londres)	eu-west-2

Nome da região	Região
Europa (Paris)	eu-west-3
Europa (Milão)	eu-south-1
Europa (Espanha)	eu-south-2
Europa (Estocolmo)	eu-north-1
América do Sul (São Paulo)	sa-east-1
Israel (Tel Aviv)	il-central-1
Oriente Médio (Barém)	me-south-1
Oriente Médio (Emirados Árabes Unidos)	me-central-1
AWS GovCloud (Leste dos EUA)	us-gov-east-1
AWS GovCloud (Oeste dos EUA)	us-gov-west-1

Contêineres do Windows no AWS Fargate

Os contêineres do Windows do Amazon ECS no AWS Fargate são compatíveis com as seguintes Regiões da AWS. Os IDs das zonas de disponibilidade compatíveis são anotados quando aplicável.

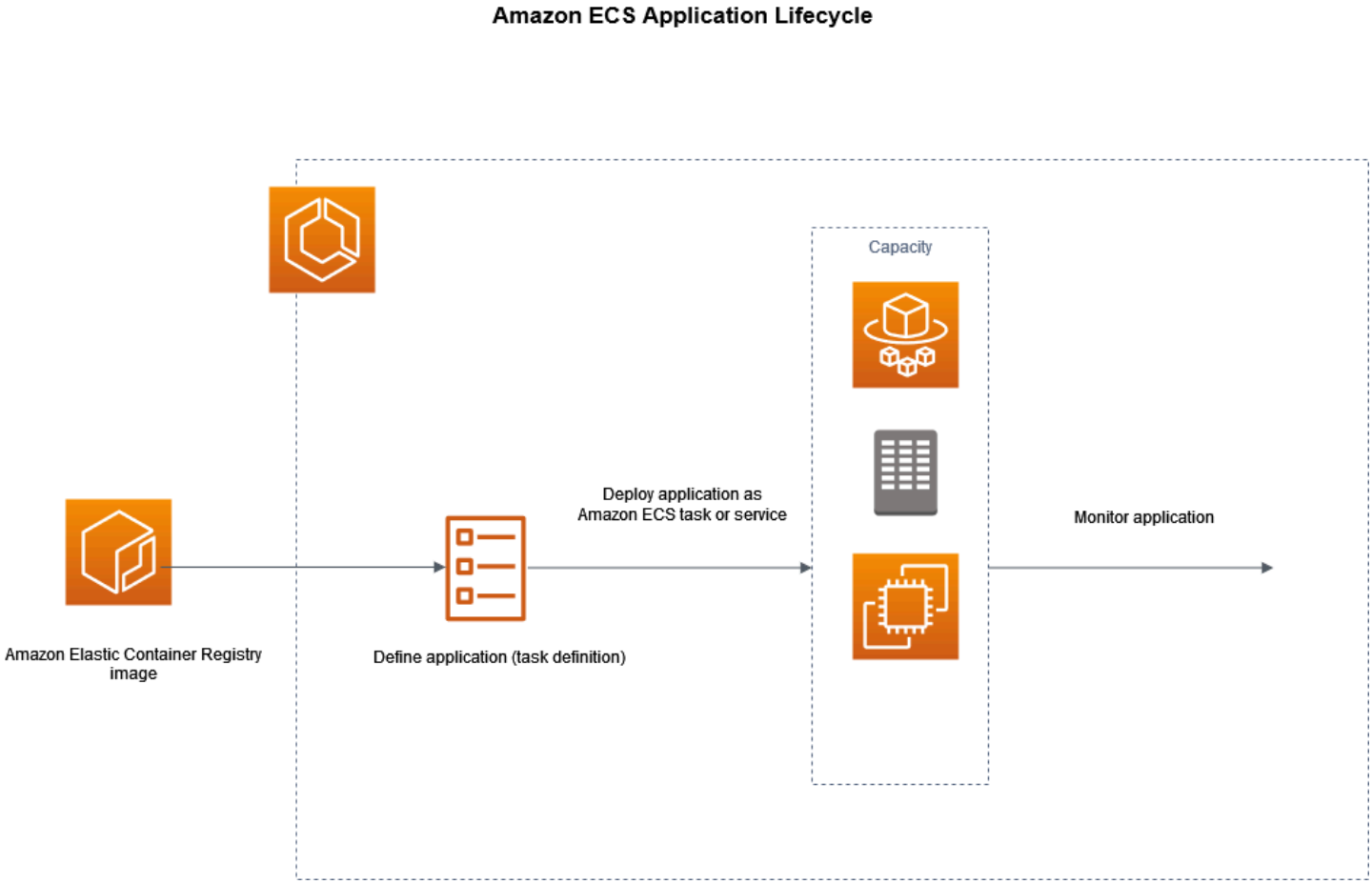
Nome da região	Região
Leste dos EUA (Ohio)	us-east-2
Leste dos EUA (Norte da Virgínia)	us-east-1 (somente `use1-az1`, `use1-az2`, `use1-az4`, `use1-az5` e `use1-az6`)
Oeste dos EUA (Norte da Califórnia)	us-west-1 (somente `usw1-az1` e `usw1-az3`)
Oeste dos EUA (Oregon)	us-west-2
África (Cidade do Cabo)	af-south-1

Nome da região	Região
Ásia-Pacífico (Hong Kong)	ap-east-1
Ásia-Pacífico (Mumbai)	ap-south-1
Ásia-Pacífico (Hyderabad)	ap-south-2
Ásia-Pacífico (Osaka)	ap-northeast-3
Ásia-Pacífico (Seul)	ap-northeast-2
Ásia-Pacífico (Singapura)	ap-southeast-1
Ásia-Pacífico (Sydney)	ap-southeast-2
Ásia-Pacífico (Melbourne)	ap-southeast-4
Ásia-Pacífico (Malásia)	ap-southeast-5
Ásia-Pacífico (Tóquio)	ap-northeast-1 (somente `apne1-az1` , `apne1-az2` e `apne1-az4`)
Canadá (Central)	ca-central-1 (somente `cac1-az1` e `cac1-az2`)
Oeste do Canadá (Calgary)	ca-west-1
China (Pequim)	cn-north-1 (somente `cnn1-az1` e `cnn1-az2`)
China (Ningxia)	cn-northwest-1
Europa (Frankfurt)	eu-central-1
Europa (Zurique)	eu-central-2
Europa (Irlanda)	eu-west-1
Europa (Londres)	eu-west-2
Europa (Paris)	eu-west-3
Europa (Milão)	eu-south-1

Nome da região	Região
Europa (Espanha)	eu-south-2
Europa (Estocolmo)	eu-north-1
América do Sul (São Paulo)	sa-east-1
Israel (Tel Aviv)	il-central-1
Oriente Médio (Emirados Árabes Unidos)	me-central-1
Oriente Médio (Bahrein)	me-south-1

Arquitetar a solução para o Amazon ECS

O diagrama a seguir mostra o ciclo de vida da aplicação e como ela funciona com os componentes do Amazon ECS.



Você deve arquitetar sua aplicações para que possam ser executadas em contêineres. Um contêiner é uma unidade padronizada de desenvolvimento de software que contém tudo o que sua aplicação de software precisa para ser executada. Isso inclui código relevante, runtime, ferramentas do sistema e bibliotecas do sistema. Os contêineres são criados a partir de um modelo somente leitura chamado imagem. Normalmente, as imagens são criadas a partir de um Dockerfile. O Dockerfile é um arquivo de texto simples que contém as instruções para criar um contêiner. Depois de serem compiladas, essas imagens são armazenadas em um registro, como o Amazon ECR, de onde elas podem ser baixadas.

Depois de criar e armazenar sua imagem, é possível criar uma definição de tarefa do Amazon ECS. Uma definição de tarefa é como um esquema para sua aplicação. É um arquivo de texto em formato JSON que descreve os parâmetros e um ou mais contêineres que formam sua aplicação.

Por exemplo, é possível usar isso para especificar parâmetros para o sistema operacional, os contêineres a serem usados, as portas a serem abertas para sua aplicação e os volumes de dados a serem usados com os contêineres na tarefa. Os parâmetros específicos disponíveis para sua definição de tarefa dependem das necessidades da aplicação específica.

Depois de configurar sua definição de tarefa, você a implanta como um serviço ou uma tarefa em seu cluster. Um cluster é um agrupamento lógico de tarefas ou serviços executados na infraestrutura de capacidade registrada em um cluster.

Uma tarefa é a instanciação de uma definição de tarefa dentro de um cluster. É possível executar uma tarefa autônoma ou executar uma tarefa como parte de um serviço. É possível usar um serviço do Amazon ECS para executar e manter simultaneamente o número desejado de tarefas em um cluster do Amazon ECS. Ele funciona de forma que, se qualquer uma de suas tarefas falharem ou pararem por algum motivo, o programador de serviço do Amazon ECS iniciará outra instância com base na sua definição de tarefa. Ele faz isso para substituí-la e, assim, manter o número desejado de tarefas no serviço.

O agente de contêiner é executado em cada instância de contêiner em um cluster do Amazon ECS. O agente envia para o Amazon ECS informações sobre as atuais tarefas em execução e a utilização dos recursos dos seus contêineres. Ele inicia e interrompe tarefas sempre que recebe uma solicitação do Amazon ECS.

Depois de implantar a tarefa ou o serviço, é possível usar qualquer uma das ferramentas a seguir para monitorar a implantação e a aplicação:

- CloudWatch
- Monitoramento de runtime

Capacidade

A capacidade é a infraestrutura onde os contêineres são executados. Estas são as opções:

- Instâncias do Amazon EC2
- Tecnologia sem servidor (AWS Fargate)
- Máquinas virtuais (VM) ou servidores on-premises

Você especifica a infraestrutura ao criar um cluster. Você também especifica o tipo de infraestrutura ao registrar uma definição de tarefa. A definição da tarefa refere-se à infraestrutura como o “tipo de

execução". Você também usa o tipo de execução ao executar uma tarefa autônoma ou implantar um serviço. Para obter informações sobre as opções de tipo de execução, consulte Tipos de inicialização do Amazon ECS.

Service endpoints

O endpoint do serviço é o URL do ponto de entrada para o Amazon ECS que você usa para se conectar ao serviço programaticamente usando o Internet Protocol versão 4 (IPv4) ou o Internet Protocol versão 6 (IPv6). Por padrão, as solicitações que você faz para se conectar ao Amazon ECS programaticamente usam endpoints de serviço que oferecem suporte somente a solicitações IPv4. Para se conectar programaticamente ao Amazon ECS usando solicitações IPv4 ou IPv6, é possível usar um endpoint de pilha dupla. Para obter informações sobre o uso de endpoints de pilha dupla, consulte Usar endpoints de pilha dupla do Amazon ECS.

Redes

Os recursos da AWS são criados em sub-redes. Ao usar instâncias do EC2, o Amazon ECS executa as instâncias na sub-rede que você especifica ao criar um cluster. Suas tarefas são executadas na sub-rede da instância. No Fargate ou em máquinas virtuais on-premises, você especifica a sub-rede ao executar uma tarefa ou criar um serviço.

Dependendo da aplicação, a sub-rede pode ser privada ou pública e estar em qualquer um dos seguintes recursos da AWS:

- Uma VPC com zonas de disponibilidade e uma zona Wavelength.
- Local Zones
- Zonas do Wavelength
- Regiões da AWS
- AWS Outposts

Para ter mais informações, consulte Aplicações do Amazon ECS em sub-redes compartilhadas, zonas locais e zonas do Wavelength ou Amazon Elastic Container Service no AWS Outposts.

Você pode fazer com que a aplicação se conecte à Internet usando um dos seguintes métodos:

- Uma sub-rede pública com um gateway da Internet

Use sub-redes públicas quando tiver aplicações públicas que exijam grandes quantidades de largura de banda ou latência mínima. Os cenários aplicáveis incluem serviços de streaming de vídeo e jogos.

- Uma sub-rede privada com um gateway NAT

  Use sub-redes privadas quando quiser proteger seus contêineres do acesso externo direto. Os cenários aplicáveis incluem sistemas de processamento de pagamentos ou contêineres que armazenam dados e senhas do usuário.

- AWS PrivateLink

  Use o AWS PrivateLink para usufruir de conectividade privada entre VPCs, serviços da AWS e suas redes on-premises sem expor seu tráfego à Internet pública.

Acesso a recursos

Você pode usar as configurações da sua conta do Amazon ECS para acessar os seguintes recursos:

- Container Insights

  O CloudWatch Container Insights coleta, agrega e resume métricas e logs das suas aplicações e microsserviços conteinerizados. As métricas incluem a utilização de recursos, como CPU, memória, disco e rede.

- Truncamento de `awsvpc`

  Em determinados tipos de instâncias do EC2, você pode ter interfaces de rede (ENIs) adicionais disponíveis em instâncias de contêiner recém-executadas.

- Autorização para atribuição de tags

  Os usuários devem ter permissões para ações que criam um recurso, como `ecsCreateCluster`. Se as tags forem especificadas na ação resource-creating, a AWS executará autorização adicional na ação `ecs:TagResource` para verificar se os usuários têm permissões para criar tags.

- Conformidade com o Fargate FIPS-140

  O Fargate oferece suporte ao Padrão Federal de Processamento de Informações (FIPS-140), que especifica os requisitos de segurança para módulos criptográficos que protejam informações confidenciais. É o padrão atual dos governos dos Estados Unidos e do Canadá e é aplicável a sistemas que precisam estar em conformidade com a Lei Federal de Gerenciamento de Segurança

da Informação (FISMA) ou com o Programa Federal de Gerenciamento de Riscos e Autorizações (FedRAMP).

- Alterações no tempo de retirada de tarefas do Fargate

  É possível configurar o período de espera antes que as tarefas do Fargate sejam retiradas para aplicação de patches.

- VPC de pilha dupla

  Permita que as tarefas se comuniquem por IPv4, IPv6 ou ambos.

- Formato do nome do recurso da Amazon (ARN)

  Alguns recursos, como autorização de marcação, exigem um novo formato de nome do recurso da Amazon (ARN).

Para obter mais informações, consulte Acesso aos recursos do Amazon ECS com as configurações de conta.

Perfis do IAM

Um perfil do IAM é uma identidade do IAM que você pode criar em sua conta que tem permissões específicas. No Amazon ECS, você pode criar perfis para conceder permissões aos recursos do Amazon ECS, como contêineres ou serviços.

Alguns recursos do Amazon ECS exigem perfis. Para obter mais informações, consulte Perfis do IAM para o Amazon ECS.

Registro em log

O registro em log e o monitoramento são aspectos importantes para manter a confiabilidade, a disponibilidade e o desempenho das workloads do Amazon ECS. As seguintes opções estão disponíveis:

- Logs do Amazon CloudWatch: encaminha os logs para o Amazon CloudWatch
- FireLens para Amazon ECS: encaminha os logs para um serviço da AWS ou destino da AWS Partner Network para armazenamento e análise de log. O AWS Partner Network é uma comunidade global de parceiros que utiliza programas, experiência e recursos para criar, comercializar e vender ofertas aos clientes.

Tipos de inicialização do Amazon ECS

O tipo de execução da definição de tarefa define em qual capacidade ela pode ser executada, por exemplo, AWS Fargate.

Depois de escolher o tipo de execução, o Amazon ECS verifica se os parâmetros de definição de tarefas configurados funcionam com o tipo de execução.

A tabela a seguir descreve os tipos de inicialização disponíveis.

Tipo de inicialização		Saiba mais
Fargate	O Fargate é um mecanismo de computação com tecnologia sem servidor e pagamento conforme o uso que permite se concentrar na criação de aplicações sem gerenciar servidores.	Tipo de inicialização do Fargate para o Amazon ECS
EC2	Você escolhe o tipo de instância, o número de instâncias e gerencia a capacidade.	Tipo de inicialização do EC2 para o Amazon ECS
Máquinas virtuais (VM) ou servidores on-premises	O Amazon ECS Anywhere permite registrar uma Instância externa, como um servidor on-premises ou uma máquina virtual (VM), no cluster.	Tipo de inicialização externa (Amazon ECS Anywhere) para o Amazon ECS

Tipo de inicialização do Fargate para o Amazon ECS

O Fargate é um mecanismo de computação com tecnologia sem servidor e pagamento conforme o uso que permite se concentrar na criação de aplicações sem gerenciar servidores. Ao escolher o Fargate, não é necessário gerenciar uma infraestrutura do EC2. Tudo o que você precisa fazer é

criar a imagem de contêiner e definir em qual cluster deseja executar as aplicações. O Fargate tem integração nativa com serviços da AWS que incluem:

- Amazon VPC
- Auto Scaling
- Elastic Load Balancing
- IAM
- Secrets Manager

Você tem mais controle com o Fargate do que com o EC2 porque seleciona a CPU e a memória exatas de que a aplicação precisa. O Fargate processa o aumento horizontal de escala da capacidade para que você não precise se preocupar com picos de tráfego. Isso significa que há menos esforço operacional com o Fargate.

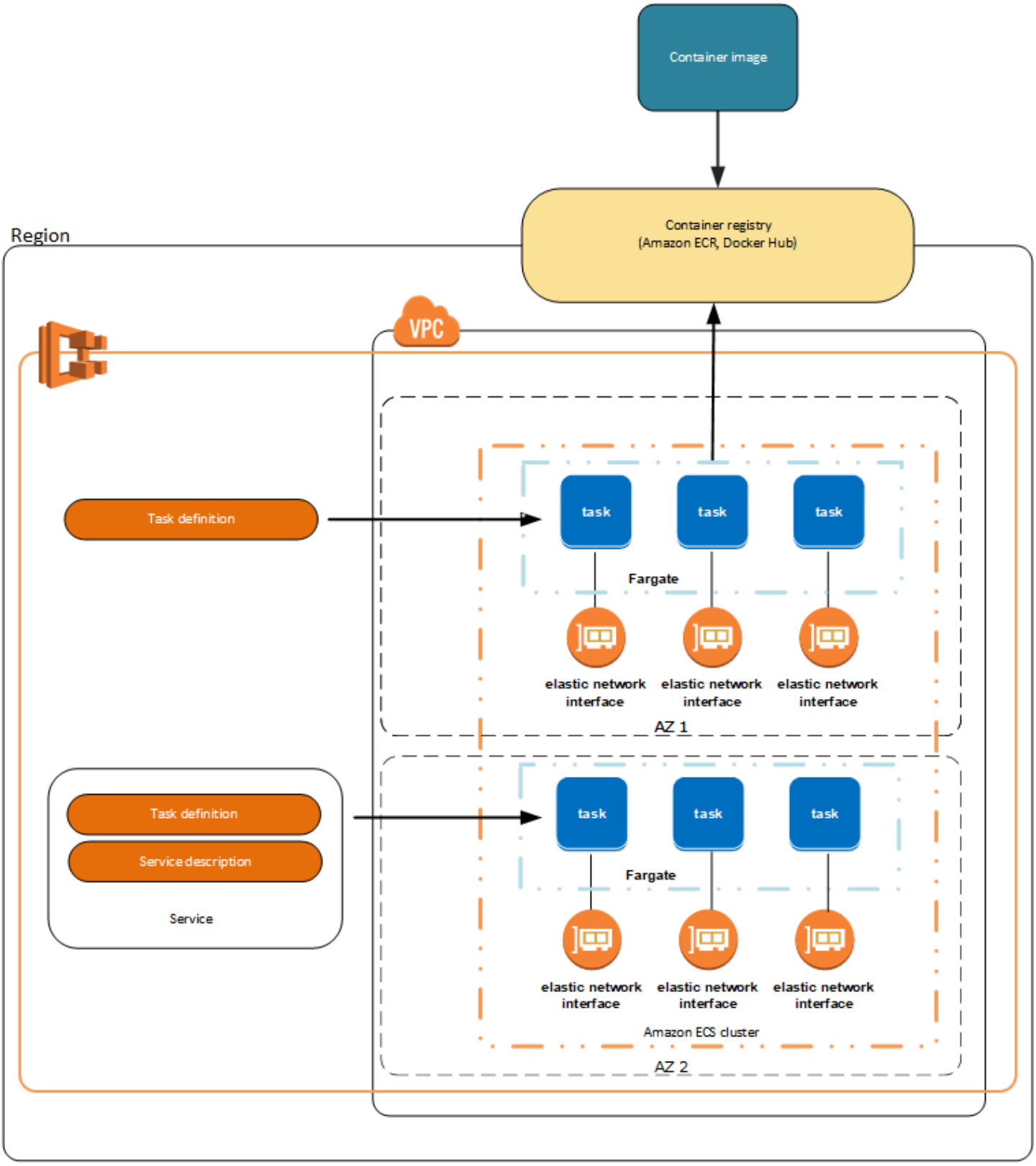
O Fargate atende aos padrões de programas de conformidade, incluindo PCI, FIPS 140-3, FedRAMP e HIPAA. Para obter mais informações, consulte Serviços da AWS no escopo por programa de conformidade.

O Fargate é adequado para as workloads a seguir:

- Workloads grandes que precisam de baixa sobrecarga operacional
- Pequenas workloads que têm explosão ocasional
- Workloads pequenas
- Workloads em batch

Para obter informações sobre as regiões que oferecem suporte ao Fargate, consulte the section called “Regiões do AWS Fargate”.

O diagrama a seguir mostra a arquitetura geral.



Para obter mais informações sobre o Amazon ECS no Fargate, consulte [AWS Fargate para o Amazon ECS](#).

Comportamento dos contêineres Linux na extração de imagens de contêiner do Fargate para o Amazon ECS

Cada tarefa do Fargate é executada em sua própria instância de uso e de locatário únicos. Quando você executa contêineres do Linux no Fargate, as imagens de contêiner ou as camadas de imagens de contêiner não são armazenadas em cache na instância. Portanto, para cada imagem de contêiner definida na tarefa, a imagem de contêiner, em sua totalidade, precisa ser extraída do registro de imagem de contêiner para cada tarefa do Fargate. O tempo necessário para extrair as imagens está diretamente relacionado ao tempo dedicado para iniciar uma tarefa do Fargate.

Considere as informações apresentadas a seguir para otimizar o tempo de extração de imagens.

Proximidade da imagem de contêiner

Para reduzir o tempo necessário para efetuar o download das imagens de contêiner, localize os dados o mais próximo possível da computação. A extração de uma imagem de contêiner usando a Internet ou entre Regiões da AWS pode afetar o tempo de download. Recomendamos armazenar a imagem de contêiner na mesma região em que a tarefa será executada. Se você armazenar a imagem de contêiner no Amazon ECR, use um endpoint da VPC de interface para reduzir ainda mais o tempo de extração da imagem. Para obter mais informações, consulte Amazon ECR interface VPC endpoints (AWS PrivateLink) no Guia do usuário do Amazon ECR.

Redução do tamanho da imagem de contêiner

O tamanho de uma imagem de contêiner afeta diretamente o tempo de download. Reduzir o tamanho da imagem de contêiner ou o número de camadas da imagem de contêiner pode reduzir o tempo de download de uma imagem. Imagens base leves (como a imagem de contêiner mínima do Amazon Linux 2023) podem ser significativamente menores do que aquelas baseadas em imagens base do sistema operacional tradicional. Para obter mais informações sobre a imagem mínima, consulte AL2023 Minimal container image no Guia do usuário do Amazon Linux 2023.

Algoritmos de compactação alternativos

Geralmente, as camadas de imagem de contêiner são compactadas quando enviadas para um registro de imagem de contêiner. A compactação da camada de imagem de contêiner reduz a quantidade de dados que precisam ser transferidos pela rede e armazenados no registro de imagens de contêiner. Depois que uma camada de imagem de contêiner for baixada para uma instância pelo runtime do contêiner, essa camada será descompactada. O algoritmo de compactação usado e a quantidade de vCPUs disponíveis para o runtime afetam o tempo dedicado para descompactar a imagem de contêiner. No Fargate, é possível aumentar o tamanho

da tarefa ou aproveitar o algoritmo de compactação zstd de maior desempenho para reduzir o tempo necessário para a descompactação. Para obter mais informações, consulte ztsd no GitHub. Para obter informações sobre como implementar as imagens para o Fargate, consulte Reducing AWS Fargate Startup Times with zstd Compressed Container Images.

Imagens de contêiner de carregamento lento

Para imagens de contêiner grandes (maiores do que 250 MB), pode ser ideal realizar o carregamento lento de uma imagem de contêiner em vez de efetuar o download da imagem de contêiner em sua totalidade. No Fargate, é possível usar o Seekable OCI (SOCI) para realizar o carregamento lento de uma imagem de contêiner de um registro de imagem de contêiner. Para obter mais informações, consulte soci-snapshotter no GitHub e Carregamento lento de imagens de contêiner usando Seekable OCI (SOCI).

Comportamento dos contêineres Windows na extração de imagens de contêiner do Fargate para o Amazon ECS

O Windows para Fargate armazena em cache a imagem de base do núcleo do servidor do mês mais recente e do mês anterior fornecida pela Microsoft. Essas imagens correspondem aos patches de KB ou número de compilação atualizados a cada Patch Tuesday. Por exemplo, em 9 de abril de 2024, a Microsoft liberou a atualização KB5036896 (17763.5696) para o Windows Server 2019. A atualização KB do mês anterior, liberada em 12 de março de 2024, foi a KB5035849 (17763.5576). Portanto, para as plataformas `WINDOWS_SERVER_2019_CORE` e `WINDOWS_SERVER_2019_FULL` as seguintes imagens de contêiner foram armazenadas em cache:

- `mcr.microsoft.com/windows/servercore:ltsc2019`
- `mcr.microsoft.com/windows/servercore:10.0.17763.5696`
- `mcr.microsoft.com/windows/servercore:10.0.17763.5576`

Além disso, em 9 de abril de 2024, a Microsoft liberou a atualização KB5036909 (20348.2402) para o Windows Server 2022. A atualização KB do mês anterior, liberada em 12 de março de 2024, foi a KB5035857 (20348.2340). Portanto, nas plataformas `WINDOWS_SERVER_2022_CORE` e `WINDOWS_SERVER_2022_FULL`, as seguintes imagens de contêiner foram armazenadas em cache:

- `mcr.microsoft.com/windows/servercore:ltsc2022`
- `mcr.microsoft.com/windows/servercore:10.0.20348.2402`
- `mcr.microsoft.com/windows/servercore:10.0.20348.2340`

Tipo de inicialização do EC2 para o Amazon ECS

O tipo de inicialização do EC2 é adequado para workloads grandes que devem ter preço otimizado.

Ao considerar como modelar definições de tarefa e serviços usando o tipo de inicialização do EC2, recomendamos que você considere quais processos devem ser executados juntos e como você pode dimensionar cada componente.

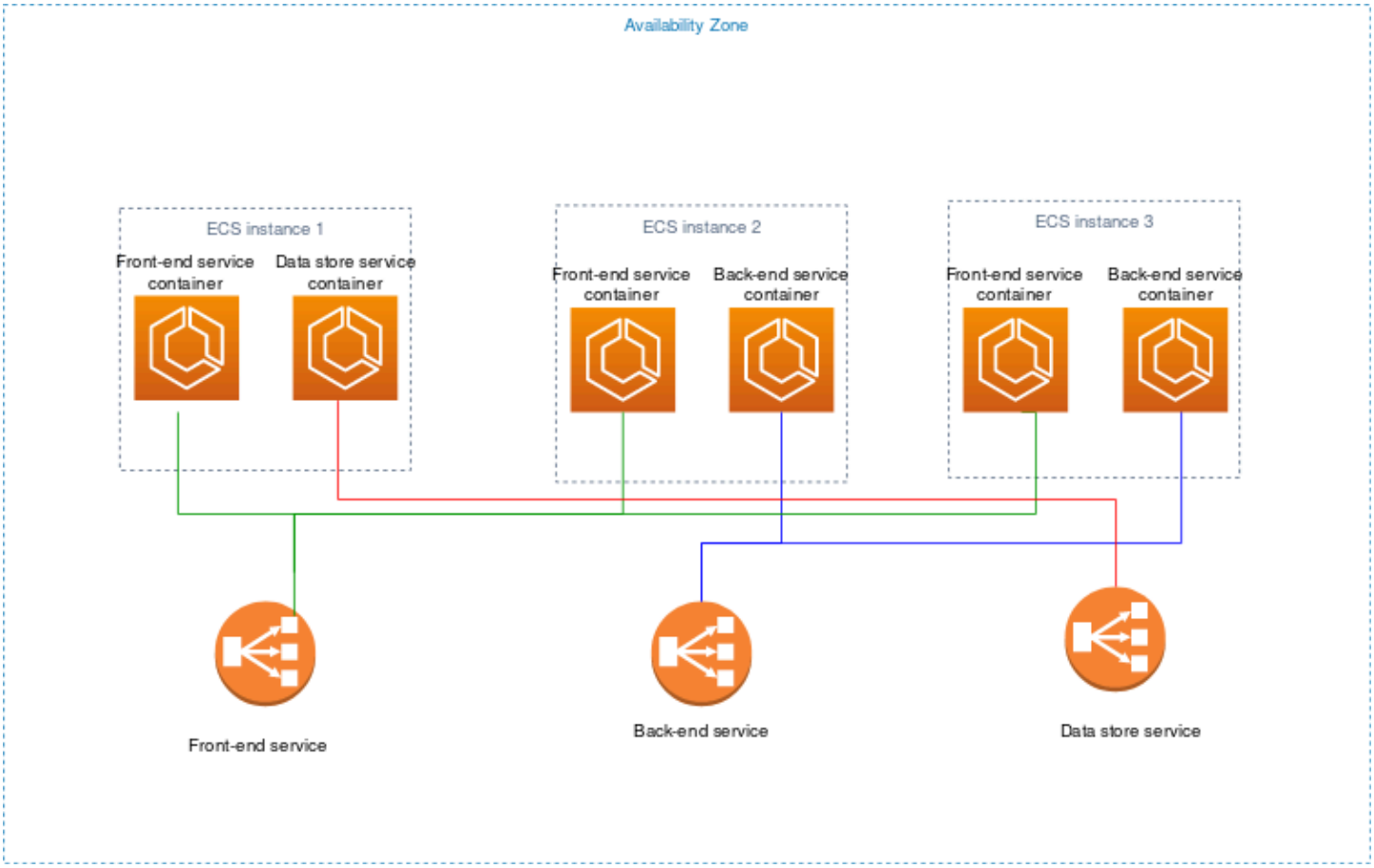
Por exemplo, suponha que uma aplicação seja formada pelos seguintes componentes:

- Um serviço frontend que exiba informações em uma página da web
- Um serviço backend que forneça APIs para o serviço frontend
- Um armazenamento físico de dados

Neste exemplo, crie definições de tarefa que agrupem contêineres utilizados para um propósito em comum. Faça a separação dos componentes distintos em várias definições de tarefa separadas. O cluster de exemplo abaixo tem três instâncias de contêiner que executam três contêineres de serviços frontend, dois contêineres de serviço backend e um contêiner de serviço de armazenamento físico de dados.

É possível agrupar contêineres relacionados em uma definição de tarefa, como contêineres vinculados que devem ser executados juntos. Por exemplo, adicione um contêiner de transmissão de log ao serviço de fron-end e inclua-o na mesma definição de tarefa.

Depois que tiver as definições de tarefa, será possível criar serviços com base nelas para manter a disponibilidade das tarefas desejadas. Para obter mais informações, consulte Criação de uma implantação de atualização contínua do Amazon ECS. Nos seus serviços, você pode associar contêineres a balanceadores de carga do Elastic Load Balancing. Para obter mais informações, consulte Uso do balanceamento de carga para distribuir o tráfego de serviço do Amazon ECS. Quando os requisitos da aplicação mudam, você pode atualizar os serviços para aumentar ou diminuir o número de tarefas desejadas. Outra opção é atualizar os serviços para implantar versões mais recentes dos contêineres nas suas tarefas. Para obter mais informações, consulte Atualizar um serviço do Amazon ECS.



Comportamento de extração de imagem de contêiner para os tipos de inicialização do EC2 e externa para o Amazon ECS

O tempo de inicialização de um contêiner varia de acordo com a imagem do contêiner subjacente. Por exemplo, uma imagem maior (versões completas do Debian, Ubuntu e Amazon1/2) pode levar mais tempo para ser inicializada porque há mais serviços executados nos contêineres em comparação com suas respectivas versões menores (Debian-slim, Ubuntu-slim e Amazon-slim) ou imagens de base menores (Alpine).

Quando o agente do Amazon ECS inicia uma tarefa, ele faz pull da imagem do Docker no registro remoto e armazena em cache uma cópia local. Quando você usa uma nova tag de imagem para cada versão da aplicação, esse comportamento é desnecessário.

O parâmetro de agente `ECS_IMAGE_PULL_BEHAVIOR` determina o comportamento de extração de imagem. As seguintes opções estão disponíveis:

- `ECS_IMAGE_PULL_BEHAVIOR`: `default`

O pull da imagem será feito remotamente. Se o pull falhar, a imagem armazenada em cache na instância é usada.

- `ECS_IMAGE_PULL_BEHAVIOR`: `always`

  O pull da imagem será feito remotamente. Se o pull falhar, a tarefa falha.

Para acelerar a implantação, defina o parâmetro do agente do Amazon ECS como um dos seguintes valores:

- `ECS_IMAGE_PULL_BEHAVIOR`: `once`

  O pull da imagem só é feito remotamente se não tiver sido feito por uma tarefa anterior na mesma instância de contêiner ou se a imagem armazenada em cache tiver sido removida pelo processo de limpeza automática de imagens. Caso contrário, a imagem armazenada em cache na instância será usada. Isso garante que nenhum pull de imagem desnecessário seja tentado.
- `ECS_IMAGE_PULL_BEHAVIOR`: `prefer-cached`

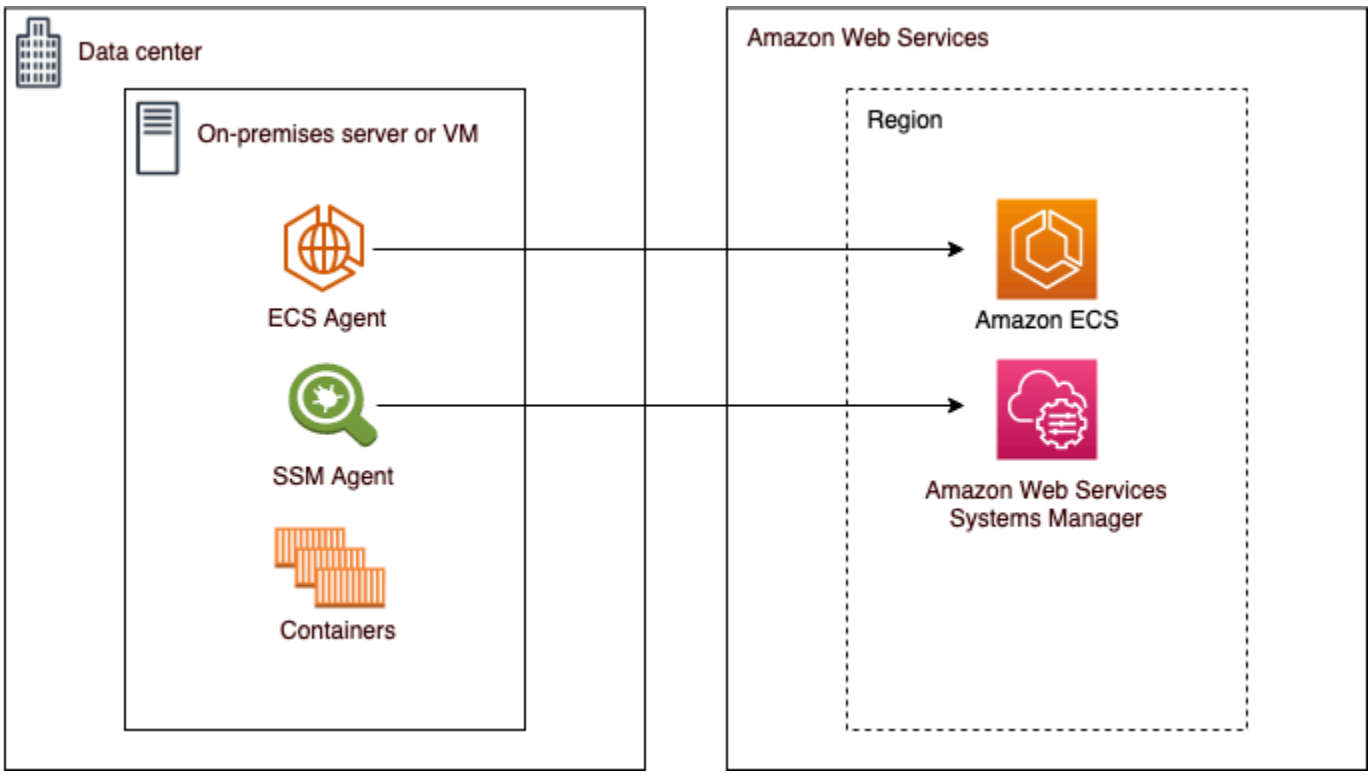
  O pull da imagem será feito remotamente se não houver uma imagem armazenada em cache. Caso contrário, a imagem armazenada em cache na instância será usada. A limpeza automática de imagens é desativada no contêiner para garantir que a imagem armazenada em cache não seja removida.

Definir o parâmetro `ECS_IMAGE_PULL_BEHAVIOR` para qualquer um dos valores anteriores pode economizar tempo porque o agente do Amazon ECS usa a imagem baixada existente. Em imagens do Docker maiores, o tempo de download pode levar de 10 a 20 segundos para fazer pull pela rede.

Tipo de inicialização externa (Amazon ECS Anywhere) para o Amazon ECS

O Amazon ECS Anywhere fornece suporte para registrar uma Instância externa, como um servidor on-premises ou uma máquina virtual (VM), no cluster do Amazon ECS. As instâncias externas são otimizadas para executar aplicações que geram tráfego de saída ou dados de processo. Se sua aplicação exigir tráfego de entrada, a falta de suporte do Elastic Load Balancing torna a execução dessas workloads menos eficiente. O Amazon ECS adicionou um novo tipo de inicialização `EXTERNAL` que você pode usar para criar serviços ou executar tarefas nas instâncias externas.

A seguir, é mostrada uma visão geral da arquitetura de sistema de alto nível do Amazon ECS Anywhere. O servidor on-premises tem o agente do Amazon ECS e o agente do SSM instalados.



Para obter mais informações, consulte Clusters do Amazon ECS para o tipo de inicialização externa.

Usar endpoints de pilha dupla do Amazon ECS

Os endpoints de pilha dupla do Amazon ECS oferecem suporte a solicitações ao Amazon ECS usando o Internet Protocol versão 4 (IPv4) e o Internet Protocol versão 6 (IPv6). Para obter uma lista dos endpoints do Amazon ECS, consulte Endpoints e cotas do Amazon ECS na Referência geral da AWS.

Ao usar a API REST, você acessa diretamente um endpoint do Amazon ECS usando o nome do endpoint (URI). O Amazon ECS oferece suporte apenas a nomes de endpoint de pilha dupla regionais, o que significa que é necessário especificar a região como parte do nome.

Use a seguinte convenção de nomenclatura para os nomes de endpoint de pilha dupla: `ecs.region.api.aws`.

Ao usar a AWS Command Line Interface (AWS CLI) e os AWS SDKs, é possível utilizar um parâmetro ou um sinalizador para mudar para um endpoint de pilha dupla. Também é possível especificar o endpoint de pilha dupla diretamente como uma substituição do endpoint do Amazon ECS no arquivo de configuração.

As seções a seguir descrevem como usar endpoints de pilha dupla via AWS CLI, AWS SDKs e API REST.

Tópicos

- Usar endpoints de pilha dupla da AWS CLI
- Usar endpoints de pilha dupla dos AWS SDKs
- Usar endpoints de pilha dupla da API REST

Usar endpoints de pilha dupla da AWS CLI

Esta seção fornece exemplos de comandos da AWS CLI usados para fazer solicitações a um endpoint de pilha dupla. Para obter mais informações sobre como instalar a AWS CLI ou atualizá-la para a versão mais recente, consulte Instalar ou atualizar para a versão mais recente da AWS CLI no Guia do usuário da versão 2 da AWS Command Line Interface.

Para usar um endpoint de pilha dupla, é possível definir o valor da configuração `use_dualstack_endpoint` como `true` no arquivo `config` para que a AWS CLI direcione todas as solicitações ao Amazon ECS feitas ao comando `ecs` da AWS CLI para o endpoint de pilha dupla da região especificada. É possível especificar a região no arquivo `config` ou em um comando usando a opção `--region`. Para obter mais informações sobre os arquivos de configuração do AWS CLI, consulte Opções do arquivo de configuração e credenciais na AWS CLI no Guia do usuário da versão 2 da AWS Command Line Interface.

Se desejar usar um endpoint de pilha dupla para comandos especificados da AWS CLI, você poderá usar qualquer um dos seguintes métodos:

- Usar o endpoint de pilha dupla por comando, definindo o parâmetro `--endpoint-url` como `https://ecs.`*`aws-region`*`.api.aws` ou `http://ecs.`*`aws-region`*`.api.aws` para qualquer comando do `ecs`.

  O comando de exemplo a seguir lista todos os clusters disponíveis e usa o endpoint de pilha dupla para a solicitação.

  ```
  $ aws ecs list-clusters --endpoint-url https://ecs.aws-region.api.aws
  ```

- Configurar perfis separados em seu arquivo do AWS Config. Por exemplo, crie um perfil que defina `use_dualstack_endpoint` como `true` e um perfil que não defina

`use_dualstack_endpoint`. Quando executar um comando, especifique qual perfil deseja usar, dependendo de querer ou não usar o endpoint de pilha dupla.

Usar endpoints de pilha dupla dos AWS SDKs

Esta seção fornece exemplos de como acessar um endpoint de pilha dupla usando os AWS SDKs.

AWS SDK for Java 2.x

O exemplo a seguir mostra como especificar um endpoint de pilha dupla para a região `us-east-1` usando AWS SDK for Java 2.x.

```
Region region = Region.US_EAST_1
EcsClient client =
 EcsClient.builder().region(region).dualstackEnabled(true).build();
```

AWS SDK para Go

O exemplo a seguir mostra como especificar um endpoint de pilha dupla para a região `us-east-1` usando AWS SDK para Go.

```
sess := session.Must(session.NewSession())
svc := ecs.New(sess, &aws.Config{
    Region: aws.String(endpoints.UsEast1RegionID),
    Endpoint: aws.String("https://ecs.us-east-1.api.aws")
})
```

Para obter mais informações, consulte Endpoints de pilha dupla e FIPS no Guia de referência de ferramentas e SDKs da AWS.

Usar endpoints de pilha dupla da API REST

Ao usar a API REST, você pode acessar diretamente um endpoint de pilha dupla especificando-o em sua solicitação. O exemplo a seguir usa o endpoint de pilha dupla para listar todos os clusters do Amazon ECS na região `us-east-1`.

```
POST / HTTP/1.1
Host: ecs.us-east-1.api.aws
Accept-Encoding: identity
```

```
Content-Length: 2
X-Amz-Target: AmazonEC2ContainerServiceV20141113.ListClusters
X-Amz-Date: 20150429T170621Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{}
```

Aplicações do Amazon ECS em sub-redes compartilhadas, zonas locais e zonas do Wavelength

O Amazon ECS é compatível com workloads que usam zonas locais, zonas do Wavelength e o AWS Outposts para quando é necessária baixa latência ou processamento de dados locais.

- É possível usar zonas locais como extensão de uma Região da AWS para colocar recursos em vários locais mais próximos dos usuários finais.
- É possível usar zonas do Wavelength para criar aplicações que oferecem baixíssimas latências para dispositivos 5G e usuários finais. O Wavelength implanta os serviços de armazenamento e computação padrão da AWS até a borda das redes 5G de operadoras de telecomunicação.
- O AWS Outposts leva serviços, infraestrutura e modelos operacionais nativos de Serviços da AWS a praticamente qualquer data center, espaço de colocalização ou instalação on-premises.

> **Important**
>
> Workloads do Amazon ECS no AWS Fargate não são compatíveis com zonas locais, com zonas do Wavelength ou com o AWS Outposts, no momento.

Para obter mais informações sobre as diferenças entre zonas locais, zonas do Wavelength e AWS Outposts, consulte Como devo pensar sobre quando usar AWS Wavelength, zonas locais da AWS ou AWS Outposts para aplicativos que exigem baixa latência ou processamento local de dados nas Perguntas frequentes sobre o AWS Wavelength.

Sub-redes compartilhadas

Você pode usar Compartilhamento de VPC para compartilhar sub-redes com outras contas da AWS dentro da mesma AWS Organizations.

Você pode usar VPCs compartilhadas no tipo de execução do EC2 com as seguintes considerações:

- O proprietário da sub-rede da VPC deve compartilhar uma sub-rede com uma conta participante antes que essa conta possa usá-la em recursos do Amazon ECS.
- Você não pode usar o grupo de segurança padrão da VPC para as instâncias de contêiner porque ele pertence ao proprietário. Além disso, os participantes não podem executar instâncias usando grupos de segurança pertencentes a outros participantes ou ao proprietário.
- Em uma sub-rede compartilhada, o participante e o proprietário controlam separadamente os grupos de segurança na respectiva conta. O proprietário da sub-rede pode visualizar esses grupos de segurança criados pelos participantes, mas não pode realizar neles nenhuma ação. Se o proprietário da sub-rede quiser remover ou modificar esses grupos de segurança, o participante que criou o grupo de segurança deve realizar a ação.
- O proprietário da VPC compartilhada não pode visualizar, atualizar ou excluir um cluster criado por um participante na sub-rede compartilhada. Isso é além dos recursos da VPC aos quais cada conta tem acesso diferente. Para obter mais informações, consulte Responsabilidades e permissões para proprietários e participantes no Manual do usuário do Amazon VPC.

Você pode usar VPCs compartilhadas no tipo de execução do Fargate com as seguintes considerações:

- O proprietário da sub-rede da VPC deve compartilhar uma sub-rede com uma conta participante antes que essa conta possa usá-la em recursos do Amazon ECS.
- Você não pode criar um serviço ou executar uma tarefa usando o grupo de segurança padrão da VPC porque ele pertence ao proprietário. Além disso, os participantes não podem criar um serviço ou executar uma tarefa usando grupos de segurança que pertencem a outros participantes ou ao proprietário.
- Em uma sub-rede compartilhada, o participante e o proprietário controlam separadamente os grupos de segurança na respectiva conta. O proprietário da sub-rede pode visualizar esses grupos de segurança criados pelos participantes, mas não pode realizar neles nenhuma ação. Se o proprietário da sub-rede quiser remover ou modificar esses grupos de segurança, o participante que criou o grupo de segurança deve realizar a ação.
- O proprietário da VPC compartilhada não pode visualizar, atualizar ou excluir um cluster criado por um participante na sub-rede compartilhada. Isso é além dos recursos da VPC aos quais cada conta tem acesso diferente. Para obter mais informações, consulte Responsabilidades e permissões para proprietários e participantes no Manual do usuário do Amazon VPC.

Para obter informações sobre o compartilhamento de sub-rede da VPC, consulte Compartilhar sua VPC com outras contas no Manual do usuário do Amazon VPC.

Zonas Locais

Uma zona local é uma extensão de uma Região da AWS geograficamente próxima dos usuários. As zonas locais têm suas próprias conexões com a Internet e suporte no AWS Direct Connect. Os recursos criados em uma zona local podem atender a usuários locais com comunicações de baixa latência. Para obter mais informações, consulte AWS Local Zones. .

Uma zona local é representada por um código de região seguido por um identificador que indica o local (por exemplo, `us-west-2-lax-1a`).

Para usar uma zona local, primeiro é necessário escolher a zona. Depois de escolher, crie uma Amazon VPC e uma sub-rede na zona local.

Inicie instâncias do Amazon EC2, servidores de arquivos do Amazon FSx e Application Load Balancers para serem usados nos seus clusters e tarefas do Amazon ECS.

Para obter mais informações, consulte O que são zonas locais da AWS? no Guia do usuário de zonas locais da AWS.

Zonas do Wavelength

É possível usar o AWS Wavelength para criar aplicações que ofereçam baixíssimas latências para dispositivos móveis e usuários finais. O Wavelength implanta os serviços de armazenamento e computação padrão da AWS até a borda das redes 5G de operadoras de telecomunicação. É possível estender uma Amazon Virtual Private Cloud (VPC) para uma ou mais zonas do Wavelength. Em seguida, é possível usar recursos da AWS, como instâncias do Amazon EC2, para executar aplicações que exijam baixíssima latência e uma conexão com Serviços da AWS na região.

Uma zona do Wavelength é uma zona isolada no local da operadora na qual a infraestrutura de Wavelength é implantada. As zonas do Wavelength estão vinculadas a uma Região da AWS. Uma zona de Wavelength é uma extensão lógica de uma região e é gerenciada pelo plano de controle na região.

Uma zona do Wavelength é representada por um código de região seguido por um identificador que indica a zona do Wavelength (por exemplo, `us-east-1-wl1-bos-wlz-1`).

Para usar uma zona do Wavelength, você deve primeiro escolher a zona. Depois de escolher, crie uma Amazon VPC e uma sub-rede na zona do Wavelength. Em seguida, inicie instâncias do Amazon EC2 na zona para serem usadas nos clusters e tarefas do Amazon ECS.

Para obter mais informações, consulte Conceitos básicos do AWS Wavelength no Guia do desenvolvedor do AWS Wavelength.

As zonas do Wavelength não estão disponíveis em todas as Regiões da AWS. Para obter informações sobre as regiões compatíveis com as zonas do Wavelength consulte Zonas do Wavelength disponíveis no Guia do desenvolvedor da AWS Wavelength.

Amazon Elastic Container Service no AWS Outposts

O AWS Outposts permite serviços nativos, infraestrutura e modelos operacionais da AWS em instalações on-premises. Em ambientes do AWS Outposts, é possível usar APIs, ferramentas e infraestrutura da AWS semelhantes às utilizadas na Nuvem AWS.

O Amazon ECS no AWS Outposts é ideal para workloads de baixa latência que precisam ser executadas perto dos dados e aplicações on-premises.

Para obter mais informações sobre o AWS Outposts, consulte o Guia do usuário do AWS Outposts.

Considerações

A seguir, apresentamos considerações sobre o uso do Amazon ECS no AWS Outposts:

- O Amazon Elastic Container Registry, o AWS Identity and Access Management e o Network Load Balancer são executados na região da AWS e não no AWS Outposts. Isto irá aumentar as latências entre esses serviços e os contêineres.
- O AWS Fargate não está disponível no AWS Outposts.

Veja a seguir as considerações sobre a conectividade de rede do AWS Outposts:

- Se a conectividade de rede entre o AWS Outposts e a região da AWS for perdida, os clusters continuarão em execução. No entanto, você não pode criar novos clusters ou executar novas ações em clusters existentes enquanto que a conectividade não for restaurada. Em caso de falhas na instância, a instância não será substituída automaticamente. O agente do CloudWatch Logs não poderá atualizar logs e dados de eventos.

- Recomendamos que você forneça conectividade confiável, altamente disponível e de baixa latência entre o AWS Outposts e a região da AWS.

Pré-requisitos

A seguir, apresentamos pré-requisitos para usar o Amazon ECS no AWS Outposts:

- Um Outpost deve estar instalado e configurado no data center on-premises.
- É necessário ter uma conexão de rede confiável entre o Outpost e a região da AWS.

Visão geral da criação de clusters em AWS Outposts

A seguir está uma visão geral da configuração:

1. Crie uma função e uma política com direitos no AWS Outposts.
2. Crie um perfil da instância do IAM com direitos no AWS Outposts.
3. Crie uma VPC ou use uma existente que esteja na mesma região que a sua AWS Outposts.
4. Crie uma sub-rede ou use uma existente que esteja associada a AWS Outposts.

   Essa é a sub-rede em que as instâncias do contêiner são executadas.
5. Criar uma regra do grupo de segurança para as instâncias de contêiner em seu cluster.
6. Crie um cluster do Amazon ECS.
7. Defina as variáveis de ambiente do agente de contêiner do Amazon ECS para iniciar a instância no cluster.
8. Execute o contêiner.

Para obter informações detalhadas sobre como integrar o Amazon ECS com AWS Outposts, consulte Extend Amazon ECS across two AWS Outposts racks.

O exemplo a seguir cria um cluster do Amazon ECS em um AWS Outposts.

1. Crie uma função e uma política com direitos no AWS Outposts.

   O arquivo `role-policy.json` é o documento de política que contém o efeito e as ações dos recursos. Para obter informações sobre o formato do arquivo, consulte PutRolePolicy na Referência de API do IAM

```
aws iam create-role --role-name ecsRole \
    --assume-role-policy-document file://ecs-policy.json
aws iam put-role-policy --role-name ecsRole --policy-name ecsRolePolicy \
    --policy-document file://role-policy.json
```

2. Crie um perfil da instância do IAM com direitos no AWS Outposts.

```
aws iam create-instance-profile --instance-profile-name outpost
aws iam add-role-to-instance-profile --instance-profile-name outpost \
    --role-name ecsRole
```

3. Crie uma VPC.

```
aws ec2 create-vpc --cidr-block 10.0.0.0/16
```

4. Crie uma sub-rede associada ao seu AWS Outposts.

```
aws ec2 create-subnet \
    --cidr-block 10.0.3.0/24 \
    --vpc-id vpc-xxxxxxxx \
    --outpost-arn arn:aws:outposts:us-west-2:123456789012:outpost/op-
xxxxxxxxxxxxxxxx \
    --availability-zone-id usw2-az1
```

5. Crie um grupo de segurança para as instâncias de contêiner, especificando o intervalo CIDR adequado para o AWS Outposts. (Essa etapa é diferente para o AWS Outposts.)

```
aws ec2 create-security-group --group-name MyOutpostSG
aws ec2 authorize-security-group-ingress --group-name MyOutpostSG --protocol tcp \
    --port 22 --cidr 10.0.3.0/24
aws ec2 authorize-security-group-ingress --group-name MyOutpostSG --protocol tcp \
    --port 80 --cidr 10.0.3.0/24
```

6. Crie o cluster.
7. Defina as variáveis de ambiente do agente de contêiner do Amazon ECS para iniciar a instância no cluster criado na etapa anterior e defina as tags que você deseja adicionar para ajudar a identificar o cluster (por exemplo, `Outpost` para indicar que o cluster é para um Outpost).

```
#! /bin/bash
cat << 'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
```

```
ECS_IMAGE_PULL_BEHAVIOR=prefer-cached
ECS_CONTAINER_INSTANCE_TAGS={“environment”: ”Outpost”}
EOF
```

> **Note**
>
> Para evitar atrasos causados pela extração de imagens de contêiner do Amazon ECR na região, use caches de imagem. Para fazer isso, cada vez que uma tarefa for executada, configure o agente do Amazon ECS como padrão para usar a imagem em cache na própria instância, definindo `ECS_IMAGE_PULL_BEHAVIOR` como `prefer-cached`.

8. Crie a instância de contêiner especificando a VPC e a sub-rede para o AWS Outposts no qual esta instância deve ser executada e um tipo de instância disponível no AWS Outposts. (Essa etapa é diferente para o AWS Outposts.)

   O arquivo `userdata.txt` contém os dados do usuário que a instância pode usar para realizar tarefas de configuração automatizadas em comum e até mesmo executar scripts após o início da instância. Para obter mais informações sobre o arquivo para chamadas de API, consulte Executar comandos na instância do Linux no lançamento no Manual do usuário do Amazon EC2.

   ```
   aws ec2 run-instances --count 1 --image-id ami-xxxxxxxx --instance-type c5.large \
       --key-name aws-outpost-key --subnet-id subnet-xxxxxxxxxxxxxxxxx \
       --iam-instance-profile Name outpost --security-group-id sg-xxxxxx \
       --associate-public-ip-address --user-data file://userdata.txt
   ```

   > **Note**
   >
   > Esse comando também é usado ao incluir instâncias adicionais no cluster. Todos os contêineres implantados no cluster serão colocados nesse AWS Outposts específico.

Práticas recomendadas de rede do Amazon ECS

As aplicações modernas geralmente são criadas a partir de vários componentes distribuídos que se comunicam uns com os outros. Por exemplo, uma aplicação móvel ou Web pode se comunicar com

um endpoint de API, e a API pode ser alimentada por vários microsserviços que se comunicam pela Internet.

Para obter informações sobre as práticas recomendadas para conectar aplicações à Internet, consulte Conexão de aplicações do Amazon ECS à Internet

Para obter mais informações sobre as práticas recomendadas para receber conexões de entrada do Amazon ECS provenientes da Internet, consulte Práticas recomendadas para receber conexões de entrada para o Amazon ECS vindas da Internet.

Para obter informações sobre as práticas recomendadas para conectar o Amazon ECS a outros serviços da AWS, consulte Práticas recomendadas para conectar o Amazon ECS a serviços da AWS de dentro da sua VPC

Para obter informações sobre as práticas recomendadas para conectar serviços em uma VPC, consulte Práticas recomendadas para conectar serviços do Amazon ECS em uma VPC.

Para obter informações sobre as práticas recomendadas para serviços de rede em Contas da AWS e VPCs, consulte Práticas recomendadas para conectar serviços do Amazon ECS em rede entre Contas da AWS e VPCs.

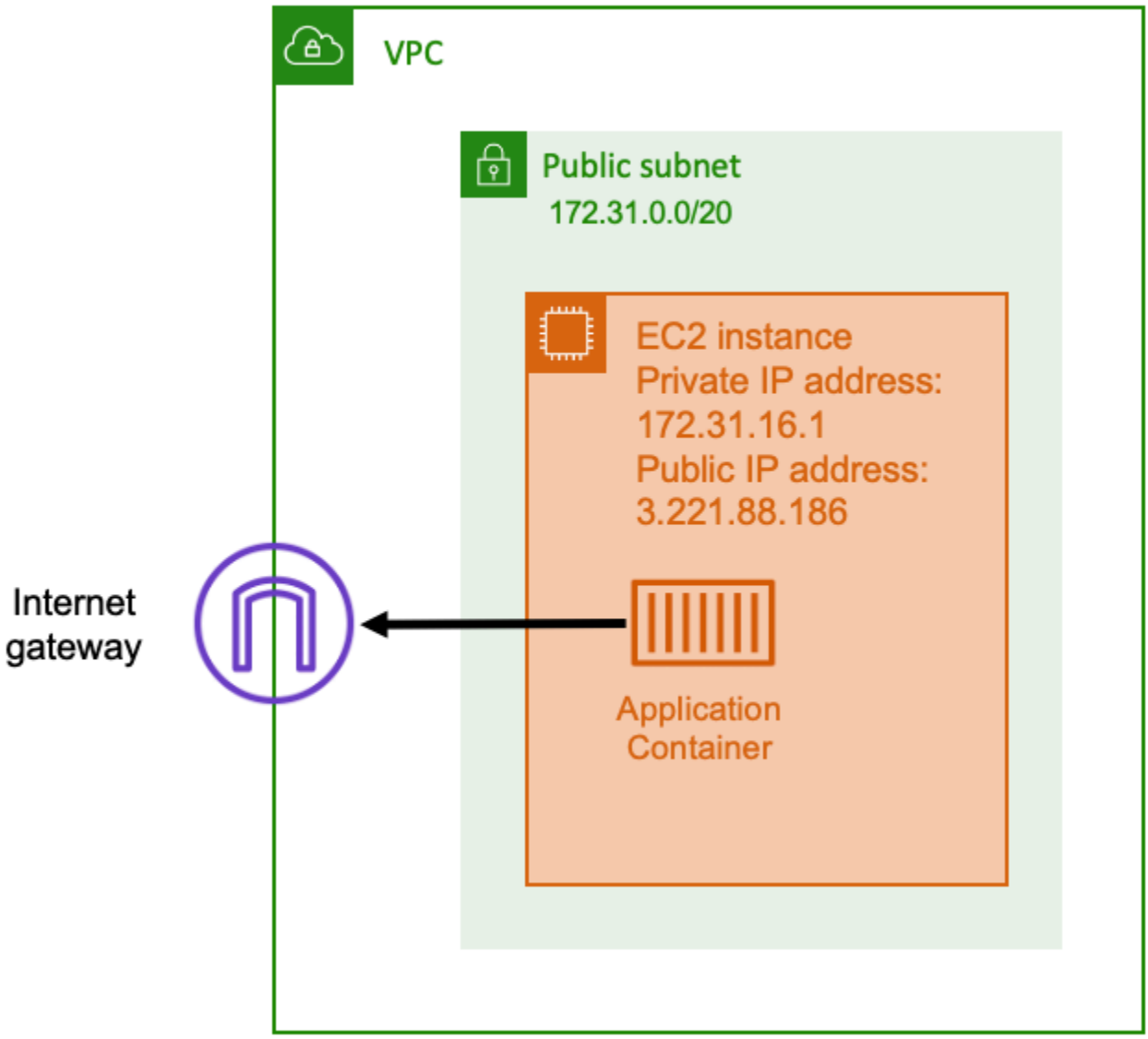
Para obter informações sobre as práticas recomendadas para serviços de solução de problemas de rede, consulte Serviços da AWS para solução de problemas de rede do Amazon ECS.

Conexão de aplicações do Amazon ECS à Internet

A maioria das aplicações em contêineres tem pelo menos alguns componentes que precisam de acesso externo à Internet. Por exemplo, o backend de uma aplicação móvel requer acesso externo às notificações push.

A Amazon Virtual Private Cloud tem dois métodos principais para facilitar a comunicação entre sua VPC e a Internet.

Sub-rede pública e gateway da Internet



Ao usar uma sub-rede pública que tem uma rota para um gateway da Internet, a aplicação em contêiner pode ser executada em um host dentro de uma VPC em uma sub-rede pública. O host que executa o contêiner recebe um endereço IP público. Esse endereço IP público é roteável pela Internet. Para obter mais informações, consulte Gateways da Internet no Guia do usuário da Amazon VPC.

Essa arquitetura de rede facilita a comunicação direta entre o host que executa a aplicação e outros hosts na Internet. A comunicação é bidirecional. Isso significa que você não só pode estabelecer uma conexão de saída com qualquer outro host na Internet, mas outros hosts na Internet também podem tentar se conectar ao seu host. Portanto, você deve prestar muita atenção ao seu grupo

de segurança e às regras de firewall. Isso garante que outros hosts na Internet não possam abrir nenhuma conexão que você não queira que seja aberta.

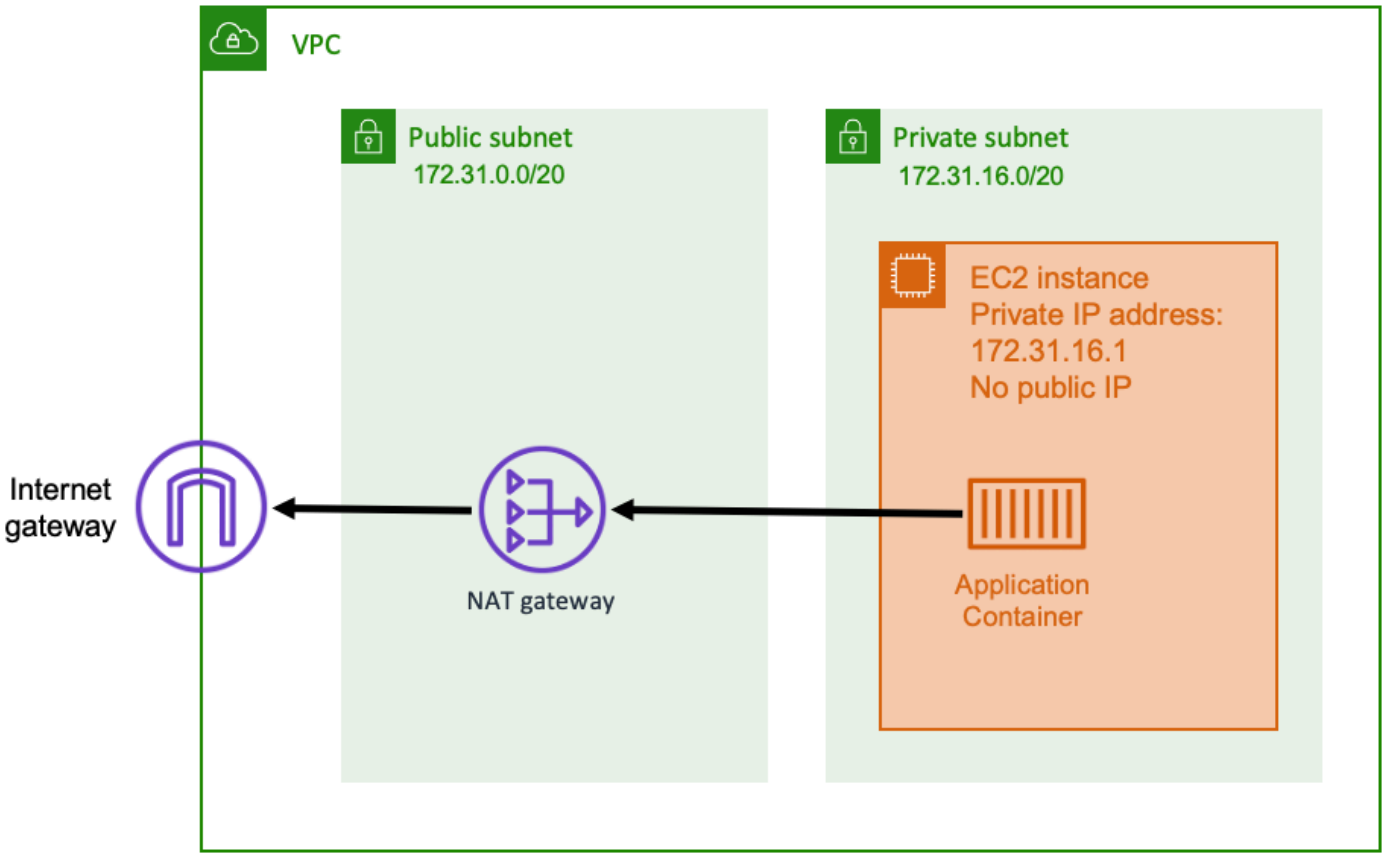
Por exemplo, se a aplicação for executada no Amazon EC2, certifique-se de que a porta 22 para acesso SSH não esteja aberta. Caso contrário, sua instância pode receber tentativas constantes de conexão SSH vindas de bots maliciosos na Internet. Esses bots vasculham endereços IP públicos. Depois de encontrarem uma porta SSH aberta, eles tentam atacar senhas com força bruta para acessar sua instância. Por causa disso, muitas organizações limitam o uso de sub-redes públicas e preferem ter a maioria, se não todos, de seus recursos dentro de sub-redes privadas.

Usar sub-redes públicas para redes é adequado em aplicações públicas que exigem grandes quantidades de largura de banda ou latência mínima. Os casos de uso aplicáveis incluem serviços de streaming de vídeo e jogos.

Essa abordagem de rede é compatível ao usar o Amazon ECS no Amazon EC2 e ao usar o Amazon EC2 no AWS Fargate.

- Amazon EC2: você pode executar instâncias do EC2 em uma sub-rede pública. O Amazon ECS usa essas instâncias do EC2 como capacidade de cluster, e qualquer contêiner que esteja sendo executado nas instâncias pode usar o endereço IP público subjacente do host para redes de saída. Isso se aplica aos modos de rede `host` e `bridge`. No entanto, o modo de rede `awsvpc` não fornece endereços IP públicos às ENIs de tarefa. Portanto, eles não podem fazer uso direto de um gateway da Internet.
- Fargate: ao criar o serviço do Amazon ECS, especifique sub-redes públicas para a configuração de rede do serviço e use a opção Atribuir endereço IP público. Cada tarefa do Fargate é conectada em rede na sub-rede pública e tem seu próprio endereço IP público para comunicação direta com a Internet.

Sub-rede privada e gateway NAT



Ao usar uma sub-rede privada e um gateway NAT, você pode executar a aplicação em contêineres em um host que esteja em uma sub-rede privada. Dessa forma, esse host tem um endereço IP privado que pode ser encaminhado dentro da VPC, mas não pode ser encaminhado pela Internet. Isso significa que outros hosts dentro da VPC podem se conectar ao host usando seu endereço IP privado, mas outros hosts na Internet não podem fazer nenhuma comunicação de entrada com o host.

Com uma sub-rede privada, você pode usar um gateway de conversão de endereços de rede (NAT) para permitir que um host dentro de uma sub-rede privada se conecte à Internet. Os hosts na Internet recebem uma conexão de entrada que parece vir do endereço IP público do gateway NAT que está dentro de uma sub-rede pública. O gateway NAT serve como uma ponte entre a Internet e a sub-rede privada. Essa configuração costuma ser preferida por motivos de segurança, pois significa que a VPC está protegida do acesso direto de invasores na Internet. Para obter mais informações, consulte Gateways NAT no Guia do usuário da Amazon VPC.

Essa abordagem baseada em rede privada é adequada para cenários em que você deseja proteger seus contêineres do acesso externo direto. Os cenários aplicáveis incluem sistemas de processamento de pagamentos ou contêineres que armazenam dados e senhas do usuário. Você será cobrado para criar e usar um gateway NAT em sua conta. Também se aplicam taxas de uso por hora do gateway NAT e de processamento de dados. Para fins de redundância, você deve ter um gateway NAT em cada zona de disponibilidade. Dessa forma, a perda de disponibilidade de uma única zona de disponibilidade não compromete a conectividade externa. Por isso, caso tenha uma workload pequena, talvez seja mais econômico usar sub-redes privadas e gateways NAT.

Essa abordagem de rede é compatível ao usar o Amazon ECS no Amazon EC2 e ao usá-lo no AWS Fargate.

- Amazon EC2: você pode executar instâncias do EC2 em uma sub-rede privada. Os contêineres executados nesses hosts do EC2 usam a rede de hosts subjacentes, e as solicitações de saída passam pelo gateway NAT.
- Fargate: ao criar o serviço do Amazon ECS, especifique sub-redes privadas para a configuração de rede do serviço e não use a opção Atribuir endereço IP público. Cada tarefa do Fargate é hospedada em uma sub-rede privada. Seu tráfego de saída é encaminhado por meio de qualquer gateway NAT que você tenha associado a essa sub-rede privada.

Práticas recomendadas para receber conexões de entrada para o Amazon ECS vindas da Internet

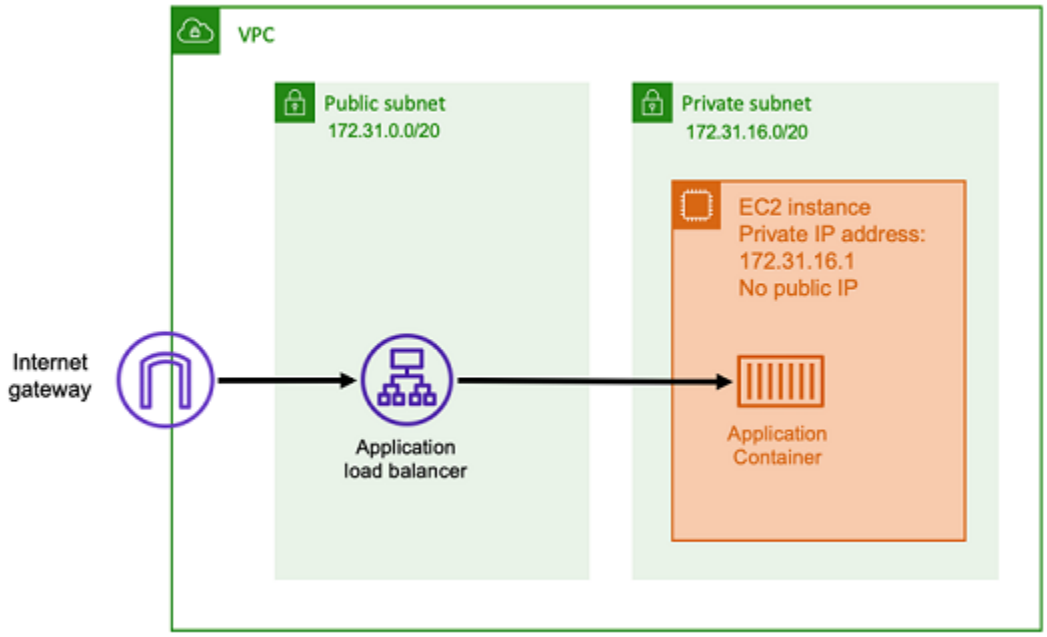
Se você administra um serviço público, deve aceitar o tráfego de entrada da Internet. Por exemplo, seu site público deve aceitar solicitações HTTP de entrada vindas de navegadores. Nesse caso, outros hosts na Internet também devem iniciar uma conexão de entrada com o host da aplicação.

Uma abordagem para esse problema é executar seus contêineres em hosts que estão em uma sub-rede pública com um endereço IP público. No entanto, não recomendamos isso em aplicações de grande escala. Para elas, uma abordagem melhor é ter uma camada de entrada escalável que fique entre a Internet e a aplicação. Para essa abordagem, você pode usar qualquer um dos serviços da AWS listados nesta seção como entrada.

Application Load Balancer

Um Application Load Balancer funciona na camada da aplicação. É a sétima camada do modelo Open Systems Interconnection (OSI). Isso torna um Application Load Balancer adequado para

serviços HTTP públicos. Se você tem um site ou uma API REST HTTP, um Application Load Balancer é um balanceador de carga adequado para essa workload. Para obter mais informações, consulte What is an Application Load Balancer? no Guia do Usuário para Application Load Balancers.



Com essa arquitetura, você cria um Application Load Balancer em uma sub-rede pública para que ele tenha um endereço IP público e possa receber conexões de entrada da Internet. Quando o Application Load Balancer recebe uma conexão de entrada ou, mais especificamente, uma solicitação HTTP, ele abre uma conexão com a aplicação usando seu endereço IP privado. Em seguida, ele encaminha a solicitação pela conexão interna.

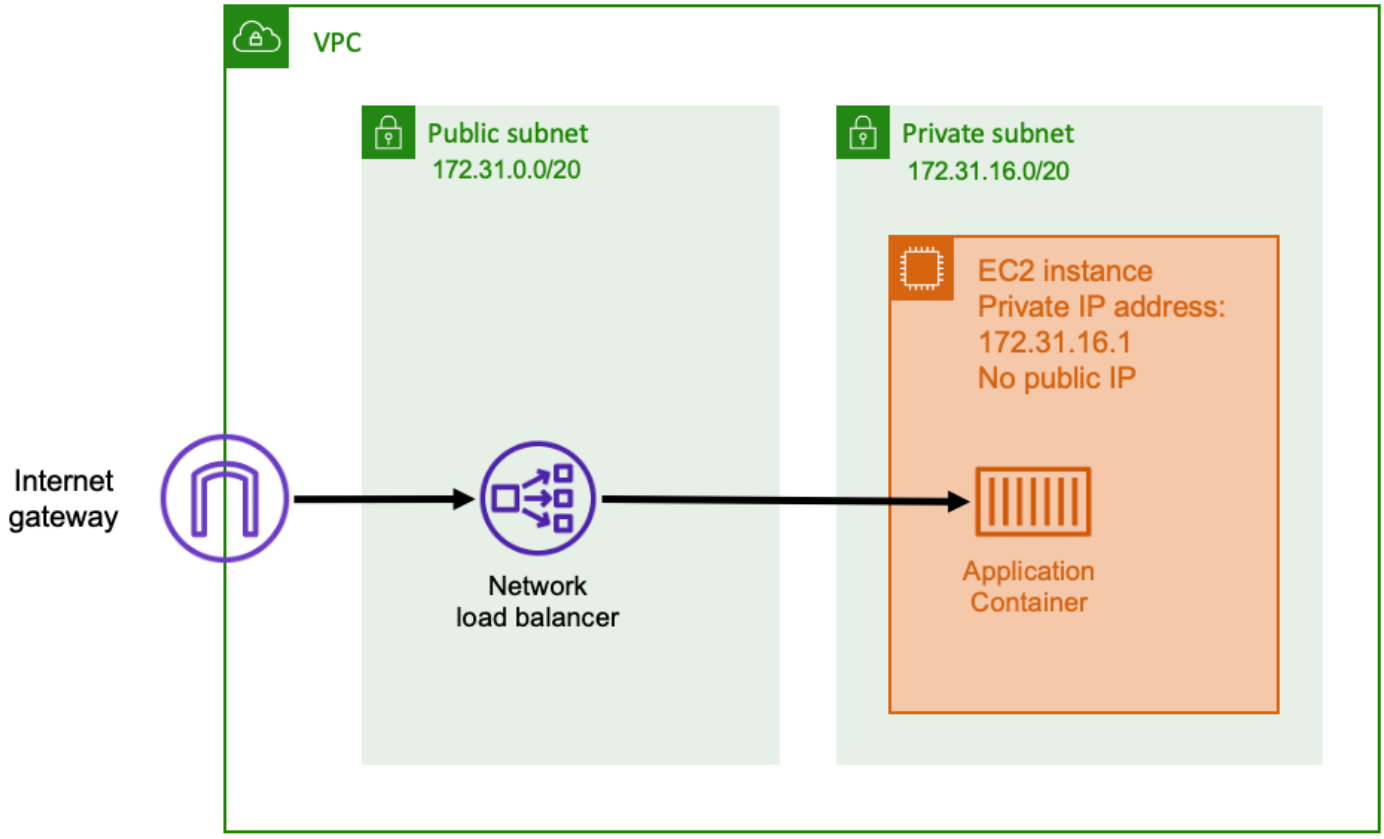
Um Application Load Balancer tem as vantagens a seguir.

- Encerramento de SSL e TLS: um Application Load Balancer pode manter comunicações e certificados HTTPS seguros para comunicações com clientes. Opcionalmente, ele pode encerrar a conexão SSL no nível do balanceador de carga para que você não precise lidar com certificados na própria aplicação.
- Encaminhamento avançado: um Application Load Balancer pode ter vários nomes de host do DNS. Ele também tem recursos avançados de encaminhamento para enviar solicitações HTTP de entrada a destinos diferentes com base em métricas, como o nome do host ou o caminho da solicitação. Isso significa que você pode usar um único Application Load Balancer como entrada em vários serviços internos diferentes ou até mesmo em microsserviços em caminhos diferentes de uma API REST.

- Suporte a gRPC e WebSockets: um Application Load Balancer pode processar mais recursos além de HTTP. Ele também pode balancear a carga de serviços baseados em gRPC e WebSocket, com suporte a HTTP/2.
- Segurança: um Application Load Balancer ajuda a proteger sua aplicação contra tráfego malicioso. Ele inclui recursos como mitigações de sincronização HTTP, sendo integrado ao AWS Web Application Firewall (AWS WAF). O AWS WAF filtra ainda mais o tráfego malicioso que pode conter padrões de ataque, como injeção de SQL ou cross-site scripting.

Network Load Balancer

Um Network Load Balancer funciona na quarta camada do modelo Open Systems Interconnection (OSI - interconexão de sistemas abertos). É adequado em protocolos não HTTP ou cenários em que a criptografia de ponta a ponta é necessária, mas não tem os mesmos recursos específicos de HTTP de um Application Load Balancer. Portanto, um Network Load Balancer é mais adequado para aplicações que não usam HTTP. Para obter mais informações, consulte What is a Network Load Balancer? no Guia do usuário para Network Load Balancers.



Quando um Network Load Balancer é usado como entrada, ele funciona de maneira semelhante a um Application Load Balancer. Isso ocorre porque ele é criado em uma sub-rede pública e tem um endereço IP público que pode ser acessado na Internet. Em seguida, o Network Load Balancer abre uma conexão com o endereço IP privado do host que executa o contêiner e envia os pacotes do lado público para o privado.

Recursos do Network Load Balancer

Como o Network Load Balancer opera em um nível inferior da pilha de rede, ele não tem o mesmo conjunto de recursos que o Application Load Balancer. No entanto, ele tem os recursos importantes descritos a seguir.

- Criptografia de ponta a ponta: como um Network Load Balancer opera na quarta camada do modelo OSI, ele não lê o conteúdo dos pacotes. Isso o torna adequado para comunicações de balanceamento de carga que precisam de criptografia de ponta a ponta.
- Criptografia TLS: além da criptografia de ponta a ponta, o Network Load Balancer pode encerrar conexões TLS. Dessa forma, as aplicações de backend não precisam implementar o próprio TLS.
- Suporte a UDP: como um Network Load Balancer opera na quarta camada do modelo OSI, ele é adequado para workloads e protocolos não HTTP que não sejam de TCP.

Encerramento de conexões

Como o Network Load Balancer não observa o protocolo da aplicação nas camadas superiores do modelo OSI, ele não pode enviar mensagens de encerramento aos clientes nesses protocolos. Diferentemente do Application Load Balancer, essas conexões precisam ser encerradas pela aplicação ou você pode configurar o Network Load Balancer para encerrar as conexões da quarta camada quando uma tarefa for interrompida ou substituída. Consulte a configuração de encerramento da conexão para os grupos de destino do Network Load Balancer na documentação do Network Load Balancer.

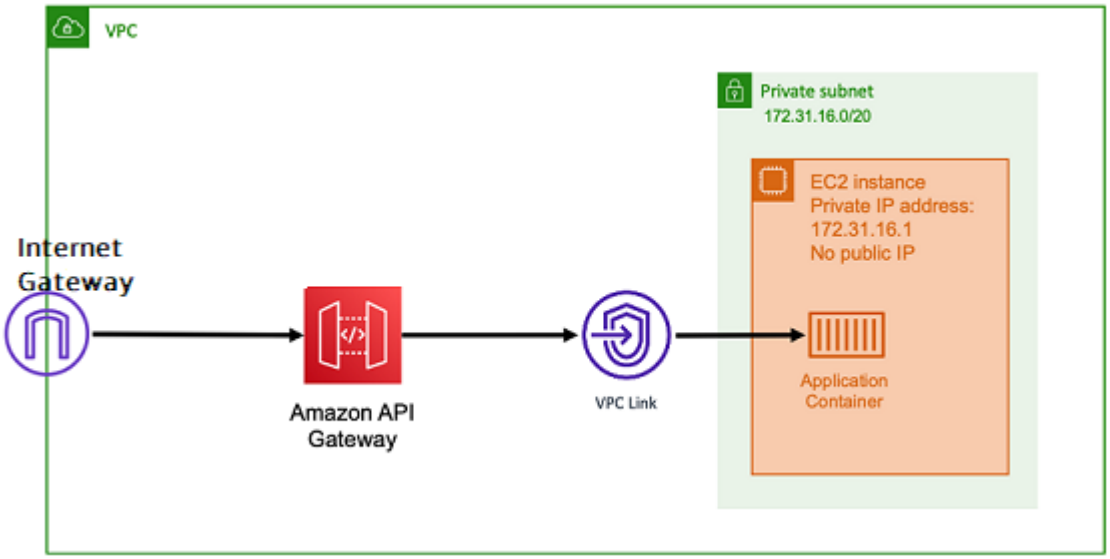
Permitir que o Network Load Balancer encerre conexões na quarta camada pode fazer com que os clientes exibam mensagens de erro indesejadas, se o cliente não processá-las. Consulte a Builders Library para obter mais informações sobre a configuração recomendada do cliente aqui .

Os métodos para encerrar conexões variam de acordo com a aplicação, no entanto, uma maneira é garantir que a meta de atraso no cancelamento de registro do Network Load Balancer seja maior do que o tempo limite de conexão do cliente. O cliente atingiria o tempo limite primeiro e se reconectaria normalmente à próxima tarefa por meio do Network Load Balancer, enquanto a tarefa

antiga esgotaria lentamente todos os clientes. Para obter mais informações sobre a meta de atraso no cancelamento de registro do Network Load Balancer, consulte a documentação do Network Load Balancer.

API HTTP do Amazon API Gateway

O Amazon API Gateway é adequado para aplicações HTTP com expansões repentinas nos volumes de solicitações ou baixos volumes de solicitações. Para obter mais informações, consulte What is Amazon API Gateway? no Guia do desenvolvedor do Amazon API Gateway.



O modelo de preços do Application Load Balancer e do Network Load Balancer inclui um preço por hora para manter os balanceadores de carga sempre disponíveis para aceitar conexões de entrada. Por outro lado, o API Gateway cobra por cada solicitação separadamente. Dessa forma, se nenhuma solicitação for recebida, não há cobranças. Sob cargas de alto tráfego, um Application Load Balancer ou Network Load Balancer pode processar um volume maior de solicitações a um preço mais barato por solicitação do que o API Gateway. No entanto, caso tenha um número baixo de solicitações em geral ou períodos de baixo tráfego, o preço cumulativo proveniente do uso do API Gateway é mais econômico do que pagar uma taxa por hora para manter um balanceador de carga que está sendo subutilizado. O API Gateway também armazena em cache as respostas da API, o que pode resultar em menores taxas de solicitação de backend.

Funções do API Gateway que usam um link de VPC que permite ao serviço gerenciado da AWS se conectar a hosts na sub-rede privada da sua VPC, usando o próprio endereço IP privado. Ele pode detectar esses endereços IP privados examinando os registros de descoberta de serviços do AWS Cloud Map gerenciados pela descoberta de serviços do Amazon ECS.

O API Gateway é compatível com os recursos a seguir.

- A operação do API Gateway é semelhante a um balanceador de carga, mas tem recursos adicionais exclusivos para o gerenciamento de API
- O API Gateway fornece recursos adicionais relacionados à autorização do cliente, aos níveis de uso e à modificação da solicitação ou resposta. Para obter mais informações, consulte Recursos do Amazon API Gateway.
- O API Gateway é compatível com endpoints de gateway de API de borda, regionais e privados. Os endpoints de borda estão disponíveis por meio de uma distribuição gerenciada do CloudFront. Os endpoints regionais e privados são locais de uma região.
- Encerramento de SSL e TLS
- Roteamento de caminhos HTTP diferentes para diferentes microsserviços de backend

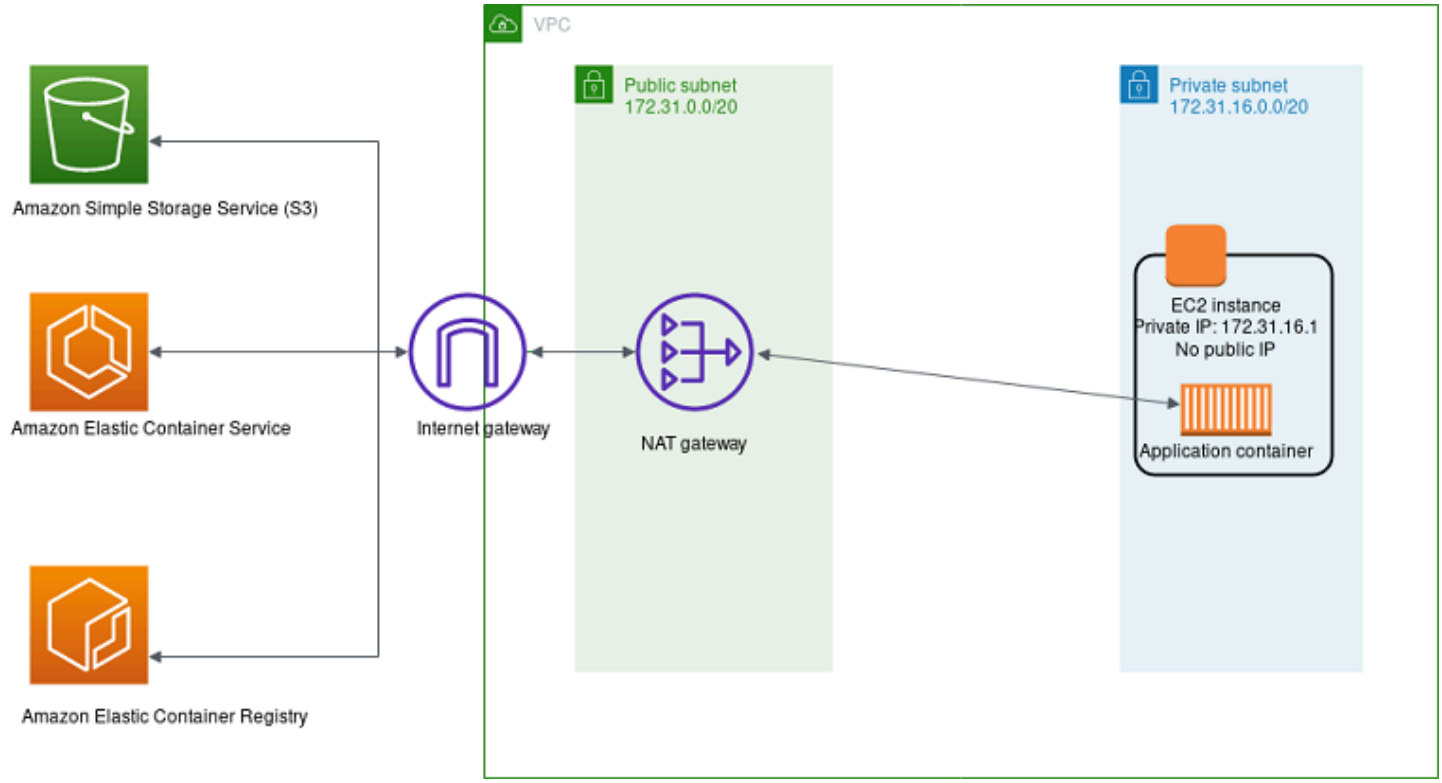
Além dos recursos anteriores, o API Gateway também oferece suporte a autorizadores do Lambda personalizados que você pode usar para proteger a API contra o uso não autorizado. Para obter mais informações, consulte Field Notes: Serverless Container-based APIs with Amazon ECS and Amazon API Gateway.

Práticas recomendadas para conectar o Amazon ECS a serviços da AWS de dentro da sua VPC

Para que o Amazon ECS funcione corretamente, o agente de contêiner do Amazon ECS executado em cada host deve se comunicar com o ambiente de gerenciamento do Amazon ECS. Se você estiver armazenando suas imagens de contêiner no Amazon ECR, os hosts do Amazon EC2 deverão se comunicar com o endpoint do serviço do Amazon ECR e com o Amazon S3, onde as camadas de imagem são armazenadas. Se você usa outros serviços da AWS para sua aplicação em contêineres, como dados persistentes armazenados no DynamoDB, verifique se esses serviços também têm o suporte de rede necessário.

NAT gateway

Usar um gateway NAT é a maneira mais fácil de garantir que suas tarefas do Amazon ECS possam acessar outros serviços da AWS. Para obter mais informações sobre essa abordagem, consulte Sub-rede privada e gateway NAT.



Estas são as desvantagens de usar esta abordagem:

- Não é possível limitar os destinos com os quais o gateway NAT pode se comunicar. Também não é possível limitar os destinos com os quais sua camada de backend pode se comunicar sem interromper todas as comunicações de saída da sua VPC.
- Os gateways NAT cobram por cada GB de dados que os atravessam. Se usar o gateway NAT para qualquer uma das seguintes operações, você será cobrado por cada GB de largura de banda:
  - Baixar arquivos grandes do Amazon S3
  - Fazer um grande volume de consultas de banco de dados no DynamoDB
  - Extrair imagens do Amazon ECR

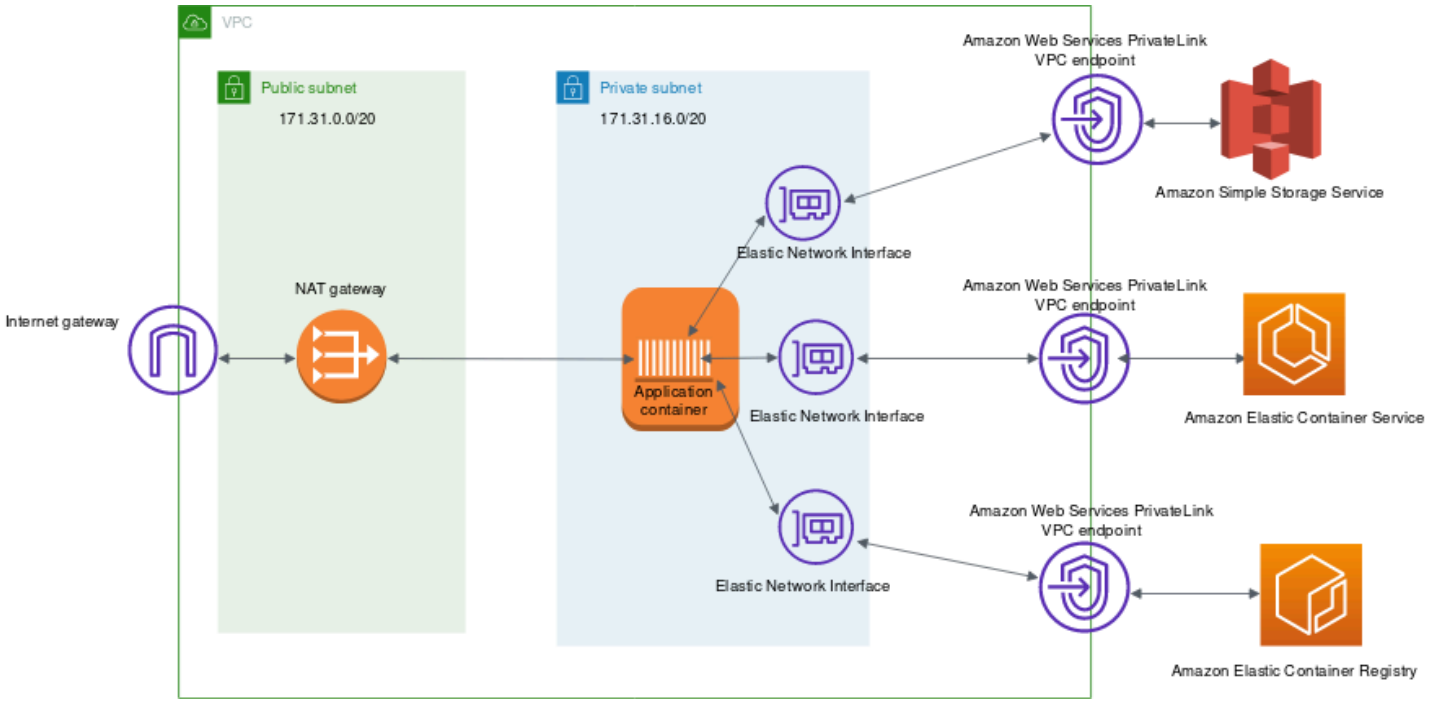
  Além disso, um gateway NAT comporta 5 Gbps de largura de banda e escala automaticamente até 45 Gbps. Se você fizer o roteamento por meio de um único gateway NAT, as aplicações que exigem conexões de largura de banda muito alta podem encontrar restrições de rede. Como solução alternativa, você pode dividir sua workload em várias sub-redes e atribuir a cada sub-rede seu próprio gateway NAT.

AWS PrivateLink

O AWS PrivateLink fornece conectividade privada entre VPCs, serviços da AWS e suas redes on-premises sem expor seu tráfego à Internet pública.

Um endpoint da VPC permite conexões privadas entre a VPC e os serviços da AWS compatíveis e serviços de endpoint da VPC. O tráfego entre a sua VPC e os outros serviços não sai da rede da Amazon. Um endpoint da VPC não exige um gateway da Internet ou um gateway privado virtual nem dispositivo NAT, conexão VPN ou conexão do AWS Direct Connect. As instâncias do Amazon EC2 na sua VPC não exigem que endereços IP públicos se comuniquem com recursos no serviço.

O diagrama a seguir mostra como a comunicação com os serviços da AWS funciona quando você usa endpoints da VPC em vez de um gateway da Internet. O AWS PrivateLink provisiona interfaces de rede elásticas (ENIs) dentro da sub-rede, e as regras de roteamento de VPC são usadas para enviar qualquer comunicação ao nome do host do serviço por meio da ENI diretamente para o serviço da AWS de destino. Esse tráfego não precisa mais usar o gateway NAT ou o gateway da Internet.



A seguir estão alguns dos endpoints da VPC comuns usados com o serviço Amazon ECS.

- Endpoints de gateway para o Amazon S3
- Endpoint da VPC do DynamoDB
- Endpoint da VPC do Amazon ECS

- Endpoint da VPC do Amazon ECR

Muitos outros serviços da AWS oferecem suporte a endpoints da VPC. Se você faz uso intenso de qualquer serviço da AWS, consulte a documentação específica desse serviço e saiba como criar um endpoint da VPC para esse tráfego.

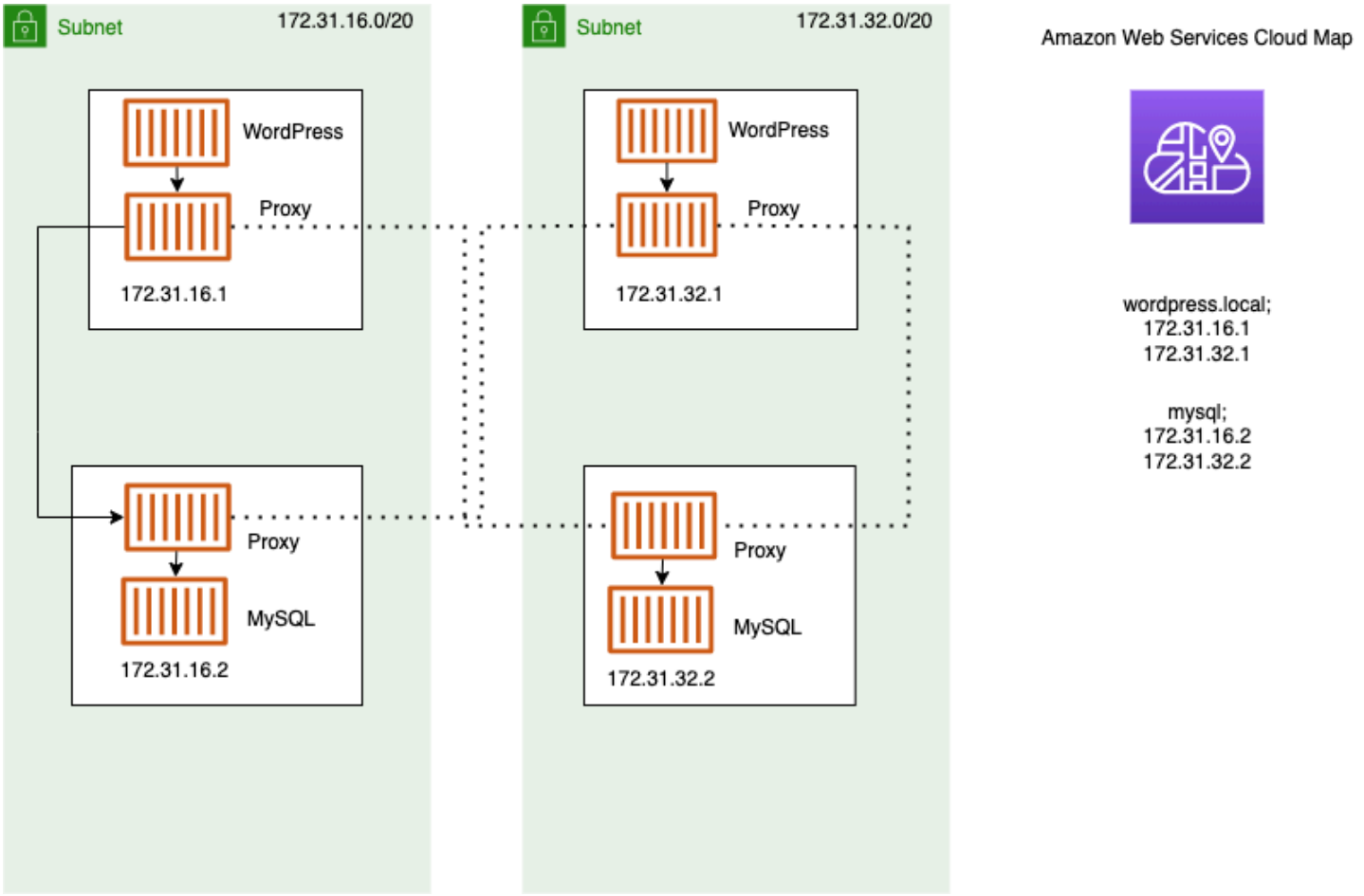
Práticas recomendadas para conectar serviços do Amazon ECS em uma VPC

Usando tarefas do Amazon ECS em uma VPC, é possível dividir aplicações monolíticas em partes separadas que podem ser implantadas e escaladas de forma independente em um ambiente seguro. Essa arquitetura é chamada de arquitetura orientada a serviços (SOA) ou microsserviços. No entanto, pode ser difícil garantir que todas essas partes, dentro e fora de uma VPC, possam se comunicar umas com as outras. Há várias abordagens para facilitar a comunicação, todas com vantagens e desvantagens diferentes.

Usar o Service Connect

Recomendamos o Service Connect, que fornece a configuração do Amazon ECS para descoberta de serviços, conectividade e monitoramento de tráfego. Com o Service Connect, as aplicações podem usar nomes curtos e portas padrão para se conectar a serviços no mesmo cluster e em outros clusters, inclusive entre VPCs na mesma região. Para obter mais informações, consulte Amazon ECS Service Connect.

Ao usar o Service Connect, o Amazon ECS gerencia todas as partes da descoberta de serviços: criar os nomes que podem ser descobertos, gerenciar dinamicamente as entradas de cada tarefa à medida que as tarefas começam e são interrompidas, executar um agente em cada tarefa configurada para descobrir os nomes. Sua aplicação pode pesquisar os nomes usando a funcionalidade padrão para nomes do DNS e fazendo conexões. Caso a aplicação já faça isso, não será necessário modificá-la para usar o Service Connect.



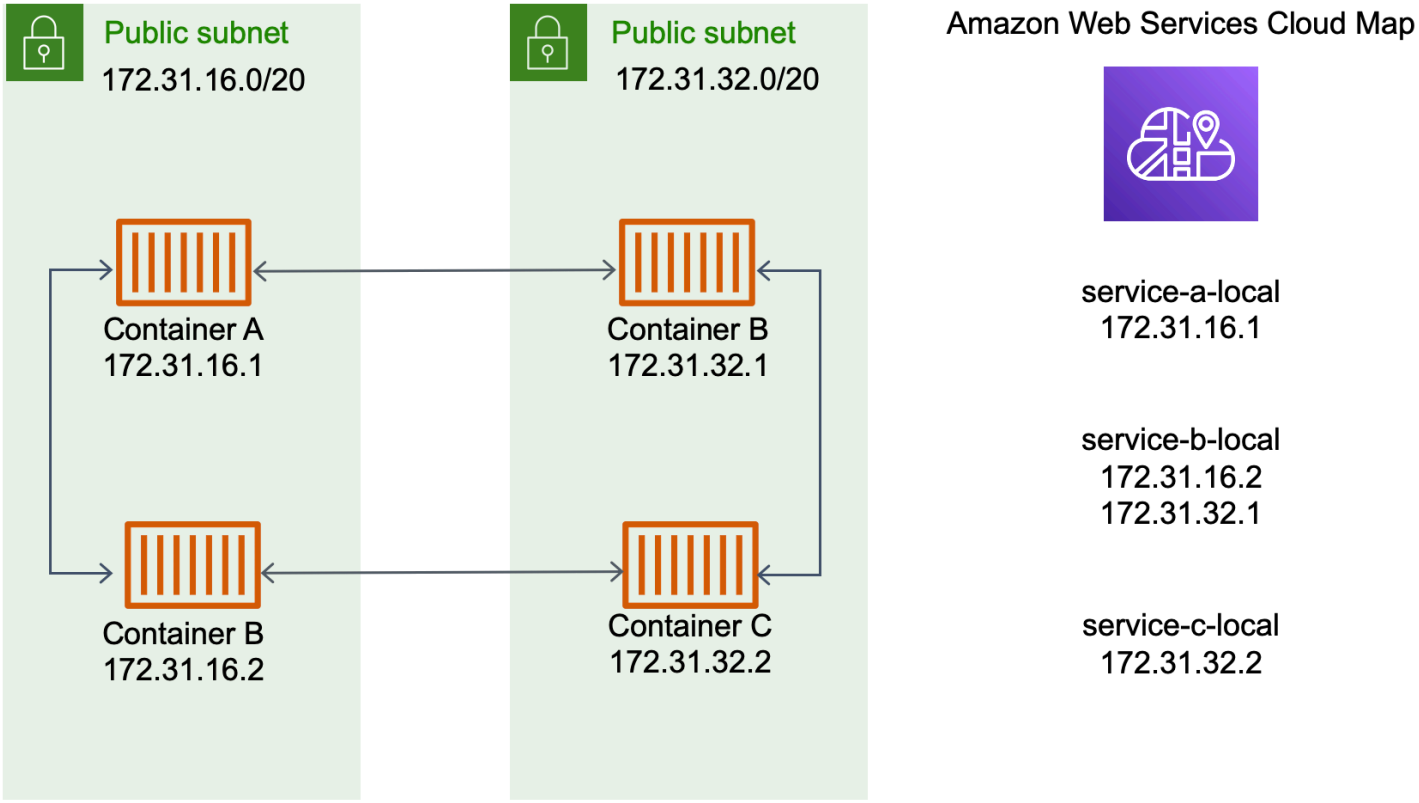
As mudanças só acontecem durante as implantações

Você fornece a configuração completa dentro de cada serviço e definição de tarefa. O Amazon ECS gerencia as alterações nessa configuração em cada implantação de serviço, para garantir que todas as tarefas em uma implantação se comportem da mesma maneira. Por exemplo, um problema comum do DNS como descoberta de serviços é controlar uma migração. Se você alterar um nome do DNS de modo a direcionar para os novos endereços IP substitutos, poderá levar o tempo máximo de TTL antes que todos os clientes comecem a usar o novo serviço. Com o Service Connect, a implantação do cliente atualiza a configuração substituindo as tarefas do cliente. Você pode configurar o disjuntor de implantação e outras configurações de implantação para afetar as alterações do Service Connect da mesma forma que qualquer outra implantação.

Usar descoberta de serviços

Outra abordagem para a comunicação entre serviços é a comunicação direta por meio da descoberta de serviços. Nessa abordagem, você pode usar a integração da descoberta de serviços do AWS Cloud Map com o Amazon ECS. Usando a descoberta de serviços, o Amazon ECS

sincroniza a lista de tarefas executadas com o AWS Cloud Map, que mantém um nome de host do DNS que resolve os endereços IP internos de uma ou mais tarefas desse serviço específico. Outros serviços na Amazon VPC podem usar esse nome de host do DNS para enviar tráfego diretamente a outro contêiner usando o endereço IP interno. Para obter mais informações, consulte Descoberta de serviços.



No diagrama anterior, há três serviços. `service-a-local` tem um contêiner e se comunica com `service-b-local`, que tem dois contêineres. `service-b-local` também deve se comunicar com `service-c-local`, que tem um contêiner. Cada contêiner em todos esses três serviços pode usar os nomes DNS internos de AWS Cloud Map para encontrar os endereços IP internos de um contêiner do serviço downstream com o qual ele precisa se comunicar.

Essa abordagem de comunicação entre serviços fornece baixa latência. À primeira vista, também é simples, pois não há componentes extras entre os contêineres. O tráfego viaja diretamente de um contêiner para o outro.

Essa abordagem é adequada ao usar o modo de rede `awsvpc`, em que cada tarefa tem seu próprio endereço IP exclusivo. A maioria dos softwares é compatível apenas com o uso de registros `A` do DNS, que resolvem diretamente os endereços IP. Ao usar o modo de rede `awsvpc`, o endereço IP de cada tarefa é um registro `A`. No entanto, se você estiver usando o modo de rede `bridge`,

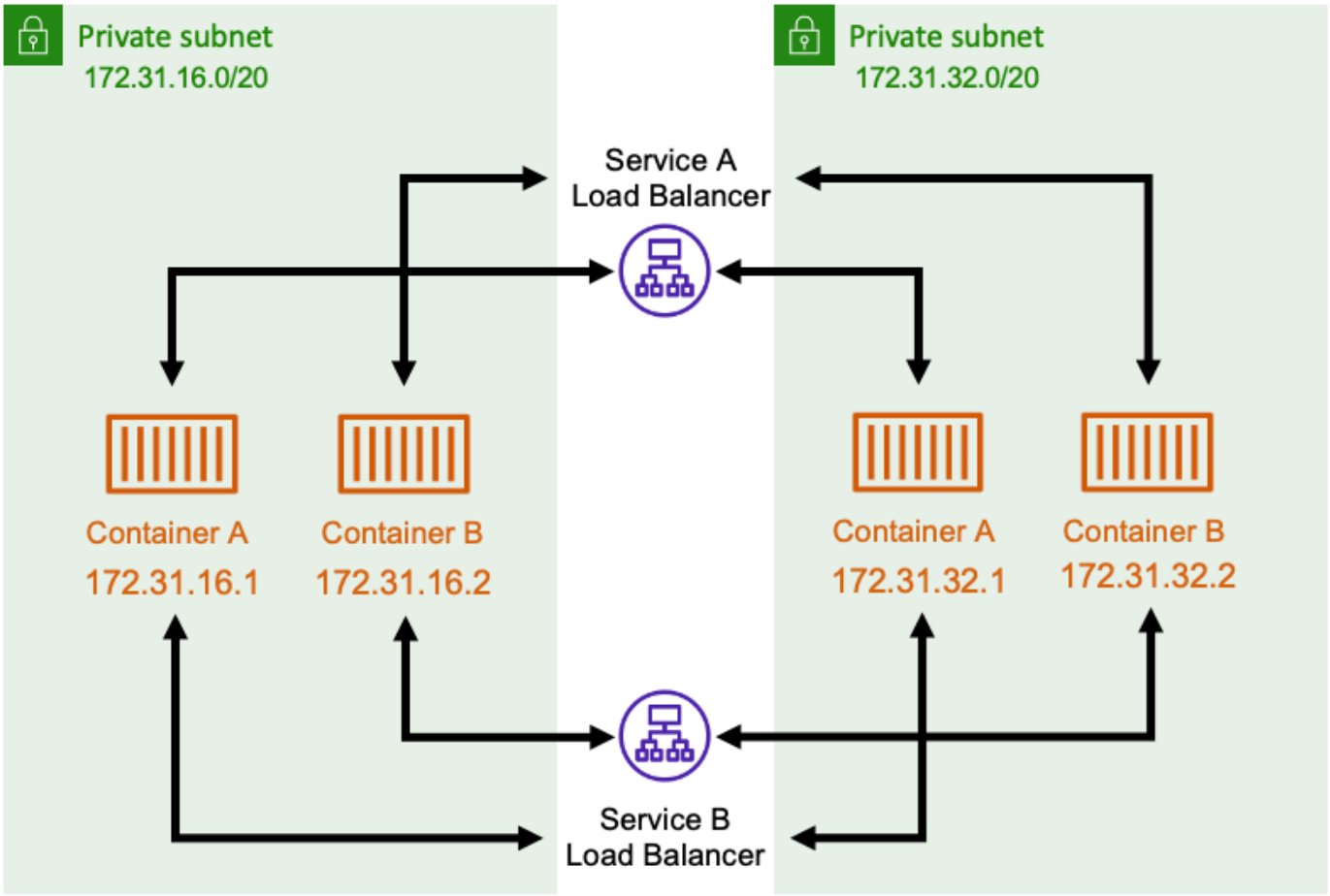
vários contêineres podem estar compartilhando o mesmo endereço IP. Além disso, os mapeamentos dinâmicos de portas fazem com que os contêineres recebam números de porta aleatoriamente nesse único endereço IP. A essa altura, um registro `A` não é mais suficiente para a descoberta de serviços. Você também deve usar um registro `SRV`. Esse tipo de registro pode rastrear endereços IP e números de porta, mas exige que você configure as aplicações adequadamente. Algumas aplicações pré-criadas que você usa podem não oferecer suporte a registros `SRV`.

Outra vantagem do modo de rede `awsvpc` é ter um grupo de segurança exclusivo para cada serviço. Você pode configurar esse grupo de segurança para permitir conexões de entrada somente dos serviços upstream específicos que precisam se comunicar com esse serviço.

A principal desvantagem da comunicação direta entre serviços usando a descoberta de serviços é que você deve implementar uma lógica extra para ter repetições e lidar com falhas de conexão. Os registros DNS têm um período de vida útil (TTL) que controla por quanto tempo são armazenados em cache. Leva algum tempo para que o registro DNS seja atualizado e o cache expire para que as aplicações possam obter a versão mais recente do registro DNS. Portanto, sua aplicação pode acabar resolvendo o registro DNS de modo a direcionar para outro contêiner que não está mais lá. A aplicação precisa processar repetições e ter uma lógica para ignorar backends problemáticos.

Usar um balanceador de carga interno

Outra abordagem à comunicação entre serviços é usar um balanceador de carga interno. Um balanceador de carga interno existe inteiramente dentro da sua VPC e só pode ser acessado por serviços dentro dela.



O balanceador de carga mantém a alta disponibilidade implantando recursos redundantes em cada sub-rede. Quando um contêiner de `serviceA` precisa se comunicar com um contêiner de `serviceB`, ele abre uma conexão para o balanceador de carga. O balanceador de carga então abre uma conexão para um contêiner a partir de `service B`. O balanceador de carga serve como um local centralizado para gerenciar todas as conexões entre cada serviço.

Se um contêiner de `serviceB` parar, o balanceador de carga poderá remover esse contêiner do grupo. O balanceador de carga também faz verificações de integridade em cada alvo downstream em seu grupo e pode remover automaticamente os alvos ruins do grupo até que eles se tornem saudáveis novamente. As aplicações não precisam mais saber quantos contêineres downstream existem. Elas simplesmente abrem suas conexões para o balanceador de carga.

Essa abordagem é vantajosa para todos os modos de rede. O balanceador de carga pode acompanhar os endereços IP da tarefa ao usar o modo de rede `awsvpc`, bem como combinações mais avançadas de endereços IP e portas ao usar o modo de rede `bridge`. Ele distribui uniformemente o tráfego em todas as combinações de endereços IP e portas, mesmo que vários

contêineres estejam realmente hospedados na mesma instância do Amazon EC2, mas apenas em portas diferentes.

A única desvantagem dessa abordagem é o custo. Para manter a alta disponibilidade, o balanceador de carga precisa ter recursos em cada zona de disponibilidade. Isso adiciona um custo extra devido à sobrecarga representada por pagar pelo balanceador de carga e pela quantidade de tráfego que passa pelo balanceador de carga.

No entanto, é possível reduzir os custos indiretos fazendo com que vários serviços compartilhem um balanceador de carga. Isso é particularmente adequado para serviços REST que usam um Application Load Balancer. É possível criar regras de roteamento baseadas em caminho que roteiam o tráfego para diferentes serviços. Por exemplo, `/api/user/*` pode rotear para um contêiner que faz parte do serviço `user`, enquanto `/api/order/*` pode rotear para o serviço `order` associado. Com essa abordagem, você paga apenas por um Application Load Balancer e tem um URL consistente para sua API. No entanto, é possível dividir o tráfego em vários microsserviços no backend.

Práticas recomendadas para conectar serviços do Amazon ECS em rede entre Contas da AWS e VPCs

Se você faz parte de uma organização com várias equipes e divisões, provavelmente implementa serviços de forma independente em VPCs separadas dentro de uma Conta da AWS compartilhada ou em VPCs associadas a várias Contas da AWS individuais. Independentemente da forma como você implanta seus serviços, recomendamos complementar seus componentes de rede para ajudar a rotear o tráfego entre as VPCs. Para isso, é possível usar vários serviços da AWS para complementar seus componentes de rede existentes.

- AWS Transit Gateway: considere esse serviço de rede primeiro. Ele atua como um hub central para rotear suas conexões entre Amazon VPCs, Contas da AWS e redes on-premises. Para obter mais informações, consulte O que é um transit gateway? no Guia de Transit Gateways da Amazon VPC.
- Suporte a Amazon VPCs e VPNs: é possível usar esse serviço para criar conexões VPN site a site para conectar redes on-premises à sua VPC. Para obter mais informações, consulte O que é o AWS Site-to-Site VPN? no Guia do usuário do AWS Site-to-Site VPN.
- Amazon VPC: use o emparelhamento da Amazon VPC para obter ajuda para conectar várias VPCs na mesma conta ou entre contas diferentes. Para obter mais informações, consulte O que é emparelhamento de VPC? no Guia de emparelhamento da Amazon VPC.

- VPCs compartilhadas: é possível usar uma VPC e sub-redes da VPC em várias contas da AWS. Para obter mais informações, consulte Trabalhar com VPCs compartilhadas no Guia do usuário da Amazon VPC.

Serviços da AWS para solução de problemas de rede do Amazon ECS

Os serviços e recursos a seguir podem ajudar você a obter informações sobre suas configurações de rede e serviços. Use essas informações para solucionar problemas de rede e otimizar melhor seus serviços.

CloudWatch Container Insights

O CloudWatch Container Insights coleta, agrega e resume métricas e logs das suas aplicações e microsserviços conteinerizados. As métricas incluem a utilização de recursos, como CPU, memória, disco e rede. Elas estão disponíveis em painéis automáticos do CloudWatch. Para obter mais informações, consulte Configurar o Container Insights no Amazon ECS no Guia do usuário do Amazon CloudWatch.

AWS X-Ray

O AWS X-Ray é um serviço de rastreamento que pode ser usado para coletar informações sobre as solicitações de rede feitas por sua aplicação. Você pode usar o SDK para instrumentar sua aplicação e capturar horários e códigos de resposta de tráfego entre seus serviços e entre seus serviços e endpoints de serviços da AWS. Para obter mais informações, consulte O que é o AWS X-Ray? no Guia do desenvolvedor do AWS X-Ray.

Você também pode explorar gráficos do AWS X-Ray de como seus serviços se relacionam uns com os outros. Ou ainda usá-los para explorar estatísticas agregadas sobre a performance de cada link entre serviços. Por fim, você pode se aprofundar em qualquer transação específica para ver como os segmentos que representam chamadas de rede estão associados a essa transação específica.

Use esses recursos para identificar se há um gargalo de rede ou se um serviço específico em sua rede não está funcionando conforme o esperado.

Logs de fluxo da VPC

É possível usar os logs de fluxo da Amazon VPC para analisar a performance da rede e depurar problemas de conectividade. Com os logs de fluxo da VPC ativados, você pode capturar um log de todas as conexões na sua VPC. Isso inclui conexões com interfaces de rede associadas ao Elastic

Load Balancing, Amazon RDS, gateways NAT e outros serviços da AWS importantes que talvez você esteja usando. Para obter mais informações, consulte Logs de fluxo da VPC no Guia do usuário do Amazon Virtual Private Cloud.

Dicas de ajuste fino da rede

É possível ajustar algumas configurações para melhorar sua rede.

nofile ulimit

Se você espera que sua aplicação trabalhe com alto volume de tráfego e gerencie muitas conexões simultâneas, leve em consideração a cota do sistema para o número de arquivos permitidos. Quando há muitos soquetes de rede abertos, cada um deve ser representado por um descritor de arquivo. Se a cota do descritor de arquivo for muito baixa, isso limitará seus soquetes de rede. Isso resulta em falhas nas conexões ou em erros. É possível atualizar a cota específica do contêiner para o número de arquivos na definição de tarefas do Amazon ECS. Se você estiver trabalhando no Amazon EC2 (em vez do AWS Fargate), talvez também seja necessário ajustar essas cotas na sua instância subjacente do Amazon EC2.

sysctl net

Outra categoria de configuração ajustável são as configurações `sysctl` net. Consulte as configurações específicas da distribuição Linux de sua escolha. Muitas dessas configurações ajustam o tamanho dos buffers de leitura e gravação. Isso pode ajudar em algumas situações na execução de instâncias do Amazon EC2 em grande escala que contêm muitos contêineres.

Otimização do tempo de inicialização da tarefa do Amazon ECS

Para acelerar a execução das tarefas, considere as recomendações a seguir.

- Cache de imagens de contêiner e instâncias de BinPack

  Caso use o tipo de execução do EC2, você pode configurar o comportamento de pull do agente de contêiner do Amazon ECS para `ECS_IMAGE_PULL_BEHAVIOR`: `prefer-cached`. O pull da imagem será feito remotamente se não houver uma imagem armazenada em cache. Caso contrário, a imagem armazenada em cache na instância será usada. A limpeza automática de imagens é desativada no contêiner para garantir que a imagem armazenada em cache não seja removida. Isso reduz o tempo de pull da imagem para execuções subsequentes. O efeito do armazenamento em cache é ainda maior quando há uma alta densidade de tarefas nas instâncias de contêiner, que você pode configurar usando a estratégia de posicionamento de `binpack`. O

armazenamento em cache de imagens de contêiner é especialmente benéfico para workloads baseadas em Windows, que costumam ter grandes tamanhos de imagem de contêiner (dezenas de GBs). Ao usar a estratégia de posicionamento de `binpack`, você também pode considerar o uso do truncamento da interface de rede elástica (ENI) para posicionar mais tarefas com o modo de rede `awsvpc` em cada instância de contêiner. O truncamento de ENI aumenta o número de tarefas que você pode executar no modo `awsvpc`. Por exemplo, uma instância c5.large que oferece suporte à execução simultânea de apenas duas tarefas pode executar até dez tarefas com o truncamento de ENI.

- Escolha um modo de rede ideal

  Embora existam muitos casos em que o modo de rede `awsvpc` é ideal, esse modo de rede pode aumentar inerentemente a latência de execução das tarefas. Isso ocorre porque, para cada tarefa no modo `awsvpc`, os fluxos de trabalho do Amazon ECS precisam provisionar e anexar uma ENI invocando as APIs do Amazon EC2, o que adiciona uma sobrecarga de vários segundos às execuções das tarefas. Por outro lado, uma vantagem importante de usar o modo de rede `awsvpc` é que cada tarefa tem um grupo de segurança para permitir ou negar tráfego. Isso significa que você tem maior flexibilidade para controlar as comunicações entre tarefas e serviços em um nível mais granular. Se a velocidade de implantação for sua prioridade, considere usar o modo `bridge` para acelerar a execução de tarefas. Para obter mais informações, consulte the section called “Modo de rede AWSVPC”.

- Acompanhe o ciclo de vida da execução de tarefas para encontrar oportunidades de otimização

  Muitas vezes, é difícil saber quanto tempo é necessário para a inicialização da aplicação. Iniciar a imagem de contêiner, executar scripts de inicialização e outras configurações durante a inicialização da aplicação pode levar bastante tempo. É possível usar o endpoint de metadados da tarefa para publicar métricas a fim de monitorar a hora de inicialização da aplicação de `StartedAt` na resposta de metadados do contêiner até a hora de `StartedAt` para sua tarefa ou serviço. Com esses dados, você entende como a aplicação está contribuindo para o tempo total de execução e encontra áreas onde é possível reduzir a sobrecarga desnecessária específica da aplicação e otimizar as imagens de contêiner. Para obter mais informações, consulte Práticas recomendadas de gerenciamento de capacidade e ajuste de escala automático do Amazon ECS.

- Escolher um tipo de instância ideal (para o tipo de inicialização do EC2)

  Escolher o tipo de instância correto baseia-se na reserva de recursos (por exemplo, CPU, memória) que você configura na tarefa. Portanto, ao dimensionar a instância, você pode calcular quantas tarefas podem ser posicionadas em uma única instância. Um exemplo simples de uma tarefa bem posicionada é hospedar quatro tarefas que exigem 0,5 vCPU e 2 GB de reservas de

memória em uma instância m5.large (com suporte para 2 vCPUs e 8 GB de memória). As reservas dessa definição de tarefa aproveitam ao máximo os recursos da instância.

Operação do Amazon ECS em grande escala

Ao começar a operar o Amazon ECS em grande escala, considere como as cotas de serviço e os controles de utilização de API do Amazon ECS e os Serviços da AWS que se integram ao Amazon ECS podem afetar você.

Tópicos

- Controles de utilização de API e cotas de serviço do Amazon ECS
- Lidar com problemas de controle de utilização do Amazon ECS

Controles de utilização de API e cotas de serviço do Amazon ECS

O Amazon ECS se integra a vários Serviços da AWS, incluindo o Elastic Load Balancing, o AWS Cloud Map e o Amazon EC2. Com essa forte integração, o Amazon ECS inclui vários recursos, como balanceamento de carga do serviço, descoberta de serviços, rede de tarefas e ajuste de escala automático de clusters. O Amazon ECS e os outros Serviços da AWS que ele integra mantêm cotas de serviço e limites de taxa de API para garantir desempenho e utilização consistentes. As cotas de serviço também evitam o provisionamento acidental de mais recursos do que o necessário e protegem contra ações maliciosas que possam aumentar sua fatura.

Ao se familiarizar com as cotas de serviço e os limites de taxa de API da AWS, você pode planejar a escalabilidade das workloads sem se preocupar com a degradação inesperada do desempenho. Para obter mais informações, consulte Cotas de serviço do Amazon ECS e Request throttling for the Amazon ECS API.

Ao escalar as workloads no Amazon ECS, considere a cota de serviço a seguir. Para instruções sobre como solicitar um aumento de cota de serviço, consulte Gerenciar cotas de serviço do Amazon ECS e do AWS Fargate no AWS Management Console.

- O AWS Fargate tem cotas que limitam o número de tarefas em execução simultânea em cada Região da AWS. Há cotas para tarefas sob demanda e do Fargate Spot no Amazon ECS. Cada cota de serviço também inclui todos os pods do Amazon EKS executados no Fargate. Para obter mais informações sobre cotas do Fargate, consulte Cotas de serviço do AWS Fargate no Guia do usuário do Amazon Elastic Container Service para AWS Fargate.

- Em tarefas executadas nas instâncias do Amazon EC2, o número máximo de instâncias do Amazon EC2 que você pode registrar em cada cluster é 5 mil. Caso use o ajuste de escala automático de cluster do Amazon ECS com um provedor de capacidade de grupo do Auto Scaling, ou caso gerencie instâncias do Amazon EC2 para o cluster sem ajuda, essa cota pode virar um gargalo de implantação. Se precisar de mais capacidade, você pode criar mais clusters ou solicitar um aumento na cota de serviço.
- Se você usa o ajuste de escala automático de cluster do Amazon ECS com um provedor de capacidade de grupo do Auto Scaling, considere a cota `Tasks in the PROVISIONING state per cluster` ao escalar os serviços. Essa cota é o número máximo de tarefas no estado `PROVISIONING` de cada cluster para as quais os provedores de capacidade podem aumentar a capacidade. Ao executar um grande número de tarefas ao mesmo tempo, você pode facilmente atingir essa cota. Um exemplo é se você implantar simultaneamente dezenas de serviços, cada um com centenas de tarefas. Quando isso acontece, o provedor de capacidade precisa executar novas instâncias de contêiner para posicionar as tarefas quando o cluster não tem capacidade suficiente. Enquanto o provedor de capacidade executa instâncias adicionais do Amazon EC2, é provável que o programador de serviços do Amazon ECS continue executando tarefas em paralelo. No entanto, essa atividade pode ter controle de utilização devido à capacidade insuficiente do cluster. O programador de serviços do Amazon ECS implementa uma estratégia de recuo e controle de utilização exponencial para repetir o posicionamento de tarefas à medida que novas instâncias de contêineres são executadas. Como resultado, você pode enfrentar tempos de implantação ou de aumento horizontal de escala mais lentos. Para evitar essa situação, você pode planejar as implantações de serviços em uma das opções a seguir. Implemente um grande número de tarefas que não exijam um aumento da capacidade de clusters, ou mantenha uma capacidade disponível de clusters para a inicialização de novas tarefas.

Além de considerar a cota de serviços do Amazon ECS ao escalar as workloads, considere também a cota de serviço dos outros Serviços da AWS integrados ao Amazon ECS. A seção a seguir aborda detalhadamente os principais limites de taxas de cada serviço e fornece recomendações para lidar com possíveis problemas de controle de utilização.

Elastic Load Balancing

Você pode configurar os serviços do Amazon ECS para usar o Elastic Load Balancing para distribuir o tráfego uniformemente entre as tarefas. Para obter mais informações sobre como escolher um balanceador de carga, consulte Uso do balanceamento de carga para distribuir o tráfego de serviço do Amazon ECS.

Cotas de serviço do Elastic Load Balancing

Ao escalar as workloads, considere as cotas de serviço do Elastic Load Balancing a seguir. A maioria das cotas de serviço do Elastic Load Balancing é ajustável, e você pode solicitar um aumento no console do Service Quotas.

Application Load Balancer

Ao usar um Application Load Balancer, dependendo do seu caso de uso, pode ser necessário solicitar um aumento de cota para:

- A cota de `Targets per Application Load Balancer`, que é o número de destinos por trás do Application Load Balancer.
- A cota de `Targets per Target Group per Region`, que é o número de destinos por trás dos grupos de destino.

Para obter mais informações, consulte Quotas for your Application Load Balancers.

Network Load Balancer

Há limitações mais rígidas no número de destinos que você pode registrar com um Network Load Balancer. Ao usar um Network Load Balancer, talvez você queira habilitar o suporte entre zonas, que vem com limitações adicionais de escalabilidade no `Targets per Availability Zone Per Network Load Balancer`, o número máximo de destinos por zona de disponibilidade para cada Network Load Balancer. Para obter mais informações, consulte Quotas for your Network Load Balancers.

Controle de utilização da API do Elastic Load Balancing

Ao configurar um serviço do Amazon ECS para usar um balanceador de carga, as verificações de integridade do grupo de destino devem ser aprovadas para que o serviço seja considerado íntegro. Para realizar essas verificações de integridade, o Amazon ECS invoca as operações de API do Elastic Load Balancing em seu nome. Caso tenha um grande número de serviços configurados com balanceadores de carga na conta, você pode tornar as implantações de serviços mais lentas devido ao possível controle de utilização específico das operações das APIs `RegisterTarget`, `DeregisterTarget` e `DescribeTargetHealth` do Elastic Load Balancing. Quando ocorre o controle de utilização, ocorrem erros de controle de utilização nas mensagens de eventos do serviço do Amazon ECS.

Caso enfrente problemas de controle de utilização na API do AWS Cloud Map, entre em contato com o Suporte para obter orientação sobre como aumentar os limites de controle de utilização na API do AWS Cloud Map. Para obter mais informações sobre como monitorar e solucionar problemas, como erros de controle de utilização, consulte Lidar com problemas de controle de utilização.

Interfaces de rede elástica

Quando as tarefas usam o modo de rede `awsvpc`, o Amazon ECS provisiona uma interface de rede elástica (ENI) exclusiva para cada tarefa. Quando os serviços do Amazon ECS usam um balanceador de carga do Elastic Load Balancing, essas interfaces de rede também são registradas como destinos no grupo de destino apropriado definido no serviço.

Cotas de serviço da interface de rede elástica

Ao executar tarefas que usam o modo de rede `awsvpc`, uma interface de rede elástica exclusiva é anexada a cada tarefa. Caso essas tarefas precisem ser acessadas pela Internet, atribua um endereço IP público à interface de rede elástica dessas tarefas. Ao escalar as workloads do Amazon ECS, considere estas duas cotas importantes:

- A cota de `Network interfaces per Region`, que é o número máximo de interfaces de rede em uma Região da AWS para sua conta.
- A cota de `Elastic IP addresses per Region`, que é o número máximo de endereços IP elásticos em uma Região da AWS.

Ambas as cotas de serviço são ajustáveis e você pode solicitar um aumento delas no console do Service Quotas. Para obter mais informações, consulte Cotas de serviço da Amazon VPC.

Em workloads do Amazon ECS hospedadas em instâncias do Amazon EC2, ao executar tarefas que usam o modo de rede `awsvpc`, considere a cota de serviço `Maximum network interfaces`, o número máximo de instâncias de rede para cada instância do Amazon EC2. Essa cota limita o número de tarefas que você pode posicionar em uma instância. Não é possível ajustar a cota e ela não está disponível no console do Service Quotas. Para obter mais informações, consulte Endereços IP por interface de rede por tipo de instância no Guia do usuário do Amazon EC2.

Embora você não possa alterar o número de interfaces de rede que podem ser anexadas a uma instância do Amazon EC2, é possível usar o recurso de truncamento da interface de rede elástica para aumentar o número de interfaces de rede disponíveis. Por exemplo, por padrão, uma instância `c5.large` pode ter até três interfaces de rede. A interface de rede primária da instância conta como

uma. Portanto, é possível anexar mais duas interfaces de rede à instância. Como cada tarefa que usa o modo de rede `awsvpc` requer uma interface de rede, normalmente só é possível executar duas dessas tarefas nesse tipo de instância. Isso pode levar à subutilização da capacidade do cluster. Se você habilitar o truncamento da interface de rede elástica, poderá aumentar a densidade da interface de rede para posicionar um número maior de tarefas em cada instância. Com o truncamento ativado, uma instância `c5.large` pode ter até 12 interfaces de rede. A instância tem a interface de rede primária e o Amazon ECS cria e anexa uma interface de rede de "truncamento" à instância. Como resultado, com essa configuração, você pode executar dez tarefas na instância em vez das duas tarefas padrão. Para obter mais informações, consulte Elastic network interface trunking.

Controle de utilização da API de interface de rede elástica

Ao executar tarefas que usam o modo de rede `awsvpc`, o Amazon ECS depende das APIs do Amazon EC2 mencionadas a seguir. Cada uma dessas APIs tem controles de utilização de API diferentes. Para obter mais informações, consulte Request throttling for the Amazon EC2 API.

- CreateNetworkInterface
- AttachNetworkInterface
- DetachNetworkInterface
- DeleteNetworkInterface
- DescribeNetworkInterfaces
- DescribeVpcs
- DescribeSubnets
- DescribeSecurityGroups
- DescribeInstances

Se as chamadas de API do Amazon EC2 receberem controle de utilização durante os fluxos de trabalho de provisionamento da interface de rede elástica, o programador de serviços do Amazon ECS repete automaticamente com recuos exponenciais. Às vezes, essas repetições automáticas podem levar a um atraso na inicialização de tarefas, resultando em velocidades de implantação mais lentas. Quando ocorrer o controle de utilização da API, você verá a mensagem `Operations are being throttled. Will try again later.` nas mensagens de eventos do serviço. Caso atenda consistentemente aos controle de utilização de API do Amazon EC2, entre em contato com o Suporte para obter orientações sobre como aumentar os limites de controle de utilização da API. Para obter mais informações sobre como monitorar e solucionar problemas de controle de utilização, consulte Handling throttling issues.

AWS Cloud Map

A descoberta de serviços do Amazon ECS usa APIs do AWS Cloud Map para gerenciar namespaces dos serviços do Amazon ECS. Se os serviços tiverem um grande número de tarefas, considere as recomendações a seguir. Para obter mais informações, consulte Considerações sobre descoberta de serviços do Amazon ECS.

Cotas de serviço do AWS Cloud Map

Quando os serviços do Amazon ECS são configurados para usar a descoberta de serviços, a cota de `Tasks per service`, que é o número máximo de tarefas do serviço, é afetada pela cota de serviço de `Instances per service` do AWS Cloud Map, que é o número máximo de instâncias desse serviço. Em particular, a cota de serviço do AWS Cloud Map diminui a quantidade de tarefas que você pode executar para, no máximo, mil tarefas por serviço. Não é possível alterar a cota do AWS Cloud Map. Para obter mais informações, consulte as Service Quotas do AWS Cloud Map.

Controle de utilização da API do AWS Cloud Map

O Amazon ECS chama as APIs `ListInstances`, `GetInstancesHealthStatus`, `RegisterInstance` e `DeregisterInstance` do AWS Cloud Map em seu nome. Elas ajudam na descoberta de serviços e realizam verificações de integridade ao executar uma tarefa. Quando vários serviços que usam a descoberta de serviços com um grande número de tarefas são implantados ao mesmo tempo, isso pode resultar na ultrapassagem dos limites de controle de utilização da API do AWS Cloud Map. Quando isso acontecer, é provável que surja a mensagem `Operations are being throttled. Will try again later` nas mensagens de eventos do serviço do Amazon ECS e que a velocidade de implantação e execução de tarefas fique mais lenta. O AWS Cloud Map não documenta os limites de controle de utilização dessas APIs. Caso enfrente problemas de controle de utilização, entre em contato com o Suporte para obter orientações sobre como aumentar os limites de controle de utilização da API. Para obter mais recomendações sobre como monitorar e solucionar problemas, como erros de controle de utilização, consulte Lidar com problemas de controle de utilização.

Lidar com problemas de controle de utilização do Amazon ECS

Os erros de controle de utilização se enquadram em duas categorias principais: controle de utilização síncrono e controle de utilização assíncrono.

Controle de utilização síncrono

Quando ocorre o controle de utilização síncrono, você recebe imediatamente uma resposta de erro do Amazon ECS. Essa categoria costuma ocorrer quando você chama as APIs do Amazon ECS enquanto executa tarefas ou cria serviços. Para obter mais informações sobre o controle de utilização envolvido e os limites relevantes do controle de utilização, consulte Request throttling for the Amazon ECS API.

Quando a aplicação inicia solicitações de API, por exemplo, usando a AWS CLI ou um AWS SDK, você pode corrigir o controle de utilização da API. Isso pode ser feito arquitetando a aplicação para lidar com os erros ou implementando uma estratégia de recuo exponencial e variação de sinal com lógica de repetição para as chamadas de API. Para obter mais informações, consulte Tempos limite, novas tentativas e recuo com variação de sinal.

Caso use um AWS SDK, a lógica de repetição automática está incorporada e é configurável.

Controle de utilização assíncrono

O controle de utilização assíncrono ocorre devido a fluxos de trabalho assíncronos em que o Amazon ECS ou o AWS CloudFormation podem estar chamando APIs em seu nome para provisionar recursos. É importante saber quais APIs da AWS o Amazon ECS invoca em seu nome. Por exemplo, a API `CreateNetworkInterface` é invocada em tarefas que usam o modo de rede `awsvpc`, e a API `DescribeTargetHealth` é invocada ao realizar verificações de integridade em tarefas registradas em um balanceador de carga.

Quando as workloads atingem uma escala considerável, o controle de utilização pode ser aplicado nessas operações de API. Ou seja, elas podem receber controle de utilização o suficiente para violar os limites impostos pelo Amazon ECS ou pelo AWS service (Serviço da AWS) que está sendo chamado. Por exemplo, se você implantar centenas de serviços, cada um com centenas de tarefas simultaneamente que usam o modo de rede `awsvpc`, o Amazon ECS invoca operações de API do Amazon EC2, como `CreateNetworkInterface`, e operações de API do Elastic Load Balancing, como `RegisterTarget` ou `DescribeTargetHealth`, para registrar a interface de rede elástica e o balanceador de carga, respectivamente. Essas chamadas de API podem exceder os limites da API, resultando em erros de controle de utilização. Veja a seguir um exemplo de erro de controle de utilização do Elastic Load Balancing incluído na mensagem de evento do serviço.

```
{
   "userIdentity":{
      "arn":"arn:aws:sts::111122223333:assumed-role/AWSServiceRoleForECS/ecs-service-
scheduler",
```

```
      "eventTime":"2022-03-21T08:11:24Z",
      "eventSource":"elasticloadbalancing.amazonaws.com",
      "eventName":" DescribeTargetHealth ",
      "awsRegion":"us-east-1",
      "sourceIPAddress":"ecs.amazonaws.com",
      "userAgent":"ecs.amazonaws.com",
      "errorCode":"ThrottlingException",
      "errorMessage":"Rate exceeded",
      "eventID":"0aeb38fc-229b-4912-8b0d-2e8315193e9c"
   }
}
```

Quando essas chamadas de API compartilham limites com outros tráfegos de API na conta, pode ser difícil monitorá-las, mesmo que sejam emitidas como eventos de serviço.

Monitorar o controle de utilização

É importante identificar quais solicitações de API recebem controle de utilização e quem as emite. Você pode usar o AWS CloudTrail, que monitora o controle de utilização e se integra ao CloudWatch, Amazon Athena e Amazon EventBridge. É possível configurar o CloudTrail para enviar eventos específicos para o CloudWatch Logs. Os insights de log do CloudWatch Logs investigam e analisam os eventos. Isso identifica detalhes em eventos de controle de utilização, como o usuário ou o perfil do IAM que fez a chamada e o número de chamadas de API que foram feitas. Para obter mais informações, consulte Monitoring CloudTrail log files with CloudWatch Logs.

Para obter mais informações sobre o CloudWatch Logs Insights e instruções sobre como consultar arquivos de log, consulte Analyzing log data with CloudWatch Logs Insights.

Com o Amazon Athena, você pode criar consultas e analisar dados usando o SQL padrão. Por exemplo, é possível criar uma tabela do Athena para analisar eventos do CloudTrail. Para obter mais informações, consulte Using the CloudTrail console to create an Athena table for CloudTrail logs.

Depois de criar uma tabela do Athena, você pode usar consultas SQL, como o exemplo a seguir, para investigar erros de `ThrottlingException`.

Substitua os valores das *`user-input`* pelos seus.

```
select eventname, errorcode,eventsource,awsregion, useragent,COUNT(*) count
FROM cloudtrail_table-name
where errorcode = 'ThrottlingException'
```

```
AND eventtime between '2024-09-24T00:00:08Z' and '2024-09-23T23:15:08Z'
group by errorcode, awsregion, eventsource, useragent, eventname
order by count desc;
```

O Amazon ECS também emite notificações de eventos para o Amazon EventBridge. Há eventos de mudança de estado do recurso e eventos de ação do serviço. Eles incluem eventos de controle de utilização de API, como `ECS_OPERATION_THROTTLED` e `SERVICE_DISCOVERY_OPERATION_THROTTLED`. Para obter mais informações, consulte Eventos de ação do serviço do Amazon ECS.

Esses eventos podem ser consumidos por um serviço, como o AWS Lambda, para realizar ações em resposta. Para obter mais informações, consulte Processo de eventos do Amazon ECS.

Se você executar tarefas autônomas, algumas operações de API, como `RunTask`, serão assíncronas, e as operações de repetição não serão executadas automaticamente. Nesses casos, você pode usar serviços, como o AWS Step Functions, com a integração do EventBridge para repetir operações de controle de utilização ou com falha. Para obter mais informações, consulte Manage a container task (Amazon ECS, Amazon SNS).

Usar o CloudWatch para monitorar controle de utilização

O CloudWatch oferece monitoramento de uso da API no namespace `Usage` em Por recurso da AWS. Essas métricas são registradas em log com o tipo API e o nome da métrica CallCount. Você pode criar alarmes para iniciar sempre que essas métricas atingirem um determinado limite. Para obter mais informações, consulte Visualizar as Service Quotas e definir alarmes.

O CloudWatch também oferece detecção de anomalias. Esse recurso usa machine learning para analisar e estabelecer linhas de base de acordo com o comportamento específico da métrica na qual você o habilitou. Se houver uma atividade incomum na API, você poderá usar esse recurso com os alarmes do CloudWatch. Para obter mais informações, consulte Usar a detecção de anomalias do CloudWatch.

Ao monitorar proativamente os erros de controle de utilização, você pode entrar em contato com o Suporte para aumentar os limites relevantes de controle de utilização e receber orientação para as necessidades exclusivas da sua aplicação.

Práticas recomendadas de gerenciamento de capacidade e ajuste de escala automático do Amazon ECS

Você pode executar workloads de aplicações em contêineres de todos os tamanhos no Amazon ECS. Isso inclui ambientes de teste mínimos e grandes ambientes de produção que operam em escala global.

Com o Amazon ECS, como em todos os serviços da AWS, você paga somente pelo que usar. Ao arquitetar sua aplicação adequadamente, você pode reduzir os custos consumindo somente os recursos necessários, quando precisar deles.

As recomendações a seguir mostram como executar suas workloads do Amazon ECS para alcançar suas metas de nível de serviço e, ao mesmo tempo, operar de forma econômica.

Tópicos

- Determinação do tamanho de tarefas do Amazon ECS
- Otimização do ajuste de escala automático de serviços do Amazon ECS
- Capacidade e disponibilidade do Amazon ECS
- Capacidade de clusters do Amazon ECS
- Escolha de tamanhos de tarefas do Fargate para o Amazon ECS
- Aceleração do provisionamento de capacidade de clusters do Amazon ECS com provedores de capacidade no Amazon EC2

Determinação do tamanho de tarefas do Amazon ECS

Uma das escolhas mais importantes a serem feitas ao implantar contêineres no Amazon ECS é sobre o tamanho deles e das tarefas. Os tamanhos de contêineres e tarefas são essenciais para o planejamento de escalabilidade e capacidade.

O Amazon ECS usa duas métricas de recursos para a capacidade: CPU e memória. O Amazon ECS mede a CPU em unidades de 1/1024 de uma vCPU completa (em que 1.024 unidades são iguais a 1 vCPU inteira). O Amazon ECS mede a memória em megabytes.

Na definição da tarefa, você pode declarar reservas e limites de recursos.

Ao declarar uma reserva, você declara a quantidade mínima de recursos que uma tarefa exige. Sua tarefa recebe, pelo menos, a quantidade de recursos solicitada. A aplicação pode conseguir usar

mais CPU ou memória do que a reserva declarada. No entanto, isso está sujeito a quaisquer limites que também tenham sido declarados.

Usar mais do que a quantidade da reserva é conhecido como intermitência. A expansão significa que sua aplicação usa mais recursos do que o reservado, mas permanece dentro dos limites declarados. O Amazon ECS garante as reservas. Por exemplo, se você usar instâncias do Amazon EC2 para fornecer capacidade, o Amazon ECS não colocará uma tarefa em uma instância em que a reserva não possa ser atendida.

Um limite é a quantidade máxima de unidades de CPU ou memória que o contêiner ou a tarefa pode usar. Se seu contêiner tentar usar mais CPU do que esse limite, o Amazon ECS controlará a utilização. Se seu contêiner tentar usar mais memória do que esse limite, o Amazon ECS interromperá seu contêiner.

Escolher esses valores pode ser um desafio. Os valores mais adequados para a aplicação dependem muito dos requisitos de recursos dela.

Testar a carga da aplicação é a chave para um planejamento bem-sucedido dos requisitos de recursos. O teste de carga ajuda você a entender melhor os requisitos da sua aplicação.

Aplicações sem estado

Em aplicações sem estado que escalam horizontalmente, como uma aplicação por trás de um balanceador de carga, recomendamos primeiro determinar a quantidade de memória que a aplicação consome ao atender às solicitações.

Para fazer isso, você pode usar ferramentas tradicionais, como `ps` ou `top`. Também é possível utilizar soluções de monitoramento, como o CloudWatch Container Insights.

Ao determinar uma reserva de CPU, considere como você deseja escalar a aplicação para atender às suas necessidades de negócios.

Você pode usar reservas de CPU menores, como 256 unidades de CPU (ou 1/4 de vCPU), para aumentar a escala horizontalmente de uma forma refinada que minimize os custos. Porém, elas podem não ser escaladas com rapidez suficiente para atender a picos significativos na demanda.

Você pode usar reservas maiores de CPU para aumentar e reduzir a escala horizontalmente com mais rapidez. Isso ajuda você a acompanhar os picos de demanda mais rapidamente. No entanto, reservas maiores de CPU são mais caras.

Outros aplicativos

Em aplicações que não são escaláveis horizontalmente, como operadores singleton ou servidores de banco de dados, a capacidade disponível e o custo representam suas considerações mais importantes.

Você deve escolher a quantidade de memória e CPU com base na necessidade indicada pelo teste de carga para atender ao seu objetivo no nível de serviço. O Amazon ECS garante que a aplicação seja colocada em um host com capacidade adequada.

Otimização do ajuste de escala automático de serviços do Amazon ECS

Um serviço do Amazon ECS é uma coleção gerenciada de tarefas. Cada serviço tem uma definição de tarefa associada, uma contagem de tarefas desejada e uma estratégia de posicionamento opcional.

O ajuste de escala automático de serviços do Amazon ECS funciona por meio do serviço Application Auto Scaling. O Application Auto Scaling usa métricas do CloudWatch como fonte para métricas de escalabilidade. Ele também usa os alarmes do CloudWatch para definir limites sobre quando aumentar ou reduzir horizontalmente a escala do serviço.

Você fornece os limites para escalabilidade. Você pode definir uma meta de métrica, chamada de escala de rastreamento de metas. Você também pode especificar limites, o que é chamado de escalabilidade de etapas.

Depois que o Application Auto Scaling é configurado, ele calcula continuamente a contagem adequada de tarefas desejadas para o serviço. Ele também notifica o Amazon ECS quando a contagem de tarefas desejada deve mudar, seja aumentando ou reduzindo a escala horizontalmente.

Para usar o ajuste de escala automático do serviço de forma eficaz, você deve escolher uma métrica de escalabilidade apropriada. Vamos abordamos como escolher uma métrica nas seções a seguir.

Caracterização da aplicação

A escalabilidade adequada de uma aplicação exige que você conheça as condições em que você deve reduzir e aumentar a escala horizontalmente dela.

Em essência, você deverá aumentar a escala horizontalmente se houver previsão de que a demanda vá ultrapassar a capacidade. Por outro lado, você poderá reduzir a escala horizontalmente da aplicação para conservar os custos quando os recursos excederem a demanda.

Identificação de uma métrica de utilização

Para escalar com eficácia, você deve identificar uma métrica que indique a utilização ou saturação. Essa métrica deve exibir as propriedades a seguir para ser útil ao escalar.

- A métrica deve estar correlacionada com a demanda. Quando os recursos são mantidos estáveis, mas a demanda muda, o valor da métrica também deve mudar. A métrica deve aumentar ou diminuir quando a demanda aumenta ou diminui.
- O valor da métrica deve reduzir a escala horizontalmente de modo proporcional à capacidade. Quando a demanda se mantém constante, a adição de mais recursos deve resultar em uma mudança proporcional no valor da métrica. Portanto, dobrar o número de tarefas deve fazer com que a métrica diminua em 50%.

A melhor maneira de identificar uma métrica de utilização é por meio de testes de carga em um ambiente de pré-produção, como um ambiente de teste. Soluções de teste de carga comerciais e de código aberto estão amplamente disponíveis. Normalmente, essas soluções podem gerar carga sintética ou simular tráfego real de usuários.

Para iniciar o processo de teste de carga, você deve primeiro criar painéis para as métricas de utilização da aplicação. Essas métricas incluem utilização da CPU, utilização da memória, operações de E/S, profundidade da fila de E/S e throughput da rede. Você pode coletar essas métricas com um serviço, como o CloudWatch Container Insights. Também é possível coletá-las usando o Amazon Managed Service for Prometheus junto com o Amazon Managed Grafana. Durante esse processo, certifique-se de coletar e traçar métricas para os tempos de resposta da aplicação ou para as taxas de conclusão do trabalho.

Ao testar a carga, comece com uma pequena taxa de solicitação ou inserção de trabalho. Mantenha essa taxa estável por alguns minutos para permitir o aquecimento da aplicação. Em seguida, aumente lentamente a taxa e mantenha-a estável por alguns minutos. Repita esse ciclo, aumentando a taxa a cada vez, até que os tempos de resposta ou conclusão da aplicação estejam muito lentos para atender aos objetivos de nível de serviço (SLOs).

Durante o teste de carga, examine cada uma das métricas de utilização. As métricas que aumentam com a carga são as principais candidatas a melhores métricas de utilização.

Em seguida, identifique o recurso que atinge a saturação. Ao mesmo tempo, também examine as métricas de utilização para ver qual delas se estabiliza primeiro em um nível alto. Ou examine qual delas atinge o pico e, em seguida, trava sua aplicação primeiro. Por exemplo, se a utilização da

CPU aumentar de 0% para 70% a 80% à medida que você adiciona carga, e permanecer nesse nível após adicionar ainda mais carga, é seguro dizer que a CPU está saturada. Dependendo da arquitetura da CPU, talvez nunca chegue a 100%. Por exemplo, suponha que a utilização da memória aumente à medida que você adiciona carga e, em seguida, a aplicação trava repentinamente ao atingir o limite de memória da tarefa ou da instância do Amazon EC2. Nessa situação, é provável que a memória tenha sido totalmente consumida. Vários recursos podem ser consumidos pela aplicação. Portanto, escolha a métrica que representa o recurso que se esgota primeiro.

Por fim, tente testar a carga novamente depois de dobrar o número de tarefas ou de instâncias do Amazon EC2. Suponha que a métrica principal aumente ou diminua pela metade da taxa de antes. Se for esse o caso, a métrica é proporcional à capacidade. Essa é uma boa métrica de utilização para ajuste de escala automático.

Agora, considere esse cenário hipotético. Suponha que você teste a carga de uma aplicação e descubra que a utilização da CPU chega a 80% com 100 solicitações por segundo. Quando mais carga é adicionada, a utilização da CPU não aumenta mais. No entanto, isso faz com que a aplicação responda mais lentamente. Em seguida, você executa o teste de carga novamente, dobrando o número de tarefas, mas mantendo a taxa no valor de pico anterior. Se você descobrir que a utilização média da CPU cai para cerca de 40%, a utilização média da CPU é uma boa candidata para uma métrica de escalabilidade. Por outro lado, se a utilização da CPU permanecer em 80% após aumentar o número de tarefas, a utilização média da CPU não é uma boa métrica de escalabilidade. Nesse caso, você precisa fazer mais pesquisa para encontrar uma métrica adequada.

Modelos de aplicações e propriedades de escala comuns

É possível executar softwares de todos os tipos na AWS. Muitas workloads são desenvolvidas internamente, enquanto outras são baseadas em softwares populares de código aberto. Independentemente de sua origem, observamos alguns padrões comuns de design de serviços. A forma eficaz de escalar depende, em grande parte, do padrão.

O servidor eficiente vinculado à CPU

O servidor eficiente vinculado à CPU não utiliza quase nenhum recurso além da CPU e do throughput da rede. Cada solicitação pode ser processada apenas pela aplicação. As solicitações não dependem de outros serviços, como bancos de dados. A aplicação pode processar centenas de milhares de solicitações simultâneas e utilizar com eficiência várias CPUs para isso. Cada solicitação é atendida por um thread dedicado com pouca sobrecarga de memória ou há um loop de

eventos assíncrono executado em cada CPU que atende às solicitações. Cada réplica da aplicação é igualmente capaz de processar uma solicitação. O único recurso que pode ser esgotado antes da CPU é a largura de banda da rede. Em serviços vinculados à CPU, a utilização da memória, mesmo no pico de throughput, é uma fração dos recursos disponíveis.

Você pode usar o ajuste de escala automático baseado em CPU para esse tipo de aplicação. A aplicação desfruta da máxima flexibilidade em termos de escala. Ela pode ser escala verticalmente, o que fornece à aplicação instâncias maiores do Amazon EC2 ou vCPUs do Fargate. Além disso, ela também pode ser escalada horizontalmente adicionando mais réplicas. Adicionar réplicas ou dobrar o tamanho da instância reduz pela metade a utilização média da CPU em relação à capacidade.

Se estiver usando a capacidade do Amazon EC2 nessa aplicação, considere colocá-la em instâncias otimizadas para computação, como a família `c5` ou `c6g`.

O servidor eficiente vinculado à memória

O servidor eficiente vinculado à memória aloca uma quantidade significativa de memória por solicitação. Na máxima simultaneidade, mas não necessariamente no throughput, a memória se esgota antes dos recursos da CPU. A memória associada a uma solicitação é liberada quando a solicitação se encerra. Solicitações adicionais podem ser aceitas desde que haja memória disponível.

Você pode usar ajuste de escala automático baseado em memória para esse tipo de aplicação. A aplicação desfruta da máxima flexibilidade em termos de escala. Você pode escalá-la verticalmente, fornecendo a ela maiores recursos de memória do Amazon EC2 ou do Fargate. Além disso, ela também pode ser escalada horizontalmente adicionando mais réplicas. Adicionar réplicas ou dobrar o tamanho da instância pode reduzir pela metade a utilização média da memória em relação à capacidade.

Se estiver usando a capacidade do Amazon EC2 nessa aplicação, considere colocá-la em instâncias otimizadas para memória, como a família `r5` ou `r6g`.

Algumas aplicações vinculadas à memória não liberam a memória associada a uma solicitação quando ela se encerra, de modo que uma redução na simultaneidade não resulta em uma redução na memória usada. Para isso, não recomendamos usar a escalabilidade baseada em memória.

O servidor baseado em operadores

O servidor baseado em operadores processa uma solicitação para cada thread de operador individual, uma após a outra. Os threads de operador podem ser leves, como os threads POSIX. Eles

também podem ser mais pesados, como os processos UNIX. Não importa qual thread eles sejam, sempre há uma simultaneidade máxima que a aplicação pode suportar. Normalmente, o limite de simultaneidade é definido proporcionalmente aos recursos de memória disponíveis. Se o limite de simultaneidade for atingido, a aplicação colocará solicitações adicionais em uma fila de backlog. Se a fila de backlog transbordar, solicitações adicionais de entrada serão imediatamente rejeitadas. As aplicações comuns que se encaixam nesse padrão incluem o servidor Web Apache e o Gunicorn.

A simultaneidade de solicitações costuma ser a melhor métrica para escalar essa aplicação. Como há um limite de simultaneidade para cada réplica, é importante aumentar a escala horizontalmente antes que o limite médio seja atingido.

A melhor maneira de obter métricas de simultaneidade de solicitações é fazer com que a aplicação as relate ao CloudWatch. Cada réplica da aplicação pode publicar o número de solicitações simultâneas como uma métrica personalizada em alta frequência. Recomendamos que a frequência seja definida para, pelo menos, uma vez a cada minuto. Depois que vários relatórios são coletados, você pode usar a simultaneidade média como métrica de escalabilidade. Essa métrica é calculada pegando a simultaneidade total e dividindo-a pelo número de réplicas. Por exemplo, se a simultaneidade total for 1.000 e o número de réplicas for 10, a simultaneidade média será 100.

Se a aplicação estiver por trás de um Application Load Balancer, você também pode usar a métrica `ActiveConnectionCount` do balanceador de carga como fator na métrica de escalabilidade. A métrica `ActiveConnectionCount` deve ser dividida pelo número de réplicas para obter um valor médio. O valor médio deve ser usado para ajuste de escala, em oposição ao valor bruto da contagem.

Para que esse projeto funcione melhor, o desvio padrão da latência de resposta deve ser pequeno em taxas de solicitação baixas. Recomendamos que, durante períodos de baixa demanda, a maioria das solicitações seja respondida em pouco tempo e que não haja muitas solicitações que demorem muito mais do que o tempo médio para serem respondidas. O tempo médio de resposta deve estar próximo ao tempo de resposta do 95º percentil. Caso contrário, podem ocorrer transbordamos de fila como resultado. Isso leva a erros. Recomendamos que você forneça réplicas adicionais quando necessário para reduzir o risco de transbordamento.

O servidor em espera

O servidor em espera realiza algum processamento para cada solicitação, mas dependente muito de um ou mais serviços downstream para funcionar. As aplicações de contêiner costumam fazer uso intenso de serviços downstream, como bancos de dados e outros serviços de API. Pode levar algum tempo para que esses serviços respondam, principalmente em cenários de alta capacidade ou alta

simultaneidade. Isso ocorre porque essas aplicações tendem a usar poucos recursos da CPU e a utilizar sua máxima simultaneidade em termos de memória disponível.

O serviço em espera é adequado tanto no padrão de servidor vinculado à memória quanto no padrão de servidor baseado em operadores, dependendo de como a aplicação foi projetada. Se a simultaneidade da aplicação for limitada somente pela memória, a utilização média da memória deve ser usada como métrica de escalabilidade. Se a simultaneidade da aplicação for baseada em um limite de operadores, a simultaneidade média deve ser usada como uma métrica de escalabilidade.

O servidor baseado em Java

Se o servidor baseado em Java estiver vinculado à CPU e escalar proporcionalmente aos recursos da CPU, ele pode ser adequado para o padrão eficiente de servidor vinculado à CPU. Se for esse o caso, a utilização média da CPU pode ser apropriada como métrica de escalabilidade. No entanto, muitas aplicações Java não são vinculadas à CPU, o que as torna difíceis de escalar.

Para obter o melhor desempenho, recomendamos alocar o máximo de memória possível no heap da Java Virtual Machine (JVM). Versões recentes da JVM, incluindo a atualização 191 ou posterior do Java 8, definem automaticamente o tamanho do heap o maior possível para caber no contêiner. Isso significa que, em Java, a utilização da memória raramente é proporcional à utilização da aplicação. À medida que a taxa de solicitações e a simultaneidade aumentam, a utilização da memória permanece constante. Por isso, não recomendamos escalar servidores baseados em Java com base na utilização da memória. Em vez disso, recomendamos escalar a utilização da CPU.

Em alguns casos, os servidores baseados em Java esgotam o heap antes de esgotar a CPU. Se a aplicação for propensa ao esgotamento de heap com alta simultaneidade, as conexões médias são a melhor métrica de escalabilidade. Se a aplicação for propensa ao esgotamento de heap com alto throughput, a taxa média de solicitações é a melhor métrica de escalabilidade.

Servidores que usam outros runtimes com coleta de resíduos

Muitas aplicações de servidor são baseadas em runtimes que realizam coleta de resíduos, como .NET e Ruby. Essas aplicações de servidor podem se encaixar em um dos padrões descritos anteriormente. No entanto, como acontece em Java, não recomendamos escalar essas aplicações com base na memória, porque a utilização média de memória observada não costuma estar correlacionada ao throughput ou à simultaneidade.

Para essas aplicações, recomendamos escalar a utilização da CPU se a aplicação estiver vinculada à CPU. Caso contrário, recomendamos escalar o throughput médio ou a simultaneidade média, com base nos resultados do teste de carga.

Processadores de trabalhos

Muitas workloads envolvem o processamento assíncrono de trabalhos. Elas incluem aplicações que não recebem solicitações em tempo real, mas se inscrevem em uma fila de trabalho para receber tarefas. Para esses tipos de aplicações, a métrica de escalabilidade adequada é quase sempre a profundidade da fila. O crescimento da fila é uma indicação de que o trabalho pendente supera a capacidade de processamento, enquanto uma fila vazia indica que há mais capacidade do que trabalho a ser feito.

Serviços de mensagens da AWS, como Amazon SQS e Amazon Kinesis Data Streams, fornecem métricas do CloudWatch que podem ser usadas para escalar. No Amazon SQS, `ApproximateNumberOfMessagesVisible` é a melhor métrica. No Kinesis Data Streams, considere usar a métrica `MillisBehindLatest`, publicada pela Kinesis Client Library (KCL). Essa métrica deve ser calculada em média entre todos os consumidores antes de usá-la para escalar.

Capacidade e disponibilidade do Amazon ECS

A disponibilidade da aplicação é crucial para fornecer uma experiência sem erros e minimizar a latência da aplicação. A disponibilidade depende de recursos acessíveis e com capacidade suficiente para atender à demanda. A AWS fornece diversos mecanismos para gerenciar a disponibilidade. Para aplicações hospedadas no Amazon ECS, isso inclui o ajuste de escala automático e as zonas de disponibilidade (AZs). O ajuste de escala automático gerencia o número de tarefas ou de instâncias com base nas métricas que você define, enquanto as zonas de disponibilidade permitem a hospedagem da aplicação em locais isolados, mas geograficamente próximos.

Assim como acontece com os tamanhos das tarefas, a capacidade e a disponibilidade apresentam determinadas compensações que você devem ser consideradas. Idealmente, a capacidade estaria perfeitamente alinhada com a demanda. Sempre haveria capacidade suficiente para atender a solicitações e processar trabalhos com a finalidade de cumprir os Objetivos de Nível de Serviço (SLOs), incluindo a baixa latência e a taxa de erro. A capacidade nunca seria excessivamente alta, resultando em custos desnecessários, nem demasiadamente baixa, causando altas latências e taxas de erro.

O ajuste de escala automático é um processo latente. Primeiro, o CloudWatch deve receber métricas em tempo real. Em seguida, o CloudWatch precisa agregá-las para análise, o que pode demorar alguns minutos, dependendo da granularidade da métrica. O CloudWatch compara as métricas com os limites de alarme para identificar uma escassez ou um excesso de recursos. Para evitar instabilidade, você deve configurar os alarmes para exigir que o limite definido seja ultrapassado

por alguns minutos antes de o alarme disparar. Além disso, demora algum tempo para efetuar o provisionamento do novas tarefas e encerrar as tarefas que não são mais necessárias.

Devido a esses possíveis atrasos no sistema, é importante que você mantenha uma margem de segurança por meio do superprovisionamento. O superprovisionamento pode ajudar a acomodar picos de demanda de curto prazo. Além disso, essa ação ajuda a aplicação a atender a solicitações adicionais sem atingir a saturação. Como uma boa prática, é possível definir a meta de ajuste de escala entre 60% e 80% de utilização. Isso auxilia a aplicação a lidar melhor com os picos de demanda extra enquanto a capacidade adicional ainda está em processo de provisionamento.

Outro motivo pelo qual recomendamos o provisionamento excessivo é para que você possa responder com rapidez às falhas da zona de disponibilidade. A AWS recomenda que as workloads de produção sejam disponibilizadas de diversas zonas de disponibilidade. Isso ocorre porque, se ocorrer uma falha na zona de disponibilidade, as tarefas que estão em execução nas zonas de disponibilidade restantes ainda poderão atender à demanda. Se a sua aplicação for executada em duas zonas de disponibilidade, você precisará dobrar a contagem normal de tarefas. Isso ocorre para que você possa fornecer capacidade imediata durante qualquer falha potencial. Se a sua aplicação for executada em três zonas de disponibilidade, recomendamos executar 1,5 vezes a contagem normal de tarefas. Em outras palavras, execute três tarefas para cada duas necessárias durante o serviço normal.

Maximização da velocidade de ajuste de escala

O ajuste de escala automático é um processo reativo que demora algum tempo para entrar em vigor. No entanto, existem algumas maneiras de ajudar a minimizar o tempo necessário para aumentar a escala horizontalmente.

Minimize o tamanho da imagem. As imagens em tamanhos maiores demoram mais para serem baixadas de um repositório de imagens e descompactadas. Portanto, manter tamanhos de imagem menores reduz o tempo que é necessário para iniciar um contêiner. Para reduzir o tamanho de uma imagem, é possível seguir estas recomendações específicas:

- Se for possível criar um binário estático ou usar a linguagem de programação Go, compile a imagem `FROM` zero e inclua somente a aplicação com o binário na imagem resultante.
- Use imagens base minimizadas de fornecedores de distribuição primários, como o Amazon Linux ou o Ubuntu.
- Não inclua nenhum artefato de compilação na imagem final. O uso de compilações em vários estágios pode ajudar nisso.

- Realize a compactação de estágios RUN sempre que possível. Cada estágio RUN cria uma nova camada de imagem, resultando em uma ida e volta adicional para efetuar o download da camada. Um único estágio RUN com diversos comandos unidos por && tem menos camadas do que um com vários estágios RUN.
- Se desejar incluir dados, como dados de inferência de ML, na imagem final, inclua somente os dados necessários para iniciar e começar a atender o tráfego. Se você precisar acessar dados sob demanda do Amazon S3 ou de outro armazenamento sem impactar o serviço, armazene esses dados localmente.

Mantenha as imagens por perto. Quanto maior a latência da rede, mais tempo demora para efetuar o download da imagem. Hospede as imagens em um repositório na mesma região da AWS em que a workload está hospedada. O Amazon ECR é um repositório de imagens de alta performance disponível em todas as regiões em que o Amazon ECS está disponível. Evite navegar pela Internet ou por um link de VPN para efetuar o download de imagens de contêiner. A hospedagem de imagens na mesma região contribui para uma maior confiabilidade geral. Isso mitiga o risco de problemas de conectividade de rede e de disponibilidade que poderiam ser causados por uma região diferente. Como alternativa, também é possível implementar a replicação entre regiões do Amazon ECR para ajudar nisso.

Reduza os limites de verificação de integridade do balanceador de carga. Os balanceadores de carga executam verificações de integridade antes de enviar tráfego para a aplicação. A configuração padrão da verificação de integridade para um grupo de destino pode demorar 90 segundos ou mais. Durante esse momento, o balanceador de carga verifica o status de integridade e recebe solicitações. A redução de tempo entre o intervalo de verificação de integridade e a contagem de limites pode fazer com que a aplicação aceite o tráfego com maior rapidez e reduza a carga em outras tarefas.

Considere o desempenho da inicialização a frio. Algumas aplicações usam runtimes, como Java, para realizar a compilação Just-In-Time (JIT). O processo de compilação, pelo menos na inicialização, pode mostrar o desempenho da aplicação. Uma solução alternativa é reescrever as partes críticas em termos de latência da workload em linguagens que não impõem uma penalidade de desempenho durante a inicialização a frio.

Use o ajuste de escala em etapas em vez de políticas de ajuste de escala com rastreamento de destino. Você tem várias opções de ajuste de escala automático de aplicação para tarefas do Amazon ECS. O rastreamento de destinos é o modo mais fácil de usar. Com isso, tudo o que você precisa fazer é definir um valor de destino para uma métrica, como a utilização média da CPU. Em

seguida, o escalador automático gerencia automaticamente o número de tarefas necessárias para atingir esse valor. Com a escalabilidade por etapas, é possível reagir mais rapidamente às mudanças na demanda, pois define os limites específicos para suas métricas de escalabilidade e quantas tarefas adicionar ou remover quando os limites são ultrapassados. E, o mais importante, é possível reagir rapidamente às mudanças na demanda minimizando a quantidade de tempo em que um alarme de limite é violado. Para obter mais informações, consulte Usar escalabilidade automática no Guia do desenvolvedor do Amazon Elastic Container Service.

Se você estiver usando instâncias do Amazon EC2 para fornecer capacidade de cluster, considere as seguintes recomendações:

Use instâncias maiores do Amazon EC2 e volumes mais rápidos do Amazon EBS. É possível aumentar as velocidades de download e de preparação de imagens ao usar uma instância maior do Amazon EC2 e um volume mais rápido do Amazon EBS. Em uma determinada família de instância do Amazon EC2, o throughput máximo da rede e do Amazon EBS aumenta à medida que o tamanho da instância aumenta (por exemplo, de `m5.xlarge` para `m5.2xlarge`). Além disso, também é possível personalizar os volumes do Amazon EBS para aumentar o throughput e as IOPS. Por exemplo, se você estiver usando volumes `gp2`, opte por usar volumes maiores que ofereçam mais throughput de linha de base. Se você estiver usando volumes `gp3`, especifique o throughput e as IOPS ao criar o volume.

Use o modo de rede bridge para tarefas em execução nas instâncias do Amazon EC2. As tarefas que usam o modo de rede `bridge` no Amazon EC2 são iniciadas com maior rapidez do que as tarefas que usam o modo de rede `awsvpc`. Quando o modo de rede `awsvpc` é usado, o Amazon ECS anexa uma interface de rede elástica (ENI) à instância antes de iniciar a tarefa. Isso introduz latência adicional. Existem várias vantagens e desvantagens no uso de redes bridge. Essas tarefas não recebem o próprio grupo de segurança e há algumas implicações para o balanceamento de carga. Para obter mais informações, consulte Load balancer target groups no Guia do usuário do Elastic Load Balancing.

Como lidar com choques de demanda

Algumas aplicações sofrem grandes choques repentinos na demanda. Isso acontece por diversos motivos, por exemplo, um evento de notícias, uma grande liquidação, um evento de mídia ou algum outro evento que se torna viral e faz com que o tráfego aumente rápida e significativamente em um espaço de tempo demasiadamente curto. Se não houver planejamento, isso pode fazer com que a demanda ultrapasse os recursos disponíveis com rapidez.

A melhor forma de lidar com choques de demanda é antecipá-los e planejar adequadamente. Como o ajuste de escala automático pode demorar algum tempo, recomendamos aumentar a escala horizontalmente da aplicação antes que o choque de demanda comece. Para obter os melhores resultados, recomendamos ter um plano de negócios que envolva uma estreita colaboração entre as equipes que usam um calendário compartilhado. A equipe que está planejando um evento deve colaborar estreitamente com a equipe responsável pela inscrição desde o início. Isso fornece à equipe tempo suficiente para ter um plano de programação claro. As pessoas da equipe podem programar a capacidade para aumentar a escala horizontalmente antes do evento e para reduzir a escala horizontalmente após o evento. Para obter mais informações, consulte Escalabilidade programada no Guia do usuário do Application Auto Scaling.

Se você tiver um plano do Enterprise Support, certifique-se de também colaborar com seu gerente técnico de contas (TAM). O TAM pode verificar suas cotas de serviço e garantir que todas as cotas necessárias sejam aumentadas antes do início do evento. Dessa forma, você evita acidentalmente atingir qualquer cota de serviço. Os gerentes técnicos de contas também podem ajudar você através da preparação de serviços, como balanceadores de carga, para garantir que seu evento ocorra sem complicações.

Lidar com choques de demanda não programados é um problema mais complexo. Os choques não programados, se forem suficientemente grandes em amplitude, podem rapidamente fazer com que a demanda ultrapasse a capacidade. Além disso, isso pode superar a capacidade de reação do ajuste de escala automático. A melhor forma de se preparar para choques não programados é provisionar recursos em excesso. Você deve ter recursos suficientes para lidar com a demanda máxima de tráfego prevista a qualquer momento.

Manter a capacidade máxima em antecipação a choques não programados na demanda pode ser dispendioso. Para mitigar o impacto nos custos, encontre uma métrica ou um evento de indicador antecedente que preveja que um grande choque de demanda é iminente. Se a métrica ou o evento fornecer um aviso prévio significativo de forma confiável, inicie o processo de aumentar a escala horizontalmente imediatamente quando o evento ocorrer ou quando a métrica ultrapassar o limite específico definido.

Se a aplicação estiver propensa a choques repentinos e não programados na demanda, considere adicionar um modo de alta performance à aplicação que sacrifique funcionalidades que não são essenciais, mas retenha funcionalidades cruciais para um cliente. Por exemplo, suponha que a aplicação possa deixar de gerar respostas personalizadas dispendiosas e passar a disponibilizar uma página de resposta estática. Nesse cenário, é possível aumentar significativamente o throughput sem a necessidade de escalar a aplicação.

Por último, você pode considerar separar os serviços monolíticos para lidar melhor com os choques de demanda. Se a aplicação for um serviço monolítico que é dispendioso para executar e demorado para escalar, você poderá extrair ou reescrever partes críticas de desempenho e executá-las como serviços separados. Estes novos serviços podem ser escalados independentemente de componentes menos críticos. Ter a flexibilidade de aumentar a escala horizontalmente da funcionalidade crítica para o desempenho separadamente de outras partes da aplicação pode reduzir o tempo necessário para adicionar capacidade e ajudar a economizar custos.

Capacidade de clusters do Amazon ECS

Você pode fornecer capacidade para um cluster do Amazon ECS de várias maneiras. Por exemplo, você pode inicializar instâncias do Amazon EC2 e registrá-las no cluster na inicialização usando o agente de contêineres do Amazon ECS. No entanto, esse método pode ser desafiador porque você precisa gerenciar a escalabilidade por conta própria. Dessa forma, recomendamos que você use provedores de capacidade do Amazon ECS. Os provedores de capacidade gerenciam a escalabilidade de recursos para você. Existem três tipos de provedores de capacidade: Amazon EC2, Fargate e Fargate Spot. Para obter mais informações sobre os provedores de capacidade do Fargate, consulte Amazon ECS clusters for Fargate launch type workloads, e para o tipo de inicialização do EC2, consulte Amazon ECS clusters for EC2 launch type workloads.

Os provedores de capacidade do Fargate e do Fargate Spot cuidam do ciclo de vida das tarefas do Fargate para você. O Fargate fornece capacidade sob demanda, e o Fargate Spot fornece capacidade spot. Quando você inicializa uma tarefa, o Amazon ECS provisiona um recurso do Fargate para você. Esse recurso do Fargate vem com as unidades de memória e CPU que correspondem diretamente aos limites do nível de tarefa que você declarou na definição da tarefa. Cada tarefa recebe seu próprio recurso do Fargate, criando uma relação 1:1 entre a tarefa e os recursos computacionais.

As tarefas executadas no Fargate Spot estão sujeitas a interrupção. As interrupções ocorrem após um aviso de dois minutos. Isso ocorre durante períodos de grande demanda. O Fargate Spot funciona melhor para workloads tolerantes a interrupções, como trabalhos em lote, ambientes de desenvolvimento ou de preparação. Elas também são adequadas para qualquer outro cenário em que a alta disponibilidade e a baixa latência não sejam um requisito.

Você pode executar tarefas do Fargate Spot junto com tarefas sob demanda do Fargate. Ao usá-los juntos, você recebe capacidade de “expansão” de provisionamento a um custo menor.

O Amazon ECS também pode gerenciar a capacidade de instâncias do Amazon EC2 para suas tarefas. Cada provedor de capacidade do Amazon EC2 é associado a um grupo do Amazon EC2

Auto Scaling especificado. Quando você usa o provedor de capacidade do Amazon EC2, o ajuste de escala automático do cluster mantém o tamanho do grupo do Amazon EC2 Auto Scaling para garantir que todas as tarefas agendadas possam ser realizadas.

Escolha de tamanhos de tarefas do Fargate para o Amazon ECS

Para definições de tarefas do AWS Fargate, é necessário especificar a CPU e a memória no nível da tarefa, sem considerar nenhuma sobrecarga. Você também pode especificar a CPU e a memória no nível do contêiner para tarefas do Fargate. Porém, isso não é necessário. Os limites de recursos devem ser maiores ou iguais às reservas declaradas. Na maioria dos casos, você pode defini-los como a soma das reservas de cada contêiner declarado na definição da tarefa. Em seguida, arredonde o número até o tamanho mais próximo do Fargate. Para obter mais informações sobre os tamanhos disponíveis, consulte CPU e memória da tarefa.

Aceleração do provisionamento de capacidade de clusters do Amazon ECS com provedores de capacidade no Amazon EC2

Os clientes que executam o Amazon ECS no Amazon EC2 podem aproveitar o Ajuste de escala automático de clusters (CAS) do Amazon ECS para gerenciar o ajuste de escala de grupos do Amazon EC2 Auto Scaling (ASG). Com o CAS, você pode configurar o Amazon ECS para escalar o ASG automaticamente e se concentrar apenas na execução das tarefas. O Amazon ECS garantirá que o ASG aumente e reduza a escala horizontalmente conforme necessário, sem necessidade de intervenção adicional. Os provedores de capacidade do Amazon ECS são usados para gerenciar a infraestrutura do cluster ao garantir que haja instâncias de contêiner suficientes para atender às demandas da aplicação. Para saber como o CAS do Amazon ECS funciona, consulte Deep Dive on Amazon ECS Cluster Auto Scaling.

Como o CAS depende de uma integração baseada no CloudWatch com o ASG para ajustar a capacidade de clusters, ele tem latência inerente associada à publicação das métricas do CloudWatch, ao tempo que a métrica `CapacityProviderReservation` demora para acionar os alarmes do CloudWatch (nos pontos alto e baixo) e ao tempo que uma instância do Amazon EC2 inicializada recentemente demora para aquecer. Você tomar as seguintes medidas para tornar o CAS mais responsivo para implantações mais rápidas:

Tamanhos do ajuste de escala para as etapas do provedor de capacidade

Os provedores de capacidade do Amazon ECS eventualmente aumentarão ou reduzirão as instâncias de contêiner para atender às demandas da aplicação. Por padrão, o número mínimo de instâncias que o Amazon ECS iniciará está definido como um. Isso poderá adicionar mais tempo às

implantações, se forem necessárias diversas instâncias para a atribuição das tarefas pendentes. É possível aumentar `minimumScalingStepSize` usando a API do Amazon ECS para elevar o número mínimo de instâncias em que o Amazon ECS aumenta ou reduz a escala horizontalmente ao mesmo tempo. Um `maximumScalingStepSize` muito baixo pode limitar quantas instâncias de contêiner terão aumento ou redução da escala horizontalmente por vez, o que pode atrasar as implantações.

> **Note**
>
> No momento, essa configuração está disponível somente usando a API `CreateCapacityProvider` ou `UpdateCapacityProvider`.

Período de aquecimento da instância

O período de aquecimento da instância corresponde ao período após o qual uma instância do Amazon EC2 iniciada recentemente pode contribuir para as métricas do CloudWatch relacionadas ao grupo do Auto Scaling. Depois que o período de aquecimento especificado expirar, a instância será contabilizada nas métricas agregadas do ASG, e o CAS continuará com sua próxima iteração de cálculos para estimar o número de instâncias necessárias.

O valor padrão para `instanceWarmupPeriod` é de 300 segundos, e você pode configurar para um valor mais baixo usando a API `CreateCapacityProvider` ou `UpdateCapacityProvider` para obter um ajuste de escala mais responsivo.

Capacidade não utilizada

Se o provedor de capacidade não tiver instâncias de contêiner disponíveis para a atribuição de tarefas, ele precisará aumentar (aumentar a escala horizontalmente) a capacidade do cluster ao iniciar instâncias do Amazon EC2 rapidamente e aguardar a inicialização antes de iniciar os contêineres nelas. Isso pode reduzir significativamente a taxa de inicialização de tarefas. Você tem duas opções disponíveis para lidar com isso.

Nesse caso, ter a capacidade não utilizada do Amazon EC2 já iniciada e pronta para executar tarefas aumentará a taxa efetiva de inicialização de tarefas. É possível usar a configuração `Target Capacity` para indicar que deseja manter a capacidade não utilizada em seus clusters. Por exemplo, ao definir a `Target Capacity` como 80%, você indica que o cluster precisa de 20% de capacidade não utilizada disponível em todos os momentos. Essa capacidade não utilizada pode permitir que qualquer tarefa independente seja iniciada imediatamente, garantindo que a

inicialização de tarefas não sofra o controle de utilização. A desvantagem desta abordagem é o potencial aumento dos custos de manutenção da capacidade não utilizada do cluster.

Uma abordagem alternativa que pode ser considerada é adicionar reserva de capacidade ao seu serviço e não ao provedor de capacidade. Isso significa que, em vez de reduzir a configuração `Target Capacity` para iniciar a capacidade não utilizada, é possível aumentar o número de réplicas em seu serviço ao modificar a métrica de ajuste de escala de rastreamento de destino ou os limites de ajuste de escala em etapas do ajuste de escala automático do serviço. Observe que essa abordagem será útil somente para workloads com picos, mas não terá efeito ao implantar novos serviços e escalar de zero a determinado número de tarefas pela primeira vez. Para obter mais informações sobre políticas de escalabilidade relacionadas, consulte Target Tracking Scaling Policies ou Step Scaling Policies

Acesso aos recursos do Amazon ECS com as configurações de conta

É possível acessar as configurações de conta Amazon ECS para optar por aceitar ou recusar recursos específicos. Para cada Região da AWS, você pode optar por aceitar ou recusar cada configuração de conta no nível da conta ou para um usuário ou perfil específico.

É possível optar por aceitar ou recusar recursos específicos se alguma das opções a seguir for relevante para você:

- Um usuário ou perfil pode optar por aceitar ou recusar configurações específicas de conta para sua conta individual.
- Um usuário ou perfil pode definir a configuração padrão de aceitação ou recusa para todos os usuários da conta.
- O usuário-raiz ou um usuário com privilégios de administrador pode aceitar ou rejeitar qualquer perfil ou usuário específico na conta. Se a configuração da conta do usuário raiz for alterada, ela definirá o padrão para todos os usuários e perfis para os quais nenhuma configuração individual de conta tenha sido selecionada.

> **Note**
>
> Os usuários federados assumem a configuração da conta do usuário raiz e não podem ter configurações explícitas de conta definidas separadamente para eles.

As seguintes configurações de conta estão disponíveis. Você deve aceitar e rejeitar separadamente cada configuração de conta.

Nome do recurso	Saiba mais
`containerInsights`	Container Insights

`serviceLongArnFormat`
`taskLongArnFormat`
`containerInstanceLongArnFormat`
	Nomes de recursos da Amazon (ARNs) e IDs

`tagResourceAuthorization`	Autorização para atribuição de tags
`fargateFIPSMode`	Conformidade do AWS Fargate com o Padrão Federal de Processamento de Informações (FIPS-140)
`fargateTaskRetirementWaitPeriod`	Tempo de espera para a retirada da tarefa do AWS Fargate
`guardDutyActivate`	Monitoramento de runtime (integração com o Amazon GuardDuty)
`dualStackIPv6`	VPC IPv6 de pilha dupla
`awsvpcTrunking`	Aumento de interfaces de rede de instância de contêiner do Linux
`defaultLogDriverMode`	Modo de driver de logs padrão

Nomes de recursos da Amazon (ARNs) e IDs

Quando recursos do Amazon ECS são criados, são atribuídos a cada recurso um nome do recurso da Amazon (ARN) e um identificador de recurso (ID) exclusivos. Se você usar uma ferramenta de linha de comando ou a API do Amazon ECS para trabalhar com o Amazon ECS, é necessário ter os ARNs ou os IDs dos recursos para determinados comandos. Por exemplo, se você usar o comando

stop-task da AWS CLI para interromper uma tarefa, será necessário especificar o ARN ou o ID da tarefa no comando.

O Amazon ECS está introduzindo um novo formato para nomes de recurso da Amazon (ARNs) e IDs de recurso nos serviços, tarefas e instâncias de contêiner do Amazon ECS. O status de aceitação para cada tipo de recurso determina o formato do nome do recurso da Amazon (ARN) usado pelo recurso. Você deve optar por aceitar o novo formato de ARN para usar recursos, como marcação de recursos, para esse tipo de recurso.

É possível optar por aceitar e recusar o novo formato do nome do recurso da Amazon (ARN) e de ID do recurso conforme a região. Atualmente, qualquer conta nova criada é aceita por padrão.

É possível optar por aceitar ou recusar o novo formato do nome de recurso da Amazon (ARN) e do ID de recurso a qualquer momento. Depois de aceitar, todos os novos recursos que você criar usarão o novo formato.

> **Note**
>
> Um ID de recursos não muda depois que é criado. Portanto, optar por aceitar ou recusar o novo formato não afeta seus IDs de recursos existentes.

As seções a seguir descrevem como os formatos de ARN e ID do recurso estão sendo alterados. Para obter mais informações sobre a transição para os novos formatos, consulte Perguntas frequentes do Amazon Elastic Container Service.

Formato do nome do recurso da Amazon (ARN)

Alguns recursos têm um nome amigável, como um serviço denominado `production`. Em outros casos, você deve especificar um recurso usando o formato de nome de recurso da Amazon (ARN). O novo formato de ARN para tarefas, serviços e instâncias de contêiner do Amazon ECS inclui o nome do cluster. Para obter detalhes sobre a aceitação do novo formato de ARN, consulte Modificar configurações de conta do Amazon ECS.

A tabela a seguir mostra o formato atual e o novo formato para cada tipo de recurso.

Tipo de recurso	ARN
Instância de contêiner	Atual: `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:container-instance/` *`container-instance-id`*

Tipo de recurso	ARN
	Novo: `arn:aws:ecs:` *region*`:`*aws_account_id* `:container-instance/` *cluster-name* `/`*container-instance-id*
Serviço do Amazon ECS	Atual: `arn:aws:ecs:` *region*`:`*aws_account_id* `:service/` *service-name* Novo: `arn:aws:ecs:` *region*`:`*aws_account_id* `:service/` *cluster-name* `/`*service-name*
Tarefa do Amazon ECS	Atual: `arn:aws:ecs:` *region*`:`*aws_account_id* `:task/`*task-id* Novo: `arn:aws:ecs:` *region*`:`*aws_account_id* `:task/`*cluster-name* `/`*task-id*

Tamanho do ID do recurso

Um ID de recurso assume a forma de uma combinação exclusiva de letras e números. Novos formatos de ID de recurso incluem IDs mais curtos para tarefas e instâncias de contêiner do Amazon ECS. O formato atual de ID de recurso tem 36 caracteres. Os novos IDs têm um formato de 32 caracteres que não inclui hifens. Para obter informações sobre a aceitação do novo formato de ID do recurso, consulte Modificar configurações de conta do Amazon ECS.

O padrão é `enabled`.

Somente recursos lançados após a aceitação recebem o novo formato do ARN e do ID do recurso. Nenhum recurso existente é afetado. Para que os serviços e tarefas do Amazon ECS façam a transição para os novos formatos de ARN e ID de recurso, será necessário recriar a tarefa ou o serviço. Para fazer a transição de uma instância de contêiner para o novo formato do ARN e do ID do recurso, a instância de contêiner deve ser drenada e uma nova instância de contêiner deve ser iniciada e registrada no cluster.

> **Note**
>
> As tarefas iniciadas por um serviço do Amazon ECS só poderão receber o novo formato do ARN e do ID do recurso se o serviço tiver sido criado em 16 de novembro de 2018 ou depois e se o usuário que criou o serviço tiver aceitado o novo formato para as tarefas.

Cronograma do formato do ARN e do ID do recurso

O cronograma para os períodos de aceitação e recusa do novo formato do nome do recurso da Amazon (ARN) e do ID do recurso para recursos do Amazon ECS terminou em 1.º de abril de 2021. Por padrão, todas as novas contas aderem ao novo formato. Todos os novos recursos criados receberão o novo formato e você não pode mais recusar.

Container Insights

Em 2 de dezembro de 2024, a AWS lançou o Container Insights com observabilidade aprimorada para o Amazon ECS. Essa versão é compatível com a observabilidade aprimorada dos clusters do Amazon ECS usando os tipos de inicialização do Amazon EC2 e do Fargate. Depois de configurar o Container Insights com observabilidade melhorada no Amazon ECS, o Container Insights recolhe automaticamente a telemetria detalhada da infraestrutura desde o nível do cluster até ao nível do contentor no seu ambiente e apresenta os dados em painéis que mostram uma variedade de métricas e dimensões. Em seguida, é possível utilizar esses painéis prontos para uso no console do Container Insights para compreender melhor a integridade e o desempenho do contêiner e atenuar problemas com mais rapidez, identificando anomalias.

Recomendamos que você use o Container Insights com observabilidade aprimorada em vez do Container Insights, pois ele fornece visibilidade detalhada no ambiente de contêiner, reduzindo o tempo médio de resolução. Para obter mais informações, consulte Métricas do Container Insights do Amazon ECS com observabilidade aprimorada, no Guia do usuário do Amazon CloudWatch.

O padrão para a configuração da conta `containerInsights` é `disabled`.

Container Insights com observabilidade aprimorada

Use o comando a seguir para ativar o Container Insights com observabilidade aprimorada.

Defina a configuração da conta `containerInsights` como `enhanced`.

```
aws ecs put-account-setting --name containerInsights --value enhanced
```

Exemplo de saída

```
{
    "setting": {
        "name": "containerInsights",
```

```
        "value": "enhanced",
        "principalArn": "arn:aws:iam::123456789012:johndoe",
         "type": user
    }
}
```

Depois de definir essa configuração de conta, todos os novos clusters usarão automaticamente o Container Insights com observabilidade aprimorada. Use o comando `update-cluster-settings` para adicionar o Container Insights com observabilidade aprimorada ao cluster existente, ou para atualizar um cluster do Container Insights para o Container Insights com observabilidade aprimorada.

```
aws ecs update-cluster-settings --cluster cluster-name --settings
 name=containerInsights,value=enhanced
```

Também é possível utilizar o console para configurar o Container Insights com observabilidade aprimorada. Para obter mais informações, consulte Modificar configurações de conta do Amazon ECS.

Container Insights

Ao definir a configuração da conta `containerInsights` como `enabled`, todos os novos clusters terão o Container Insights habilitado por padrão. É possível modificar clusters existentes com o uso do `update-cluster-settings`.

Para usar o Container Insights, defina a configuração da conta `containerInsights` como `enabled`. Use o comando a seguir para ativar o Container Insights.

```
aws ecs put-account-setting --name containerInsights --value enabled
```

Exemplo de saída

```
{
    "setting": {
        "name": "containerInsights",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:johndoe",
         "type": user
    }
}
```

Ao definir a configuração da conta `containerInsights` como `enabled`, todos os novos clusters terão o Container Insights habilitado por padrão. Use o comando `update-cluster-settings` para adicionar o Container Insights a um cluster existente.

```
aws ecs update-cluster-settings --cluster cluster-name --settings
 name=containerInsights,value=enabled
```

Também é possível utilizar o console para configurar o Container Insights. Para obter mais informações, consulte Modificar configurações de conta do Amazon ECS.

Conformidade do AWS Fargate com o Padrão Federal de Processamento de Informações (FIPS-140)

O Fargate oferece suporte ao Padrão Federal de Processamento de Informações (FIPS-140), que especifica os requisitos de segurança para módulos criptográficos que protejam informações confidenciais. É o padrão atual dos governos dos Estados Unidos e do Canadá e é aplicável a sistemas que precisam estar em conformidade com a Lei Federal de Gerenciamento de Segurança da Informação (FISMA) ou com o Programa Federal de Gerenciamento de Riscos e Autorizações (FedRAMP).

O nome do recurso é `fargateFIPSMode`.

O padrão é `disabled`.

Você deve ativar a conformidade com o Padrão Federal de Processamento de Informações (FIPS-140) no Fargate. Para obter mais informações, consulte the section called “Conformidade com o FIPS-140 do AWS Fargate”.

> **Important**
>
> A configuração da conta do `fargateFIPSMode` só pode ser alterada por meio da API do Amazon ECS ou da AWS CLI. Para obter mais informações, consulte Modificar configurações de conta do Amazon ECS.

Execute `put-account-setting-default` com a opção `fargateFIPSMode` definida como `enabled`. Para obter mais informações, consulte put-account-setting-default na Referência de API do Amazon Elastic Container Service.

- Você pode usar o comando a seguir para ativar a conformidade com FIPS-140.

```
aws ecs put-account-setting-default --name fargateFIPSMode --value enabled
```

Exemplo de saída

```
{
    "setting": {
        "name": "fargateFIPSMode",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:root",
         "type": user
    }
}
```

É possível executar `list-account-settings` para visualizar o status atual de conformidade com o FIPS-140. Use a opção `effective-settings` para visualizar as configurações do nível da conta.

```
aws ecs list-account-settings --effective-settings
```

Autorização para atribuição de tags

O Amazon ECS está introduzindo a autorização para a atribuição de tags para a criação de recursos. Os usuários devem ter permissões de marcação para ações que criam o recurso, como `ecsCreateCluster`. Quando você cria um recurso e especifica tags para esse recurso, a AWS executa uma autorização adicional para verificar se há permissões para criar tags. Portanto, é preciso conceder permissões explícitas para usar a ação `ecs:TagResource`. Para obter mais informações, consulte the section called “Recursos de tags durante a criação”.

Para optar pela autorização para a atribuição de tags, execute `put-account-setting-default` com a opção `tagResourceAuthorization` definida como `enable`. Para obter mais informações, consulte put-account-setting-default na Referência de API do Amazon Elastic Container Service. É possível executar `list-account-settings` para ver o status atual da autorização para atribuição de tags.

- Você pode usar o comando a seguir para habilitar a autorização de marcação.

```
aws ecs put-account-setting-default --name tagResourceAuthorization --value on --
region region
```

Exemplo de saída

```
{
    "setting": {
        "name": "tagResourceAuthorization",
        "value": "on",
        "principalArn": "arn:aws:iam::123456789012:root",
        "type": user
    }
}
```

Depois de habilitar a autorização de marcação, você deve configurar as permissões apropriadas para permitir que os usuários marquem recursos na criação. Para obter mais informações, consulte the section called “Recursos de tags durante a criação”.

É possível executar `list-account-settings` para ver o status atual da autorização para atribuição de tags. Use a opção `effective-settings` para visualizar as configurações do nível da conta.

```
aws ecs list-account-settings --effective-settings
```

Cronograma de autorização para atribuição de tags

É possível confirmar se a autorização para atribuição de tags está ativa executando `list-account-settings` para visualizar o valor `tagResourceAuthorization`. Quando o valor for on, isso significa que a autorização para atribuição de tags está em uso. Para obter mais informações, consulte list-account-settings na Referência de API do Amazon Elastic Container Service.

Veja a seguir as datas importantes relacionadas à autorização para atribuição de tags.

- 18 de abril de 2023: Introdução da autorização para atribuição de tags. Todas as contas novas e existentes devem optar por usar o recurso. É possível começar a usar a autorização de marcação. Ao optar, conceda as permissões apropriadas.

- De 9 de fevereiro de 2024 a 6 de março de 2024: todas as contas novas e contas existentes não afetadas têm a autorização de marcação ativada por padrão. Você pode habilitar ou desabilitar a configuração da conta `tagResourceAuthorization` para verificar a política do IAM.

  A AWS notificou as contas afetadas.

  Para desabilitar o recurso, execute `put-account-setting-default` com a opção `tagResourceAuthorization` definida como `off`.
- 7 de março de 2024: caso tenha habilitado a autorização de marcação, não poderá mais desabilitar a configuração da conta.

  Recomendamos que você conclua o teste da política do IAM antes dessa data.
- 29 de março de 2024: todas as contas usam autorização de marcação. A configuração no nível da conta não estará mais disponível no console do Amazon ECS ou na AWS CLI.

Tempo de espera para a retirada da tarefa do AWS Fargate

A AWS envia notificações quando você tiver tarefas do Fargate em execução em uma revisão de versão da plataforma marcada para ser retirada. Para obter mais informações, consulte Retirada e manutenção de tarefas para o AWS Fargate no Amazon ECS .

A AWS é responsável por aplicar patches e manter a infraestrutura subjacente do AWS Fargate. Quando a AWS determina que é necessário fazer uma atualização de segurança ou de infraestrutura para uma tarefa do Amazon ECS hospedada no Fargate, é necessário interromper as tarefas e iniciar novas tarefas para substituí-las. É possível configurar o período de espera antes que as tarefas sejam retiradas para aplicação de patches. Você tem a opção de retirar a tarefa imediatamente, esperar 7 dias corridos ou esperar 14 dias corridos.

Essa configuração está no nível da conta.

É possível configurar a hora em que o Fargate inicia a retirada da tarefa. Para workloads que exijam a aplicação imediata das atualizações, escolha a configuração imediata (`0`). Quando precisar de mais controle, por exemplo, quando uma tarefa só puder ser interrompida durante uma determinada janela, configure a opção de 7 dias (`7`) ou 14 dias (`14`).

Recomendamos que você escolha um período de espera mais curto para receber as revisões das versões da plataforma mais recentes mais cedo.

Configure o período de espera executando `put-account-setting-default` ou `put-account-setting` como usuário-raiz ou um usuário administrativo. Use a opção `fargateTaskRetirementWaitPeriod` para o `name` e a opção `value` definida como um dos valores a seguir:

- `0`: A AWS envia a notificação e imediatamente começa a retirar as tarefas afetadas.
- `7`: A AWS envia a notificação e aguarda 7 dias corridos antes de começar a retirar as tarefas afetadas.
- `14`: A AWS envia a notificação e aguarda 14 dias corridos antes de começar a retirar as tarefas afetadas.

O padrão são 7 dias.

Para obter mais informações, consulte put-account-setting-default e put-account-setting na Referência de API do Amazon Elastic Container Service.

Você pode executar o comando a seguir para definir o período de espera para 14 dias.

```
aws ecs put-account-setting-default --name fargateTaskRetirementWaitPeriod --value 14
```

Exemplo de saída

```
{
    "setting": {
        "name": "fargateTaskRetirementWaitPeriod",
        "value": "14",
        "principalArn": "arn:aws:iam::123456789012:root",
        "type: user"
    }
}
```

É possível executar `list-account-settings` para visualizar o tempo de espera atual da retirada da tarefa do Fargate. Use a opção `effective-settings`.

```
aws ecs list-account-settings --effective-settings
```

Aumento de interfaces de rede de instância de contêiner do Linux

Cada tarefa do Amazon ECS que usa o modo de rede `awsvpc` recebe a própria interface de rede elástica (ENI), que está anexada à instância de contêiner que a hospeda. Existe um limite padrão para o número de interfaces de rede que podem ser anexadas a uma instância do Amazon EC2, e a interface de rede primária conta como uma delas. Por exemplo, por padrão, uma instância `c5.large` pode ter até três ENIs associadas a ela. A interface de rede principal para a instância conta como uma, por isso você pode associar mais duas ENIs à instância. Como cada tarefa que usa o modo de rede `awsvpc` exige uma ENI, em geral, você pode executar somente duas dessas tarefas nesse tipo de instância.

O Amazon ECS oferece suporte à execução de instâncias de contêiner com maior densidade de ENI usando tipos de instâncias do Amazon EC2 compatíveis. Quando você usa esses tipos de instância e ativa a configuração da conta `awsvpcTrunking`, ENIs adicionais ficam disponíveis em instâncias de contêiner iniciadas recentemente. Essa configuração permite que você coloque mais tarefas em cada instância de contêiner.

Por exemplo, uma instância `c5.large` com `awsvpcTrunking` tem um limite de ENI aumentado para doze. A instância de contêiner terá a interface de rede prmária e o Amazon ECS cria e anexa uma interface de rede "tronco" à instância de contêiner. Portanto, essa configuração permite que você inicie 10 tarefas na instância de contêiner, em vez das duas tarefas atuais.

Monitoramento de runtime (integração com o Amazon GuardDuty)

O monitoramento de runtime é um serviço inteligente de detecção de ameaças que protege as workloads em execução nas instâncias de contêiner do Fargate e do EC2, monitorando continuamente as atividades de log e rede da AWS para identificar comportamentos maliciosos ou não autorizados.

O parâmetro `guardDutyActivate` é somente leitura no Amazon ECS e indica se o monitoramento de runtime está ativado ou desativado pelo administrador de segurança na sua conta do Amazon ECS. O GuardDuty controla essa configuração de conta em seu nome. Para obter mais informações, consulte Protecting Amazon ECS workloads with Runtime Monitoring.

Você pode executar `list-account-settings` para visualizar a configuração atual de integração do GuardDuty.

```
aws ecs list-account-settings
```

Exemplo de saída

```
{
    "setting": {
        "name": "guardDutyActivate",
        "value": "on",
        "principalArn": "arn:aws:iam::123456789012:doej",
        "type": aws-managed"
    }
}
```

VPC IPv6 de pilha dupla

O Amazon ECS oferece suporte ao fornecimento de tarefas com um endereço IPv6 além do endereço IPv4 privado primário.

Para que as tarefas recebam um endereço IPv6, a tarefa deve usar o modo de rede `awsvpc`, deve ser iniciada em uma VPC configurada para o modo de pilha dupla e a configuração da conta do `dualStackIPv6` deve estar habilitada. Para obter mais informações sobre outros requisitos, consulte Usar uma VPC no modo de pilha dupla para o tipo de execução do EC2 e Usar uma VPC no modo de pilha dupla para o tipo de execução do Fargate.

> **Important**
>
> A configuração da conta do `dualStackIPv6` só pode ser alterada por meio da API do Amazon ECS ou da AWS CLI. Para obter mais informações, consulte Modificar configurações de conta do Amazon ECS.

Se você tinha uma tarefa em execução usando o modo de rede `awsvpc` em uma sub-rede habilitada para IPv6 entre 1.º de outubro de 2020 e 2 de novembro de 2020, a configuração padrão `dualStackIPv6` da conta na região em que a tarefa estava sendo executada é `disabled`. Se essa condição não for atendida, a configuração padrão `dualStackIPv6` na região será `enabled`.

O padrão é `disabled`.

Modo de driver de logs padrão

O Amazon ECS permite definir um modo padrão de entrega de mensagens de log enviadas de um contêiner para o driver de logs escolhido. O modo de entrega afeta a estabilidade da aplicação quando o fluxo de logs do contêiner para o driver de logs é interrompido.

A configuração `defaultLogDriverMode` aceita dois valores: `blocking` e `non-blocking`. Para obter informações sobre esses modos de entrega, consulte LogConfiguration na Referência da API do Amazon Elastic Container Service.

Se você não especificar um modo de entrega na `logConfiguration` da definição do contêiner, o modo especificado usando essa configuração de conta será usado como padrão.

O modo de entrega padrão é `non-blocking`.

> **Note**
>
> Em 25 de junho de 2025, o Amazon ECS alterou o modo padrão do driver de log de `blocking` para `non-blocking` a fim de priorizar a disponibilidade das tarefas em relação ao registro em log. Para continuar usando o modo `blocking` após essa alteração, siga uma destas opções:
>
> - Defina a opção `mode` na definição `logConfiguration` do seu contêiner como `blocking`.
> - Defina a configuração da conta `defaultLogDriverMode` como `blocking`.

Para definir o modo padrão do driver de logs como `blocking`, você pode executar o comando a seguir.

```
aws ecs put-account-setting-default --name defaultLogDriverMode --value "blocking"
```

Visualizar configurações de conta do Amazon ECS usando o console

Visualize as configurações da sua conta no console para ver a quais atributos você tem acesso.

> **Important**
>
> As configurações da conta `dualStackIPv6`, `fargateFIPSMode` e `fargateTaskRetirementWaitPeriod` só podem ser visualizadas ou alteradas por meio da AWS CLI.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação na parte superior, selecione a região da qual você deseja visualizar as configurações da conta.
3. Na página de navegação, selecione Account Settings (Configurações da conta).

Modificar configurações de conta do Amazon ECS

Modifique as configurações da sua conta para acessar os atributos do Amazon ECS.

O parâmetro `guardDutyActivate` é somente leitura no Amazon ECS e indica se o monitoramento de runtime está ativado ou desativado pelo administrador de segurança na sua conta do Amazon ECS. O GuardDuty controla essa configuração de conta em seu nome. Para obter mais informações, consulte Protecting Amazon ECS workloads with Runtime Monitoring.

> **Important**
>
> As configurações da conta `dualStackIPv6`, `fargateFIPSMode` e `fargateTaskRetirementWaitPeriod` só podem ser visualizadas ou alteradas por meio da AWS CLI.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação na parte superior, selecione a região da qual você deseja visualizar as configurações da conta.
3. Na página de navegação, selecione Account Settings (Configurações da conta).
4. Selecione Atualizar.
5. (Opcional) Para aumentar ou diminuir o número de tarefas passíveis de execução no modo de rede awsvpc para cada instância do EC2, em Entroncamento AWSVPC, selecione Entroncamento AWSVPC.

6. (Opcional) Para utilizar ou parar de utilizar o CloudWatch Container Insights por padrão para clusters, em Observabilidade do CloudWatch Container Insights, escolha uma destas opções:
   - Para usar o Container Insights com observabilidade aprimorada, escolha Container Insights com observabilidade aprimorada.
   - Para usar o Container Insights, escolha Container Insights.
   - Para parar de utilizar o Container Insights, escolha Desativado.
7. (Opcional) Para habilitar ou desabilitar a autorização de marcação, em Autorização de marcação de recursos, marque ou desmarque Autorização de marcação de recursos.
8. (Opcional) Para configurar um modo de driver de log padrão para quando um modo de entrega de log não estiver definido em um contêiner`logConfiguration`, em Modo de driver de log padrão, escolha uma das seguintes opções:
   - Para definir o modo padrão do driver de log como `blocking`, escolha Bloquear.
   - Para definir o modo padrão do driver de log como `non-blocking`, escolha Não bloquear.
9. Escolha Salvar alterações.
10. Na tela de confirmação, escolha Confirm (Confirmar) para salvar a seleção.

Próximas etapas

Se você ativou o Container Insights com observabilidade aprimorada ou o Container Insights, existe a opção de atualizar seus clusters existentes para utilizar o recurso. Para obter mais informações, consulte Atualização de um cluster do Amazon ECS.

Reverter para as configurações padrão da conta do Amazon ECS

É possível usar o AWS Management Console para reverter as configurações da conta do Amazon ECS para o padrão.

A opção Revert to account default (Reverter para o padrão da conta) só está disponível quando as configurações da sua conta não são mais as configurações padrão.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação na parte superior, selecione a região da qual você deseja visualizar as configurações da conta.
3. Na página de navegação, selecione Account Settings (Configurações da conta).

4. Selecione Atualizar.
5. Escolha Revert to account default (Reverter para o padrão da conta).
6. Na tela de confirmação, escolha Confirm (Confirmar) para salvar a seleção.

Gerenciar configurações de conta do Amazon ECS usando o AWS CLI

É possível gerenciar suas configurações de conta usando a API do Amazon ECS, a AWS CLI ou os SDKs. As configurações da conta `dualStackIPv6`, `fargateFIPSMode` e `fargateTaskRetirementWaitPeriod` só podem ser visualizadas ou alteradas por meio dessas ferramentas.

> **Note**
>
> É possível usar endpoints de serviço de pilha dupla para interagir com o Amazon ECS via AWS CLI, SDKs e API do Amazon ECS sobre IPv4 e IPv6. Para obter mais informações, consulte Usar endpoints de pilha dupla do Amazon ECS.

Para obter informações sobre as ações de API disponíveis para definições de tarefa, consulte Ações de configuração de conta na Referência de API do Amazon Elastic Container Service.

Use um dos comando a seguir para modificar a configuração de conta padrão para todos os usuários ou funções do na sua conta. Essas alterações se aplicarão a toda a conta da AWS, a menos que o usuário ou o perfil cancele explicitamente essas configurações por conta própria.

- put-account-setting-default (AWS CLI)

```
aws ecs put-account-setting-default --name serviceLongArnFormat --value enabled --
region us-east-2
```

  Você também pode usar esse comando para modificar outras configurações de conta. Para fazer isso, substitua o parâmetro `name` pela configuração de conta correspondente.

- Write-ECSAccountSetting (AWS Tools for Windows PowerShell)

```
Write-ECSAccountSettingDefault -Name serviceLongArnFormat -Value enabled -Region us-
east-1 -Force
```

Para modificar as configurações da sua conta de usuário (AWS CLI)

Use um dos comandos a seguir para modificar configurações de conta para o seu usuário do . Se você estiver usando esses comandos como o usuário raiz, as alterações se aplicarão a toda a conta da AWS, a menos que um usuário ou um perfil cancele explicitamente essas configurações por conta própria.

- put-account-setting (AWS CLI)

```
aws ecs put-account-setting --name serviceLongArnFormat --value enabled --region us-east-1
```

  Você também pode usar esse comando para modificar outras configurações de conta. Para fazer isso, substitua o parâmetro `name` pela configuração de conta correspondente.

- Write-ECSAccountSetting (AWS Tools for Windows PowerShell)

```
Write-ECSAccountSetting -Name serviceLongArnFormat -Value enabled -Force
```

Para modificar as configurações de conta para um usuário ou perfil específico (AWS CLI)

Use um dos comandos a seguir e especifique o ARN de um usuário, perfil ou usuário raiz na solicitação para modificar as configurações de conta para um usuário ou perfil específico.

- put-account-setting (AWS CLI)

```
aws ecs put-account-setting --name serviceLongArnFormat --value enabled --principal-arn arn:aws:iam::aws_account_id:user/principalName --region us-east-1
```

  Você também pode usar esse comando para modificar outras configurações de conta. Para fazer isso, substitua o parâmetro `name` pela configuração de conta correspondente.

- Write-ECSAccountSetting (AWS Tools for Windows PowerShell)

```
Write-ECSAccountSetting -Name serviceLongArnFormat -Value enabled -PrincipalArn
 arn:aws:iam::aws_account_id:user/principalName -Region us-east-1 -Force
```

Perfis do IAM para o Amazon ECS

Um perfil do IAM é uma identidade do IAM que você pode criar em sua conta que tem permissões específicas. No Amazon ECS, você pode criar perfis para conceder permissões aos recursos do Amazon ECS, como contêineres ou serviços.

Os perfis exigidos pelo Amazon ECS dependem do tipo de execução da definição da tarefa e dos recursos usados. Use a tabela a seguir para determinar quais perfis do IAM são necessários no Amazon ECS.

Função	Definição	Quando necessário	Mais informações
Função de execução de tarefas	Esse perfil permite que o Amazon ECS use outros serviços da AWS em seu nome.	Sua tarefa está hospedada no AWS Fargate ou em instâncias externas e: • extrai uma imagem de contêiner de um repositório privado do Amazon ECR. • extrai uma imagem de contêiner de um repositório privado do Amazon ECR em uma conta diferente daquela que executa a tarefa. • envia logs de contêiner ao CloudWatch Logs usando o driver de log `awslogs`. Sua tarefa está hospedada no AWS	Função do IAM de execução de tarefas do Amazon ECS

Função	Definição	Quando necessário	Mais informações
		Fargate ou em instâncias do Amazon EC2 e: • usa autenticação de registro privado. • usa monitoramento de runtime. • a definição de tarefa faz referênci a a dados confidenc iais usando segredos do Secrets Manager ou parâmetros do AWS Systems Manager Parameter Store.	
Função da tarefa	Esse perfil permite que o código da aplicação (no contêiner) use outros serviços da AWS.	Sua aplicação acessa outros serviços da AWS, como o Amazon S3.	Perfil do IAM para tarefas do Amazon ECS
Função de instância de contêiner	Esse perfil permite que as instâncias do EC2 ou instâncias externas se registrem no cluster.	Sua tarefa está hospedada em instâncias do Amazon EC2 ou em uma instância externa.	Função do IAM de instância de contêiner do Amazon ECS
Perfil do Amazon ECS Anywhere	Esse perfil permite que as instâncias externas acessem as APIs da AWS.	Sua tarefa está hospedada em instâncias externas.	Perfil do IAM para o Amazon ECS Anywhere

Função	Definição	Quando necessário	Mais informações
Infraestrutura do Amazon ECS para o perfil de balanceadores de carga	Esse perfil permite que o Amazon ECS gerencie recursos de balanceadores de carga nos clusters em seu nome para implantações azul/verde.	Você pode querer usar implantações azul/verde do Amazon ECS.	Perfil do IAM da infraestrutura do Amazon ECS para balanceadores de carga
Perfil para CodeDeploy do Amazon ECS	Esse perfil permite que o CodeDeploy faça atualizações nos serviços.	Use o tipo de implantação azul/verde do CodeDeDeploy para implantar serviços.	Função do IAM para CodeDeploy do Amazon ECS
Perfil para EventBridge do Amazon ECS	Esse perfil permite que o EventBridge faça atualizações nos serviços.	Use regras e destinos do EventBridge para agendar as tarefas.	Perfil do IAM para EventBridge do Amazon ECS

Função	Definição	Quando necessário	Mais informações
Perfil de infraestrutura do Amazon ECS	Esse perfil permite que o Amazon ECS gerencie recursos de infraestrutura nos clusters.	• Você deseja anexar volumes do Amazon EBS às tarefas do tipo de execução do Fargate ou EC2 do Amazon ECS. O perfil de infraestrutura permite que o Amazon ECS gerencie os volumes do Amazon EBS para as tarefas. • Você deseja usar o Transport Layer Security (TLS) para criptografar o tráfego entre os serviços do Amazon ECS Service Connect. • Você deseja criar grupos de destino do VPC Lattice.	Perfil do IAM de infraestrutura do Amazon ECS

Definições de tarefa do Amazon ECS

Uma definição de tarefa é como um esquema para sua aplicação. É um arquivo de texto em formato JSON que descreve os parâmetros e um ou mais contêineres que formam sua aplicação.

Veja a seguir alguns dos parâmetros que você pode especificar em uma definição de tarefa.

- O tipo de inicialização a ser usado, que determina a infraestrutura na qual as tarefas são hospedadas
- A imagem do Docker a ser usada com cada contêiner em sua tarefa
- A CPU e a memória a serem usadas com cada tarefa ou cada contêiner dentro de uma tarefa
- Os requisitos de memória e CPU
- O sistema operacional do contêiner no qual a tarefa é executada
- O modo de rede do Docker a ser usado para os contêineres na tarefa
- A configuração de registro em log a ser usada para suas tarefas
- Se a tarefa deve continuar sendo executada caso o contêiner seja concluído ou falhe
- O comando que o contêiner executa quando é iniciado
- Eventuais volumes de dados que são usados com os contêineres na tarefa
- O perfil do IAM usado pelas suas tarefas

Para obter uma lista completa de parâmetros de definição de tarefa, consulte Parâmetros de definição de tarefa do Amazon ECS para o tipo de inicialização do Fargate.

Depois de criar uma definição de tarefa, você pode executá-la como uma tarefa ou um serviço.

- Uma tarefa é a instanciação de uma definição de tarefa dentro de um cluster. Depois de criar uma definição de tarefa para a aplicação no Amazon ECS, é possível especificar o número de tarefas que serão executadas no cluster.
- Um serviço do Amazon ECS executa e mantém simultaneamente o número desejado de tarefas em um cluster do Amazon ECS. Ele funciona de forma que, se qualquer uma de suas tarefas falharem ou pararem por algum motivo, o programador de serviço do Amazon ECS iniciará outra instância com base na sua definição de tarefa. Ele faz isso para substituí-la e, assim, manter o número desejado de tarefas no serviço.

Tópicos

- Estados de definição de tarefa do Amazon ECS
- Projetar sua aplicação para o Amazon ECS
- Criar uma definição de tarefa do Amazon ECS usando o console
- Atualizar uma definição de tarefa do Amazon ECS usando o console
- Cancelar registro de uma revisão de definição de tarefa do Amazon ECS usando o console
- Exclusão de uma revisão de definição de tarefa do Amazon ECS usando o console
- Casos de uso de definição de tarefa do Amazon ECS
- Parâmetros de definição de tarefa do Amazon ECS para o tipo de inicialização do Fargate
- Parâmetros de definição de tarefa do Amazon ECS para o tipo de inicialização do EC2
- Modelo de definição de tarefa do Amazon ECS
- Exemplos de definições de tarefa do Amazon ECS

Estados de definição de tarefa do Amazon ECS

A definição de uma tarefa muda de estado ao criá-la, cancelar o registro ou exclui-la. É possível visualizar o estado da definição de tarefa no console ou usando `DescribeTaskDefinition`.

Veja a seguir os estados possíveis para uma definição de tarefa:

ACTIVE

A definição de uma tarefa é `ACTIVE` após ser registrada no Amazon ECS. É possível usar as definições de tarefa no estado `ACTIVE` para executar tarefas ou criar serviços.

INACTIVE

Quando você cancela o registro de uma definição de tarefa, uma definição de tarefa passa do estado `ACTIVE` para o estado `INACTIVE`. É possível recuperar uma definição de tarefa `INACTIVE` chamando `DescribeTaskDefinition`. Não é possível executar novas tarefas ou criar novos serviços com uma definição de tarefa no estado `INACTIVE`. Não há impacto sobre os serviços ou tarefas existentes.

DELETE_IN_PROGRESS

Depois que você envia a definição da tarefa para exclusão, uma definição de tarefa passa do estado `INACTIVE` para o estado `DELETE_IN_PROGRESS`. Depois que a definição da tarefa estiver no estado `DELETE_IN_PROGRESS`, o Amazon ECS verificará periodicamente se a definição da tarefa de destino não está sendo mencionada por nenhuma tarefa ou

implantação ativa e, em seguida, excluirá a definição da tarefa permanentemente. Não é possível executar novas tarefas ou criar novos serviços com uma definição de tarefa no estado `DELETE_IN_PROGRESS`. É possível enviar uma definição de tarefa para exclusão a qualquer momento sem afetar tarefas e serviços existentes.

É possível visualizar no console as definições de tarefa que estão no estado `DELETE_IN_PROGRESS`, e a definição da tarefa pode ser recuperada chamando `DescribeTaskDefinition`.

Quando você exclui todas as revisões de definição da tarefa `INACTIVE`, o nome da definição da tarefa não é exibido no console e não é retornado na API. Se uma revisão de definição de tarefa estiver no estado `DELETE_IN_PROGRESS`, o nome da definição de tarefa é exibido no console e retornado na API. O nome da definição da tarefa é mantido pelo Amazon ECS e a revisão será incrementada na próxima vez que você criar uma definição de tarefa com esse nome.

Se você usar o AWS Config para gerenciar suas definições de tarefa, o AWS Config cobrará por todos os registros de definição de tarefa. Só haverá cobrança pelo cancelamento do registro da definição de tarefa `ACTIVE` mais recente. Não haverá cobrança pela exclusão de uma definição de tarefa. Para obter mais informações sobre definição de preço, consulte Preços do AWS Config.

Recursos do Amazon ECS que podem bloquear uma exclusão

Uma solicitação de exclusão da definição de tarefa não será concluída se houver algum recurso do Amazon ECS que dependa da revisão da definição de tarefa. Os recursos a seguir podem impedir que uma definição de tarefa seja excluída:

- Tarefas autônomas do Amazon ECS: a definição da tarefa é necessária para que a tarefa permaneça íntegra.
- Tarefas de serviço do Amazon ECS: a definição da tarefa é necessária para que a tarefa permaneça íntegra.
- Implantações e conjuntos de tarefas do serviço do Amazon ECS: a definição da tarefa é necessária quando um evento de escalabilidade é iniciado para uma implantação ou conjunto de tarefas do Amazon ECS.

Se sua definição de tarefa permanecer no estado `DELETE_IN_PROGRESS`, será possível usar o console ou a AWS CLI para identificar e interromper os recursos que bloqueiem a exclusão da definição de tarefa.

Exclusão da definição de tarefa após a remoção do recurso bloqueado

As regras a seguir se aplicam depois que você remove os recursos que bloqueiam a exclusão da definição da tarefa:

- Tarefas do Amazon ECS: a exclusão da definição da tarefa pode levar até uma hora para ser concluída após a interrupção da tarefa.
- Implantações e conjuntos de tarefas do serviço do Amazon ECS: a exclusão da definição da tarefa pode levar até 24 horas para ser concluída após a exclusão da implantação ou do conjunto de tarefas.

Projetar sua aplicação para o Amazon ECS

Você arquiteta a aplicação criando uma definição de tarefa para ela. A definição da tarefa contém os parâmetros que definem as informações sobre a aplicação, incluindo:

- O tipo de execução a ser usado, que determina a infraestrutura na qual as tarefas são hospedadas.

  Ao usar o tipo de execução do EC2, você também escolhe o tipo de instância. Em alguns tipos de instância, como GPU, é necessário definir parâmetros adicionais. Para obter mais informações, consulte Casos de uso de definição de tarefa do Amazon ECS.
- A imagem de contêiner, que contém o código da aplicação e todas as dependências que o código da aplicação requer para ser executado.
- O modo de rede a ser usado para os contêineres na sua tarefa.

  O modo de rede determina como a tarefa se comunica pela rede.

  Em tarefas executadas nas instâncias do EC2, há várias opções, mas recomendamos usar o modo de rede `awsvpc`. O modo de rede `awsvpc` simplifica a rede de contêineres, oferecendo mais controle sobre como as aplicações se comunicam entre si e com outros serviços dentro das VPCs.

  Em tarefas executadas no Fargate, você deve usar o modo de rede `awsvpc`.
- A configuração de registro em log a ser usada nas tarefas.
- Quaisquer volumes de dados que são usados com os contêineres na tarefa.

Para obter uma lista completa de parâmetros de definição de tarefa, consulte Parâmetros de definição de tarefa do Amazon ECS para o tipo de inicialização do Fargate.

Use as diretrizes a seguir ao criar suas definições de tarefas:

- Use cada família de definição de tarefas para apenas uma finalidade comercial.

  Se você agrupar vários tipos de contêineres de aplicações na mesma definição de tarefa, não poderá escalar esses contêineres de forma independente. Por exemplo, um site e uma API normalmente exigem padrões de escalabilidade diferentes. À medida que o tráfego aumenta, você talvez vá precisar de um número diferente de contêineres da Web do que o de contêineres de API. Se esses dois contêineres forem implantados na mesma definição de tarefa, todas as tarefas executarão o mesmo número de contêineres da Web e contêineres de API.
- Combine cada versão da aplicação com uma revisão de definição de tarefa dentro de uma família de definição de tarefa.

  Dentro de uma família de definições de tarefas, cada revisão de definição de tarefa representa um snapshot pontual das configurações de uma imagem de contêiner específica. Isso é semelhante à forma como o contêiner é um snapshot de todos os componentes que são necessários para executar uma versão específica do código da sua aplicação.

  Crie um mapeamento individual entre uma versão do código da aplicação, uma tag de imagem de contêiner e uma revisão da definição da tarefa. Um processo de lançamento típico envolve um git commit que é transformado em uma imagem de contêiner com tag atribuída do SHA do git commit. Em seguida, essa tag de imagem de contêiner recebe sua própria revisão de definição de tarefa do Amazon ECS. Por último, o serviço do Amazon ECS é atualizado para implantar a nova revisão da definição da tarefa.
- Use perfis do IAM diferentes para cada família de definição de tarefa.

  Defina cada definição de tarefa com seu próprio perfil do IAM. Implemente essa prática e forneça a cada componente de negócio sua própria família de definições de tarefas. Ao implementar essas duas práticas recomendadas, é possível limitar o acesso que cada serviço tem aos recursos da sua conta da AWS. Por exemplo, é possível conceder acesso ao seu serviço de autenticação para se conectar ao seu banco de dados de senhas. Ao mesmo tempo, você também pode garantir que somente o serviço de pedidos tenha acesso às informações de pagamento do cartão de crédito.

Práticas recomendadas para imagens de contêiner do Amazon ECS

Uma imagem de contêiner é um conjunto de instruções sobre como criar o contêiner. Uma imagem de contêiner contém o código da aplicação e todas as dependências que o código da aplicação requer para ser executado. As dependências da aplicação incluem os pacotes de código-fonte dos quais seu código de aplicação depende, um runtime de linguagem para linguagens interpretadas e pacotes binários dos quais seu código vinculado dinamicamente depende.

Use as diretrizes a seguir ao projetar e criar suas imagens de contêiner:

- Complete suas imagens de contêiner armazenando todas as dependências da aplicação como arquivos estáticos dentro da imagem do contêiner.

  Se você alterar algo na imagem do contêiner, crie uma imagem de contêiner com as alterações.
- Execute um único processo de aplicação em um contêiner.

  A vida útil do contêiner é tão longa quanto a execução do processo da aplicação. O Amazon ECS substitui processos com falha e determina onde executar o processo de substituição. Uma imagem completa torna a implantação geral mais resiliente.
- Faça com que a aplicação processe `SIGTERM`.

  Quando o Amazon ECS interrompe uma tarefa, primeiro envia um sinal SIGTERM à tarefa para notificar que a aplicação precisa terminar e desligar. Em seguida, o Amazon ECS envia uma mensagem de `SIGKILL`. Quando as aplicações ignoram o `SIGTERM`, o serviço do Amazon ECS deve esperar para enviar o sinal `SIGKILL` antes de encerrar o processo.

  Você precisa identificar quanto tempo leva para a aplicação concluir o trabalho e garantir que as aplicações processem o sinal `SIGTERM`. O tratamento de sinais da aplicação precisa impedir que a aplicação realize um novo trabalho, concluindo o trabalho que está em andamento, ou salvar o trabalho inacabado no armazenamento fora da tarefa, caso demore muito para ser concluído.
- Configure aplicações contêineres para gravar logs em `stdout` e `stderr`.

  Desacoplar o tratamento de logs do código da aplicação oferece flexibilidade para ajustar o tratamento de logs no nível da infraestrutura. Um exemplo disso é alterar seu sistema de registro em log. Em vez de modificar seus serviços e criar e implantar uma imagem de contêiner, você pode ajustar as configurações.
- Use tags para criar versões das imagens do contêiner.

As imagens do contêiner são armazenadas em um registro do contêiner. Cada imagem em um registro é identificada por uma tag. Há uma tag chamada `latest`. Essa tag funciona como um ponteiro para a versão mais recente da imagem do contêiner da aplicação, semelhante à `HEAD` de um repositório git. Recomendamos usar a tag `latest` apenas para fins de teste. Como prática recomendada, atribua tags a imagens de contêiner usando uma tag exclusiva para cada compilação. Recomendamos que você atribua tags a suas imagens usando o SHA do git para o git commit que foi usado para criar a imagem.

Não será necessário criar uma imagem de contêiner para cada commit. No entanto, recomendamos que você crie uma nova imagem de contêiner sempre que liberar uma confirmação de código específica para o ambiente de produção. Também recomendamos que você atribua uma tag à imagem usando uma tag que corresponda ao git commit do código que está dentro da imagem. Se você atribuiu uma tag à imagem com o git commit, poderá descobrir mais rapidamente qual versão do código a imagem está executando.

Também recomendamos que você ative as tags de imagem imutáveis no Amazon Elastic Container Registry. Com essa configuração, você não pode alterar a imagem do contêiner para a qual uma tag aponta. Em vez disso, o Amazon ECR impõe que uma nova imagem seja carregada em uma nova tag. Para obter mais informações, consulte Mutabilidade da tag de imagem no Guia do usuário do Amazon ECR.

Ao arquitetar a aplicação para execução no AWS Fargate, você deve optar entre implantar vários contêineres na mesma definição de tarefa e implantar contêineres separadamente em várias definições de tarefa. Se as seguintes condições forem necessárias, recomendamos implantar vários contêineres na mesma definição de tarefa:

- Os contêineres compartilham um ciclo de vida comum (ou seja, eles são iniciados e encerrados juntos).
- Os contêineres precisam ser executados no mesmo host subjacente (ou seja, um contêiner faz referência ao outro na porta localhost).
- Seus contêineres compartilham recursos.
- Os contêineres compartilham volumes de dados.

Se essas condições não forem necessárias, recomendamos implantar contêineres separadamente em várias definições de tarefa. Isso permite escalar, provisionar e desprovisionar os contêineres separadamente.

Práticas recomendadas para tamanhos de tarefas do Amazon ECS

Os tamanhos de contêineres e tarefas são essenciais para o planejamento de ajuste de escala e de capacidade. No Amazon ECS, CPU e memória são as duas métricas de recursos usadas para capacidade. A CPU é medida em unidades de 1/1024 de uma vCPU completa (1.024 unidades são iguais a 1 vCPU inteira). A memória é medida em megabytes. Na definição da tarefa, é possível configurar reservas e limites de recursos.

Ao configurar uma reserva, você está definindo a quantidade mínima de recursos que uma tarefa exige. Sua tarefa recebe, pelo menos, a quantidade de recursos solicitada. A aplicação pode conseguir usar mais CPU ou memória do que a reserva declarada. No entanto, isso está sujeito a quaisquer limites que também foram declarados. Usar mais do que a quantidade da reserva é conhecido como intermitência. No Amazon ECS, as reservas são garantidas. Por exemplo, se você usar instâncias do Amazon EC2 para fornecer capacidade, o Amazon ECS não coloca uma tarefa em uma instância em que a reserva não possa ser atendida.

Um limite é a quantidade máxima de unidades de CPU ou memória que o contêiner ou a tarefa pode usar. Qualquer tentativa de usar mais CPU além desse limite resulta em controle de utilização. Qualquer tentativa de usar mais memória faz com que o contêiner seja interrompido.

Escolher esses valores pode ser um desafio. Isso ocorre porque os valores mais adequados para a aplicação dependem muito dos requisitos de recursos da aplicação. Testar a carga da aplicação é a chave para um planejamento bem-sucedido dos requisitos de recursos e para uma melhor compreensão dos requisitos da aplicação.

Aplicações sem estado

Em aplicações sem estado que escalam horizontalmente, como uma aplicação por trás de um balanceador de carga, recomendamos primeiro determinar a quantidade de memória que a aplicação consome ao atender às solicitações. Para isso, você pode usar ferramentas tradicionais, como `ps` ou `top`, ou soluções de monitoramento, como o CloudWatch Container Insights.

Ao determinar uma reserva de CPU, considere como você deseja escalar a aplicação para atender às suas necessidades de negócios. Você pode usar reservas de CPU menores, como 256 unidades de CPU (ou 1/4 de vCPU), para aumentar a escala horizontalmente de uma forma refinada que minimize os custos. Porém, elas podem não ser escaladas com rapidez suficiente para atender a picos significativos na demanda. Você pode usar reservas maiores de CPU para aumentar e reduzir a escala horizontalmente com mais rapidez e, portanto, atender aos picos de demanda com mais rapidez. No entanto, reservas maiores de CPU são mais caras.

Outros aplicativos

Em aplicações que não são escaláveis horizontalmente, como operadores únicos ou servidores de banco de dados, a capacidade disponível e o custo representam suas considerações mais importantes. Você deve escolher a quantidade de memória e CPU com base na necessidade indicada pelo teste de carga para fornecer um tráfego que atinja seu objetivo no nível de serviço. O Amazon ECS garante que a aplicação seja colocada em um host com capacidade adequada.

Opções de redes de tarefas do Amazon ECS para o tipo de inicialização do EC2

O comportamento de rede das tarefas do Amazon ECS hospedadas em instâncias do Amazon EC2 depende do modo de rede especificado na definição de tarefa. Recomendamos que você use o modo de rede `awsvpc`, a menos que tenha uma necessidade específica de usar um modo de rede diferente.

Veja a seguir os modos de rede disponíveis.

Modo de rede	Contêineres do Linux no EC2	Contêineres do Windows no EC2	Descrição
`awsvpc`	Sim	Sim	A tarefa recebe sua própria interface de rede elástica (ENI) e um endereço IPv4 privado primário. Isso dá à tarefa as mesmas proprieda des de rede que as instâncias do Amazon EC2.
`bridge`	Sim	Não	A tarefa usa a rede virtual integrada do Docker no Linux, que é executada em cada instância do Amazon EC2 que hospeda a tarefa. A rede virtual integrada no Linux usa o driver de rede `bridge` do Docker. Esse é o modo de rede padrão no Linux quando um modo de rede não é especificado na definição de tarefa.
`host`	Sim	Não	A tarefa usa a rede do host que ignora a rede virtual integrada do Docker mapeando as portas do contêiner diretamente para a ENI

Modo de rede	Contêineres do Linux no EC2	Contêineres do Windows no EC2	Descrição
			da instância do Amazon EC2 que hospeda a tarefa. Os mapeamentos dinâmicos de portas não podem ser usados nesse modo de rede. Um contêiner em uma definição de tarefa que use esse modo deve especificar um número de `hostPort` específico. Um número de porta em um host não pode ser usado por várias tarefas. Como resultado, não é possível executar várias tarefas da mesma definição de tarefa em uma única instância do Amazon EC2.
`none`	Sim	Não	A tarefa não tem conectividade com rede externa.
`default`	Não	Sim	A tarefa usa a rede virtual integrada do Docker no Windows, que é executada em cada instância do Amazon EC2 que hospeda a tarefa. A rede virtual integrada no Windows usa o driver de rede `nat` do Docker. Esse é o modo de rede padrão no Windows quando um modo de rede não é especificado na definição de tarefa.

Para obter mais informações sobre as redes do Docker no Linus, consulte Networking overview (Visão geral de redes) na documentação do Docker.

Para obter mais informações sobre as redes Docker no Windows, consulte Rede de contêineres do Windows na documentação sobre containers do Windows da Microsoft.

Alocar uma interface de rede para uma tarefa do Amazon ECS

Os recursos de redes de tarefas fornecidos pelo modo de rede `awsvpc` dão às tarefas do Amazon ECS as mesmas propriedades de redes que as instâncias do Amazon EC2. Usar o modo de rede `awsvpc` simplifica a rede de contêineres. Você tem mais controle sobre como as aplicações se

comunicam entre si e com outros serviços dentro das VPCs. O modo de rede `awsvpc` também fornece maior segurança para os contêineres, permitindo que você use grupos de segurança e ferramentas de monitoramento de rede em um nível mais granular dentro das tarefas. Você também pode usar outros recursos de rede do Amazon EC2, como o VPC Flow Logs, para monitorar o tráfego que entra e sai das tarefas. Além disso, os contêineres que pertencem à mesma tarefa podem se comunicar por meio da interface `localhost`.

A interface de rede elástica (ENI) da tarefa é um recurso totalmente gerenciado do Amazon ECS. O Amazon ECS cria a ENI e a anexa à instância host do Amazon EC2 host com o grupo de segurança especificado. A tarefa envia e recebe tráfego de rede na ENI da mesma maneira que as instâncias do Amazon EC2 fazem com suas interfaces de rede primárias. A cada ENI de tarefa é atribuído um endereço IPv4 privado, por padrão. Se a VPC estiver habilitada para o modo de pilha dupla e você usar uma sub-rede com um bloco CIDR IPv6, a ENI da tarefa também receberá um endereço IPv6. Cada tarefa pode ter apenas uma ENI.

Essas ENIs ficam visíveis no console do Amazon EC2 para sua conta. Sua conta não pode desanexar ou modificar as ENIs. Isso visa a evitar a exclusão acidental de uma ENI associada a uma tarefa em execução. É possível visualizar as informações do anexo da ENI para tarefas no console do Amazon ECS ou com a operação da API DescribeTasks. Quando a tarefa for interrompida ou se o serviço sofrer redução de escala vertical, a ENI da tarefa será desvinculada e excluída.

Quando precisar aumentar a densidade da ENI, use a configuração de conta `awsvpcTrunking`. O Amazon ECS também cria e anexa uma interface de rede “tronco” para a instância de contêiner. A rede tronco é totalmente gerenciada pelo Amazon ECS. A ENI do tronco é excluída quando você encerra ou cancela o registro da instância de contêiner do cluster do Amazon ECS. Para obter mais informações sobre a configuração de conta de `awsvpcTrunking`, consulte Pré-requisitos.

Você especifica `awsvpc` no parâmetro `networkMode` da definição de tarefa. Para obter mais informações, consulte Modo de rede.

Depois, ao executar uma tarefa ou criar um serviço, use o parâmetro `networkConfiguration` que inclui uma ou mais sub-redes para colocar as tarefas e um ou mais grupos de segurança para anexar a uma ENI. Para obter mais informações, consulte Configuração de rede. As tarefas são posicionadas em instâncias do Amazon EC2 nas mesmas zonas de disponibilidade que as dessas sub-redes e os grupos de segurança especificados são associados à ENI provisionada para a tarefa.

Considerações sobre o Linux

Considere o seguinte ao utilizar o sistema operacional Linux.

- Se você usar uma instância p5.48xlarge no modo `awsvpc`, não será possível executar mais de uma tarefa na instância.
- As tarefas e os serviços que usam o modo de rede `awsvpc` exigem que a função vinculada ao serviço do Amazon ECS forneça ao Amazon ECS as permissões para fazer chamadas a outros serviços da AWS em seu nome. Essa função será automaticamente criada quando você criar um cluster ou se criar ou atualizar um serviço no AWS Management Console. Para obter mais informações, consulte Uso de perfis vinculados ao serviço para o Amazon ECS. Você também pode criar a função vinculada ao serviço com o seguinte comando da AWS CLI:

```
aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
```

- A instância do Linux do Amazon EC2 requer a versão `1.15.0` ou posterior do agente de contêiner para executar tarefas que usam o modo de rede `awsvpc`. Se você estiver usando a AMI do Linux otimizada para o Amazon ECS, sua instância também precisará, pelo menos, da versão `1.15.0-4` do pacote `ecs-init`.
- O Amazon ECS preenche o nome de host da tarefa usando a rede de tarefas com um nome de host DNS (interno) fornecido pela Amazon quando as opções `enableDnsHostnames` e `enableDnsSupport` estão habilitadas na sua VPC. Se essas opções não estiverem habilitadas, o nome de host DNS da tarefa será definido como um nome de host aleatório. Para obter mais informações sobre as configurações de DNS para uma VPC, consulte Usar DNS com sua VPC no Guia do usuário da Amazon VPC.
- Cada tarefa do Amazon ECS que usa o modo de rede `awsvpc` recebe sua própria interface de rede elástica (ENI), que está anexada à instância do Amazon EC2 que a hospeda. Existe uma cota padrão para o número de interfaces de rede que é possível anexar a uma instância Linux do Amazon EC2. A interface de rede primária conta para essa cota. Por exemplo, por padrão, uma instância `c5.large` só pode ter até três ENIs associadas a ela. A interface de rede primária da instância conta como uma. É possível associar mais duas ENIs à instância. Como cada tarefa que usa o modo de rede `awsvpc` exige uma ENI, em geral, você pode executar somente duas dessas tarefas nesse tipo de instância. Para obter mais informações sobre os limites de ENI padrão para cada tipo de instância, consulte Endereços IP por interface de rede por tipo de instância no Guia do usuário do Amazon EC2.
- O Amazon ECS oferece suporte ao lançamento de instâncias do Linux do Amazon EC2 usando tipos de instâncias compatíveis, com maior densidade de ENI. Quando você aceitar a configuração de conta `awsvpcTrunking` e registrar instâncias do Linux do Amazon EC2 usando esses tipos de instância no seu cluster, estas instâncias terão cotas de ENI mais altas. Utilizar essas instâncias com a cota mais alta significa que é possível colocar mais tarefas em cada instância

Linux do Amazon EC2. Para usar a maior densidade de ENI com o recurso de entroncamento, sua instância do Amazon EC2 deve usar a versão `1.28.1` ou posterior do agente de contêiner. Se você estiver usando a AMI otimizada para o Amazon ECS, sua instância precisará, pelo menos, da versão `1.28.1-2` do pacote `ecs-init`. Para obter mais informações sobre como optar pela configuração de conta `awsvpcTrunking`, consulte Acesso aos recursos do Amazon ECS com as configurações de conta. Para obter mais informações sobre o entroncamento de ENI, consulte Aumento das interfaces de rede de instâncias de contêiner do Linux no Amazon ECS.

- Ao hospedar tarefas que usam o modo de rede `awsvpc` em instâncias do Linux do Amazon EC2, suas ENIs de tarefa não recebem endereços IP públicos. Para acessar a Internet, as tarefas devem ser executadas em uma sub-rede privada configurada para usar um gateway NAT. Para obter mais informações, consulte Gateways NAT no Guia do usuário da Amazon VPC. O acesso à rede de entrada deve ocorrer de dentro da VPC que usa o endereço IP privado ou roteado por um balanceador de carga de dentro da VPC. As tarefas inicializadas em sub-redes públicas não têm acesso à Internet.
- O Amazon ECS reconhece somente as ENIs que ele anexa às suas instâncias Linux do Amazon EC2. Se você tiver anexado ENIs manualmente às suas instâncias, o Amazon ECS poderá tentar adicionar uma tarefa a uma instância que não tenha adaptadores de rede suficientes. Isso pode fazer com que a tarefa atinja o tempo limite, passe para um status em desprovisionamento e, em seguida, para um status de interrompida. Recomendamos que você não anexe ENIs às suas instâncias manualmente.
- As instâncias do Linux do Amazon EC2 devem ser registradas com o recurso `ecs.capability.task-eni` para serem consideradas para posicionamento de tarefas com o modo de rede `awsvpc`. As instâncias que executam a versão `1.15.0-4` ou posterior do `ecs-init` são automaticamente registradas com esse atributo.
- As ENIs criadas e anexadas às suas instâncias do Linux do Amazon EC2 não podem ser desvinculadas manualmente nem modificadas pela sua conta. Isso visa a evitar a exclusão acidental de uma ENI associada a uma tarefa em execução. Para liberar as ENIs para uma tarefa, interrompa-a.
- Há um limite de 16 sub-redes e 5 grupos de segurança que podem ser especificados na `awsVpcConfiguration` ao executar uma tarefa ou ao criar um serviço que usa o modo de rede `awsvpc`. Para obter mais informações, consulte AwsVpcConfiguration na Referência da API de serviço do contêiner do Amazon Elastic Container Service.
- Quando uma tarefa é iniciada com o modo de rede `awsvpc`, o agente de contêiner do Amazon ECS cria um contêiner `pause` adicional para cada tarefa antes de iniciar os contêineres na definição de tarefa. Em seguida, ele configura o namespace de rede do contêiner `pause` ao

executar os plug-ins do CNI amazon-ecs-cni-plugins. O agente inicia então o restante dos contêineres na tarefa para que eles compartilhem a pilha de rede do contêiner `pause`. Isso significa que todos os contêineres em uma tarefa são endereçáveis por endereços IP da ENI, e que eles podem se comunicar entre eles por meio da interface `localhost`.

- Serviços com tarefas que usam o modo de rede `awsvpc` oferecem suporte somente a Application Load Balancers e Network Load Balancers. Ao criar grupos de destino para esses serviços, você precisa escolher `ip` como o tipo de destino. Não use `instance`. Isso ocorre porque as tarefas que usam o modo de rede `awsvpc` são associadas a uma ENI, não a uma instância do Linux do Amazon EC2. Para obter mais informações, consulte Uso do balanceamento de carga para distribuir o tráfego de serviço do Amazon ECS.
- Se a sua VPC for atualizada para alterar o conjunto de opções de DHCP que ela utiliza, não será possível aplicar essas alterações às tarefas existentes. Inicie novas tarefas com essas alterações aplicadas, verifique se elas estão funcionando corretamente e, em seguida, interrompa tarefas existentes para modificar essas configurações de rede com segurança.

Considerações sobre Windows

Veja a seguir considerações quando for usado o sistema operacional Windos.

- Instâncias de contêiner que usam a AMI do Windows Server 2016 otimizada para o Amazon ECS não podem hospedar tarefas que usam o modo de rede `awsvpc`. Se você tiver um cluster que contém AMIs do Windows Server 2016 otimizadas para o Amazon ECS e AMIs do Windows compatíveis com o modo de rede `awsvpc`, tarefas que usam o modo de rede `awsvpc` não serão iniciadas nas instâncias do Windows Server 2016. Em vez disso, elas são iniciadas em instâncias com suporte para o modo de rede `awsvpc`.
- Sua instância do Windows do Amazon EC2 requer a versão `1.57.1` ou posterior do agente de contêiner para usar métricas do CloudWatch para contêineres do Windows que usem o modo de rede `awsvpc`.
- As tarefas e os serviços que usam o modo de rede `awsvpc` exigem que a função vinculada ao serviço do Amazon ECS forneça ao Amazon ECS as permissões para fazer chamadas a outros serviços da AWS em seu nome. Essa função será criada automaticamente quando você criar um cluster ou se criar ou atualizar um serviço no AWS Management Console. Para obter mais informações, consulte Uso de perfis vinculados ao serviço para o Amazon ECS. Também é possível criar a função vinculada ao serviço com o comando da AWS CLI a seguir.

```
aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
```

- Sua instância do Windows do Amazon EC2 requer a versão `1.54.0` ou posterior do agente de contêiner para executar tarefas que usam o modo de rede `awsvpc`. Ao inicializar a instância, você deve configurar as opções necessárias para o modo de rede `awsvpc`. Para obter mais informações, consulte the section called “Inicialização de instâncias de contêiner”.
- O Amazon ECS preenche o nome de host da tarefa usando a rede de tarefas com um nome de host DNS (interno) fornecido pela Amazon quando as opções `enableDnsHostnames` e `enableDnsSupport` estão habilitadas na sua VPC. Se essas opções não estiverem habilitadas, o nome de host DNS da tarefa será um nome de host aleatório. Para obter mais informações sobre as configurações de DNS para uma VPC, consulte Usar DNS com sua VPC no Guia do usuário da Amazon VPC.
- Cada tarefa do Amazon ECS que usa o modo de rede `awsvpc` recebe sua própria interface de rede elástica (ENI), que está anexada à instância do Windows do Amazon EC2 que a hospeda. Existe uma cota padrão para o número de interfaces de rede que é possível anexar a uma instância Windows do Amazon EC2. A interface de rede primária conta para essa cota. Por exemplo, por padrão, uma instância `c5.large` só pode ter até três ENIs associadas a ela. A interface de rede primária da instância conta como uma delas. É possível associar mais duas ENIs à instância. Como cada tarefa que usa o modo de rede `awsvpc` exige uma ENI, em geral, você pode executar somente duas dessas tarefas nesse tipo de instância. Para obter mais informações sobre os limites de ENI padrão para cada tipo de instância, consulte Endereços IP por interface de rede por tipo de instância no Guia do usuário do Amazon EC2.
- Ao hospedar tarefas que usam o modo de rede `awsvpc` em instâncias do Windows do Amazon EC2, suas ENIs de tarefa não recebem endereços IP públicos. Para acessar a Internet, inicie tarefas em uma sub-rede privada configurada para usar um gateway NAT. Para obter mais informações, consulte Gateways NAT no Guia do usuário da Amazon VPC. O acesso à rede de entrada deve ocorrer de dentro da VPC usando o endereço IP privado ou roteado por um load balancer de dentro da VPC. As tarefas inicializadas em sub-redes públicas não têm acesso à Internet.
- O Amazon ECS reconhece somente as ENIs que ele anexa às suas instâncias Windows do Amazon EC2. Se você tiver anexado ENIs manualmente às suas instâncias, o Amazon ECS poderá tentar adicionar uma tarefa a uma instância que não tenha adaptadores de rede suficientes. Isso pode fazer com que a tarefa atinja o tempo limite, passe para um status em desprovisionamento e, em seguida, para um status de interrompida. Recomendamos que você não anexe ENIs às suas instâncias manualmente.

- As instâncias do Windows do Amazon EC2 devem ser registradas com o recurso `ecs.capability.task-eni` para serem consideradas para posicionamento de tarefas com o modo de rede `awsvpc`.
- É possível modificar ou desanexar manualmente as ENIs criadas e anexadas às suas instâncias do Windows do Amazon EC2. Isso evita que você exclua acidentalmente uma ENI associada a uma tarefa em execução. Para liberar as ENIs para uma tarefa, interrompa-a.
- É possível especificar até 16 sub-redes e 5 grupos de segurança em `awsVpcConfiguration` ao executar uma tarefa ou criar um serviço que usa o modo de rede `awsvpc`. Para obter mais informações, consulte AwsVpcConfiguration na Referência da API de serviço do contêiner do Amazon Elastic Container Service.
- Quando uma tarefa é iniciada com o modo de rede `awsvpc`, o agente de contêiner do Amazon ECS cria um contêiner `pause` adicional para cada tarefa antes de iniciar os contêineres na definição de tarefa. Em seguida, ele configura o namespace de rede do contêiner `pause` ao executar os plug-ins do CNI amazon-ecs-cni-plugins. O agente inicia então o restante dos contêineres na tarefa para que eles compartilhem a pilha de rede do contêiner `pause`. Isso significa que todos os contêineres em uma tarefa são endereçáveis por endereços IP da ENI, e que eles podem se comunicar entre eles por meio da interface `localhost`.
- Serviços com tarefas que usam o modo de rede `awsvpc` oferecem suporte somente a Application Load Balancers e Network Load Balancers. Ao criar grupos de destino para esses serviços, você precisa escolher `ip` como o tipo de destino, e não `instance`. Isso ocorre porque as tarefas que usam o modo de rede `awsvpc` são associadas a uma ENI, não a uma instância do Windows do Amazon EC2. Para obter mais informações, consulte Uso do balanceamento de carga para distribuir o tráfego de serviço do Amazon ECS.
- Se a sua VPC for atualizada para alterar o conjunto de opções de DHCP que ela utiliza, não será possível aplicar essas alterações às tarefas existentes. Inicie novas tarefas com essas alterações aplicadas, verifique se elas estão funcionando corretamente e, em seguida, interrompa tarefas existentes para modificar essas configurações de rede com segurança.
- Não há suporte para os seguintes recursos quando você usa o modo de rede `awsvpc` em uma configuração de EC2 do Windows:
  - Configuração de pilha dupla
  - IPv6
  - Entroncamento ENI

Usar uma VPC no modo de pilha dupla

Ao usar uma VPC no modo de pilha dupla, as tarefas podem se comunicar por IPv4, IPv6 ou ambos. Os endereços IPv4 e IPv6 são independentes um do outro. Portanto, é preciso configurar o encaminhamento e a segurança na VPC separadamente para IPv4 e IPv6. Para obter mais informações sobre como configurar a VPC para o modo de pilha dupla, consulte Migrar para IPv6 no Guia do usuário da Amazon VPC.

Se tiver configurado sua VPC com um gateway da Internet ou um gateway da Internet somente saída, será possível utilizar sua VPC no modo de pilha dupla. Fazendo isso, as tarefas que obtêm um endereço IPv6 podem acessar a Internet por meio de um gateway da Internet ou um gateway da Internet somente saída. Gateways NAT são opcionais. Para obter mais informações, consulte Gateways da Internet e Gateways da Internet somente de saída no Guia do usuário da Amazon VPC.

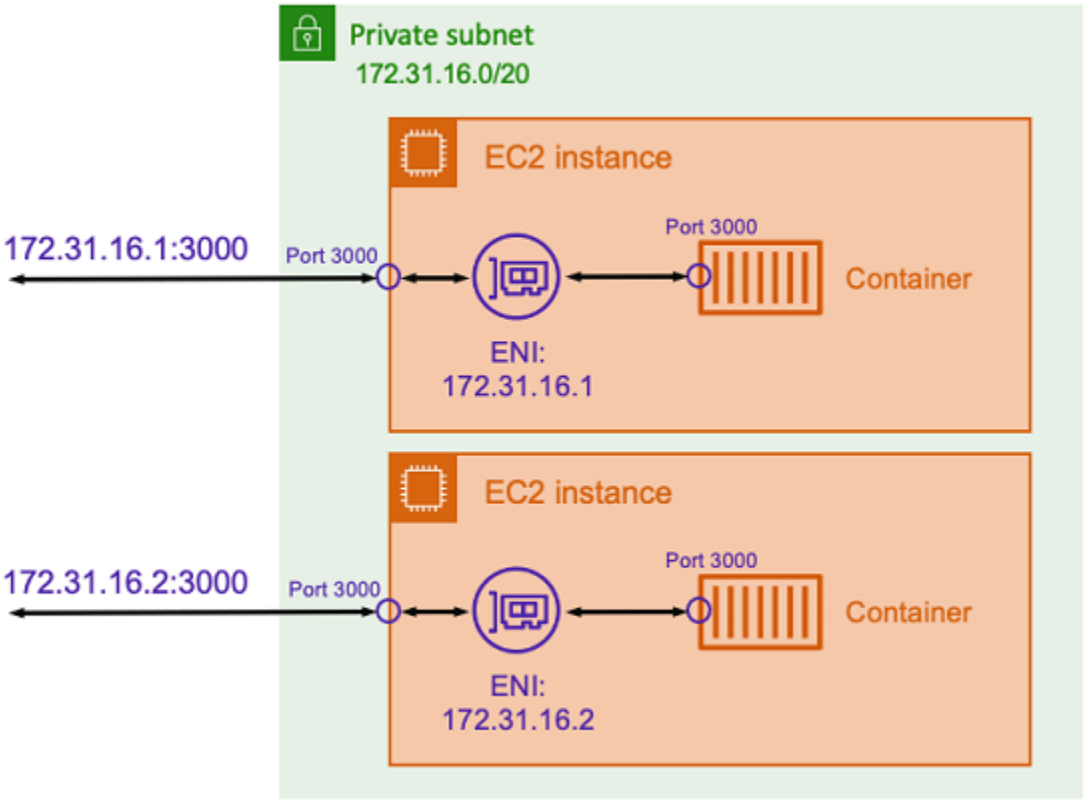
As tarefas do Amazon ECS receberão um endereço IPv6 se as seguintes condições forem atendidas:

- A instância Linux do Amazon EC2 que hospeda a tarefa está usando a versão `1.45.0` ou posterior do agente de contêiner. Para obter informações sobre como verificar a versão do agente que a instância está usando e atualizá-la, se necessário, consulte Atualizar o agente de contêiner do Amazon ECS.
- A configuração `dualStackIPv6` da conta está habilitada. Para obter mais informações, consulte Acesso aos recursos do Amazon ECS com as configurações de conta.
- Sua tarefa está usando o modo de rede `awsvpc`.
- Sua VPC e sua sub-rede estão configuradas para IPv6. A configuração inclui as interfaces de rede criadas na sub-rede especificada. Para obter mais informações sobre como configurar a VPC para o modo de pilha dupla, consulte Migrar para IPv6 e Modificar o atributo de endereçamento IPv6 para a sub-rede no Guia do usuário da Amazon VPC.

Mapeie as portas de contêiner do Amazon ECS para a interface de rede de instância do EC2

Só há suporte para o modo de rede `host` com tarefas do Amazon ECS hospedadas em instâncias do Amazon EC2. Não há suporte com o uso do Amazon ECS no Fargate.

O modo de rede `host` é o modo de rede mais básico com suporte no Amazon ECS. Com o uso do modo host, a rede do contêiner é vinculada diretamente ao host subjacente que está executando o contêiner.



Suponha que você esteja executando um contêiner Node.js com uma aplicação Express que escuta em uma porta `3000` semelhante à ilustrada no diagrama anterior. Quando o modo de rede `host` é usado, o contêiner recebe tráfego na porta 3000 usando o endereço IP da instância de host subjacente do Amazon EC2. Não é recomendável usar esse modo.

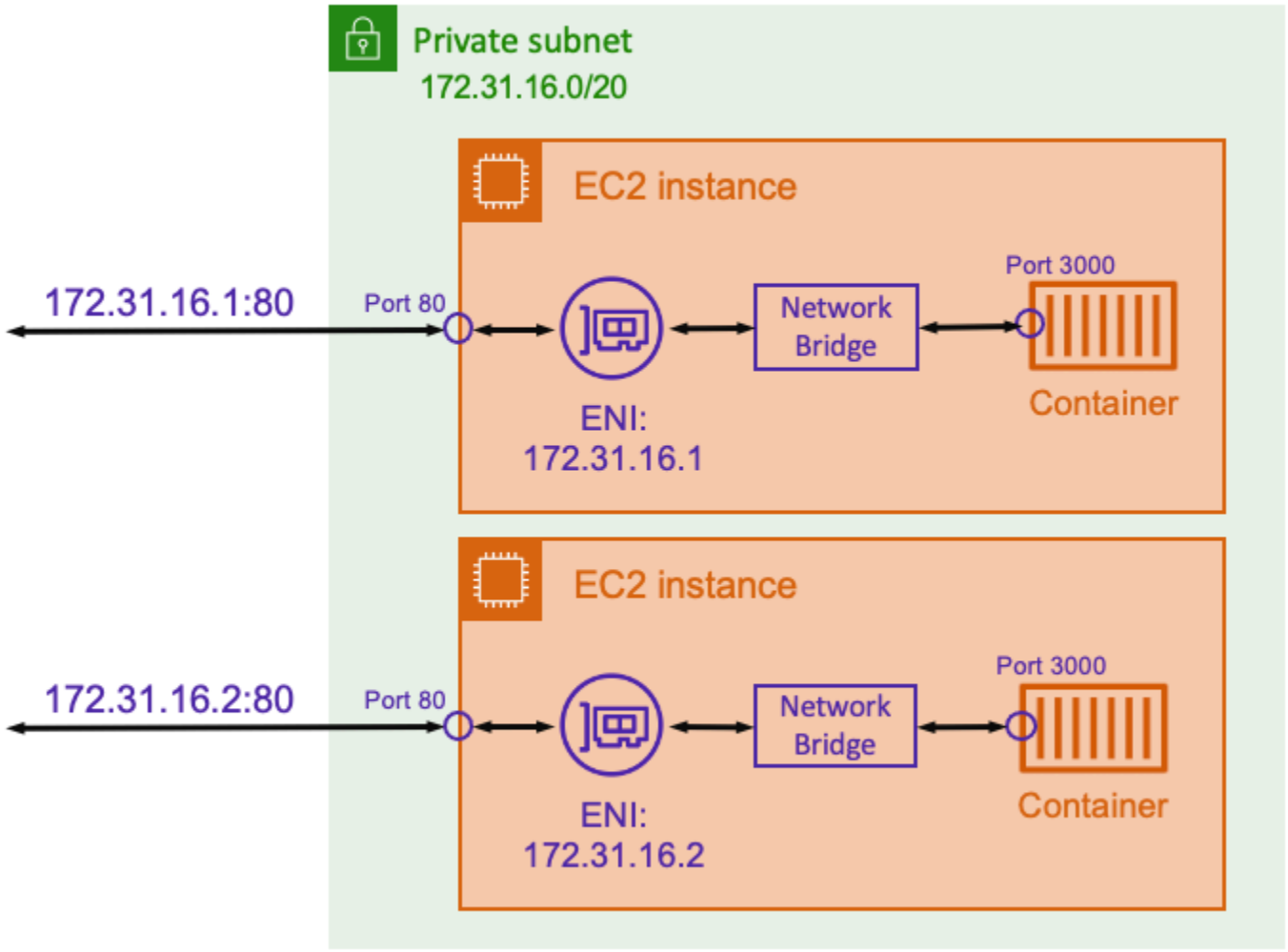
Existem desvantagens significativas em usar esse modo de rede. Você não pode executar mais do que uma única instanciação de uma tarefa em cada host. Isso ocorre porque somente a primeira tarefa pode ser vinculada à porta necessária na instância do Amazon EC2. Também não há como remapear uma porta de contêiner quando ela estiver usando o modo de rede `host`. Por exemplo, se uma aplicação precisar escutar um determinado número de porta, você não poderá remapear o número da porta diretamente. Em vez disso, você deverá gerenciar quaisquer conflitos de portas alterando a configuração da aplicação.

Também há implicações de segurança com o uso do modo de rede `host`. Esse modo permite que os contêineres representem o host e permite que os contêineres se conectem a serviços de rede de loopback privados no host.

Use a rede virtual do Docker para tarefas do Linux no Amazon ECS

Só há suporte para o modo de rede `bridge` com tarefas do Amazon ECS hospedadas em instâncias do Amazon EC2.

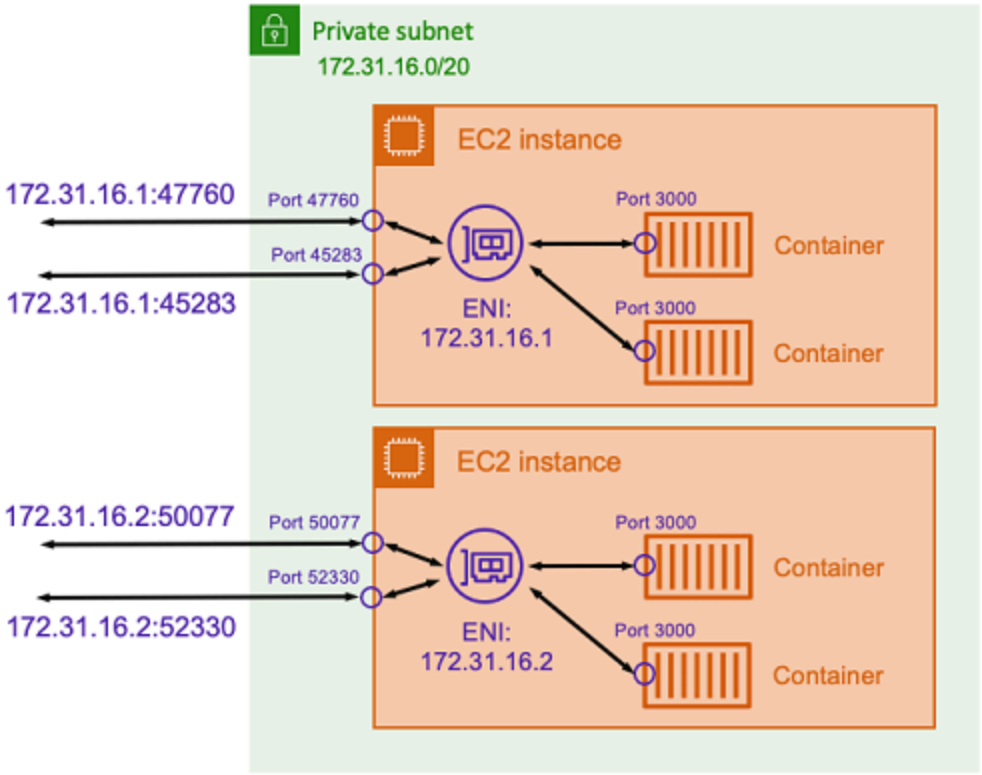
Com o modo `bridge`, você está usando uma ponte de rede virtual para criar uma camada entre o host e a rede do contêiner. Dessa forma, é possível criar mapeamentos de portas que remapeiem uma porta de host para uma porta de contêiner. Os mapeamentos podem ser estáticos ou dinâmicos.



Com um mapeamento estático de portas, é possível definir explicitamente qual porta de host deseja mapear para uma porta de contêiner. Usando o exemplo acima, a porta `80` do host está sendo mapeada para a porta `3000` do contêiner. Para se comunicar com a aplicação em contêiner, você envia tráfego para a porta `80` no endereço IP da instância do Amazon EC2. Do ponto de vista da aplicação em contêineres, ela vê esse tráfego de entrada na porta `3000`.

Se você quiser alterar apenas a porta de tráfego, os mapeamentos de portas estáticos são adequados. No entanto, isso ainda tem a mesma desvantagem de usar o modo de rede `host`. Você não pode executar mais do que uma única instanciação de uma tarefa em cada host. Isso ocorre porque um mapeamento estático de portas permite que apenas um único contêiner seja mapeado para a porta 80.

Para resolver esse problema, considere usar o modo de rede `bridge` com um mapeamento dinâmico de portas, conforme mostrado no diagrama a seguir.



Ao não especificar uma porta de host no mapeamento de portas, é possível fazer com que o Docker escolha uma porta aleatória e não usada do intervalo de portas temporárias e a atribua como porta de host pública para o contêiner. Por exemplo, a aplicação Node.js receptora na porta `3000` do contêiner pode receber uma porta aleatória de número alto, como `47760`, no host do Amazon EC2. Isso significa que é possível executar várias cópias desse contêiner no host. Além disso, cada contêiner pode ter sua própria porta atribuída no host. Cada cópia do contêiner recebe tráfego na porta `3000`. Contudo, os clientes que enviam tráfego para esses contêineres usam as portas de host atribuídas aleatoriamente.

O Amazon ECS ajuda você a acompanhar as portas atribuídas aleatoriamente para cada tarefa. Isso é feito atualizando automaticamente os grupos-alvo do balanceador de carga e a descoberta de

serviços do AWS Cloud Map para ter a lista de portas e endereços IP de tarefas. Isso facilita o uso de serviços em operação usando o modo `bridge` com portas dinâmicas.

No entanto, uma desvantagem de se usar o modo de rede `bridge` é que é difícil bloquear as comunicações entre serviços. Como os serviços podem ser atribuídos a qualquer porta aleatória e não usada, é necessário abrir amplos intervalos de portas entre os hosts. Contudo, não é fácil criar regras específicas para que um determinado serviço só possa se comunicar com outro serviço específico. Os serviços não têm portas específicas para usar nas regras de rede de grupos de segurança.

Opções de redes de tarefas do Amazon ECS para o tipo de inicialização do Fargate

Por padrão, todas as tarefas do Amazon ECS no Fargate recebem uma interface de rede elástica (ENI) com um endereço IP privado primário. Ao usar uma sub-rede pública, você pode opcionalmente atribuir um endereço IP público à ENI da tarefa. Se a VPC estiver configurada para o modo de pilha dupla e você usar uma sub-rede com um bloco CIDR IPv6, a ENI da tarefa também receberá um endereço IPv6. Uma tarefa só pode ter uma ENI associada a ela de cada vez. Os contêineres que pertencem à mesma tarefa também podem se comunicar por meio da interface `localhost`. Para obter mais informações sobre VPCs e sub-redes, consulte Como a Amazon VPC funciona no Guia do usuário da Amazon VPC.

Para que uma tarefa no Fargate possa extrair uma imagem de contêiner, a tarefa deve ter uma rota para a Internet. Veja a seguir como verificar se a sua tarefa tem uma rota para a Internet.

- Quando você usa uma sub-rede pública, pode atribuir um endereço IP público à ENI da tarefa.
- Ao usar uma sub-rede privada, a sub-rede pode ter um gateway NAT anexado.
- Ao usar imagens de contêiner hospedadas no Amazon ECR, você pode configurar o Amazon ECR para usar um endpoint da VPC de interface e a extração da imagem ocorre no endereço IPv4 privado da tarefa. Para obter mais informações, consulte Endpoints da VPC de interface do Amazon ECR (AWS PrivateLink) no Guia do usuário do Amazon Elastic Container Registry.

Como cada tarefa tem sua própria ENI, você pode usar recursos de rede, como os logs de fluxo de VPC, que podem ser usados para monitorar o tráfego de e para suas tarefas. Para obter mais informações, consulte Logs de fluxo da VPC no Guia do usuário do Amazon Virtual Private Cloud.

Você também pode utilizar AWS PrivateLink. É possível configurar um endpoint de interface VPC para que você possa acessar APIs do Amazon ECS por meio de endereços IP privados. AWS

PrivateLink restringe todo o tráfego de rede entre sua VPC e o Amazon ECS para a rede da Amazon. Você não precisa de um gateway da Internet, de um dispositivo NAT ou de um gateway privado virtual. Para obter mais informações, consulte VPC endpoint de interface do Amazon ECS (AWS PrivateLink).

Consulte exemplos de como usar o recurso `NetworkConfiguration` com o AWS CloudFormation, consulte the section called “Exemplos de modelos do AWS CloudFormation”.

As ENIs que são criadas são totalmente gerenciadas pelo AWS Fargate. Além disso, existe uma política do IAM associada que é utilizada para conceder permissões para o Fargate. Para tarefas que usam a versão `1.4.0` ou posterior da plataforma do Fargate, a tarefa recebe uma única ENI (conhecida como ENI de tarefa), e todo o tráfego de rede flui por essa ENI dentro da VPC. Esse tráfego é registrado nos seus logs de fluxo da VPC. Para tarefas que usam a versão `1.3.0` e anterior da plataforma, além da ENI de tarefa, a tarefa também recebe uma ENI de propriedade do Fargate separada, que é usada para algum tráfego de rede que não é visível nos logs de fluxo da VPC. A tabela a seguir descreve o comportamento do tráfego de rede e a política do IAM necessária para cada versão da plataforma.

Ação	Fluxo de tráfego com versão **`1.3.0`** e anterior da plataforma Linux	Fluxo de tráfego com versão **`1.4.0`** da plataforma Linux	Fluxo de tráfego com versão **`1.0.0`** da plataforma Windows	Permissão do IAM
Recuperar credenciais de login do Amazon ECR	ENI de propriedade do Fargate	ENI de tarefa	ENI de tarefa	Função do IAM de execução de tarefas
Extração de imagem	ENI de tarefa	ENI de tarefa	ENI de tarefa	Função do IAM de execução de tarefas
Como enviar logs por meio de um driver de log	ENI de tarefa	ENI de tarefa	ENI de tarefa	Função do IAM de execução de tarefas

Ação	Fluxo de tráfego com versão **1.3.0** e anterior da plataforma Linux	Fluxo de tráfego com versão **1.4.0** da plataforma Linux	Fluxo de tráfego com versão **1.0.0** da plataforma Windows	Permissão do IAM
Enviar logs pelo FireLens para Amazon ECS	ENI de tarefa	ENI de tarefa	ENI de tarefa	Função do IAM de tarefa
Recuperar segredos do Secrets Manager ou Systems Manager	ENI de proprieda de do Fargate	ENI de tarefa	ENI de tarefa	Função do IAM de execução de tarefas
Tráfego do sistema de arquivos do Amazon EFS	Indisponível	ENI de tarefa	ENI de tarefa	Função do IAM de tarefa
Tráfego de aplicativos	ENI de tarefa	ENI de tarefa	ENI de tarefa	Função do IAM de tarefa

Considerações

Considere o seguinte ao usar redes de tarefas:

- É necessário que a função vinculada ao serviço do Amazon ECS forneça ao Amazon ECS as permissões para fazer chamadas a outros serviços da AWS em seu nome. Essa função será criada quando você criar um cluster ou se criar ou atualizar um serviço no AWS Management Console. Para obter mais informações, consulte Uso de perfis vinculados ao serviço para o Amazon ECS. Também é possível criar a função vinculada ao serviço usando o comando da AWS CLI a seguir.

```
aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
```

- O Amazon ECS preenche o nome de host da tarefa usando a rede de tarefas com um nome de host DNS (interno) fornecido pela Amazon quando as opções `enableDnsHostnames` e `enableDnsSupport` estão habilitadas na sua VPC. Se essas opções não estiverem habilitadas, o nome de host DNS da tarefa será definido como um nome de host aleatório. Para obter mais informações sobre as configurações de DNS para uma VPC, consulte Usar DNS com sua VPC no Guia do usuário da Amazon VPC.
- É possível especificar até 16 sub-redes e 5 grupos de segurança para `awsVpcConfiguration`. Para obter mais informações, consulte AwsVpcConfiguration na Referência da API de serviço do contêiner do Amazon Elastic Container Service.
- Você não pode desanexar ou modificar manualmente as ENIs criadas e anexadas pelo Fargate. Isso visa a evitar a exclusão acidental de uma ENI associada a uma tarefa em execução. Para liberar as ENIs para uma tarefa, interrompa-a.
- Se uma sub-rede de VPC for atualizada para alterar o conjunto de opções de DHCP que ela utiliza, não será possível aplicar essas alterações às tarefas existentes que usarem a VPC. Inicie novas tarefas, que receberão a nova configuração para uma migração tranquila ao testar a nova alteração e, em seguida, interrompa as antigas caso nenhuma reversão seja necessária.
- O seguinte se aplica às tarefas executadas na plataforma Fargate versão `1.4.0` ou posterior para Linux, ou `1.0.0` para Windows. Tarefas iniciadas em sub-redes de pilha dupla recebem um endereço IPv4 e um endereço IPv6.
- Para tarefas que utilizam versão da plataforma `1.4.0` ou posterior para Linux ou `1.0.0` para Windows, as ENIs de tarefas oferecem suporte a frames jumbo. As interfaces de rede são configuradas com uma unidade de transmissão máxima (MTU), que se refere ao tamanho da maior carga útil que cabe em um único quadro. Quanto maior a MTU, maior será a carga útil do aplicativo que pode caber em um único quadro, o que reduz a sobrecarga por quadro e aumenta a eficiência. O suporte a quadros jumbo reduz a sobrecarga quando o caminho de rede entre a tarefa e o destino oferece suporte a quadros jumbo.
- Os serviços com tarefas que usam o tipo de inicialização do Fargate) só oferecem suporte a Application Load Balancers e Network Load Balancers. Não há suporte para Classic Load Balancers. Ao criar grupos de destino, você precisa escolher `ip` como o tipo de destino, e não `instance`. Para obter mais informações, consulte Uso do balanceamento de carga para distribuir o tráfego de serviço do Amazon ECS.

Usar uma VPC no modo de pilha dupla

Ao usar uma VPC no modo de pilha dupla, suas tarefas podem se comunicar por IPv4, IPv6 ou ambos. Os endereços IPv4 e IPv6 são independentes um do outro e você pode configurar o roteamento e a segurança na VPC separadamente para IPv4 e IPv6. Para obter mais informações sobre como configurar a VPC para o modo de pilha dupla, consulte Migrar para IPv6 no Guia do usuário da Amazon VPC.

As tarefas do Amazon ECS no Fargate receberão um endereço IPv6 se as seguintes condições forem atendidas:

- A configuração de conta `dualStackIPv6` do Amazon ECS está ativada (`enabled`) para que a entidade principal do IAM execute as tarefas na região em que você está executando suas tarefas. Essa configuração só pode ser modificada usando a API ou a AWS CLI. Você tem a opção de ativar essa configuração para uma entidade principal do IAM específica na conta ou para toda a conta, definindo a configuração padrão da conta. Para obter mais informações, consulte Acesso aos recursos do Amazon ECS com as configurações de conta.
- Sua VPC e sua sub-rede estão habilitadas para IPv6. Para obter mais informações sobre como configurar a VPC para o modo de pilha dupla, consulte Migrar para IPv6 no Guia do usuário da Amazon VPC.
- Sua sub-rede está habilitada para atribuição automática de endereços IPv6. Para obter mais informações sobre como configurar sua sub-rede, consulte Modificar o atributo de endereçamento IPv6 para a sua sub-rede no Guia do usuário da Amazon VPC.
- A tarefa ou o serviço utiliza a versão `1.4.0` ou posterior da plataforma Fargate para Linux.

Se você configurar sua VPC com um gateway da Internet ou um gateway da Internet somente saída, as tarefas do Amazon ECS no Fargate que receberem um endereço IPv6 poderão acessar a Internet. Não são necessários gateways NAT. Para obter mais informações, consulte Gateways da Internet e Gateways da Internet somente de saída no Guia do usuário da Amazon VPC.

Opções de armazenamento para tarefas do Amazon ECS

O Amazon ECS fornece opções de armazenamento de dados flexíveis, econômicas e fáceis de usar, dependendo das suas necessidades.. O Amazon ECS oferece suporte às seguintes opções de volume de dados para contêineres:

Volume de dados	Tipos de execução compatíveis	Sistemas operacionais compatíveis	Persistência de armazenamento	Casos de uso
Amazon Elastic Block Store (Amazon EBS)	Fargate, Amazon EC2	Linux	Pode ser persistido quando anexado a uma tarefa autônoma. Temporário quando anexado a uma tarefa mantida por um serviço.	Os volumes do Amazon EBS fornecem armazenamento em blocos econômico, durável e de alto desempenho para workloads em contêineres com uso intenso de dados. Os casos de uso comuns incluem workloads transacionais, como bancos de dados, áreas de trabalho virtuais e volumes raiz, e workloads com alto throughput, como processamento de logs e workloads ETL. Para obter mais informações, consulte Uso de volumes do Amazon EBS com o Amazon ECS.

Volume de dados	Tipos de execução compatíveis	Sistemas operacionais compatíveis	Persistência de armazenamento	Casos de uso
Amazon Elastic File System (Amazon EFS)	Fargate, Amazon EC2	Linux	Persistente	Os volumes do Amazon EFS fornecem armazenamento de arquivos compartilhado simples, escalável e persistente para uso com tarefas do Amazon ECS, que cresce e diminui automaticamente à medida que arquivos são adicionados e removidos. Os volumes do Amazon EFS oferecem suporte à concorrência e são úteis para aplicações em contêineres que escalam horizontalmente e precisam de funcionalidades de armazenamento, como baixa

Volume de dados	Tipos de execução compatíveis	Sistemas operacionais compatíveis	Persistência de armazenamento	Casos de uso
				latência, alto throughput e consistência de leitura após gravação. Os casos de uso comuns incluem workloads, como data analytics , processam ento de mídia, gerenciamento de conteúdo e distribuição pela Web. Para obter mais informaçõ es, consulte Uso de volumes do Amazon EFS com o Amazon ECS.

Volume de dados	Tipos de execução compatíveis	Sistemas operacionais compatíveis	Persistência de armazenamento	Casos de uso
Amazon FSx para Windows File Server	Amazon EC2	Windows	Persistente	Os volumes do FSx para Windows File Server fornecem servidores de arquivos do Windows totalmente gerenciados que podem ser usados para provisionar tarefas do Windows que precisam de armazenamento de arquivos persistente, distribuído, compartilhado e estático. Casos de uso comuns incluem aplicações .NET que podem exigir pastas locais como armazenamento persistente para salvar as saídas da aplicação. O Amazon FSx

Volume de dados	Tipos de execução compatíveis	Sistemas operacionais compatíveis	Persistência de armazenamento	Casos de uso
				para Windows File Server oferece uma pasta local no contêiner que permite a leitura e gravação de vários contêiner es no mesmo sistema de arquivos apoiado por um compartil hamento SMB. Para obter mais informações, consulte Uso de volumes do FSx para Windows File Server com Amazon ECS.

Volume de dados	Tipos de execução compatíveis	Sistemas operacionais compatíveis	Persistência de armazenamento	Casos de uso
Volumes do Docker	Amazon EC2	Windows, Linux	Persistente	Os volumes do Docker são um recurso de runtime do contêiner do Docker que permite que os contêineres persistam os dados montando um diretório no sistema de arquivos do host. Drivers de volume do Docker (também conhecidos como plug-ins) são usados para integrar os volumes do contêiner com os sistemas de armazenam ento externos. Os volumes do Docker podem ser gerenciados por drivers de terceiros ou pelo driver `local` integrado. Casos

Volume de dados	Tipos de execução compatíveis	Sistemas operacionais compatíveis	Persistência de armazenamento	Casos de uso
				de uso comuns para volumes do Docker incluem fornecer volumes de dados persistentes ou compartilhar volumes em diferentes locais em diferentes contêineres na mesma instância de contêiner. Para obter mais informações, consulte Uso de volumes do Docker com o Amazon ECS.

Volume de dados	Tipos de execução compatíveis	Sistemas operacionais compatíveis	Persistência de armazenamento	Casos de uso
Montagens bind	Fargate, Amazon EC2	Windows, Linux	Temporário	As montagens vinculadas consistem em um arquivo ou diretório no host, como uma instância do Amazon EC2 ou o AWS Fargate, que é montado em um contêiner. Casos de uso comuns para montagens vinculadas incluem compartilhar um volume de um contêiner de origem com outros contêineres na mesma tarefa ou montar um volume de host ou um volume vazio em um ou mais contêineres. Para obter mais informações, consulte Uso de montagens

Volume de dados	Tipos de execução compatíveis	Sistemas operacionais compatíveis	Persistência de armazenamento	Casos de uso
				vinculadas com o Amazon ECS.

Uso de volumes do Amazon EBS com o Amazon ECS

Os volumes do Amazon Elastic Block Store (Amazon EBS) fornecem armazenamento em bloco de alta disponibilidade, econômico, durável e de alta performance para workloads com uso intenso de dados. Os volumes do Amazon EBS podem ser usados com tarefas do Amazon ECS em aplicações de alto throughput e uso intenso de transações.

Durante a execução de tarefas autônomas, você pode fornecer a configuração que será usada para anexar um volume do EBS à tarefa. Durante a criação ou atualização do serviço, você pode fornecer a configuração que será usada para anexar um volume do EBS por tarefa a cada tarefa gerenciada pelo serviço do Amazon ECS.

Ao fornecer a configuração do volume no momento da execução e não na definição da tarefa, você cria definições de tarefas que não estão restritas a um tipo específico de volume de dados ou a configurações específicas de volume do EBS. Em seguida, você pode reutilizar as definições de tarefas em diferentes ambientes de runtime. Por exemplo, você pode fornecer mais throughput durante a implantação para as workloads de produção do que para os ambientes de pré-produção.

Os volumes do Amazon EBS anexados às tarefas do Amazon ECS são gerenciados pelo Amazon ECS em seu nome. Os volumes podem ser criptografados com chaves do AWS Key Management Service (AWS KMS) para proteger seus dados. Você pode configurar volumes novos e vazios para anexação ou usar snapshots para carregar dados de volumes existentes. Ao usar snapshots para configurar volumes, você pode especificar uma `volumeInitializationRate`, em MiB/s, no qual os dados são buscados do snapshot para criar volumes totalmente inicializados em um período previsível. Para obter mais informações sobre a inicialização de volumes, consulte Inicializar volumes do Amazon EBS no Guia do usuário do Amazon EBS. Para obter mais informações sobre a configuração de volumes do Amazon EBS, consulte Adiar a configuração de volumes para o momento da inicialização na definição de tarefa do Amazon ECS e Especificar a configuração de volumes do Amazon EBS na implantação do Amazon ECS.

Para monitorar o desempenho do volume, você também pode usar as métricas do Amazon CloudWatch. Para obter mais informações sobre as métricas do Amazon ECS para volumes do Amazon EBS, consulte Métricas do Amazon ECS CloudWatch e Métricas do Amazon ECS Container Insights.

A anexação de um volume do Amazon EBS a uma tarefa é compatível com todas as Regiões da AWS de empresas comerciais e na China que oferecem suporte ao Amazon ECS.

Para obter mais informações sobre volumes do Amazon EBS, consulte Amazon EBS volumes no Guia do usuário do Amazon EBS.

Sistemas operacionais e tipos de mídia com suporte

A tabela a seguir fornece as configurações com suporte de sistema operacional e tipo de inicialização.

Tipo de inicialização	Linux	Windows
Fargate	Os volumes do Amazon EBS são compatíveis com a versão da plataforma `1.4.0` ou posterior (Linux). Para obter mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.	Não compatível
EC2	Tarefas hospedadas em instâncias Linux baseadas em Nitro com imagens de máquina da Amazon (AMIs) otimizadas para Amazon ECS. Para obter mais informações sobre os tipos de instância, consulte Tipos de instância no Guia do usuário do Amazon EC2. Os volumes do Amazon EBS são compatíveis com	Tarefas hospedadas em instâncias Linux baseadas em Nitro com imagens de máquina da Amazon (AMIs) otimizadas para Amazon ECS. Para obter mais informações sobre os tipos de instância, consulte Tipos de instância no Guia do usuário do Amazon EC2. Os volumes do Amazon EBS são compatíveis com

Tipo de inicialização	Linux	Windows
	a AMI otimizada para ECS `20231219` ou posterior. Para obter mais informaçõ es, consulte Recuperar os metadados da AMI otimizada para Amazon ECS.	a AMI otimizada para ECS `20241017` ou posterior. Para obter mais informaçõ es, consulte Recuperar os metadados da AMI do Windows otimizada para Amazon ECS.

Considerações

Considere o seguinte ao usar volumes do Amazon EBS:

- Não é possível configurar volumes do Amazon EBS para serem anexados às tarefas do Amazon ECS de tipo de inicialização do Fargate nas zonas de disponibilidade `use1-az3`.
- O tipo de volume magnético (`standard`) do Amazon EBS não é compatível com tarefas hospedadas no Fargate. Para obter mais informações sobre os tipos de volumes do Amazon EBS, consulte Amazon EBS volumes no Guia do usuário do Amazon EC2.
- Um perfil do IAM na infraestrutura do Amazon ECS é necessário ao criar um serviço ou uma tarefa autônoma que esteja configurando um volume na implantação. Você pode anexar a política do IAM `AmazonECSInfrastructureRolePolicyForVolumes` gerenciada pela AWS ao perfil ou usar a política gerenciada como um guia para criar e anexar sua própria política com permissões que atendam às suas necessidades específicas. Para obter mais informações, consulte Perfil do IAM de infraestrutura do Amazon ECS.
- Você pode anexar, no máximo, um volume do Amazon EBS a cada tarefa do Amazon ECS, e ele deve ser novo. Não é possível anexar um volume existente do Amazon EBS a uma tarefa. No entanto, você pode configurar um novo volume do Amazon EBS na implantação usando o snapshot de um volume existente.
- Para usar os volumes do Amazon EBS com os serviços do Amazon ECS, o controlador de implantação deve ser “ECS”. Tanto a estratégia de implantação azul/verde como a contínua são compatíveis ao usar esse controlador de implantação.
- Para que um contêiner na tarefa grave no volume montado do Amazon EBS, você deve executar o contêiner como usuário-raiz.
- O Amazon ECS adiciona automaticamente as tags reservadas `AmazonECSCreated` e `AmazonECSManaged` ao volume anexado. Se você remover essas tags do volume, o Amazon

ECS não poderá gerenciá-lo em seu nome. Para obter mais informações sobre marcação de volumes do Amazon EBS, consulte Tagging Amazon EBS volumes. Para obter mais informações sobre marcação de recursos do Amazon ECS, consulte Tagging your Amazon ECS resources.

- Não há suporte para o provisionamento de volumes usando um snapshot de um volume do Amazon EBS que contém partições.
- Os volumes anexados às tarefas gerenciadas por um serviço não são preservados e sempre são excluídos após o encerramento da tarefa.
- Não é possível configurar volumes do Amazon EBS para anexar tarefas do Amazon ECS em execução no AWS Outposts.

Adiar a configuração de volumes para o momento da inicialização na definição de tarefa do Amazon ECS

Para configurar um volume do Amazon EBS para anexar à tarefa, você deve especificar a configuração do ponto de montagem na definição da tarefa e nomear o volume. Você também deve definir `configuredAtLaunch` como `true` porque os volumes do Amazon EBS não podem ser configurados para anexação na definição da tarefa. Em vez disso, os volumes do Amazon EBS são configurados para associação durante a implantação.

Para registrar a definição de tarefa usando a AWS Command Line Interface (AWS CLI), salve o modelo como arquivo JSON e passe o arquivo como entrada para o comando `register-task-definition`.

Para criar e registrar uma definição de tarefa usando o AWS Management Console, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

A definição de tarefa a seguir mostra a sintaxe dos objetos `mountPoints` e `volumes` na definição de tarefa. Para obter mais informações sobre os parâmetros de definição de tarefa, consulte Parâmetros de definição de tarefa do Amazon ECS para o tipo de inicialização do Fargate. Para usar esse exemplo, substitua os *`user input placeholders`* por suas próprias informações.

Linux

```
{
    "family": "mytaskdef",
    "containerDefinitions": [
        {
            "name": "nginx",
```

```
            "image": "public.ecr.aws/nginx/nginx:latest",
            "networkMode": "awsvpc",
           "portMappings": [
                {
                    "name": "nginx-80-tcp",
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp",
                    "appProtocol": "http"
                }
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEBSVolume",
                    "containerPath": "/mount/ebs",
                    "readOnly": true
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "myEBSVolume",
            "configuredAtLaunch": true
        }
    ],
    "requiresCompatibilities": [
        "FARGATE", "EC2"
    ],
    "cpu": "1024",
    "memory": "3072",
    "networkMode": "awsvpc"
}
```

Windows

```
{
    "family": "mytaskdef",
     "memory": "4096",
     "cpu": "2048",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
```

```
    "requiresCompatibilities": ["EC2"]
    "containerDefinitions": [
        {
             "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file
 -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>'; C:\\ServiceMonitor.exe
 w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 443,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEBSVolume",
                    "containerPath": "drive:\ebs",
                    "readOnly": true
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "myEBSVolume",
            "configuredAtLaunch": true
        }
    ],
    "requiresCompatibilities": [
        "FARGATE", "EC2"
    ],
```

```
    "cpu": "1024",
    "memory": "3072",
    "networkMode": "awsvpc"
}
```

`mountPoints`

Tipo: Matriz de objeto

Obrigatório: Não

Os pontos de montagem dos volumes de dados no contêiner. Esse parâmetro é mapeado em `Volumes` na API create-container do Docker na opção `--volume` de execução do docker.

Os contêineres do Windows podem montar diretórios inteiros na mesma unidade como `$env:ProgramData`. Os contêineres do Windows não podem montar diretórios em uma unidade diferente, e os pontos de montagem não podem ser usados entre unidades. Você deve especificar pontos de montagem para anexar um volume do Amazon EBS diretamente a uma tarefa do Amazon ECS.

`sourceVolume`

Tipo: String

Exigido: Sim, quando `mountPoints` são usados

O nome do volume a ser montado.

`containerPath`

Tipo: String

Exigido: Sim, quando `mountPoints` são usados

O caminho no contêiner onde o volume será montado.

`readOnly`

Tipo: booliano

Obrigatório: Não

Caso o valor seja `true`, o contêiner tem acesso somente leitura ao volume. Caso esse valor seja `false`, o contêiner pode gravar no volume. O valor padrão é `false`.

Para tarefas executadas em instâncias do EC2 que executam o sistema operacional Windows, deixe o valor como o padrão de `false`.

`name`

Tipo: String

Obrigatório: Não

O nome do volume. São permitidos até 255 letras (caixa alta e baixa), números, hífens (`-`) e sublinhados (`_`). Esse nome é referenciado no parâmetro `sourceVolume` do objeto `mountPoints` de definição do contêiner.

`configuredAtLaunch`

Tipo: booliano

Obrigatório: sim, quando você deseja anexar um volume do EBS diretamente a uma tarefa.

Especifica se um volume é configurável na execução. Quando definido como `true`, você pode configurar o volume ao executar uma tarefa autônoma ou ao criar ou atualizar um serviço. Quando definido como `false`, não será possível fornecer outra configuração de volume na definição da tarefa. Esse parâmetro deve ser fornecido e definido como `true` para configurar um volume do Amazon EBS para anexar a uma tarefa.

Criptografar dados armazenados em volumes do Amazon EBS para tarefas do Amazon ECS

Você pode usar o AWS Key Management Service (AWS KMS) para criar e gerenciar chaves criptográficas que protegem seus dados. Os volumes do Amazon EBS são criptografados em repouso usando o AWS KMS keys. Os seguintes tipos de dados são criptografados:

- Dados armazenados em repouso no volume
- E/S de disco
- Snapshots criados no volume
- Novos volumes criados de snapshots criptografados

Os volumes do Amazon EBS associados a tarefas podem ser criptografados usando uma Chave gerenciada pela AWS padrão com o alias `alias/aws/ebs` ou uma chave simétrica gerenciada pelo cliente especificada na configuração do volume. As Chaves gerenciadas pela AWS padrão são exclusivas para cada Conta da AWS por Região da AWS e são criadas automaticamente. Para criar

uma chave simétrica gerenciada pelo cliente, siga as etapas em Creating symmetric encryption KMS keys no Guia do desenvolvedor do AWS KMS.

É possível configurar a criptografia do Amazon EBS por padrão para que todos os novos volumes criados e anexados a uma Região da AWS específica sejam criptografados usando a chave do KMS especificada para a sua conta. Para obter mais informações sobre criptografia do Amazon EBS e criptografia por padrão, consulte Criptografia do Amazon EBS no Guia do usuário do Amazon EBS.

Você também pode configurar a criptografia em nível de cluster do Amazon ECS para armazenamento gerenciado do Amazon ECS ao criar ou atualizar um cluster. A criptografia em nível de cluster pode ser usada para criptografar todos os volumes do Amazon EBS associados a tarefas executadas em um cluster específico usando a chave do KMS especificada no nível do cluster. Para obter mais informações sobre como configurar a criptografia no nível do cluster, consulte ManagedStorageConfiguration na Referência da API do Amazon ECS.

É possível configurar uma combinação qualquer dessas chaves. A ordem de precedência das chaves do KMS é a seguinte:

1. A chave do KMS especificada na configuração do volume. Quando você especifica uma chave KMS na configuração do volume, ela substitui o padrão do Amazon EBS e qualquer chave do KMS especificada no nível do cluster.
2. A chave do KMS especificada no nível do cluster. Quando você especifica uma chave do KMS para criptografia no nível de cluster do armazenamento gerenciado do Amazon ECS, ela substitui a criptografia padrão do Amazon EBS, mas não substitui nenhuma chave do KMS especificada na configuração do volume.
3. Criptografia padrão do Amazon EBS. A criptografia padrão se aplica quando você não especifica uma chave do KMS em nível do cluster ou uma chave na configuração do volume. Se você habilitar a criptografia do Amazon EBS por padrão, o padrão será a chave do KMS que você especifica para criptografia por padrão. Caso contrário, o padrão é o Chave gerenciada pela AWS com o alias `alias/aws/ebs`.

> **Note**
>
> Se você definir `encrypted` como `false` na configuração do volume, não especificar nenhuma chave do KMS em nível de cluster e habilitar a criptografia do Amazon EBS por padrão, o volume ainda será criptografado com a chave especificada para a criptografia do Amazon EBS por padrão.

Política de chave do KMS gerenciada pelo cliente

Para criptografar um volume do EBS anexado à tarefa usando uma chave gerenciada pelo cliente, você deve configurar sua política de chave do KMS para garantir que o perfil do IAM usado na configuração do volume tenha as permissões necessárias para usar a chave. A política de chave deve incluir as permissões `kms:CreateGrant` e `kms:GenerateDataKey*`. As permissões `kms:ReEncryptTo` e `kms:ReEncryptFrom` são necessárias para criptografar volumes criados usando snapshots. Se quiser configurar e criptografar somente volumes novos e vazios para anexação, você pode excluir as permissões `kms:ReEncryptTo` e `kms:ReEncryptFrom`.

O trecho JSON a seguir mostra as instruções de política de chave que você pode anexar à sua política de chave do KMS. O uso dessas instruções fornecerá acesso ao Amazon ECS para usar a chave para criptografar o volume do EBS. Para usar os exemplos de instruções de política, substitua os *`user input placeholders`* por suas próprias informações. Como sempre, configure apenas as permissões de que você precisa.

```
{
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": "kms:DescribeKey",
      "Resource":"*"
    },
    {
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": [
      "kms:GenerateDataKey*",
      "kms:ReEncryptTo",
      "kms:ReEncryptFrom"
      ],
      "Resource":"*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "aws_account_id",
          "kms:ViaService": "ec2.region.amazonaws.com"
        },
        "ForAnyValue:StringEquals": {
          "kms:EncryptionContextKeys": "aws:ebs:id"
        }
      }
    },
    {
```

```
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": "kms:CreateGrant",
      "Resource":"*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "aws_account_id",
          "kms:ViaService": "ec2.region.amazonaws.com"
        },
        "ForAnyValue:StringEquals": {
          "kms:EncryptionContextKeys": "aws:ebs:id"
        },
        "Bool": {
          "kms:GrantIsForAWSResource": true
        }
      }
    }
```

Para obter mais informações sobre políticas e permissões de chave, consulte Key policies in AWS KMS e AWS KMS permissions no Guia do desenvolvedor do AWS KMS. Para solucionar problemas de anexação de volumes do EBS relacionados às permissões de chave, consulte Solução de problemas de anexações de volumes do Amazon EBS às tarefas do Amazon ECS .

Especificar a configuração de volumes do Amazon EBS na implantação do Amazon ECS

Depois de registrar uma definição de tarefa com o parâmetro `configuredAtLaunch` definido como `true`, você pode configurar um volume do Amazon EBS na implantação ao executar uma tarefa autônoma ou ao criar ou atualizar um serviço.

Para configurar um volume, você pode usar as APIs do Amazon ECS ou passar um arquivo JSON como entrada para os seguintes comandos da AWS CLI:

- `run-task` para executar uma tarefa autônoma do ECS.
- `start-task` para executar uma tarefa autônoma do ECS em uma instância de contêiner específica. Esse comando não é aplicável às tarefas do tipo de execução do Fargate.
- `create-service` para criar um serviço do ECS.
- `update-service` para atualizar um serviço existente.

> **Note**
>
> Para que um contêiner na tarefa grave no volume montado do Amazon EBS, você deve executar o contêiner como usuário-raiz.

Você também pode configurar um volume do Amazon EBS usando o AWS Management Console. Para ter mais informações, consulte Execução de uma aplicação como uma tarefa do Amazon ECS, Criação de uma implantação de atualização contínua do Amazon ECS e Atualizar um serviço do Amazon ECS.

O trecho JSON a seguir mostra todos os parâmetros de um volume do Amazon EBS que podem ser configurados na implantação. Para usar esses parâmetros na configuração de volume, substitua os *user input placeholders* por suas próprias informações. Para obter mais informações sobre esses parâmetros, consulte Volume configurations.

```
"volumeConfigurations": [
        {
            "name": "ebs-volume",
            "managedEBSVolume": {
                "encrypted": true,
                "kmsKeyId": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
                "volumeType": "gp3",
                "sizeInGiB": 10,
                "snapshotId": "snap-12345",
                "volumeInitializationRate":100,
                "iops": 3000,
                "throughput": 125,
                "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
                            {
                                "key": "key1",
                                "value": "value1"
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
```

```
                "roleArn": "arn:aws:iam::1111222333:role/ecsInfrastructureRole",
                 "terminationPolicy": {
                    "deleteOnTermination": true//can't be configured for service-
managed tasks, always true
                 },
                 "filesystemType": "ext4"
            }
        }
    ]
```

> **Important**
>
> Certifique-se de que o `volumeName` especificado na configuração seja o mesmo `volumeName` especificado na definição de tarefa.

Para obter informações sobre como verificar o status de anexação de volumes, consulte Solução de problemas de anexações de volumes do Amazon EBS às tarefas do Amazon ECS . Para obter informações sobre o perfil do AWS Identity and Access Management (IAM) na infraestrutura do Amazon ECS necessário para a anexação de volumes do EBS, consulte Perfil do IAM de infraestrutura do Amazon ECS.

Veja a seguir exemplos de trechos JSON que mostram a configuração dos volumes do Amazon EBS. Esses exemplos podem ser usados salvando os trechos em arquivos JSON e passando os arquivos como parâmetros (usando o parâmetro `--cli-input-json file://filename`) para comandos da AWS CLI. Substitua os *`user input placeholders`* por suas próprias informações.

Configuração de um volume para uma tarefa autônoma

O trecho a seguir mostra a sintaxe para configurar volumes do Amazon EBS para anexação a uma tarefa autônoma. O trecho JSON a seguir mostra a sintaxe para definir as configurações de `volumeType`, `sizeInGiB`, `encrypted` e `kmsKeyId`. A configuração especificada no arquivo JSON é usada para criar e anexar um volume do EBS à tarefa autônoma.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "volumeConfigurations": [
        {
            "name": "datadir",
```

```
            "managedEBSVolume": {
                "volumeType": "gp3",
                "sizeInGiB": 100,
                "roleArn":"arn:aws:iam::1111222333:role/ecsInfrastructureRole",
                "encrypted": true,
                "kmsKeyId":
 "arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
            }
        }
    ]
}
```

Configuração de um volume na criação do serviço

O trecho a seguir mostra a sintaxe para configurar volumes do Amazon EBS para anexação a tarefas gerenciadas por um serviço. Os volumes são baseados no snapshot especificado usando o parâmetro `snapshotId` a uma taxa de 200 MiB/s. A configuração especificada no arquivo JSON é usada para criar e anexar um volume do EBS a cada tarefa gerenciada pelo serviço.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "serviceName": "mysvc",
   "desiredCount": 2,
   "volumeConfigurations": [
        {
            "name": "myEbsVolume",
            "managedEBSVolume": {
              "roleArn":"arn:aws:iam::1111222333:role/ecsInfrastructureRole",
              "snapshotId": "snap-12345",
              "volumeInitializationRate": 200
            }
        }
   ]
}
```

Configuração de um volume na atualização do serviço

O trecho JSON a seguir mostra a sintaxe para atualizar um serviço que anteriormente não tinha volumes do Amazon EBS configurados para anexar a tarefas. Você deve fornecer o ARN de uma revisão de definição de tarefa com `configuredAtLaunch` definido como `true`. O trecho JSON a seguir mostra a sintaxe para definir `volumeType`, `sizeInGiB`, `throughput` e `iops` e as

configurações de `filesystemType`. Essa configuração é usada para criar e anexar um volume do EBS a cada tarefa gerenciada pelo serviço.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "service": "mysvc",
   "desiredCount": 2,
   "volumeConfigurations": [
        {
            "name": "myEbsVolume",
            "managedEBSVolume": {
              "roleArn":"arn:aws:iam::1111222333:role/ecsInfrastructureRole",
               "volumeType": "gp3",
                "sizeInGiB": 100,
                 "iops": 3000,
                "throughput": 125,
                "filesystemType": "ext4"
            }
        }
   ]
}
```

Configuração de um serviço para não utilizar mais os volumes do Amazon EBS

O trecho JSON a seguir mostra a sintaxe para atualizar um serviço para não utilizar mais os volumes do Amazon EBS. Você deve fornecer o ARN de uma definição de tarefa com `configuredAtLaunch` definido como `false` ou de uma definição de tarefa sem o parâmetro `configuredAtLaunch`. Você também deve fornecer um objeto `volumeConfigurations` vazio.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "service": "mysvc",
   "desiredCount": 2,
   "volumeConfigurations": []
}
```

Política de encerramento para volumes do Amazon EBS

Quando uma tarefa do Amazon ECS é encerrada, o Amazon ECS usa o valor de `deleteOnTermination` para determinar se o volume do Amazon EBS associado à tarefa

encerrada deve ser excluído. Por padrão, os volumes do EBS anexados às tarefas são excluídos quando a tarefa é encerrada. Em tarefas autônomas, você pode alterar essa configuração para preservar o volume após o encerramento da tarefa.

> **Note**
>
> Os volumes anexados às tarefas gerenciadas por um serviço não são preservados e sempre são excluídos após o encerramento da tarefa.

Tags em volumes do Amazon EBS

Você pode marcar volumes do Amazon EBS usando o objeto `tagSpecifications`. Usando o objeto, você pode fornecer suas próprias tags e definir a propagação delas na definição da tarefa ou do serviço, dependendo se o volume está anexado a uma tarefa autônoma ou a uma tarefa em um serviço. O número máximo de tags que podem ser anexadas a um volume é 50.

> **Important**
>
> O Amazon ECS anexa automaticamente as tags reservadas `AmazonECSCreated` e `AmazonECSManaged` a um volume do Amazon EBS. Isso significa que você pode controlar a anexação de no máximo 48 tags adicionais a um volume. Essas tags adicionais podem ser definidas pelo usuário, gerenciadas pelo ECS ou propagadas.

Se quiser adicionar tags gerenciadas pelo Amazon ECS ao volume, você deve definir `enableECSManagedTags` como `true` na chamada de `UpdateService`, `CreateService`, `RunTask` ou `StartTask`. Se você ativar as tags gerenciadas pelo Amazon ECS, o Amazon ECS marca o volume automaticamente com informações de cluster e serviço (`aws:ecs:`*`clusterName`* e `aws:ecs:`*`serviceName`*). Para obter mais informações sobre marcação de recursos do Amazon ECS, consulte Tagging your Amazon ECS resources.

O trecho JSON a seguir mostra a sintaxe para marcar cada volume do Amazon EBS anexado a cada tarefa em um serviço com uma tag definida pelo usuário. Para usar esse exemplo para criar um serviço, substitua os *`user input placeholders`* pelas suas próprias informações.

```
{
   "cluster": "mycluster",
```

```
    "taskDefinition": "mytaskdef",
    "serviceName": "mysvc",
    "desiredCount": 2,
    "enableECSManagedTags": true,
    "volumeConfigurations": [
         {
             "name": "datadir",
             "managedEBSVolume": {
                 "volumeType": "gp3",
                 "sizeInGiB": 100,
                  "tagSpecifications": [
                     {
                         "resourceType": "volume",
                         "tags": [
                             {
                                 "key": "key1",
                                 "value": "value1"
                             }
                         ],
                         "propagateTags": "NONE"
                     }
                 ],
                 "roleArn":"arn:aws:iam:1111222333:role/ecsInfrastructureRole",
                 "encrypted": true,
                 "kmsKeyId":
  "arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
             }
         }
    ]
}
```

> **Important**
>
> Você deve especificar um tipo de recurso de `volume` para marcar os volumes do Amazon EBS.

Desempenho de volumes do Amazon EBS para tarefas sob demanda do Fargate

As IOPS e o throughput básico de volume do Amazon EBS disponíveis para uma tarefa sob demanda do Fargate dependem do total de unidades de CPU solicitadas para a tarefa. Se você solicitar 0,25, 0,5 ou 1 unidade de CPU virtual (vCPU) para a tarefa do Fargate, recomendamos

configurar um volume SSD de uso geral (gp2 ou gp3) ou um volume de unidade de disco rígido (HDD) (st1 ou sc1). Se você solicitar mais de 1 vCPU para a tarefa do Fargate, os limites básicos de desempenho a seguir se aplicam a um volume do Amazon EBS anexado à tarefa. Você pode obter temporariamente um desempenho do EBS superior aos limites a seguir. No entanto, é recomendável planejar a workload com base nesses limites.

Unidades de CPU solicitadas (em vCPUs)	IOPS básicas do Amazon EBS (E/S de 16 KiB)	Throughput básico do Amazon EBS (em MiBps, E/S de 128 KiB)	Largura de banda básica (Mbps)
2	3.000	75	360
4	5.000	120	1.150
8	10.000	250	2.300
16	15.000	500	4.500

> **Note**
>
> Ao configurar um volume do Amazon EBS para anexação a uma tarefa do Fargate, o limite de desempenho do Amazon EBS para a tarefa do Fargate é compartilhado entre o armazenamento temporário da tarefa e o volume anexado.

Desempenho dos volumes do Amazon EBS para tarefas do EC2

O Amazon EBS fornece tipos de volume que diferem em características de performance e preço para que você possa adaptar o custo e a performance de armazenamento às necessidades das aplicações. Para obter informações sobre performance, incluindo IOPS por volume e throughput por volume, consulte Tipos de volume do Amazon EBS no Guia do usuário do Amazon Elastic Block Store.

Solução de problemas de anexações de volumes do Amazon EBS às tarefas do Amazon ECS

Talvez seja necessário solucionar problemas ou verificar a anexação dos volumes do Amazon EBS às tarefas do Amazon ECS.

Verificação do status de anexação de volumes

Você pode usar o AWS Management Console para visualizar o status de anexação de um volume do Amazon EBS a uma tarefa do Amazon ECS. Se a tarefa for iniciada e a anexação falhar, você também verá um motivo do status que pode ser usado para solucionar problemas. O volume criado será excluído e a tarefa será interrompida. Para obter mais informações sobre os motivos de status, consulte Motivos do status para anexação do volume do Amazon EBS às tarefas do Amazon ECS.

Para visualizar o status de anexação e o motivo do status de um volume usando o console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, escolha o cluster em que a tarefa está sendo executada. A página de detalhes do cluster é exibida.
3. Na página de detalhes do cluster, escolha a guia Tarefas.
4. Escolha a tarefa cujo status de anexação de volumes você deseja visualizar. Talvez seja necessário usar Filtrar o status desejado e escolher Interrompida se a tarefa que você deseja examinar tiver sido interrompida.
5. Na página de detalhes da tarefa, escolha a guia Volumes. Você poderá ver o status de anexação do volume do Amazon EBS em Status do anexo. Se o volume apresentar falhas ao ser anexado à tarefa, você poderá escolher o status em Status do anexo para exibir a causa da falha.

Você também pode visualizar o status de anexação de volumes de uma tarefa e o motivo associado do status usando a API DescribeTasks.

Falhas de serviço e tarefas

Você pode encontrar falhas em serviços ou tarefas que não são específicas dos volumes do Amazon EBS e que podem afetar a anexação do volume. Para obter mais informações, consulte .

- Mensagens de evento de serviço
- Códigos de erro de tarefas interrompidas
- Motivos de falha da API

Motivos do status para anexação do volume do Amazon EBS às tarefas do Amazon ECS

Use a referência a seguir para corrigir problemas que você possa encontrar na forma de motivos de status no AWS Management Console ao configurar volumes do Amazon EBS para anexação às tarefas do Amazon ECS. Para obter mais informações sobre como localizar esses motivos de status no console, consulte Verificação do status de anexação de volumes.

O ECS não conseguiu assumir o perfil de infraestrutura do ECS configurado 'arn:aws:iam::*111122223333*:role/*ecsInfrastructureRole*'. Verifique se o perfil que está sendo passado tem a relação de confiança adequada com o Amazon ECS

Esse motivo de status aparece nos cenários a seguir.

- Você fornece um perfil do IAM sem a política de confiança necessária anexada. O Amazon ECS não poderá acessar o perfil do IAM de infraestrutura do Amazon ECS fornecida se o perfil não tiver a política de confiança necessária. A tarefa pode ficar presa no estado `DEPROVISIONING`. Para obter mais informações sobre a política de confiança necessária, consulte Perfil do IAM de infraestrutura do Amazon ECS.
- O usuário do IAM não tem permissão para passar o perfil de infraestrutura do Amazon ECS para o Amazon ECS. A tarefa pode ficar presa no estado `DEPROVISIONING`. Para evitar esse problema, você pode anexar a permissão `PassRole` ao usuário. Para obter mais informações, consulte Perfil do IAM de infraestrutura do Amazon ECS.
- O perfil do IAM não tem as permissões necessárias para anexação de volumes do Amazon EBS. A tarefa pode ficar presa no estado `DEPROVISIONING`. Para obter mais informações sobre as permissões específicas necessárias para anexação de volumes do Amazon EBS às tarefas, consulte Perfil do IAM de infraestrutura do Amazon ECS.

> **Note**
>
> Você também pode ver essa mensagem de erro devido a um atraso na propagação do perfil. Se tentar usar o perfil novamente depois de esperar alguns minutos não resolver o problema, você pode ter configurado incorretamente a política de confiança do perfil.

O ECS não conseguiu configurar o volume do EBS. IdempotentParameterMismatch" encontrado; "O token do cliente que você forneceu está associado a um recurso que já foi excluído. Use um token de cliente diferente."

Os seguintes cenários envolvendo a chave do AWS KMS podem fazer com que uma mensagem de `IdempotentParameterMismatch` apareça:

- Você especifica um ARN, ID ou alias da chave do KMS que não é válido. Nesse cenário, a tarefa pode parecer ter sido executada com êxito, mas ela falhará, porque a AWS autentica a chave do KMS de forma assíncrona. Para obter mais informações, consulte Amazon EBS encryption no Guia do usuário do Amazon EC2.
- Você fornece uma chave gerenciada pelo cliente sem as permissões para que o perfil do IAM de infraestrutura do Amazon ECS use a chave para criptografia. Para evitar problemas na permissão da política de chave, consulte o exemplo política de chave do AWS KMS em Data encryption for Amazon EBS volumes.

Você pode configurar o Amazon EventBridge para enviar eventos de volumes do Amazon EBS e eventos de alteração do estado de tarefas do Amazon ECS para um destino, como grupos do Amazon CloudWatch. Em seguida, você pode usar esses eventos para identificar o problema específico relacionado à chave gerenciada pelo cliente que afetou a anexação do volume. Para obter mais informações, consulte .

- Como posso criar um grupo de logs do CloudWatch para usar como destino para uma regra do EventBridge? no AWS re:Post.
- Task state change events.
- Eventos do Amazon EventBridge para Amazon EBS no Guia do usuário do Amazon EBS.

O ECS atingiu o tempo limite ao configurar a anexação de volumes do EBS à tarefa.

Os cenários a seguir, relacionados ao formato do sistema de arquivos, resultam nessa mensagem.

- O formato do sistema de arquivos que você especifica durante a configuração não é compatível com o sistema operacional da tarefa.
- Você configura um volume do Amazon EBS para ser criado de um snapshot, e o formato do sistema de arquivos do snapshot não é compatível com o sistema operacional da tarefa. Em volumes criados de um snapshot, você deve especificar o mesmo tipo de sistema de arquivos que o volume estava usando quando o snapshot foi criado.

Você pode utilizar os logs do agente de contêiner do Amazon ECS para solucionar essa mensagem em tarefas do tipo de execução do Amazon EC2. Para obter mais informações, consulte Amazon ECS log file locations e Amazon ECS log collector.

Uso de volumes do Amazon EFS com o Amazon ECS

O Amazon Elastic File System (Amazon EFS) oferece armazenamento simples e escalável de arquivos para uso com tarefas do Amazon ECS. Com o Amazon EFS, a capacidade de armazenamento é elástica. Ela aumenta e diminui automaticamente à medida que arquivos são adicionados e removidos. Suas aplicações poderão ter o armazenamento que precisarem e quando precisarem.

Use os sistemas de arquivos do Amazon EFS com o Amazon ECS para exportar dados do sistema de arquivos em sua frota de instâncias de contêiner. Desse modo, suas tarefas terão acesso ao mesmo armazenamento persistente, não importa a instância em que estejam. Suas definições de tarefa devem fazer referências a montagens de volume na instância de contêiner para usar o sistema de arquivos.

Para ver um tutorial, consulte Configurar os sistemas de arquivos do Amazon EFS para o Amazon ECS usando o console.

Considerações

Considere o seguinte ao usar volumes do Amazon EFS:

- Para tarefas que usam o tipo de inicialização do EC2, o suporte ao sistema de arquivos do Amazon EFS foi adicionado como uma visualização prévia pública com a versão `20191212` da AMI otimizada para Amazon ECS com a versão 1.35.0 do agente de contêiner. No entanto, o suporte ao sistema de arquivos do Amazon EFS se tornou disponível para o público com a versão `20200319` da AMI otimizada para Amazon ECS com a versão 1.38.0 do agente de contêiner, que continha os recursos de ponto de acesso do Amazon EFS e de autorização do IAM. Recomendamos o uso da versão `20200319` ou posterior da AMI otimizada para Amazon ECS ou posterior para usar esses recursos. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.

  > **Note**
  >
  > Se você criar sua própria AMI, deverá usar o agente de contêiner 1.38.0 ou posterior, `ecs-init` versão 1.38.0-1 ou posterior e executar os comandos a seguir na instância do

Amazon EC2 para habilitar o plug-in de volume do Amazon ECS. Os comandos dependem de você estar usando o Amazon Linux 2 ou o Amazon Linux como imagem base.
Amazon Linux 2

```
yum install amazon-efs-utils
systemctl enable --now amazon-ecs-volume-plugin
```

Amazon Linux

```
yum install amazon-efs-utils
sudo shutdown -r now
```

- Para tarefas hospedadas no Fargate, os sistemas de arquivos do Amazon EFS são compatíveis com a versão 1.4.0 ou posterior (Linux) da plataforma. Para obter mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.
- Ao usar volumes do Amazon EFS para tarefas hospedadas no Fargate, o Fargate cria um contêiner do supervisor responsável pelo gerenciamento do volume do Amazon EFS. O contêiner do supervisor usa uma pequena quantidade de memória da tarefa e CPU. O contêiner supervisor fica visível ao consultar a versão 4 do endpoint de metadados de tarefa. Além disso, ele é visível no CloudWatch Container Insights como o nome de contêiner `aws-fargate-supervisor`. Para obter mais informações ao usar o tipo de execução do Amazon EC2, consulte Endpoint de metadados de tarefas do Amazon ECS versão 4. Para obter mais informações ao usar o tipo de execução do Fargate, consulte Endpoint de metadados de tarefas do Amazon ECS versão 4 para tarefas no Fargate.
- Não há suporte para o uso de volumes do Amazon EFS ou a especificação de um `EFSVolumeConfiguration` em instâncias externas.
- Recomendamos que você defina o parâmetro `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION` no arquivo de configuração do agente como um valor menor do que o padrão (cerca de 1 hora). Essa alteração ajuda a evitar a expiração da credencial de montagem do EFS e permite limpar as montagens que não estão em uso. Para obter mais informações, consulte Configuração do agente de contêiner do Amazon ECS.

Uso dos pontos de acesso do Amazon EFS

Os pontos de acesso do Amazon EFS são pontos de entrada específicos da aplicação para um sistema de arquivos do EFS para gerenciar o acesso de aplicações a conjuntos de dados

compartilhados. Para obter mais informações sobre pontos de acesso do Amazon EFS e como controlar o acesso a eles, consulte Trabalhar com pontos de acesso do Amazon EFS no Guia do usuário do Amazon Elastic File System.

Os pontos de acesso podem impor uma identidade de usuário, inclusive grupos POSIX do usuário, para todas as solicitações do sistema de arquivamento feitas por meio do ponto de acesso. Pontos de acesso também podem impor um diretório raiz distinto para o sistema de arquivos. Isso é feito para que os clientes apenas possam acessar dados no diretório definido ou em seus subdiretórios.

> **Note**
>
> Ao criar um ponto de acesso do EFS, você especifica um caminho no sistema de arquivos para servir como o diretório raiz. Ao fazer referência ao sistema de arquivos do EFS com um ID de ponto de acesso em sua definição de tarefa do Amazon ECS, o diretório raiz deve ser omitido ou definido como `/`, o que impõe o caminho definido no ponto de acesso do EFS.

É possível usar uma função do IAM da tarefa do Amazon ECS para obrigar que aplicações específicas usem um ponto de acesso específico. Ao combinar políticas do IAM com pontos de acesso, você pode fornecer facilmente acesso seguro a conjuntos de dados específicos para as suas aplicações. Para obter mais informações sobre como usar perfis do IAM de tarefa, consulte Perfil do IAM para tarefas do Amazon ECS.

Práticas recomendadas para usar volumes do Amazon EFS com o Amazon ECS

Anote as práticas recomendadas a seguir ao usar o Amazon EFS com o Amazon ECS.

Controles de segurança e acesso para volumes do Amazon EFS

O Amazon EFS oferece recursos de controle de acesso para garantir que os dados armazenados em um sistema de arquivos do Amazon EFS estejam seguros e acessíveis somente em aplicações que precisam deles. Você pode proteger os dados habilitando a criptografia em repouso e em trânsito. Para obter mais informações, consulte Criptografia de dados no Amazon EFS no Manual do usuário do Amazon Elastic File System.

Além da criptografia de dados, você pode usar o Amazon EFS para restringir o acesso a um sistema de arquivos. Há três maneiras de implementar o controle de acesso no EFS.

- Grupos de segurança: com os destinos de montagem do Amazon EFS, você pode configurar um grupo de segurança usado para permitir e negar tráfego de rede. É possível configurar o grupo

de segurança anexado ao Amazon EFS para permitir o tráfego do NFS (porta 2049) do grupo de segurança que está conectado às instâncias do Amazon ECS ou, ao usar o modo de rede `awsvpc`, da tarefa do Amazon ECS.

- IAM: você pode restringir o acesso a um sistema de arquivos do Amazon EFS usando o IAM. Quando configuradas, as tarefas do Amazon ECS exigem um perfil do IAM para acessar o sistema de arquivos e montar um sistema de arquivos do EFS. Para obter mais informações, consulte Usar o IAM para controlar o acesso de dados do system de arquivos no Guia do usuário do Amazon Elastic File System.

  As políticas do IAM também podem impor condições predefinidas, como exigir que um cliente use TLS ao se conectar a um sistema de arquivos do Amazon EFS. Para obter mais informações, consulte Amazon EFS condition keys for clients no Guia do usuário do Amazon Elastic File System.
- Pontos de acesso do Amazon EFS: os pontos de acesso do Amazon EFS são pontos de entrada específicos da aplicação em um sistema de arquivos do Amazon EFS. Você pode usar os pontos de acesso para impor uma identidade de usuário, incluindo grupos POSIX do usuário, a todas as solicitações do sistema de arquivos feitas por meio do ponto de acesso. Pontos de acesso também podem impor um diretório raiz distinto para o sistema de arquivos. Isso é feito para que os clientes só possam acessar dados no diretório especificado ou em seus subdiretórios.

Políticas do IAM

É posssível usar políticas do IAM para controlar o acesso ao sistema de arquivos do Amazon EFS.

É possível especificar as ações a seguir para clientes NFS em um sistema de arquivos usando uma política de sistema de arquivos.

Ação	Descrição
`elasticfilesystem:ClientMount`	Fornece acesso somente leitura a um sistema de arquivos para um cliente NFS.
`elasticfilesystem:ClientWrite`	Fornece permissões de gravação em um sistema de arquivos.
`elasticfilesystem:ClientRootAccess`	Fornece o uso do usuário raiz ao acessar um sistema de arquivos.

Você precisa especificar cada ação em uma política. É possível definir políticas das seguintes maneiras:

- Baseada no cliente: anexe a política ao perfil da tarefa

  Defina a opção Autorização do IAM ao criar a definição da tarefa.
- Baseada em recursos: anexe a política ao sistema de arquivos do Amazon EFS

  Se a política baseada em recursos não existir, por padrão, na criação do sistema de arquivos, o acesso será concedido a todas as entidades principais (*).

Quando você define a opção Autorização do IAM, mesclamos a política associada ao perfil da tarefa e a baseada em recursos do Amazon EFS. A opção Autorização do IAM passa a identidade da tarefa (o perfil da tarefa) com a política para o Amazon EFS. Isso permite que a política baseada em recursos do Amazon EFS tenha contexto para o usuário ou perfil do IAM especificado na política. Se você não definir a opção, a política em nível de recurso do Amazon EFS identificará o usuário do IAM como "anônimo".

Considere implementar todos os três controles de acesso em um sistema de arquivos do Amazon EFS para obter segurança máxima. Por exemplo, você pode configurar o grupo de segurança anexado a um ponto de montagem do Amazon EFS para permitir somente a entrada de tráfego do NFS de um grupo de segurança associado à instância de contêiner ou tarefa do Amazon ECS. Além disso, você pode configurar o Amazon EFS para exigir um perfil do IAM para acessar o sistema de arquivos, mesmo que a conexão seja originada de um grupo de segurança permitido. Por último, você pode usar os pontos de acesso do Amazon EFS para impor permissões de usuário POSIX e especificar diretórios raiz para aplicações.

O trecho de definição de tarefa a seguir mostra como montar um sistema de arquivos do Amazon EFS usando um ponto de acesso.

```
"volumes": [
    {
      "efsVolumeConfiguration": {
        "fileSystemId": "fs-1234",
        "authorizationConfig": {
          "accessPointId": "fsap-1234",
          "iam": "ENABLED"
        },
        "transitEncryption": "ENABLED",
        "rootDirectory": ""
```

```
      },
      "name": "my-filesystem"
    }
]
```

Performance de volumes do Amazon EFS

O Amazon EFS oferece dois modos de desempenho: uso geral e E/S máxima. O uso geral é adequado para aplicações sensíveis à latência, como sistemas de gerenciamento de conteúdo e ferramentas de CI/CD. Por outro lado, os sistemas de arquivos com E/S máxima são adequados para workloads como data analytics, processamento de mídia e machine learning. Essas workloads precisam realizar operações paralelas de centenas ou até milhares de contêineres e exigem o maior throughput agregado e IOPS possíveis. Para obter mais informações, consulte Amazon EFS performance modes no Guia do usuário do Amazon Elastic File System.

Algumas workloads sensíveis à latência exigem os níveis mais altos de E/S fornecidos pelo modo de desempenho de E/S máxima e a latência mais baixa fornecida pelo modo de desempenho de uso geral. Para esse tipo de workload, recomendamos a criação de vários sistemas de arquivos do modo de desempenho de uso geral. Nesse caso, recomendamos distribuir a workload da aplicação em todos esses sistemas de arquivos, desde que a workload e as aplicações possam oferecer suporte a ela.

Throughput dos volumes do Amazon EFS

Todos os sistemas de arquivos do Amazon EFS têm um throughput associado medido, determinado pela quantidade de throughput provisionado para sistemas de arquivos que usam Throughput provisionado ou pela quantidade de dados armazenados na classe de armazenamento EFS Standard ou One Zone para sistemas de arquivos que usam Throughput intermitente. Para obter mais informações, consulte Understanding metered throughput no Guia do usuário do Amazon Elastic File System.

O modo de throughput padrão dos sistemas de arquivos do Amazon EFS é o modo intermitente. Com o modo intermitente, o throughput disponível para um sistema de arquivos aumenta ou reduz a escala verticalmente à medida que o sistema de arquivos cresce. Como workloads baseadas em arquivos costumam apresentar picos, exigindo altos níveis de throughput por um tempo e níveis mais baixos de throughput no restante, o Amazon EFS foi criado para intermitência, permitindo altos níveis de throughput por alguns períodos. Além disso, como várias workloads exigem muita leitura, as operações de leitura são medidas na proporção de 1:3 em relação a outras operações de NFS (como gravação).

Todos os sistemas de arquivos do Amazon EFS oferecem um desempenho básico consistente de 50 MB/s para cada TB de armazenamento no Amazon EFS Standard ou Amazon EFS One Zone. Todos os sistemas de arquivos (independentemente do tamanho) podem gerar intermitências a 100 MiB/s. Sistemas de arquivos com mais de 1 TB de armazenamento EFS Standard ou EFS One Zone podem gerar intermitência a 100 MB/s para cada TB. Como as operações de leitura são medidas na proporção de 1:3, você pode gerar até 300 MiBs/s para cada TiB de throughput de leitura. Conforme você adiciona dados ao sistema de arquivos, o throughput máximo disponível para ele é escalado de forma linear e automática com seu armazenamento na classe Amazon EFS Standard. Se precisar de mais throughput do que pode obter com a quantidade de dados armazenados, você pode configurar o throughput provisionado de acordo com a quantidade específica que a workload exige.

O throughput do sistema de arquivos é compartilhado por todas as instâncias do Amazon EC2 conectadas a um sistema de arquivos. Por exemplo, um sistema de arquivos de 1 TB que atinge intermitência a 100 MB/s de throughput pode gerar 100 MB/s de uma única instância do Amazon EC2, cada uma com 10 MB/s. Para obter mais informações, consulte Modos de performance no Guia do usuário do Amazon Elastic File System.

Otimização de custos para volumes do Amazon EFS

O Amazon EFS simplifica o ajuste de escala do armazenamento para você. Os sistemas de arquivos do Amazon EFS crescem automaticamente à medida que você adiciona mais dados. Principalmente com o modo Throughput intermitente do Amazon EFS, o throughput no Amazon EFS escala à medida que o tamanho do sistema de arquivos na classe de armazenamento padrão aumenta. Para melhorar o throughput sem pagar um custo adicional pelo throughput provisionado em um sistema de arquivos do EFS, você pode compartilhar um sistema de arquivos do Amazon EFS com várias aplicações. Usando pontos de acesso do Amazon EFS, você pode implementar o isolamento de armazenamento em sistemas de arquivos compartilhados do Amazon EFS. Ao fazer isso, mesmo que as aplicações ainda compartilhem o mesmo sistema de arquivos, elas não podem acessar os dados a menos que você as autorize.

À medida que os dados crescem, o Amazon EFS ajuda você a mover automaticamente arquivos acessados com pouca frequência para uma classe de armazenamento inferior. A classe de armazenamento Amazon EFS Standard-Infrequent Access (IA) reduz os custos de armazenamento de arquivos que não são acessados todos os dias. O EFS faz isso sem sacrificar a alta disponibilidade, a alta durabilidade, a elasticidade e o acesso ao sistema de arquivos POSIX que o Amazon EFS fornece. Para obter mais informações, consulte Classes de armazenamento do Amazon EFS no Guia do usuário do Amazon Elastic File System.

Considere usar as políticas de ciclo de vida do Amazon EFS para economizar dinheiro transferindo automaticamente arquivos acessados com pouca frequência para o armazenamento Amazon EFS IA. Para obter mais informações, consulte Amazon EFS lifecycle management (Gerenciamento de ciclo de vida do Amazon EFS) no Guia do usuário do Amazon Elastic File System.

Ao criar um sistema de arquivos do Amazon EFS, você pode escolher se o Amazon EFS replica seus dados em várias zonas de disponibilidade (padrão) ou os armazena de forma redundante em uma única zona de disponibilidade. A classe de armazenamento Amazon EFS One Zone pode reduzir os custos de armazenamento em uma margem significativa em comparação com as classes de armazenamento Amazon EFS Standard. Considere usar a classe de armazenamento Amazon EFS One Zone para workloads que não exigem resiliência Multi-AZ. Você pode reduzir ainda mais o custo do armazenamento Amazon EFS One Zone transferindo arquivos acessados com pouca frequência para o Amazon EFS One Zone-Infrequent Access. Para obter mais informações, consulte o Amazon EFS Infrequent Access.

Proteção de dados de volume do Amazon EFS

O Amazon EFS armazena os dados de forma redundante em várias zonas de disponibilidade em sistemas de arquivos que usam classes de armazenamento padrão. Ao selecionar as classes de armazenamento Amazon EFS One Zone, seus dados são armazenados de forma redundante em uma única zona de disponibilidade. Além disso, o Amazon EFS foi projetado para fornecer 99.999999999% (11 noves) de durabilidade em um determinado ano.

Como em qualquer ambiente, é uma prática recomendada ter um backup e criar proteções contra a exclusão acidental. Para dados do Amazon EFS, essa prática recomendada inclui um backup funcional e testado regularmente usando o AWS Backup. Os sistemas de arquivos que usam as classes de armazenamento Amazon EFS One Zone são configurados para fazer backup automático dos arquivos por padrão na criação do sistema de arquivos, a menos que você desabilite essa funcionalidade. Para obter mais informações, consulte Backup de sistemas de arquivos EFS no Guia do usuário do Amazon Elastic File System.

Especificar um sistema de arquivos do Amazon EFS na definição de tarefa do Amazon ECS

Para usar volumes do sistema de arquivos do Amazon EFS para seus contêineres, você deve especificar as configurações de volume e ponto de montagem na definição de tarefa. O trecho JSON de definição de tarefa a seguir mostra a sintaxe para os objetos `volumes` e `mountPoints` de um contêiner.

```
{
```

```
    "containerDefinitions": [
        {
            "name": "container-using-efs",
            "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
            "entryPoint": [
                "sh",
                "-c"
            ],
            "command": [
                "ls -la /mount/efs"
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEfsVolume",
                    "containerPath": "/mount/efs",
                    "readOnly": true
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "myEfsVolume",
            "efsVolumeConfiguration": {
                "fileSystemId": "fs-1234",
                "rootDirectory": "/path/to/my/data",
                "transitEncryption": "ENABLED",
                "transitEncryptionPort": integer,
                "authorizationConfig": {
                    "accessPointId": "fsap-1234",
                    "iam": "ENABLED"
                }
            }
        }
    ]
}
```

`efsVolumeConfiguration`

Tipo: Objeto

Obrigatório: Não

Esse parâmetro é especificado quando volumes Amazon EFS são usados.

`fileSystemId`

Tipo: string

Obrigatório: Sim

A ID do sistema de arquivamento Amazon EFS a ser usada.

`rootDirectory`

Tipo: string

Obrigatório: Não

O diretório dentro do sistema de arquivamento Amazon EFS a ser montado como diretório raiz dentro do host. Caso esse parâmetro seja omitido, a raiz do volume Amazon EFS será usada. Especificar `/` tem o mesmo efeito que omitir esse parâmetro.

> **Important**
>
> Se um ponto de acesso do EFS for especificado no `authorizationConfig`, o parâmetro do diretório raiz deverá ser omitido ou definido como `/`, o que imporá o caminho definido no ponto de acesso do EFS.

`transitEncryption`

Tipo: String

Valores válidos: `ENABLED` | `DISABLED`

Obrigatório: Não

Especifica se a criptografia deve ou não ser habilitada para dados do Amazon EFS em trânsito entre o host do Amazon ECS e o servidor do Amazon EFS. A criptografia de trânsito deverá ser habilitada se a autorização do IAM do Amazon EFS for usada. Se o parâmetro for omitido, o valor padrão `DISABLED` será usado. Para obter mais informações, consulte Criptografar dados em trânsito no Guia do usuário do Amazon Elastic File System.

`transitEncryptionPort`

Tipo: inteiro

Obrigatório: Não

A porta a ser usada ao enviar dados criptografados entre o host do Amazon ECS e o servidor do Amazon EFS. Se você não especificar uma porta de criptografia em trânsito, será usada a estratégia de seleção de porta usada pelo assistente de montagem do Amazon EFS. Para mais informações, consulte Auxiliar de Montagem EFS no Guia de Usuário Amazon Elastic File System.

`authorizationConfig`

Tipo: Objeto

Obrigatório: Não

Detalhes de configuração de autorização do sistema de arquivamento Amazon EFS.

`accessPointId`

Tipo: Sequência

Obrigatório: Não

ID do ponto de acesso a ser usado. Se um ponto de acesso for especificado, o valor do diretório raiz no `efsVolumeConfiguration` deve ser omitido ou definido como `/`, que impõe o caminho definido no ponto de acesso do EFS. Se um ponto de acesso for usado, a criptografia em trânsito deverá ser habilitada em `EFSVolumeConfiguration`. Para mais informações, consulte Trabalhando com Pontos de Acesso Amazon EFS no Guia de Usuário Amazon Elastic File System.

`iam`

Tipo: String

Valores válidos: `ENABLED` | `DISABLED`

Obrigatório: Não

Especifica se é necessário ou não usar o perfil do IAM para a tarefa do Amazon ECS estabelecida em uma definição de tarefa ao montar o sistema de arquivos do Amazon EFS. Em caso positivo, a criptografia de trânsito deve estar habilitada em `EFSVolumeConfiguration`. Se o parâmetro for omitido, o valor padrão `DISABLED` será usado. Para obter mais informações, consulte Funções do IAM para tarefas.

Configurar os sistemas de arquivos do Amazon EFS para o Amazon ECS usando o console

Saiba como usar os sistemas de arquivos do Amazon Elastic File System (Amazon EFS) com o Amazon ECS.

Etapa 1: criar um cluster do Amazon ECS

Use as etapas a seguir para criar um cluster do Amazon ECS.

Para criar um novo cluster (console do Amazon ECS)

Antes de começar, atribua a permissão apropriada do IAM. Para obter mais informações, consulte the section called “Exemplos de clusters do Amazon ECS”.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, selecione a Região a ser usada.
3. No painel de navegação, escolha Clusters.
4. Na página Clusters, escolha Create Cluster (Criar cluster).
5. Em Configuração do cluster, em Nome do cluster, insira `EFS-tutorial` para o nome do cluster.
6. (Opcional) Para alterar a VPC e as sub-redes onde suas tarefas e serviços são iniciados, em Networking (Redes), execute qualquer uma das operações a seguir:
   - Para remover uma sub-rede, em Subnets (Sub-redes), escolha X para cada sub-rede que você deseja remover.
   - Para mudar para uma VPC diferente da VPC default (padrão), em VPC, escolha uma VPC existente e, depois, Subnets (Sub-redes), e selecione cada sub-rede.
7. Para adicionar instâncias do Amazon EC2 ao seu cluster, expanda Infraestrutura e selecione Instâncias do Amazon EC2. Em seguida, configure o grupo do Auto Scaling que atua como o provedor de capacidade:
   - Para criar um grupo do Auto Scaling, a partir de Auto Scaling group (ASG) (Grupo do Auto Scaling (ASG)), selecione Create new group (Criar novo grupo) e, em seguida, forneça os seguintes detalhes sobre o grupo:
     - Em Sistema/arquitetura operacional, escolha Amazon Linux 2.
     - Em EC2 instance type (Tipo de instância do EC2), escolha `t2.micro`.

Em SSH key pair (Par de chaves de SSH), escolha o par que prova sua identidade quando você se conecta à instância.

- Em Capacidade, insira 1.

8. Escolha Criar.

Etapa 2: criar um grupo de segurança para instâncias do Amazon EC2 e o sistema de arquivos Amazon EFS

Nesta etapa, você criará um grupo de segurança para suas instâncias do Amazon EC2 que permite tráfego de rede de entrada na porta 80 e seu sistema de arquivos Amazon EFS que permite o acesso de entrada das suas instâncias de contêiner.

Crie um grupo de segurança para suas instâncias do Amazon EC2 com as opções a seguir:

- Nome do grupo de segurança: um nome exclusivo para seu grupo de segurança.
- VP: a VPC identificada anteriormente para o cluster.
- Regra de entrada
  - Tipo: HTTP
  - Origem: 0.0.0.0/0.

Crie um grupo de segurança seu sistema de arquivos Amazon EFS com as opções a seguir:

- Nome do grupo de segurança: um nome exclusivo para seu grupo de segurança. Por exemplo, `.EFS-access-for-sg-`*`dc025fa2`*
- VP: a VPC identificada anteriormente para o cluster.
- Regra de entrada
  - Tipo: NFS
  - Origem: Personalizada com o ID do grupo de segurança que você criou para suas instâncias.

Para obter informações sobre como criar um grupo de segurança, consulte Criar um grupo de segurança para sua instância do Amazon EC2 no Guia do usuário do Amazon EC2.

Etapa 3: criar um sistema de arquivos do Amazon EFS

Nesta etapa, você cria um sistema de arquivos do Amazon EFS.

Para criar um sistema de arquivos do Amazon EFS para tarefas do Amazon ECS.

1. Abra o console do Amazon Elastic File System em https://console.aws.amazon.com/efs/.
2. Escolha Create file system (Criar sistema de arquivos).
3. Insira um nome para seu sistema de arquivos e escolha a VPC na qual suas instâncias de contêiner estão hospedadas. Por padrão, cada sub-rede na VPC especificada recebe um destino de montagem que usa o grupo de segurança padrão para essa VPC. Em seguida, escolha Personalizar.

   > **Note**
   >
   > Este tutorial pressupõe que o sistema de arquivos do Amazon EFS, o cluster do Amazon ECS, as instâncias de contêiner e as tarefas estejam na mesma VPC. Para obter mais informações sobre a montagem de um sistema de arquivos em uma VPC diferente, consulte Walkthrough: Mount a file system from a different VPC no Guia do usuário do Amazon EFS.

4. Na página Configurações do sistema de arquivos, defina as configurações opcionais e, em Configurações de performance, escolha o modo de throughput Intermitente para seu sistema de arquivos. Depois de definir as configurações, selecione Avançar.

   a. (Opcional) Adicione tags ao sistema de arquivos. Por exemplo, você pode especificar um nome exclusivo para o sistema de arquivos inserindo esse nome na coluna Value (Valor) ao lado da chave Name (Nome).

   b. (Opcional) Habilite o gerenciamento do ciclo de vida para economizar dinheiro em armazenamento acessado com pouca frequência. Para obter mais informações, consulte Gerenciamento de ciclo de vida do EFS no Guia do usuário do Amazon Elastic File System.

   c. (Opcional) Habilitar criptografia. Marque a caixa de seleção para habilitar a criptografia do sistema de arquivos do Amazon EFS em repouso.

5. Na página Acesso à rede, em Montar destinos, substitua a configuração do grupo de segurança existente para cada zona de disponibilidade pelo grupo de segurança que você criou para o sistema de arquivos Etapa 2: criar um grupo de segurança para instâncias do Amazon EC2 e o sistema de arquivos Amazon EFS e escolha Avançar.
6. Não é preciso configurar a Política do sistema de arquivos para este tutorial, de modo que é possível pular a seção escolhendo Avançar.
7. Revise as opções do sistema de arquivos e escolha Criar para concluir o processo.

8. Na tela Sistemas de arquivo, registre o ID do sistema de arquivos. Na próxima etapa, você fará referência a esse valor na definição de tarefa do Amazon ECS.

Etapa 4: adicionar conteúdo ao sistema de arquivos do Amazon EFS

Nesta etapa, você monta o sistema de arquivos do Amazon EFS em uma instância do Amazon EC2 e adiciona conteúdo a ela. Esta etapa é para fins de teste neste tutorial, a fim de ilustrar a natureza persistente dos dados. Ao usar esse recurso, você normalmente teria sua aplicação ou outro método de gravação de dados no seu sistema de arquivos do Amazon EFS.

Para criar uma instância do Amazon EC2 e montar o sistema de arquivos do Amazon EFS

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. Escolha Executar instância.
3. Em Imagens de aplicações e sistemas operacionais (imagem de máquina da Amazon), selecione a AMI do Amazon Linux 2 (HVM).
4. Em Tipo de instância, mantenha o tipo de instância padrão, `t2.micro`.
5. Em Par de chaves (login), selecione um par de chaves para acesso por SSH à instância.
6. Em Configurações de rede, selecione a VPC especificada para o sistema de arquivos Amazon EFS e o cluster do Amazon ECS. Selecione uma sub-rede e o grupo de segurança da instância criado em Etapa 2: criar um grupo de segurança para instâncias do Amazon EC2 e o sistema de arquivos Amazon EFS. Configure o grupo de segurança da instância. Verifique se Atribuir automaticamente IP público está habilitado.
7. Em Configurar armazenamento, escolha o botão Editar para sistemas de arquivos e, em seguida, escolha EFS. Selecione o sistema de arquivos que você criou em Etapa 3: criar um sistema de arquivos do Amazon EFS. Opcionalmente, é possível alterar o ponto de montagem ou deixar o valor padrão.

   > **Important**
   >
   > É necessário selecionar uma sub-rede antes de adicionar um sistema de arquivos à instância.

8. Desmarque Criar e anexar grupos de segurança automaticamente. Deixe a outra caixa de seleção marcada. Selecione Add shared file system (Adicionar sistema de arquivos compartilhado).

9. Em Advanced Details (Detalhes avançados), certifique-se de que o script de dados do usuário seja preenchido automaticamente com as etapas de montagem do sistema de arquivos do Amazon EFS.
10. Em Resumo, verifique se o Número de instâncias é 1. Escolha Iniciar instância.
11. Na página Iniciar uma instância escolha Visualizar todas instâncias para ver o status das suas instâncias. Inicialmente, o Estado da instância é `PENDING`. Depois que o estado mudar para `RUNNING` e a instância passar por todas as verificações de status, a instância estará pronta para uso.

Agora, você se conecta à instância do Amazon EC2 e adiciona conteúdo ao sistema de arquivos do Amazon EFS.

Para conectar-se à instância do Amazon EC2 e adicionar conteúdo ao sistema de arquivos do Amazon EFS

1. SSH para a instância do Amazon EC2 que você criou. Para obter mais informações, consulte Conectar-se à instância do Linux usando SSH no Guia do usuário do Amazon EC2.
2. Na janela do terminal, execute o comando df -T para verificar se o sistema de arquivos Amazon EFS está montado. Na saída a seguir, destacamos a montagem do sistema de arquivos do Amazon EFS.

```
$ df -T
Filesystem     Type            1K-blocks    Used        Available Use% Mounted on
devtmpfs       devtmpfs           485468       0           485468   0% /dev
tmpfs          tmpfs              503480       0           503480   0% /dev/shm
tmpfs          tmpfs              503480     424           503056   1% /run
tmpfs          tmpfs              503480       0           503480   0% /sys/fs/
cgroup
/dev/xvda1     xfs               8376300 1310952          7065348  16% /
127.0.0.1:/    nfs4     9007199254739968       0 9007199254739968   0% /mnt/efs/fs1
tmpfs          tmpfs              100700       0           100700   0% /run/
user/1000
```

3. Navegue até o diretório no qual o sistema de arquivos do Amazon EFS está montado. No exemplo acima, ele é `/mnt/efs/fs1`.
4. Crie um arquivo chamado `index.html` com o seguinte conteúdo:

```
<html>
```

```
    <body>
        <h1>It Works!</h1>
        <p>You are using an Amazon EFS file system for persistent container
 storage.</p>
    </body>
</html>
```

Etapa 5: criar uma definição de tarefa

A definição de tarefa a seguir cria um volume de dados chamado `efs-html`. O contêiner `nginx` monta o volume de dados do host na raiz de NGINX, `/usr/share/nginx/html`.

Para criar uma nova definição de tarefa usando o console do Amazon ECS

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Task definitions (Definições de tarefa).
3. Escolha Create new task definition (Criar nova definição de tarefa), Create new task definition with JSON (Criar nova definição de tarefa com JSON).
4. Na caixa do editor JSON, copie e cole o seguinte texto JSON, substituindo `fileSystemId` pelo ID do sistema de arquivos do Amazon EFS.

```
{
    "containerDefinitions": [
        {
            "memory": 128,
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "mountPoints": [
                {
                    "containerPath": "/usr/share/nginx/html",
                    "sourceVolume": "efs-html"
                }
            ],
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest"
```

```
        }
    ],
    "volumes": [
        {
            "name": "efs-html",
            "efsVolumeConfiguration": {
                "fileSystemId": "fs-1324abcd",
                "transitEncryption": "ENABLED"
            }
        }
    ],
    "family": "efs-tutorial",
    "executionRoleArn":"arn:aws:iam::111122223333:role/ecsTaskExecutionRole"
}
```

> **Note**
>
> O perfil do IAM de execução de tarefas do Amazon ECS não exige nenhuma permissão específica relacionada ao Amazon EFS para montar um sistema de arquivos do Amazon EFS. Por padrão, se a política baseada em recursos do Amazon EFS não existir, o acesso será concedido a todas as entidades principais (*) na criação do sistema de arquivos.
>
> O perfil da tarefa do Amazon ECS será necessário apenas se a “autorização do IAM para EFS” estiver habilitada na definição da tarefa do Amazon ECS. Quando habilitada, a identidade do perfil da tarefa deverá ter o acesso permitido ao sistema de arquivos do Amazon EFS na política baseada em recursos do Amazon EFS, e o acesso anônimo deverá ser desabilitado.

5. Escolha Criar.

Etapa 6: executar uma tarefa e visualizar os resultados

Agora que o sistema de arquivos do Amazon EFS foi criado e há conteúdo da Web para o contêiner NGINX atender, você pode executar uma tarefa usando a definição de tarefa criada. Os servidores web do NGINX fornecem sua página HTML simples. Se você atualizar o conteúdo no sistema de arquivos do Amazon EFS, essas alterações serão propagadas para todos os contêineres que também tenham montado este sistema de arquivos.

A tarefa é executada na sub-rede que você definiu para o cluster.

Para executar uma tarefa e visualizar os resultados usando o console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, selecione o cluster que contém a tarefa autônoma.

   Determine o recurso no qual você inicia o serviço.

Para iniciar um serviço em	Etapas
Clusters	a. Na página Clusters, selecione o cluster no qual o serviço será criado. b. Na guia Tasks (Tarefas), escolha Run new task (Executar nova tarefa).
Tipo de inicialização	a. Na página Task (Tarefa), escolha a definição de tarefa. b. Se houver mais de uma revisão, selecione a revisão. c. Escolha Create (Criar), Run task (Executar tarefa).

3. (Opcional) Escolha como a tarefa programada é distribuída em toda a infraestrutura do cluster. Expanda Compute configuration (Configuração de computação) e realize estas ações:

Método de distribuição	Etapas
Tipo de inicialização	a. Na seção Compute options (Opções de computação), selecione Launch type (Tipo de inicialização).

Método de distribuição	Etapas
	b. Em Tipo de inicialização, selecione EC2.

4. Em Application type (Tipo de aplicação), escolha Task (Tarefa).
5. Em Definição de tarefa, escolha a definição de tarefa `efs-tutorial` que você criou anteriormente.
6. Em Tarefas desejadas, insira 1.
7. Escolha Criar.
8. Na página Cluster, escolha Infraestrutura.
9. Em Instâncias de contêiner, escolha a instância de contêiner com a qual deseja estabelecer conexão.
10. Na página Instância de contêiner, Em Redes, registre o IP público para sua instância.
11. Abra um navegador e insira o endereço IP público. Você deve ver a mensagem a seguir:

```
It works!
You are using an Amazon EFS file system for persistent container storage.
```

> **Note**
>
> Se você não vir a mensagem, verifique se o grupo de segurança para suas instâncias de contêiner permite o tráfego de rede de entrada na porta 80 e se o grupo de segurança do seu sistema de arquivos permite acesso de entrada a partir da instância de contêiner.

Uso de volumes do FSx para Windows File Server com Amazon ECS

O FSx para Windows File Server fornece servidores de arquivos Windows totalmente gerenciados, baseados em um sistema de arquivos Windows. Ao usar o FSx for Windows File Server junto com o ECS, você pode provisionar tarefas do Windows com armazenamento de arquivos persistente, distribuído, compartilhado e estático. Para obter mais informações, consulte o O que é FSx for Windows File Server?.

> **Note**
>
> As instâncias do EC2 que usam a AMI completa do Windows Server 2016 otimizada para o Amazon ECS não oferecem suporte a volumes de tarefas do ECS do FSx for Windows File Server.
>
> Não é possível usar volumes do FSx para Windows File Server em contêineres do Windows na configuração do Fargate. Em vez disso, você pode modificar os contêineres para montá-los no startup.

É possível usar o FSx for Windows File Server para implantar workloads do Windows que exigem acesso ao armazenamento externo compartilhado, ao armazenamento regional de alta disponibilidade ou ao armazenamento de alta throughput. É possível montar um ou mais volumes do sistema de arquivos do FSx para Windows File Server em um contêiner do Amazon ECS executado em uma instância do Amazon ECS. É possível compartilhar volumes do sistema de arquivos do FSx para Windows File Server entre vários contêineres do Amazon ECS em uma única tarefa do Amazon ECS.

Para permitir o uso do FSx para Windows File Server com o ECS, inclua o ID do sistema de arquivos FSx para Windows File Server e as informações relacionadas em uma definição de tarefa. Isso está no exemplo a seguir de trecho JSON de definição de tarefa. Antes de criar e executar uma definição de tarefa, você precisa do seguinte.

- Uma instância Windows do EC2 do ECS que é unida a um domínio válido. Ela pode ser hospedada por um AWS Directory Service for Microsoft Active Directory, Active Directory on-premises ou Active Directory auto-hospedado no Amazon EC2.
- Um segredo do AWS Secrets Manager ou um parâmetro do Systems Manager que contém as credenciais usadas para associação ao domínio do Active Directory e anexar o sistema de arquivos do FSx para Windows File Server. Os valores de credencial são as credenciais de nome e senha que você inseriu ao criar o diretório ativo.

Para ver um tutorial relacionado, consulte Saiba como configurar sistemas de arquivos do FSx para Windows File Server para o Amazon ECS.

Considerações

Considere o seguinte ao usar volumes do FSx for Windows File Server.

- O FSx for Windows File Server com Amazon ECS só oferece suporte a instâncias do Windows do Amazon EC2. As instâncias do Linux do Amazon EC2 não são compatíveis.
- O FSx for Windows File Server com o Amazon ECS não oferece suporte ao AWS Fargate.
- O FSx for Windows File Server com o Amazon ECS, com modo de rede `awsvpc`, exige a versão `1.54.0` ou posterior do agente de contêiner.
- O número máximo de letras de unidade que podem ser usadas para uma tarefa do Amazon ECS é 23. Cada tarefa com um volume do FSx for Windows File Server obtém uma letra de unidade atribuída a ela.
- Por padrão, o tempo de limpeza de recursos de tarefas é de três horas após o término da tarefa. Mesmo que nenhuma tarefa o use, um mapeamento de arquivos criado por uma tarefa persiste por três horas. O tempo de limpeza padrão pode ser configurado usando a variável de ambiente `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION` do Amazon ECS. Para obter mais informações, consulte Configuração do agente de contêiner do Amazon ECS.
- Normalmente, as tarefas somente são executadas na mesma VPC do sistema de arquivos do FSx for Windows File Server. No entanto, é possível ter suporte entre VPCs se houver uma conectividade de rede estabelecida entre a VPC do cluster do Amazon ECS e o sistema de arquivos do FSx for Windows File Server por meio de emparelhamento da VPC.
- Controle o acesso a um sistema de arquivos do FSx for Windows File Server no nível da rede configurando os grupos de segurança da VPC. Somente as tarefas hospedadas em instâncias do EC2, associadas ao domínio do Active Directory com grupos de segurança do Active Directory configurados corretamente, podem acessar o compartilhamento de arquivos do FSx para Windows File Server. Se os grupos de segurança estiverem configurados incorretamente, o Amazon ECS falhará na inicialização da tarefa e apresentará a seguinte mensagem de erro: `unable to mount file system` *`fs-id`*”.
- O FSx for Windows File Server é integrado ao AWS Identity and Access Management (IAM) para controlar as ações que os usuários e grupos do IAM podem executar em recursos específicos do FSx for Windows File Server. Com autorização do cliente, os clientes podem definir funções do IAM que permitem ou negam acesso a sistemas de arquivos específicos do FSx for Windows File Server, opcionalmente exigir acesso somente leitura e, opcionalmente, permitir ou impedir acesso raiz ao sistema de arquivos do cliente. Para obter mais informações, consulte Segurança no Guia do usuário do Amazon FSx for Windows.

Práticas recomendadas ao usar o FSx para Windows File Server com o Amazon ECS

Anote as práticas recomendadas a seguir ao usar o FSx para Windows File Server com o Amazon ECS.

Segurança e controles de acesso para o FSx para Windows File Server

O FSx para Windows File Server oferece os recursos de controle de acesso a seguir, usados para garantir que os dados armazenados em um sistema de arquivos do FSx para Windows File Server estejam seguros e acessíveis somente pelas aplicações que precisam deles.

Criptografia de dados para volumes do FSx para Windows File Server

O FSx para Windows File Server oferece suporte a duas formas de criptografia para sistemas de arquivos. São elas a criptografia de dados em trânsito e a criptografia em repouso. A criptografia de dados em trânsito é compatível com compartilhamentos de arquivos mapeados em uma instância de contêiner que seja compatível com o protocolo SMB 3.0 ou mais recente. A criptografia de dados em repouso é habilitada automaticamente ao criar um sistema de arquivos do Amazon FSx. O Amazon FSx criptografa automaticamente os dados em trânsito usando a criptografia SMB à medida que você acessa seu sistema de arquivos, sem a necessidade de modificar suas aplicações. Para obter mais informações, consulte Data encryption in Amazon FSx no Guia do usuário do Amazon FSx para Windows File Server.

Uso de ACLs do Windows para controle de acesso em nível de pasta

A instância do Windows para Amazon EC2 acessa os compartilhamentos de arquivos do Amazon FSx usando as credenciais do Active Directory. Ela usa listas de controle de acesso (ACLs) padrão do Windows para controle de acesso refinado em nível de arquivo e pasta. Você pode criar várias credenciais, cada uma para uma pasta específica dentro do compartilhamento que mapeia para uma tarefa específica.

No exemplo a seguir, a tarefa tem acesso à pasta `App01` usando uma credencial salva no Secrets Manager. Seu nome do recurso da Amazon (ARN) é `1234`.

```
"rootDirectory": "\\path\\to\\my\\data\App01",
"credentialsParameter": "arn-1234",
"domain": "corp.fullyqualified.com",
```

Em outro exemplo, uma tarefa tem acesso à pasta `App02` usando uma credencial salva no Secrets Manager. Seu ARN é `6789`.

```
"rootDirectory": "\\path\\to\\my\\data\App02",
"credentialsParameter": "arn-6789",
"domain": "corp.fullyqualified.com",
```

Especificar um sistema de arquivos do FSx para Windows File Server em uma definição de tarefa do Amazon ECS

Para usar volumes do sistema de arquivos do FSx for Windows File Server para seus contêineres, especifique as configurações de volume e ponto de montagem na definição de tarefa. O trecho JSON de definição de tarefa a seguir mostra a sintaxe para os objetos `volumes` e `mountPoints` de um contêiner.

```
{
    "containerDefinitions": [
        {
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "portMappings": [],
            "command": ["New-Item -Path C:\\fsx-windows-dir\\index.html -ItemType file
 -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>It Works!</h2> <p>You are using Amazon
 FSx for Windows File Server file system for persistent container storage.</p>' -
Force"],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": false,
            "name": "container1",
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
                }
            ]
        },
        {
            "entryPoint": [
```

```
                "powershell",
                "-Command"
            ],
            "portMappings": [
                {
                    "hostPort": 443,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "command": ["Remove-Item -Recurse C:\\inetpub\\wwwroot\\* -Force; Start-
Sleep -Seconds 120; Move-Item -Path C:\\fsx-windows-dir\\index.html -Destination C:\
\inetpub\\wwwroot\\index.html -Force; C:\\ServiceMonitor.exe w3svc"],
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
                }
            ],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": true,
            "name": "container2"
        }
    ],
    "family": "fsx-windows",
    "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
    "volumes": [
        {
            "name": "fsx-windows-dir",
            "fsxWindowsFileServerVolumeConfiguration": {
                "fileSystemId": "fs-0eeb5730b2EXAMPLE",
                "authorizationConfig": {
                    "domain": "example.com",
                    "credentialsParameter": "arn:arn-1234"
                },
                "rootDirectory": "share"
            }
        }
    ]
```

```
}
```

`FSxWindowsFileServerVolumeConfiguration`

Tipo: Objeto

Obrigatório: Não

Esse parâmetro é especificado quando você está usando o sistema de arquivos do FSx for Windows File Server para armazenamento de tarefas.

`fileSystemId`

Tipo: string

Obrigatório: Sim

O ID do sistema de arquivos do FSx for Windows File Server a ser usado.

`rootDirectory`

Tipo: string

Obrigatório: Sim

O diretório no sistema de arquivos do FSx for Windows File Server que deve ser montado como o diretório raiz dentro do host.

`authorizationConfig`

`credentialsParameter`

Tipo: string

Obrigatório: Sim

As opções de credencial de autorização:

- Nome do recurso da Amazon (ARN) de um segredo do Secrets Manager.
- Nome do recurso da Amazon (ARN) de um parâmetro do Systems Manager.

`domain`

Tipo: string

Obrigatório: Sim

Um nome de domínio totalmente qualificado hospedado por um diretório do AWS Directory Service for Microsoft Active Directory (AWS Managed Microsoft AD) ou por um Active Directory do EC2 auto-hospedado.

Métodos para o armazenamento de credenciais de volume do FSx para Windows File Server

Existem dois métodos diferentes de armazenamento de credenciais para uso com o parâmetro de credenciais.

- Segredo do AWS Secrets Manager

  Essa credencial pode ser criada no console do AWS Secrets Manager usando a categoria Other type of secret (Outro tipo de segredo). Você adiciona uma linha para cada par chave-valor, nome de usuário/administrador e senha/*senha*.

- Parâmetro do Systems Manager

  Essa credencial pode ser criada no console de parâmetros do Systems Manager inserindo texto no formulário mostrado no trecho do código de exemplo a seguir.

```
{
  "username": "admin",
  "password": "password"
}
```

O `credentialsParameter` no parâmetro `FSxWindowsFileServerVolumeConfiguration` da definição de tarefa mantém o ARN do segredo ou o ARN do parâmetro do Systems Manager. Para obter mais informações, consulte O que é o AWS Secrets Manager no Guia do usuário do Secrets Manager e Systems Manager Parameter Store no Guia do usuário do Systems Manager.

Saiba como configurar sistemas de arquivos do FSx para Windows File Server para o Amazon ECS

Saiba como iniciar uma instância do Windows otimizada para o Amazon ECS que hospeda um sistema de arquivos do FSx para Windows File Server e contêineres que podem acessar o sistema de arquivos. Para fazer isso, você primeiro criará um Microsoft Active Directory gerenciado pela AWS do AWS Directory Service. Em seguida, você cria um sistema de arquivos do FSx para Windows File Server e um cluster com uma instância do Amazon EC2 e uma definição de tarefa. Configure

a definição de tarefa para os contêineres usarem o sistema de arquivos do FSx for Windows File Server. Finalmente, teste o sistema de arquivos.

Entre 20 e 45 minutos são necessários toda vez que você inicia ou exclui o Active Directory ou o sistema de arquivos do FSx for Windows File Server. Esteja preparado para reservar pelo menos 90 minutos para concluir o tutorial ou conclua o tutorial em algumas sessões.

Pré-requisitos para o tutorial

- Um usuário administrativo. Consulte Configuração para usar o Amazon ECS.
- (Opcional) Um par de chaves PEM para se conectar à instância do Windows do EC2 por acesso RDP. Para obter informações sobre como criar pares de chaves, consulte Pares de chaves do Amazon EC2 e instâncias do Amazon EC2 no Guia do usuário do Amazon EC2.
- Uma VPC com pelo menos uma sub-rede pública e uma privada e um grupo de segurança. É possível usar a VPC padrão. Você não precisa de um gateway ou dispositivo NAT. O AWS Directory Service não oferece suporte à conversão de endereços de rede (NAT) com o Active Directory. Para que isso funcione, o Active Directory, o sistema de arquivos do FSx for Windows File Server, o cluster do ECS e a instância do EC2 devem estar localizados na VPC. Para obter mais informações sobre VPCs e Active Directories, consulte Criar uma VPC e Pré-requisitos para criar um Microsoft AD gerenciado pela AWS.
- As permissões do IAM ecsInstanceRole e ecsTaskExecutionRole estão associadas à sua conta. Esses perfis vinculados ao serviço permitem que os serviços façam chamadas de API e acessem contêineres, segredos, diretórios e servidores de arquivos em seu nome.

Etapa 1: criar funções de acesso do IAM

Crie um cluster com o AWS Management Console

1. Consulte Função do IAM de instância de contêiner do Amazon ECS para verificar se você tem uma ecsInstanceRole e ver como será possível criar uma, caso não tenha.
2. Recomendamos que as políticas de função sejam personalizadas para permissões mínimas em um ambiente de produção real. Com a finalidade de trabalhar neste tutorial, verifique se a política gerenciada pela AWS a seguir está anexada a ecsInstanceRole. Anexe a política, caso ela ainda não esteja anexada.
   - AmazonEC2ContainerServiceforEC2Role
   - AmazonSSMManagedInstanceCore

- AmazonSSMDirectoryServiceAccess

Para anexar políticas gerenciadas pela AWS

a. Abra o console do IAM.
b. No painel de navegação, selecione Perfis.
c. Escolha um perfil gerenciado da AWS.
d. Escolha Permissões, Anexar políticas.
e. Para restringir as políticas disponíveis a serem anexadas, use Filter.
f. Selecione a política apropriada e escolha Attach policy (Anexar política).

3. Consulte Função do IAM de execução de tarefas do Amazon ECS para verificar se você tem uma ecsTaskExecutionRole e ver como será possível criar uma, caso não tenha.

   Recomendamos que as políticas de função sejam personalizadas para permissões mínimas em um ambiente de produção real. Com a finalidade de trabalhar neste tutorial, verifique se as políticas gerenciadas pela AWS a seguir estão anexadas a ecsTaskExecutionRole Anexe as políticas, caso elas ainda não estejam anexadas. Use o procedimento indicado na seção anterior para anexar as políticas gerenciadas pela AWS.

   - SecretsManagerReadWrite
   - AmazonFSxReadOnlyAccess
   - AmazonSSMReadOnlyAccess
   - AmazonECSTaskExecutionRolePolicy

Etapa 2: criar o Windows Active Directory (AD)

1. Siga as etapas descritas em Criar seu Microsoft AD gerenciado pela AWS no Guia de administração do AWS Directory Service. Use a VPC que você designou para este tutorial. Na etapa 3 de Criar seu Microsoft AD gerenciado pela AWS, salve o nome do usuário e a senha de administrador para uso em uma etapa posterior. Além disso, anote o nome de DNS totalmente qualificado para etapas futuras. É possível concluir a etapa a seguir enquanto o Active Directory está sendo criado.
2. Crie um segredo do AWS Secrets Manager a ser usado nas etapas a seguir. Para obter mais informações, consulte Conceitos básicos do Secrets Manager no Guia do usuário do AWS Secrets Manager.

a. Abra o console do Secrets Manager.

b. Clique em Store a new secret (Armazenar um novo segredo).

c. Selecione Other type of secrets (Outro tipo de segredos).

d. Em Secret key/value (Chave/valor secreto), na primeira linha, crie uma chave **`username`** com valor **`admin`**. Clique em + Add row (+ Adicionar linha).

e. Na nova linha, crie uma chave **`password`**. Em valor, digite a senha inserida na Etapa 3 de Criar seu diretório do AD gerenciado pela AWS.

f. Clique no botão Next (Avançar).

g. Forneça um nome e uma descrição do segredo. Clique em Next.

h. Clique em Next. Clique em Store (Armazenar).

i. Na lista da página Segredos, clique no segredo que você acabou de criar.

j. Salve o ARN do novo segredo para uso nas próximas etapas,

k. É possível ir para a próxima etapa enquanto seu Active Directory está sendo criado.

Etapa 3: verificar e atualizar o grupo de segurança

Nesta etapa, você verificará e atualizará as regras do grupo de segurança que você estiver usando. Para isso, você pode usar o grupo de segurança padrão que foi criado para a VPC.

Verifique e atualize o grupo de segurança.

Você precisa criar ou editar o grupo de segurança para enviar dados de e para as portas, que são descritas em Grupos de segurança da Amazon VPC no Guia do usuário do FSx for Windows File Server. É possível fazer isso criando a regra de entrada do grupo de segurança mostrada na primeira linha da tabela de regras de entrada a seguir. Essa regra permite tráfego de entrada de interfaces de rede (e das instâncias associadas) atribuídas ao grupo de segurança. Todos os recursos de nuvem criados estão dentro da mesma VPC e anexados ao mesmo grupo de segurança. Portanto, essa regra permite o tráfego entre o sistema de arquivos do FSx for Windows File Server, o Active Directory e a instância do ECS conforme necessário. As outras regras de entrada permitem que o tráfego atenda ao site e ao acesso RDP para conexão à instância do ECS.

A tabela a seguir mostra quais regras de entrada do grupo de segurança são necessárias para este tutorial.

Tipo	Protocolo	Intervalo de portas	Origem
Todo o tráfego	Tudo	Todos	*sg-securitygroup*
HTTPS	TCP	443	0.0.0.0/0
RDP	TCP	3389	endereço IP do seu laptop

A tabela a seguir mostra quais regras de saída do grupo de segurança são necessárias para este tutorial.

Tipo	Protocolo	Intervalo de portas	Destino
Todo o tráfego	Tudo	Tudo	0.0.0.0/0

1. Abra o Console do EC2 e selecione Security groups (Grupos de segurança) no menu à esquerda.
2. Na lista de grupos de segurança exibida, marque a caixa de seleção à esquerda do grupo de segurança que você está usando para este tutorial.

   Os detalhes do security group são exibidos.
3. Edite as regras de entrada e de saída selecionando as guias Inbound rules (Regras de entrada) ou Outbound rules (Regras de saída) e escolhendo os botões Edit inbound rules (Editar regras de entrada) ou Edit outbound rules (Editar regras de saída). Edite as regras para corresponder às regras exibidas nas tabelas anteriores. Depois de criar a instância do EC2 posteriormente neste tutorial, edite a origem RDP da regra de entrada com o endereço IP público da instância do EC2, conforme descrito em Conectar-se à sua instância do Windows usando RDP no Guia do usuário do Amazon EC2.

Etapa 4: criar um sistema de arquivos do FSx for Windows File Server

Depois que o grupo de segurança for verificado e atualizado e o Active Directory for criado e estiver no status ativo, crie o sistema de arquivos do FSx for Windows File Server na mesma VPC do Active Directory. Use as etapas a seguir para criar um sistema de arquivos do FSx for Windows File Server para as tarefas do Windows.

Crie seu primeiro sistema de arquivos.

1. Abra o console do Amazon FSx.
2. No painel, escolha Create file system (Criar sistema de arquivos) para iniciar o assistente de criação de sistemas de arquivos.
3. Na página Selecionar tipo de sistema de arquivos, escolha FSx for Windows File Server e, em seguida, selecione Next (Avançar). A página Criar sistema de arquivos é exibida.
4. Na seção Detalhes do sistema de arquivos, forneça um nome para o sistema de arquivos. Nomear seus sistemas de arquivos torna mais fácil encontrar e gerenciar seus sistemas. Podem ser usados até 256 caracteres Unicode. Os caracteres permitidos são letras, números, espaços e os caracteres especiais sinal de mais (+). sinal de menos (-), sinal de igual (=), ponto (.), sublinhado (_), dois-pontos (:) e barra (/).
5. Em Deployment type (Tipo de implantação), escolha Single-AZ para implantar um sistema de arquivos implantado em uma única zona de disponibilidade. Single-AZ 2 é a geração mais recente de sistemas de arquivos de zona de disponibilidade única. É compatível com armazenamento SSD e HDD.
6. Em Storage type (Tipo de armazenamento), escolha HDD.
7. Em Storage capacity (Capacidade de armazenamento), insira a capacidade mínima de armazenamento.
8. Mantenha a Capacidade de Throughput na configuração padrão.
9. Na seção Rede e segurança, escolha a mesma Amazon VPC que você escolheu para o diretório do AWS Directory Service.
10. Em VPC Security Groups (Grupos de segurança da VPC), escolha o grupo de segurança que você verificou na Etapa 3: verificar e atualizar o grupo de segurança.
11. Em Windows authentication (Autenticação do Windows), escolha AWS Managed Microsoft Active Directory (Microsoft Active Directory gerenciado pela) e, depois, escolha o diretório do AWS Directory Service na lista.

12. Em Encryption (Criptografia), mantenha a configuração padrão da Encryption key (Chave de criptografia) como aws/fsx (default).
13. Mantenha as configurações padrão para Preferências de manutenção.
14. Clique no botão Next (Avançar).
15. Verifique a configuração do sistema de arquivos mostrada na página Criar sistema de arquivos. Para sua referência, anote quais configurações do sistema de arquivos você pode modificar após a criação do sistema de arquivos. Escolha Create file system (Criar sistema de arquivos).
16. Anote o ID do sistema de arquivos. Você precisará usá-lo em uma etapa posterior.

    É possível seguir para as próximas etapas para criar um cluster e uma instância do EC2 enquanto o sistema de arquivos do FSX for Windows File Server está sendo criado.

Etapa 5: criar um cluster do Amazon ECS

Criar um cluster usando o console do Amazon ECS

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, selecione a Região a ser usada.
3. No painel de navegação, escolha Clusters.
4. Na página Clusters, escolha Create Cluster (Criar cluster).
5. Em Configuração do cluster), para Nome do cluster, insira windows-fsx-cluster.
6. Expanda Infraestrutura, desmarque AWS Fargate (tecnologia sem servidor) e selecione Instâncias do Amazon EC2.

    - Para criar um grupo do Auto Scaling, a partir de Auto Scaling group (ASG) (Grupo do Auto Scaling (ASG)), selecione Create new group (Criar novo grupo) e, em seguida, forneça os seguintes detalhes sobre o grupo:

        - Em Sistema operacional/Arquitetura, escolha Windows Server 2019 Core.
        - Em Tipo de instância do EC2, escolha t2.medium ou t2.micro.
7. Escolha Criar.

Etapa 6: criar uma instância do Amazon ECS otimizada para o Amazon EC2

Crie uma instância de contêiner do Windows do Amazon ECS.

Para criar uma instância do Amazon ECS

1. Use o comando `aws ssm get-parameters` para recuperar o nome da AMI para a região que hospeda sua VPC. Para obter mais informações, consulte Recuperar os metadados da AMI otimizada para Amazon ECS.
2. Use o console do Amazon EC2 para iniciar a instância.
   a. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
   b. Na barra de navegação, selecione a Região a ser usada.
   c. No Painel do EC2, escolha Launch Instance (Iniciar instância).
   d. Em Name (Nome), insira um nome exclusivo.
   e. Em Imagens de aplicações e SO (Amazon Machine Image), no campo pesquisar, insira o nome da AMI que você recuperou.
   f. Em Tipo de instância, escolha t2.medium ou t2.micro.
   g. Em Key pair (login) (Par de chaves [login]), escolha um par de chaves. Se você não especificar um par de chaves, você
   h. Em Configurações de rede, para VPC e sub-rede, escolha sua VPC e uma sub-rede pública.
   i. Em Network settings (Configurações de rede), em Security group (Grupo de segurança), selecione um grupo de segurança existente ou crie outro. Certifique-se de que o grupo de segurança escolhido tenha as regras de entrada e saída definidas em Pré-requisitos para o tutorial
   j. Em Network settings (Configurações de rede), em Auto-assign Public IPC (Atribuir IP público automaticamente), selecione Enable (Habilitar).
   k. Expanda Detalhes avançados, e em Associar diretório a domínio, selecione o ID do Active Directory que você criou. Esse domínio se junta ao AD quando a instância do EC2 é iniciada.
   l. Em Advanced details (Detalhes avançados), em IAM instance profile (Perfil de instância do IAM), escolha ecsInstanceRole.
   m. Configure a instância de contêiner do Amazon ECS com os dados de usuário a seguir. Em Advanced Details (Detalhes avançados), cole o seguinte script no campo User data (Dados do usuário), substituindo *cluster_name* pelo nome do cluster.

```
<powershell>
Initialize-ECSAgent -Cluster windows-fsx-cluster -EnableTaskIAMRole
```

```
</powershell>
```

n. Quando estiver pronto, selecione o campo de confirmação e, então, escolha Launch Instances.

o. Uma página de confirmação informa que sua instância está sendo executada. Selecione Visualizar instâncias para fechar a página de confirmação e voltar ao console.

3. Abra o console em https://console.aws.amazon.com/ecs/v2.
4. No painel de navegação, escolha Clusters e, em seguida, escolha windows-fsx-cluster.
5. Selecione a guia Infraestrutura e verifique se a sua instância foi registrada no cluster windows-fsx-cluster.

Etapa 7: registrar uma definição de tarefa do Windows

Para executar contêineres do Windows no cluster do Amazon ECS, você deve registrar uma definição de tarefa. O exemplo de definição de tarefa a seguir exibe uma página da Web simples. A tarefa inicia dois contêineres que têm acesso ao sistema de arquivos do FSx. O primeiro contêiner grava um arquivo HTML no sistema de arquivos. O segundo contêiner baixa o arquivo HTML do sistema de arquivos e serve a página da Web.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Task definitions (Definições de tarefa).
3. Escolha Create new task definition (Criar nova definição de tarefa), Create new task definition with JSON (Criar nova definição de tarefa com JSON).
4. NA caixa do editor JSON, substitua os valores do seu perfil de execução da tarefa e os detalhes sobre o sistema de arquivos FSx e, em seguida, escolha Salvar.

```
{
    "containerDefinitions": [
        {
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "portMappings": [],
            "command": ["New-Item -Path C:\\fsx-windows-dir\\index.html -ItemType
 file -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body
 {margin-top: 40px; background-color: #333;} </style> </head><body> <div
 style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1> <h2>It
```

```
 Works!</h2> <p>You are using Amazon FSx for Windows File Server file system for
 persistent container storage.</p>' -Force"],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": false,
            "name": "container1",
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
                }
            ]
        },
        {
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "portMappings": [
                {
                    "hostPort": 443,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "command": ["Remove-Item -Recurse C:\\inetpub\\wwwroot\\* -Force;
 Start-Sleep -Seconds 120; Move-Item -Path C:\\fsx-windows-dir\\index.html -
Destination C:\\inetpub\\wwwroot\\index.html -Force; C:\\ServiceMonitor.exe
 w3svc"],
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
                }
            ],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": true,
```

```
            "name": "container2"
        }
    ],
    "family": "fsx-windows",
    "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
    "volumes": [
        {
            "name": "fsx-windows-dir",
            "fsxWindowsFileServerVolumeConfiguration": {
                "fileSystemId": "fs-0eeb5730b2EXAMPLE",
                "authorizationConfig": {
                    "domain": "example.com",
                    "credentialsParameter": "arn:arn-1234"
                },
                "rootDirectory": "share"
            }
        }
    ]
}
```

Etapa 8: executar uma tarefa e visualizar os resultados

Antes de executar a tarefa, verifique se o status do sistema de arquivos do FSx for Windows File Server está Available (Disponível). Depois que ele estiver disponível, será possível executar uma tarefa usando a definição de tarefa criada. A tarefa começa ao criar contêineres que embaralham um arquivo HTML entre eles usando o sistema de arquivos. Após o embaralhamento, um servidor Web atende à página HTML simples.

> **Note**
>
> Talvez você não seja capaz de se conectar ao site a partir de uma VPN.

Execute uma tarefa e veja os resultados no console do Amazon ECS.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Clusters e, em seguida, escolha windows-fsx-cluster.
3. Escolha a guia Tarefas e, em seguida, escolha Executar nova tarefa.
4. Em Launch Type, selecione EC2.

5. Em Configuração de implantação, para Definição de tarefa, escolha fsx-windows e, em seguida, escolha Criar.
6. Quando o status da tarefa for EM EXECUÇÃO, escolha o ID da tarefa.
7. Em Contêineres, quando o status do contêiner1 for INTERROMPIDO, selecione contêiner2 para ver os detalhes do contêiner.
8. Em Detalhes de contêiner para o contêiner2, selecione Ligações de rede e clique no endereço IP externo associado ao contêiner. O navegador abrirá e exibirá a mensagem a seguir.

```
Amazon ECS Sample App
It Works!
You are using Amazon FSx for Windows File Server file system for persistent
 container storage.
```

> **Note**
>
> Pode demorar alguns minutos para que a mensagem seja exibida. Se você não vir a mensagem após alguns minutos, certifique-se de que as execuções não estejam ocorrendo em uma VPN e de que o grupo de segurança das instâncias de contêiner permita o tráfego HTTP da rede de entrada na porta 443.

Etapa 9: limpar

> **Note**
>
> Demora de 20 a 45 minutos para excluir o sistema de arquivos do FSx for Windows File Server ou o AD. Você deve aguardar até que as operações de exclusão do sistema de arquivos do FSx for Windows File Server sejam concluídas antes de iniciar as operações de exclusão do AD.

Exclua o sistema de arquivos do FSx for Windows File Server

1. Abra o console do Amazon FSx.
2. Escolha o botão de opção à esquerda do sistema de arquivos do FSx for Windows File Server que você acabou de criar.
3. Escolha Ações.

4. Selecione Delete file system (Excluir sistema de arquivos).

Excluir AD.

1. Abra o console de AWS Directory Service.
2. Escolha o botão de opção à esquerda do AD que você acabou de criar.
3. Escolha Ações.
4. Selecione Delete directory (Excluir diretório).

Excluir o cluster.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Clusters e, em seguida, escolha windows-fsx-cluster.
3. Escolha Delete Cluster (Excluir cluster).
4. Insira a frase e, em seguida, escolha Excluir.

Encerra a instância do EC2.

1. Abra o console do Amazon EC2.
2. No menu à esquerda, selecione Instances (Instâncias).
3. Marque a caixa de seleção à esquerda da instância do EC2 que você criou.
4. Clique no Estado da instância, Encerrar instância.

Excluir segredo.

1. Abra o console do Secrets Manager.
2. Selecione o segredo que você criou para esta demonstração.
3. Clique em Actions (Ações)
4. Selecione Delete secret Excluir segredo).

Uso de volumes do Docker com o Amazon ECS

Ao usar os volumes do Docker, o driver `local` integrado ou um driver de volume de terceiros poderão ser usados. Os volumes do Docker são gerenciados pelo Docker e um diretório é criado em `/var/lib/docker/volumes` na instância de contêiner que contém os dados de volume.

Para usar os volumes do Docker, especifique uma `dockerVolumeConfiguration` em sua definição de tarefa. Para obter mais informações, consulte Volumes na documentação do Docker.

Alguns casos de uso comuns para os volumes do Docker são os seguintes:

- Para fornecer volumes de dados persistentes para o uso com contêineres
- Para compartilhar um volume de dados definido em locais diferentes em contêineres diferentes na mesma instância de contêiner
- Para definir um volume de dados não persistente e vazio, além de montá-lo em vários contêineres na mesma tarefa
- Para fornecer um volume de dados para sua tarefa gerenciada por um driver de terceiros

Considerações sobre o uso de volumes do Docker

Considere o seguinte ao usar volumes do Docker:

- Só há suporte para os volumes do Docker com o uso do tipo de inicialização do EC2 ou instâncias externas.
- Os contêineres do Windows só são compatíveis com o uso do driver `local`.
- Se um driver de terceiros for usado, verifique se ele está instalado e ativo na instância do contêiner antes que o agente de contêiner seja iniciado. Se o driver de terceiros não estiver ativo antes da inicialização do agente, será possível reiniciar o agente do contêiner usando um dos comandos a seguir:
  - Para a AMI do Amazon Linux 2 otimizada para o Amazon ECS:

    ```
    sudo systemctl restart ecs
    ```

  - Para a AMI do Amazon Linux otimizada para o Amazon ECS:

    ```
    sudo stop ecs && sudo start ecs
    ```

Para obter informações sobre como especificar um volume do Docker em uma definição de tarefa, consulte Especificar um volume do Docker em uma definição de tarefa do Amazon ECS.

Especificar um volume do Docker em uma definição de tarefa do Amazon ECS

Antes de seus contêineres poderem usar volumes de dados, você deve especificar as configurações de volume e ponto de montagem em sua definição de tarefa. Esta seção descreve a configuração do volume de um contêiner. Para tarefas que usam um volume do Docker, especifique um `dockerVolumeConfiguration`. Para tarefas que usam um volume de host de montagem bind, especifique um `host` e `sourcePath` opcional.

O JSON de definição de tarefa a seguir mostra a sintaxe para os objetos `volumes` e `mountPoints` de um contêiner.

```
{
    "containerDefinitions": [
        {
            "mountPoints": [
                {
                    "sourceVolume": "string",
                    "containerPath": "/path/to/mount_volume",
                    "readOnly": boolean
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "string",
            "dockerVolumeConfiguration": {
                "scope": "string",
                "autoprovision": boolean,
                "driver": "string",
                "driverOpts": {
                    "key": "value"
                },
                "labels": {
                    "key": "value"
                }
            }
        }
    ]
```

```
}
```

`name`

Tipo: String

Obrigatório: Não

O nome do volume. São permitidos até 255 letras (caixa alta e baixa), números, hífens (`-`) e sublinhados (_). Esse nome é referenciado no parâmetro `sourceVolume` do objeto `mountPoints` de definição do contêiner.

`dockerVolumeConfiguration`

Tipo: objeto DockerVolumeConfiguration

Obrigatório: Não

Esse parâmetro é especificado ao usar volumes do Docker. Os volumes do Docker só são compatíveis ao executar tarefas em instâncias do EC2. Os contêineres do Windows só são compatíveis com o uso do driver `local`. Para usar montagens bind, em vez disso, especifique um `host`.

`scope`

Tipo: String

Valores válidos: `task` | `shared`

Obrigatório: Não

O escopo para o volume do Docker, que determina o ciclo de vida. Os volumes do Docker que são delimitados para uma `task` são provisionados automaticamente quando a tarefa é iniciada e destruídos quando a tarefa é interrompida. Volumes do Docker delimitados como `shared` são mantidos após a interrupção da tarefa.

`autoprovision`

Tipo: booliano

Valor padrão: `false`

Obrigatório: Não

Se o valor for `true`, o volume de Docker será criado se ele ainda não existir. Esse campo só será usado se `scope` for `shared`. Se `scope` for `task`, esse parâmetro deverá ser omitido.

`driver`

Tipo: String

Obrigatório: Não

O driver do volume do Docker a ser usado. O valor do driver deve corresponder ao nome do driver fornecido pelo Docker porque esse nome é usado no posicionamento de tarefas. Se o driver foi instalado usando a CLI de plug-in do Docker, use `docker plugin ls` para recuperar o nome do driver na instância de contêiner. Se o driver foi instalado usando outro método, use a descoberta de plug-in do Docker para recuperar o nome do driver.

`driverOpts`

Tipo: String

Obrigatório: Não

Um mapa de opções específicas do driver do Docker pelas quais passar. Esse parâmetro é mapeado para `DriverOpts` na seção Criar um volume da API remota do Docker.

`labels`

Tipo: String

Obrigatório: Não

Metadados personalizados para adicionar ao volume do Docker.

`mountPoints`

Tipo: Matriz de objeto

Obrigatório: Não

Os pontos de montagem dos volumes de dados no contêiner. Esse parâmetro é mapeado em `Volumes` na API create-container do Docker na opção `--volume` de execução do docker.

Os contêineres do Windows podem montar diretórios inteiros na mesma unidade como `$env:ProgramData`. Os contêineres do Windows não podem montar diretórios em uma unidade diferente, e os pontos de montagem não podem ser usados entre unidades. Você deve especificar pontos de montagem para anexar um volume do Amazon EBS diretamente a uma tarefa do Amazon ECS.

`sourceVolume`

Tipo: String

Exigido: Sim, quando `mountPoints` são usados

O nome do volume a ser montado.

`containerPath`

Tipo: String

Exigido: Sim, quando `mountPoints` são usados

O caminho no contêiner onde o volume será montado.

`readOnly`

Tipo: booliano

Obrigatório: Não

Caso o valor seja `true`, o contêiner tem acesso somente leitura ao volume. Caso esse valor seja `false`, o contêiner pode gravar no volume. O valor padrão é `false`.

Para tarefas executadas em instâncias do EC2 que executam o sistema operacional Windows, deixe o valor como o padrão de `false`.

Exemplos de volume do Docker para o Amazon ECS

Os exemplos a seguir mostram como fornecer armazenamento efêmero para um contêiner, como fornecer um volume compartilhado para vários contêineres e como fornecer armazenamento persistente NFS para um contêiner.

Para fornecer armazenamento temporário em um contêiner usando um volume do Docker

Neste exemplo, um contêiner utiliza um volume de dados vazio que é descartado após o término da tarefa. Um exemplo de caso de uso é que você pode ter um contêiner que precisa acessar algum local de armazenamento de arquivos temporários durante uma tarefa. Essa tarefa pode ser realizada usando um volume do Docker.

1. Na seção `volumes` de definição de tarefa, defina um volume de dados com os valores `name` e `DockerVolumeConfiguration`. Neste exemplo, especificamos o escopo como `task` para que o volume seja excluído depois que a tarefa é interrompida e use o driver `local` integrado.

```
"volumes": [
    {
        "name": "scratch",
        "dockerVolumeConfiguration" : {
            "scope": "task",
            "driver": "local",
            "labels": {
                "scratch": "space"
            }
        }
    }
]
```

2. Na seção `containerDefinitions`, defina um contêiner com valores `mountPoints` que faz referência ao nome do volume definido e o valor `containerPath` para montar o volume no contêiner.

```
"containerDefinitions": [
    {
        "name": "container-1",
        "mountPoints": [
            {
              "sourceVolume": "scratch",
              "containerPath": "/var/scratch"
            }
        ]
    }
]
```

Para fornecer armazenamento persistente para vários contêiner usando um volume do Docker

Neste exemplo, você quer um volume compartilhado para vários contêineres usarem e deseja que ele persista após a interrupção de qualquer tarefa única que o esteja usando. O driver `local` integrado está sendo usado. Isso acontece para que o volume permaneça vinculado ao ciclo de vida da instância de contêiner.

1. Na seção `volumes` de definição de tarefa, defina um volume de dados com os valores `name` e `DockerVolumeConfiguration`. Neste exemplo, especifique um escopo `shared` de forma que o volume persista e defina a provisão automática como `true`. Isso acontece para que o volume seja criado para uso. Então, utilize também o driver `local` incorporado.

```
"volumes": [
    {
        "name": "database",
        "dockerVolumeConfiguration" : {
            "scope": "shared",
            "autoprovision": true,
            "driver": "local",
            "labels": {
                "database": "database_name"
            }
        }
    }
]
```

2. Na seção `containerDefinitions`, defina um contêiner com valores `mountPoints` que faz referência ao nome do volume definido e o valor `containerPath` para montar o volume no contêiner.

```
"containerDefinitions": [
    {
        "name": "container-1",
        "mountPoints": [
        {
          "sourceVolume": "database",
          "containerPath": "/var/database"
        }
      ]
    },
    {
      "name": "container-2",
      "mountPoints": [
        {
          "sourceVolume": "database",
          "containerPath": "/var/database"
        }
      ]
    }
  ]
```

Para fornecer armazenamento NFS persistente para um contêiner usando um volume do Docker

Neste exemplo, um contêiner utiliza um volume de dados do NFS que é montado automaticamente quando a tarefa é iniciada e desmontado quando a tarefa é interrompida. Isso usa o driver `local` integrado do Docker. Um exemplo de caso de uso: você pode ter um armazenamento NFS local e precisa acessá-lo por meio de uma tarefa do ECS Anywhere. Isso pode ser feito usando uma opção de volume do Docker com driver NFS.

1. Na seção `volumes` de definição de tarefa, defina um volume de dados com os valores `name` e `DockerVolumeConfiguration`. Neste exemplo, especifique um escopo de `task` para que o volume seja desmontado após o término da tarefa. Use o driver `local` e configure o `driverOpts` com as opções `type`, `device` e o de acordo. Substitua `NFS_SERVER` pelo endpoint do servidor NFS.

```
"volumes": [
       {
           "name": "NFS",
           "dockerVolumeConfiguration" : {
               "scope": "task",
               "driver": "local",
               "driverOpts": {
                   "type": "nfs",
                   "device": "$NFS_SERVER:/mnt/nfs",
                   "o": "addr=$NFS_SERVER"
               }
           }
       }
   ]
```

2. Na seção `containerDefinitions`, defina um contêiner com valores `mountPoints` que faça referência ao nome do volume definido e o valor `containerPath` para montar o volume no contêiner.

```
"containerDefinitions": [
       {
           "name": "container-1",
           "mountPoints": [
               {
                 "sourceVolume": "NFS",
                 "containerPath": "/var/nfsmount"
               }
```

```
            ]
        }
    ]
```

Uso de montagens vinculadas com o Amazon ECS

Com montagens vinculadas, um arquivo ou diretório em um host, como uma instância do Amazon EC2, é montado em um contêiner. As montagens bind são compatíveis com tarefas hospedadas em instâncias do Fargate e do Amazon EC2. Montagens vinculadas estão associadas ao ciclo de vida do contêiner que as usa. A partir do momento em que todos os contêineres que utilizam uma montagem bind forem interrompidos, por exemplo, quando uma tarefa é interrompida, os dados serão removidos. Para tarefas hospedadas em instâncias do Amazon EC2, os dados podem ser vinculados ao ciclo de vida da instância do Amazon EC2 do host especificando um `host` e um valor opcional de `sourcePath` na definição de tarefa. Para obter mais informações, consulte Montagens bind na documentação do Docker.

Veja a seguir alguns casos de uso comuns de montagens bind.

- Para fornecer um volume de dados vazio para ser montado em um ou mais contêineres.
- Para montar um volume de dados de host em um ou mais contêineres.
- Para compartilhar um volume de dados de um contêiner de origem com outros contêineres na mesma tarefa.
- Para expor um caminho e seu conteúdo de um Dockerfile para um ou mais contêineres.

Considerações quando forem usadas montagens bind

Ao usar montagens bind, considere o seguinte.

- Por padrão, as tarefas hospedadas no AWS Fargate usando a versão da plataforma `1.4.0` ou posterior (Linux) ou `1.0.0` ou posterior (Windows) recebem, no mínimo, 20 GiB de armazenamento temporário para montagens vinculadas. É possível aumentar a quantidade total de armazenamento temporário, até um máximo de 200 GiB, com a especificação do parâmetro `ephemeralStorage` na definição da tarefa.
- Para expor arquivos de um Dockerfile a um volume de dados quando uma tarefa é executada, o plano de dados do Amazon ECS procura uma diretiva `VOLUME`. Se o caminho absoluto especificado na diretiva `VOLUME` é o mesmo que o `containerPath` especificado na definição de tarefa, os dados no caminho da diretiva `VOLUME` são copiados para o volume de dados. No

exemplo de Dockerfile a seguir, um arquivo denominado `examplefile` do diretório `/var/log/exported` é gravado no host e, em seguida, montado no contêiner.

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
RUN mkdir -p /var/log/exported
RUN touch /var/log/exported/examplefile
VOLUME ["/var/log/exported"]
```

Por padrão, as permissões de volume são definidas como `0755` e o proprietário como `root`. É possível personalizar essas permissões no Dockerfile. O exemplo a seguir define o proprietário do diretório como `node`.

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
RUN yum install -y shadow-utils && yum clean all
RUN useradd node
RUN mkdir -p /var/log/exported && chown node:node /var/log/exported
RUN touch /var/log/exported/examplefile
USER node
VOLUME ["/var/log/exported"]
```

- Para tarefas hospedadas em instâncias do Amazon EC2, quando um`host` e `sourcePath` não são especificados, o daemon do Docker gerencia a montagem bind para você. Quando nenhum contêiner fizer referência a essa montagem bind, o serviço de limpeza de tarefas do agente de contêiner do Amazon ECS acabará excluindo-a. Isso ocorre por padrão três horas após o encerramento do contêiner. Porém, é possível configurar essa duração com a variável de agente `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION`. Para obter mais informações, consulte Configuração do agente de contêiner do Amazon ECS. Se você precisar que esses dados persistam além do ciclo de vida do contêiner, especifique um valor `sourcePath` para a montagem bind.

Especificar uma montagem associada em uma definição de tarefa do Amazon ECS

Em tarefas do Amazon ECS hospedadas em instâncias do Fargate ou do Amazon EC2, o trecho JSON de definição de tarefa a seguir mostra a sintaxe dos objetos `volumes`, `mountPoints` e `ephemeralStorage` para uma definição de tarefa.

```
{
   "family": "",
   ...
```

```
   "containerDefinitions" : [
      {
         "mountPoints" : [
            {
               "containerPath" : "/path/to/mount_volume",
               "sourceVolume" : "string"
            }
          ],
          "name" : "string"
       }
    ],
    ...
    "volumes" : [
       {
          "name" : "string"
       }
    ],
    "ephemeralStorage": {
    "sizeInGiB": integer
    }
}
```

Para tarefas do Amazon ECS que são hospedadas em instâncias do Amazon EC2, é possível utilizar o parâmetro `host` e um `sourcePath` ao especificar detalhes do volume da tarefa. Quando especificado, ele associa a montagem bind ao ciclo de vida da tarefa no lugar do contêiner.

```
"volumes" : [
    {
        "host" : {
            "sourcePath" : "string"
        },
        "name" : "string"
    }
]
```

A seguir, é descrito em mais detalhes cada parâmetro de definição de tarefa.

`name`

Tipo: String

Obrigatório: Não

O nome do volume. São permitidos até 255 letras (caixa alta e baixa), números, hífens (-) e sublinhados (_). Esse nome é referenciado no parâmetro `sourceVolume` do objeto `mountPoints` de definição do contêiner.

`host`

Obrigatório: Não

O parâmetro `host` é usado para vincular o ciclo de vida da montagem bind à instância host do Amazon EC2, em vez da tarefa, e onde ela está armazenada. Caso o parâmetro `host` esteja vazio, o daemon do Docker atribui um caminho host para o volume de dados, mas os dados não têm garantia de persistir depois que os contêineres associados deixarem de ser executados.

Os contêineres do Windows podem montar diretórios inteiros na mesma unidade como `$env:ProgramData`.

> **Note**
>
> O parâmetro `sourcePath` só é compatível ao usar tarefas hospedadas em instâncias do Amazon EC2.

`sourcePath`

Tipo: String

Obrigatório: Não

Quando o parâmetro `host` é usado, especifique um `sourcePath` para declarar o caminho na instância host do Amazon EC2 que é apresentada ao contêiner. Caso esse parâmetro esteja vazio, o daemon do Docker atribui um caminho host para você. Caso o parâmetro `host` contenha um local de arquivo `sourcePath`, o volume de dados persistirá no local especificado na instância host do Amazon EC2 até você excluí-lo manualmente. Caso o valor `sourcePath` não exista na instância host do Amazon EC2, o daemon do Docker o criará. Caso o local exista, o conteúdo da pasta do caminho de origem é exportado.

`mountPoints`

Tipo: Matriz de objeto

Obrigatório: Não

Os pontos de montagem dos volumes de dados no contêiner. Esse parâmetro é mapeado em `Volumes` na API create-container do Docker na opção `--volume` de execução do docker.

Os contêineres do Windows podem montar diretórios inteiros na mesma unidade como `$env:ProgramData`. Os contêineres do Windows não podem montar diretórios em uma unidade diferente, e os pontos de montagem não podem ser usados entre unidades. Você deve especificar pontos de montagem para anexar um volume do Amazon EBS diretamente a uma tarefa do Amazon ECS.

`sourceVolume`

Tipo: String

Exigido: Sim, quando `mountPoints` são usados

O nome do volume a ser montado.

`containerPath`

Tipo: String

Exigido: Sim, quando `mountPoints` são usados

O caminho no contêiner onde o volume será montado.

`readOnly`

Tipo: booliano

Obrigatório: Não

Caso o valor seja `true`, o contêiner tem acesso somente leitura ao volume. Caso esse valor seja `false`, o contêiner pode gravar no volume. O valor padrão é `false`.

Para tarefas executadas em instâncias do EC2 que executam o sistema operacional Windows, deixe o valor como o padrão de `false`.

`ephemeralStorage`

Tipo: Objeto

Obrigatório: Não

A quantidade de armazenamento temporário a ser alocada para a tarefa. Esse parâmetro é usado para expandir a quantidade total de armazenamento temporário, além da quantidade padrão,

para tarefas hospedadas no AWS Fargate, usando a versão `1.4.0` ou posterior (Linux) ou `1.0.0` ou posterior (Windows) da plataforma.

É possível usar a CLI do Copilot, o CloudFormation, o AWS SDK ou a CLI para especificar o armazenamento temporário para uma montagem bind.

Exemplos de montagem de associação para o Amazon ECS

Os exemplos a seguir abrangem casos de uso comuns de uma montagem bind para contêineres.

Para alocar uma quantidade maior de espaço de armazenamento temporário para uma tarefa do Fargate

Para tarefas do Amazon ECS hospedadas no Fargate usando a versão `1.4.0` ou posterior (Linux) ou `1.0.0` (windows) da plataforma, você pode alocar mais do que a quantidade padrão de armazenamento temporário para os contêineres da tarefa que serão usados. Esse exemplo pode ser incorporado aos outros exemplos para alocar mais armazenamento temporário para suas tarefas do Fargate.

- Na definição da tarefa, defina um objeto `ephemeralStorage`. O `sizeInGiB` deve ser um número inteiro entre os valores de `21` e `200` e é expresso em GiB.

```
"ephemeralStorage": {
    "sizeInGiB": integer
}
```

Para fornecer um volume de dados vazio para um ou mais contêineres

Em alguns casos, você pode querer fornecer aos contêineres de uma tarefa algum espaço temporário. Por exemplo, você pode ter dois contêineres de banco de dados que precisam acessar o mesmo local de armazenamento de arquivos temporários durante uma tarefa. Isso pode ser feito usando uma montagem bind.

1. Na seção `volumes` da definição de tarefa, defina uma montagem bind com o nome `database_scratch`.

```
  "volumes": [
    {
      "name": "database_scratch"
```

```
    }
  ]
```

2. Na seção `containerDefinitions`, crie as definições de contêiner de banco de dados. Isso deve ser feito para que elas montem o volume.

```
"containerDefinitions": [
    {
      "name": "database1",
      "image": "my-repo/database",
      "cpu": 100,
      "memory": 100,
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "database_scratch",
          "containerPath": "/var/scratch"
        }
      ]
    },
    {
      "name": "database2",
      "image": "my-repo/database",
      "cpu": 100,
      "memory": 100,
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "database_scratch",
          "containerPath": "/var/scratch"
        }
      ]
    }
  ]
```

Para expor um caminho e seu conteúdo em um Dockerfile para um contêiner

Neste exemplo, você tem um Dockerfile que grava dados que você quer montar em um contêiner. Este exemplo funciona para tarefas hospedadas em instâncias do Fargate ou do Amazon EC2.

1. Crie um Dockerfile. O exemplo a seguir usa a imagem pública do contêiner do Amazon Linux 2 e cria um arquivo denominado `examplefile` no diretório `/var/log/exportedque` queremos montar dentro do contêiner. A diretiva `VOLUME` deve especificar um caminho absoluto.

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
RUN mkdir -p /var/log/exported
RUN touch /var/log/exported/examplefile
VOLUME ["/var/log/exported"]
```

   Por padrão, as permissões de volume são definidas como `0755` e o proprietário como `root`. Essas permissões podem ser alteradas no Dockerfile. No exemplo a seguir, o proprietário do diretório `/var/log/exported` é definido como `node`.

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
RUN yum install -y shadow-utils && yum clean all
RUN useradd node
RUN mkdir -p /var/log/exported && chown node:node /var/log/exported
USER node
RUN touch /var/log/exported/examplefile
VOLUME ["/var/log/exported"]
```

2. Na seção `volumes` da definição da tarefa, defina um volume com o nome `application_logs`.

```
"volumes": [
  {
    "name": "application_logs"
  }
]
```

3. Na seção `containerDefinitions`, crie as definições do contêiner da aplicação. Isso deve ser feito para que elas montem o armazenamento. O valor `containerPath` deve corresponder ao caminho absoluto especificado na diretiva `VOLUME` do Dockerfile.

```
"containerDefinitions": [
  {
    "name": "application1",
    "image": "my-repo/application",
    "cpu": 100,
    "memory": 100,
    "essential": true,
    "mountPoints": [
```

```
        {
          "sourceVolume": "application_logs",
          "containerPath": "/var/log/exported"
        }
      ]
    },
    {
      "name": "application2",
      "image": "my-repo/application",
      "cpu": 100,
      "memory": 100,
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "application_logs",
          "containerPath": "/var/log/exported"
        }
      ]
    }
]
```

Para fornecer um volume de dados vazio para um contêiner vinculado ao ciclo de vida da instância host do Amazon EC2

Para tarefas hospedadas em instâncias do Amazon EC2, você pode usar montagens bind e ter os dados vinculados ao ciclo de vida da instância host do Amazon EC2. É possível fazer isso usando o parâmetro `host` e especificando um valor `sourcePath`. Todos os arquivos existentes em `sourcePath` são apresentados aos contêineres no valor `containerPath`. Todos os arquivos que são gravados no valor `containerPath` são gravados no valor `sourcePath` na instância do host do Amazon EC2.

> **Important**
>
> O Amazon ECS não sincroniza seu armazenamento entre instâncias do Amazon EC2. As tarefas que usam armazenamento persistente podem ser colocadas em qualquer instância do Amazon EC2, no cluster que tiver capacidade disponível. Se suas tarefas exigirem armazenamento persistente após a interrupção e a reinicialização, sempre especifique a mesma instância do Amazon EC2 no momento da inicialização da tarefa com o comando start-task da AWS CLI. Também podem ser usados volumes do Amazon EFS

para armazenamento persistente. Para obter mais informações, consulte Uso de volumes do Amazon EFS com o Amazon ECS.

1. Na seção `volumes` da definição de tarefa, defina uma montagem bind com os valores `name` e `sourcePath`. No exemplo a seguir, a instância host do Amazon EC2 contém dados em `/ecs/webdata` que você quer montar no contêiner.

```
"volumes": [
  {
    "name": "webdata",
    "host": {
      "sourcePath": "/ecs/webdata"
    }
  }
]
```

2. Na seção `containerDefinitions`, defina um contêiner com um valor `mountPoints` que faça referência ao nome da montagem bind definida e ao valor `containerPath` para montar a montagem bind no contêiner.

```
"containerDefinitions": [
  {
    "name": "web",
    "image": "public.ecr.aws/docker/library/nginx:latest",
    "cpu": 99,
    "memory": 100,
    "portMappings": [
      {
        "containerPort": 80,
        "hostPort": 80
      }
    ],
    "essential": true,
    "mountPoints": [
      {
        "sourceVolume": "webdata",
        "containerPath": "/usr/share/nginx/html"
      }
    ]
  }
```

```
]
```

Para montar um volume definido em vários contêineres em diferentes locais

É possível definir um volume de dados em uma definição de tarefa e montar esse volume em locais diferentes em diferentes contêineres. Por exemplo, seu contêiner host tem uma pasta de dados de site em `/data/webroot`. É possível montar esse volume de dados como somente leitura em dois servidores Web diferentes com raízes de documentos distintas.

1. Na seção `volumes` de definição de tarefa, defina um volume de dados com o nome `webroot` e o caminho de origem `/data/webroot`.

```
"volumes": [
  {
    "name": "webroot",
    "host": {
      "sourcePath": "/data/webroot"
    }
  }
]
```

2. Na seção `containerDefinitions`, defina um contêiner para cada servidor da Web com valores `mountPoints` que associam o volume `webroot` com o valor `containerPath` apontando para a raiz do documento desse contêiner.

```
"containerDefinitions": [
  {
    "name": "web-server-1",
    "image": "my-repo/ubuntu-apache",
    "cpu": 100,
    "memory": 100,
    "portMappings": [
      {
        "containerPort": 80,
        "hostPort": 80
      }
    ],
    "essential": true,
    "mountPoints": [
      {
        "sourceVolume": "webroot",
```

```
          "containerPath": "/var/www/html",
          "readOnly": true
        }
      ]
    },
    {
      "name": "web-server-2",
      "image": "my-repo/sles11-apache",
      "cpu": 100,
      "memory": 100,
      "portMappings": [
        {
          "containerPort": 8080,
          "hostPort": 8080
        }
      ],
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "webroot",
          "containerPath": "/srv/www/htdocs",
          "readOnly": true
        }
      ]
    }
  ]
```

Para montar volumes de outro contêiner usando **`volumesFrom`**

Para tarefas hospedadas em instâncias do Amazon EC2, é possível definir um ou mais volumes em um contêiner e, em seguida, usar o parâmetro `volumesFrom` em uma definição de contêiner diferente (dentro da mesma tarefa) para montar todos os volumes do `sourceContainer` em seus pontos de montagem definidos originalmente. O parâmetro `volumesFrom` se aplica a volumes determinados na definição de tarefa, e os que são incorporados na imagem com um Dockerfile.

1. (Opcional) Para compartilhar um volume incorporado em uma imagem, utilize a instrução `VOLUME` no arquivo Dockerfile. O Dockerfile de exemplo a seguir usa uma imagem `httpd` e, em seguida, adiciona um volume e monta-o em `dockerfile_volume` na raiz do documento Apache. É a pasta usada pelo servidor Web `httpd`.

   ```
   FROM httpd
   ```

```
VOLUME ["/usr/local/apache2/htdocs/dockerfile_volume"]
```

É possível incorporar uma imagem com esse Dockerfile e enviá-la para um repositório, como o Docker Hub, e usá-la em sua definição de tarefa. A imagem de exemplo `my-repo/httpd_dockerfile_volume` usada nas etapas a seguir foi criada com o Dockerfile anterior.

2. Crie uma definição de tarefa que defina seus outros volumes e pontos de montagem para os contêineres. Nesta seção `volumes` de exemplo, você cria um volume vazio chamado `empty`, que o daemon do Docker gerencia. Há também um volume host definido que se chama `host_etc`. Ele exporta a pasta `/etc` na instância de contêiner host.

```
{
  "family": "test-volumes-from",
  "volumes": [
    {
      "name": "empty",
      "host": {}
    },
    {
      "name": "host_etc",
      "host": {
        "sourcePath": "/etc"
      }
    }
  ],
```

Na seção definições de contêiner, crie um contêiner que monte os volumes definidos anteriormente. Neste exemplo, o contêiner `web` monta os volumes `empty` e `host_etc`. Este é o contêiner que utiliza a imagem criada com um volume no Dockerfile.

```
"containerDefinitions": [
    {
      "name": "web",
      "image": "my-repo/httpd_dockerfile_volume",
      "cpu": 100,
      "memory": 500,
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 80
        }
```

```
      ],
      "mountPoints": [
        {
          "sourceVolume": "empty",
          "containerPath": "/usr/local/apache2/htdocs/empty_volume"
        },
        {
          "sourceVolume": "host_etc",
          "containerPath": "/usr/local/apache2/htdocs/host_etc"
        }
      ],
      "essential": true
    },
```

Crie outro contêiner que use `volumesFrom` para montar todos os volumes que estão associados com o contêiner `web`. Todos os volumes no contêiner `web` também são montados no contêiner `busybox`. Isso inclui o volume especificado no Dockerfile que foi utilizado para criar a imagem `my-repo/httpd_dockerfile_volume`.

```
    {
      "name": "busybox",
      "image": "busybox",
      "volumesFrom": [
        {
          "sourceContainer": "web"
        }
      ],
      "cpu": 100,
      "memory": 500,
      "entryPoint": [
        "sh",
        "-c"
      ],
      "command": [
        "echo $(date) > /usr/local/apache2/htdocs/empty_volume/date && echo $(date)
 > /usr/local/apache2/htdocs/host_etc/date && echo $(date) > /usr/local/apache2/
htdocs/dockerfile_volume/date"
      ],
      "essential": false
    }
  ]
}
```

Quando essa tarefa é executada, os dois contêineres montam os volumes, e o `command` no contêiner `busybox` grava a data e a hora em um arquivo. Esse arquivo se chama `date` em cada uma das pastas de volume. As pastas ficam visíveis no site exibido pelo contêiner `web`.

> **Note**
>
> Como o contêiner `busybox` executa um comando rápido e, em seguida, é encerrado, ele deve ser definido como `"essential": false` na definição do contêiner. Caso contrário, ele interromperá toda a tarefa quando for encerrado.

Gerenciar o espaço de memória de troca de contêiner no Amazon ECS

Com o Amazon ECS, você pode controlar o uso do espaço de memória de troca em suas instâncias do Amazon EC2 baseadas em Linux no nível de contêiner. Utilizando uma configuração de troca de contêiner, cada contêiner de uma definição de tarefa pode ter a troca habilitada ou desabilitada. Para aqueles com troca habilitada, a quantidade máxima de espaço de troca utilizada pode ser limitada. Por exemplo, contêineres críticos para latência podem ter a troca desabilitada. Por outro lado, contêineres com altas demandas de memória transitória podem ter a troca habilitada para reduzir as chances de erros de falta de memória quando o contêiner está sob carga.

A configuração de troca para um contêiner é gerenciada pelos seguintes parâmetros de definição de contêiner.

`maxSwap`

A quantidade total de memória de troca (em MiB) que um contêiner pode usar. Esse parâmetro será convertido na opção `--memory-swap` para execução do docker em que o valor é a soma da memória do contêiner mais o valor de `maxSwap`.

Se um valor `maxSwap` de `0` for especificado, o contêiner não usará a troca. Os valores aceitos são `0` ou qualquer número inteiro positivo. Se o parâmetro `maxSwap` for omitido, o contêiner usará a configuração de troca para a instância de contêiner na qual ele está sendo executado. Um valor `maxSwap` deve ser definido para que o parâmetro `swappiness` seja usado.

`swappiness`

É possível usar isso para ajustar o comportamento de troca de memória de um contêiner. Um valor `swappiness` igual a `0` faz com que a troca não ocorra, a menos que ela seja necessária.

Um valor `swappiness` de `100` fará com que as páginas sejam trocadas de forma agressiva. Os valores aceitos são números inteiros entre `0` e `100`. Se o parâmetro `swappiness` não for especificado, será usado um valor padrão de `60`. Se nenhum valor for especificado para `maxSwap`, esse parâmetro será ignorado. Esse parâmetro é mapeado para a opção `--memory-swappiness` de execução do docker.

No exemplo abaixo, é fornecida a sintaxe JSON.

```
"containerDefinitions": [{
        ...
        "linuxParameters": {
            "maxSwap": integer,
            "swappiness": integer
        },
        ...
}]
```

Considerações

Considere o seguinte ao usar uma configuração de troca de contêiner.

- O espaço de troca deve ser habilitado e alocado na instância do Amazon EC2 que hospeda suas tarefas para os contêineres usarem. As AMIs otimizadas para o Amazon ECS não têm a troca habilitada por padrão. É necessário habilitar a troca na instância para usar esse recurso. Para obter mais informações, consulte Volumes de troca de armazenamento de instâncias no Manual do usuário do Amazon EC2 ou Como alocar memória para atuar como espaço de troca em uma instância do Amazon EC2?.
- Os parâmetros de definição do contêiner do espaço de troca são compatíveis somente com definições de tarefa que especificam o tipo de inicialização do EC2. Esses parâmetros não são compatíveis com definições de tarefa destinadas somente ao uso do Amazon ECS no Fargate.
- Esse recurso é compatível somente com contêineres do Linux. No momento, não há compatibilidade com contêineres do Windows.
- Se os parâmetros `maxSwap` e `swappiness` de definição de contêiner forem omitidos de uma definição de tarefa, cada contêiner terá um valor `swappiness` padrão de `60`. Além disso, o limite de uso total de trocas é de duas vezes a memória do contêiner.
- Se você estiver usando tarefas no Amazon Linux 2023, não haverá suporte para o parâmetro `swappiness`.

Diferenças de definição de tarefa do Amazon ECS para o tipo de inicialização do Fargate

Para usar o Fargate, você precisa configurar a definição de tarefa para usar o tipo de execução do Fargate. Há considerações adicionais ao usar o Fargate.

Parâmetros de definição de tarefa

As tarefas que usam o tipo de inicialização do Fargate não são compatíveis com todos os parâmetros de definição de tarefa do Amazon ECS disponíveis. Alguns parâmetros são totalmente incompatíveis e outros se comportam de maneira diferente nas tarefas do Fargate.

Os seguintes parâmetros de definição de tarefa não são válidos nas tarefas do Fargate:

- `disableNetworking`
- `dnsSearchDomains`
- `dnsServers`
- `dockerSecurityOptions`
- `extraHosts`
- `gpu`
- `ipcMode`
- `links`
- `placementConstraints`
- `privileged`
- `maxSwap`
- `swappiness`

Os seguintes parâmetros de definição de tarefa são válidos em tarefas do Fargate, mas têm limitações que devem ser observadas:

- `linuxParameters`: ao definir opções específicas do Linux que são aplicadas ao contêiner, para `capabilities`, o único recurso possível de ser adicionado é `CAP_SYS_PTRACE`. Os parâmetros `devices`, `sharedMemorySize` e `tmpfs` não têm suporte. Para obter mais informações, consulte Parâmetros do Linux.

- `volumes`: as tarefas do Fargate só oferecem suporte aos volumes do host de montagem bind e, portanto, não há suporte para o parâmetro `dockerVolumeConfiguration`. Para obter mais informações, consulte Volumes.
- `cpu`: para contêineres do Windows no AWS Fargate, o valor não pode ser menor que 1 vCPU.
- `networkConfiguration`: as tarefas do Fargate sempre usam o modo de rede `awsvpc`.

Para garantir que sua definição de tarefa seja válida para uso com o Fargate, é possível especificar o seguinte quando você registrar a definição de tarefa:

- No AWS Management Console, para o campo Requires Compatibilities (Requer compatibilidades), especifique `FARGATE`.
- Na AWS CLI, especifique a opção `--requires-compatibilities`.
- Na API do Amazon ECS, especifique o sinalizador `requiresCompatibilities`.

Arquiteturas e sistemas operacionais

Quando você configura uma definição de tarefa e de contêiner para AWS Fargate, você deve especificar o sistema operacional que o contêiner executa. Há suporte para os seguintes sistemas operacionais para AWS Fargate:

- Amazon Linux 2

> **Note**
>
> Os contêineres do Linux usam apenas o kernel e a configuração do kernel do sistema operacional do host. Por exemplo, a configuração do kernel inclui os controles do sistema `sysctl`. Uma imagem de contêiner de Linux pode ser criada a partir de uma imagem base que contenha os arquivos e programas de qualquer distribuição do Linux. Se a arquitetura da CPU corresponder, será possível executar contêineres a partir de qualquer imagem de contêiner de Linux em qualquer sistema operacional.

- Windows Server 2019 Full
- Windows Server 2019 Core
- Windows Server 2022 Full
- Windows Server 2022 Core

Quando você executar contêineres do Windows no AWS Fargate, você deverá ter uma arquitetura de CPU X86_64.

Quando você executar contêineres do Linux no AWS Fargate, você pode usar a arquitetura de CPU X86_64 ou a arquitetura ARM64 para suas aplicações baseadas em ARM. Para obter mais informações, consulte the section called “Definições de tarefa para workloads do ARM de 64 bits”.

CPU e memória da tarefa

As definições de tarefa do Amazon ECS para AWS Fargate exigem que você especifique a CPU e a memória no nível da tarefa. A maioria dos casos de uso são atendidos com a especificação desses recursos somente no nível de tarefa. A tabela a seguir mostra as combinações válidas para CPU e memória em nível de tarefa. É possível especificar valores de memória na definição de tarefa como uma string em MiB ou GB. Por exemplo, você pode especificar um valor de memória como `3072` em MiB ou `3 GB` em GB. Você pode especificar valores de CPU no arquivo JSON como uma string em unidades de CPU ou em CPUs virtuais (vCPUs). Por exemplo, você pode especificar um valor de CPU como `1024` em unidades de CPU ou `1 vCPU` em vCPUs.

Valor de CPU	Valor de memória	Sistemas operacionais com suporte para o AWS Fargate
256 (0,25 vCPU)	512 MiB, 1 GB, 2 GB	Linux
512 (0,5 vCPU)	1 GB, 2 GB, 3 GB, 4 GB	Linux
1024 (1 vCPU)	2 GB, 3 GB, 4 GB, 5 GB, 6 GB, 7 GB, 8 GB	Linux, Windows
2048 (2 vCPU)	Entre 4 GB e 16 GB em incrementos de 1 GB	Linux, Windows
4096 (4 vCPU)	Entre 8 GB e 30 GB em incrementos de 1 GB	Linux, Windows
8192 (8 vCPU)	Entre 16 GB e 60 GB em incrementos de 4 GB	Linux

Valor de CPU	Valor de memória	Sistemas operacionais com suporte para o AWS Fargate
Note Essa opção requer a plataforma Linux `1.4.0` ou posterior.		
16384 (16 vCPU) **Note** Essa opção requer a plataforma Linux `1.4.0` ou posterior.	Entre 32 GB e 120 GB em incrementos de 8 GB	Linux

Redes de tarefas

As tarefas do Amazon ECS para AWS Fargate exigem o modo de rede `awsvpc`, que fornece uma interface de rede elástica para cada tarefa. Ao executar uma tarefa ou criar um serviço com esse modo de rede, você deve especificar uma ou mais sub-redes para anexar à interface de rede e um ou mais security groups para aplicar à interface de rede.

Se você estiver usando sub-redes públicas, decida se um endereço IP público será fornecido à interface de rede. Para uma tarefa do Fargate em uma sub-rede pública, para extrair imagens do contêiner, será preciso que um endereço IP público seja atribuído à interface de rede elástica da tarefa, com uma rota para a Internet ou um gateway NAT que possa encaminhar as solicitações para a Internet. Para que uma tarefa do Fargate em uma sub-rede privada extraia imagens de contêiner, é necessário um gateway NAT na sub-rede para encaminhar solicitações para a Internet. Ao hospedar imagens de contêiner no Amazon ECR, você pode configurar o Amazon ECR para usar um endpoint da VPC de interface. Nesse caso, o endereço IPv4 privado da tarefa é usado para a extração da imagem. Para obter mais informações sobre endpoints de interface do Amazon ECR, consulte Endpoints da VPC de interface do Amazon ECR (AWS PrivateLink) no Guia do usuário do Amazon Elastic Container Registry.

Este é um exemplo da seção `networkConfiguration` para um serviço do Fargate:

```
"networkConfiguration": {
   "awsvpcConfiguration": {
      "assignPublicIp": "ENABLED",
      "securityGroups": [ "sg-12345678" ],
      "subnets": [ "subnet-12345678" ]
   }
}
```

Limites de recursos de tarefa

As definições de tarefa do Amazon ECS para contêineres do Linux no AWS Fargate oferecem suporte ao parâmetro `ulimits` para definir os limites de recursos a serem estabelecidos para um contêiner.

As definições de tarefa do Amazon ECS para Windows no AWS Fargate não oferecem suporte ao parâmetro `ulimits` para definir os limites de recursos a serem estabelecidos para um contêiner.

As tarefas do Amazon ECS hospedadas no Fargate usam os valores de limite de recursos padrão limitados pelo sistema operacional, com exceção do parâmetro de limite de recursos `nofile`. O limite de recursos `nofile` define uma restrição sobre o número de arquivos abertos que um contêiner pode usar. No Fargate, o limite flexível padrão de `nofile` é `65535`, e o limite rígido é `65535`. É possível definir os valores de ambos os limites até `1048576`.

Veja a seguir um exemplo de trecho de definição de tarefa que mostra como definir um limite `nofile` personalizado que tenha sido duplicado:

```
"ulimits": [
    {
       "name": "nofile",
       "softLimit": 2048,
       "hardLimit": 8192
    }
]
```

Para obter mais informações sobre os outros limites de recursos que podem ser ajustados, consulte Limites de recurso.

Registro em log

Registro em log de eventos

O Amazon ECS registra em log as ações que realiza no EventBridge. É possível usar os eventos do Amazon ECS para o Eventbridge receber notificações quase em tempo real quanto ao estado atual dos clusters, serviços e tarefas do Amazon ECS. Além disso, é possível automatizar ações para responder a esses eventos. Para obter mais informações, consulte Automatização de respostas a erros do Amazon ECS usando o EventBridge.

Registro em log do ciclo de vida da tarefa

As tarefas executadas no Fargate publicam registros de data e hora para rastrear a tarefa nos estados do ciclo de vida da tarefa. É possível ver os registros de data e hora nos detalhes da tarefa no AWS Management Console, e com a descrição da tarefa na AWS CLI e nos SDKs. Por exemplo, é possível usar os carimbos de data/hora para avaliar quanto tempo a tarefa gastou baixando as imagens do contêiner e decidir se você deve otimizar o tamanho da imagem do contêiner ou usar índices Seekable OCI. Para obter mais informações sobre práticas de imagens de contêiner, consulte Práticas recomendadas para imagens de contêiner do Amazon ECS.

Registro em log de aplicações

As definições de tarefa do Amazon ECS para AWS Fargate oferecem suporte aos drivers de log `awslogs`, `splunk` e `awsfirelens` para a configuração do log.

O driver de log `awslogs` configura as tarefas do Fargate para enviar informações de log ao Amazon CloudWatch Logs. A tabela a seguir mostra um trecho de uma definição de tarefa em que o driver de log `awslogs` está configurado:

```
"logConfiguration": {
   "logDriver": "awslogs",
   "options": {
      "awslogs-group" : "/ecs/fargate-task-definition",
      "awslogs-region": "us-east-1",
      "awslogs-stream-prefix": "ecs"
   }
}
```

Para obter mais informações sobre como usar o driver de log `awslogs` em uma definição de tarefa para enviar os logs de contêiner para o CloudWatch Logs, consulte Envio de logs do Amazon ECS para o CloudWatch .

Para obter mais informações sobre o driver de log `awsfirelens` em uma definição de tarefa, consulte Envio de logs do Amazon ECS para um serviço da AWS ou para uma AWS Partner.

Para obter mais informações sobre como usar o driver de log `splunk` em uma definição de tarefa, consulte Driver de log do `splunk`.

Armazenamento de tarefas

Para tarefas do Amazon ECS hospedadas no Fargate, os seguintes tipos de armazenamento são compatíveis:

- Os volumes do Amazon EBS fornecem armazenamento em blocos econômico, durável e de alto desempenho para workloads em contêineres com uso intenso de dados. Para obter mais informações, consulte Uso de volumes do Amazon EBS com o Amazon ECS.
- Os volumes do Amazon EFS para armazenamento persistente. Para obter mais informações, consulte Uso de volumes do Amazon EFS com o Amazon ECS.
- Montagens bind para armazenamento temporário. Para obter mais informações, consulte Uso de montagens vinculadas com o Amazon ECS.

Carregamento lento de imagens de contêiner usando Seekable OCI (SOCI)

As tarefas do Amazon ECS no Fargate que usam a versão da plataforma Linux `1.4.0` podem usar o Seekable OCI (SOCI) para ajudar a iniciar tarefas mais rapidamente. Com o SOCI, os contêineres gastam apenas alguns segundos na extração da imagem antes de serem iniciados, fornecendo tempo para a configuração do ambiente e a instanciação da aplicação enquanto a imagem é baixada em segundo plano. Isso é chamado de carregamento lento. Quando o Fargate inicia uma tarefa do Amazon ECS, o Fargate detecta automaticamente se existe um índice SOCI para uma imagem na tarefa e inicia o contêiner sem esperar que a imagem inteira seja baixada.

Para contêineres executados sem índices SOCI, as imagens do contêiner são baixadas completamente antes de o contêiner ser iniciado. Esse comportamento é o mesmo em todas as outras versões da plataforma do Fargate e na AMI otimizada para Amazon ECS em instâncias do Amazon EC2.

O Seekable OCI (SOCI) é uma tecnologia de código aberto desenvolvida pela AWS que pode iniciar contêineres mais rapidamente carregando lentamente a imagem do contêiner. O SOCI funciona criando um índice (Índice SOCI) dos arquivos em uma imagem de contêiner existente. Esse índice ajuda a iniciar contêineres mais rapidamente, fornecendo a capacidade de extrair um arquivo

individual de uma imagem de contêiner antes de baixar a imagem inteira. O índice SOCI deve ser armazenado como um artefato no mesmo repositório da imagem no registro do contêiner. Você só deve usar índices SOCI de fontes confiáveis, pois o índice é a fonte autorizada do conteúdo da imagem. Para obter mais informações, consulte Introdução ao Seekable OCI para o carregamento lento de imagens de contêiner.

Os clientes que quiserem usar o SOCI só poderão usar o SOCI Index Manifest v2. Os clientes existentes que já tiverem usado o SOCI no Fargate poderão continuar usando o SOCI Index Manifest v1, no entanto, recomendamos fortemente que esses clientes migrem para o SOCI Index Manifest v2. O SOCI Index Manifest v2 cria uma relação explícita entre imagens de contêiner e seus índices SOCI para garantir implantações consistentes.

Considerações

Se você quiser que o Fargate use um índice SOCI para carregar imagens de contêiner lentamente em uma tarefa, considere o seguinte:

- Somente tarefas executadas na versão da plataforma Linux `1.4.0` podem usar índices SOCI. Não há suporte para tarefas executadas em contêineres do Windows no Fargate.
- Não há suporte para tarefas executadas nas arquitetura de CPU X86_64 ou ARM64.
- As imagens de contêiner na definição da tarefa devem ser armazenadas em um registro de imagens compatível. A seguir, são listados os registros compatíveis:
  - Registros privados do Amazon ECR.
- Só há suporte para as imagens de contêiner que usem compactação gzip ou não sejam compactadas. Não há suporte para imagens de contêiner que usem compactação zstd.
- Para o SOCI Index Manifest v2, a geração de um manifesto do índice SOCI modifica o manifesto da imagem do contêiner conforme adicionamos uma anotação ao índice SOCI. Isso resulta em um novo resumo da imagem do contêiner. O conteúdo das camadas do sistema de arquivos das imagens do contêiner não muda.
- Para o SOCI Index Manifest v2, quando a imagem do contêiner já tiver sido armazenada no repositório de imagens do contêiner, depois que um índice SOCI tiver sido gerado, você precisará reenviar a imagem do contêiner. O reenvio de uma imagem de contêiner não aumentará os custos de armazenamento ao duplicar as camadas do sistema de arquivos, apenas carregará um novo arquivo de manifesto.
- Recomendamos que você experimente o carregamento lento com imagens de contêiner maiores que 250 MiB compactadas em tamanho. É menos provável que você veja uma redução no tempo de carregamento de imagens menores.

- Como o carregamento lento pode alterar o tempo de início das tarefas, talvez seja necessário alterar vários tempos limite, como o período de carência da verificação de integridade do Elastic Load Balancing.
- Se você quiser evitar que uma imagem de contêiner seja carregada lentamente, a imagem do contêiner precisará ser reenviada sem um índice SOCI anexado.

Criação de um índice Seekable OCI

Para que uma imagem de contêiner seja carregada lentamente, ela precisa de um índice SOCI (um arquivo de metadados) criado e armazenado no repositório de imagens do contêiner com a imagem do contêiner. Para criar e fazer push de um índice SOCI, é possível usar a ferramenta de CLI soci-snapshotter de código aberto no GitHub. Você também pode implantar o AWS SOCI Index Builder do CloudFormation. Essa é uma solução sem servidor que cria e faz push de um índice SOCI automaticamente quando uma imagem de contêiner é enviada por push para o Amazon ECR. Para obter mais informações sobre a solução e as etapas de instalação, consulte CloudFormation AWS SOCI Index Builder no GitHub. O AWS SOCI Index Builder do CloudFormation é uma maneira de automatizar a introdução ao SOCI, enquanto a ferramenta SOCI de código aberto tem mais flexibilidade na geração de índices e a capacidade de integrar a geração de índices nos pipelines de integração contínua e entrega contínua (CI/CD).

> **Note**
>
> Para que o índice SOCI seja criado para uma imagem, a imagem deve existir no armazenamento de imagens do containerd no computador com o `soci-snapshotter` em execução. Se a imagem estiver no armazenamento de imagens do Docker, ela não poderá ser encontrada.

Verificação se uma tarefa usou carregamento lento

Para verificar se uma tarefa foi carregada lentamente usando o SOCI, verifique o endpoint de metadados da tarefa de dentro da tarefa. Quando você consulta o endpoint de metadados de tarefa versão 4, há um campo `Snapshotter` no caminho padrão do contêiner do qual você está consultando. Além disso, há campos `Snapshotter` para cada contêiner no caminho `/task`. O valor padrão para esse campo é `overlayfs`, e esse campo será definido como `soci` se o SOCI for usado. Para verificar se uma imagem de contêiner tem um SOCI Index Manifest v2 anexado, você pode recuperar o índice de imagem do Amazon ECR usando o AWS CLI.

```
IMAGE_REPOSITORY=r
IMAGE_TAG=latest

aws ecr batch-get-image \
    --repository-name=$IMAGE_REPOSITORY \
    --image-ids imageTag=$IMAGE_TAG \
    --query 'images[0].imageManifest' --output text | jq -r '.manifests[] |
 select(.artifactType=="application/vnd.amazon.soci.index.v2+json")'
```

Para verificar se uma imagem de contêiner tem um SOCI Index Manifest v1 anexado, você pode usar a API OCI Referrers.

```
ACCOUNT_ID=111222333444
AWS_REGION=us-east-1
IMAGE_REPOSITORY=nginx-demo
IMAGE_TAG=latest
IMAGE_DIGEST=$(aws ecr describe-images --repository-name $IMAGE_REPOSITORY --image-ids
 imageTag=$IMAGE_TAG --query 'imageDetails[0].imageDigest' --output text)
ECR_PASSWORD=$(aws ecr get-login-password)

curl \
    --silent \
    --user AWS:$ECR_PASSWORD \
    https://$ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/v2/$IMAGE_REPOSITORY/
referrers/$IMAGE_DIGEST?artifactType=application%2Fvnd.amazon.soci.index.v1%2Bjson | jq
 -r '.'
```

Diferenças de definição de tarefa do Amazon ECS para instâncias do EC2 executando o Windows

As tarefas executadas em instâncias do EC2 do Windows não oferecem suporte a todos os parâmetros de definição de tarefa disponíveis do Amazon ECS. Alguns parâmetros são totalmente incompatíveis e outros se comportam de maneira diferente.

Os seguintes parâmetros de definição de tarefa não são compatíveis com definições de tarefa do Amazon EC2 do Windows:

- `containerDefinitions`

  - `disableNetworking`
  - `dnsServers`
  - `dnsSearchDomains`
  - `extraHosts`
  - `links`
  - `linuxParameters`
  - `privileged`
  - `readonlyRootFilesystem`
  - `user`
  - `ulimits`
- `volumes`
  - `dockerVolumeConfiguration`
- `cpu`

  É recomendável especificar a CPU em nível de contêiner para contêineres do Windows.
- `memory`

  É recomendável especificar a memória em nível de contêiner para contêineres do Windows.
- `proxyConfiguration`
- `ipcMode`
- `pidMode`
- `taskRoleArn`

  Os perfis do IAM para tarefas em recursos de instâncias do Windows para EC2 exigem uma configuração adicional, mas boa parte dela é semelhante à configuração de perfis do IAM para tarefas em instâncias de contêiner do Linux. Para obter mais informações, consulte the section called “ Configuração adicional de instância do Windows no Amazon EC2”.

Criar uma definição de tarefa do Amazon ECS usando o console

Crie uma definição de tarefa para poder definir a aplicação que você executa como tarefa ou serviço.

Ao criar uma definição de tarefa para o tipo de execução externo, é necessário criar a definição da tarefa usando o editor JSON e definir o parâmetro `requireCapabilities` como `EXTERNAL`.

É possível criar uma definição de tarefa usando a experiência do console ou especificando um arquivo JSON.

Validação de JSON

O editor de JSON do console do Amazon ECS valida o seguinte no arquivo JSON:

- O arquivo é um arquivo JSON válido.
- O arquivo não contém nenhuma chave estranha.
- O arquivo contém o parâmetro `familyName`.
- Há pelo menos uma entrada em `containerDefinitions`.

Pilhas do AWS CloudFormation

O comportamento a seguir é aplicável às definições de tarefa criadas no novo console do Amazon ECS antes de 12 de janeiro de 2023.

Ao criar uma definição de tarefa, o console do Amazon ECS cria automaticamente uma pilha do CloudFormation com um nome que começa com `ECS-Console-V2-TaskDefinition-`. Se tiver usado a AWS CLI ou um AWS SDK para cancelar o registro da definição de tarefa, você deverá excluir manualmente a pilha de definições de tarefa. Para obter mais informações, consulte Excluir uma pilha no Guia do usuário do AWS CloudFormation.

As definições de tarefa criadas após 12 de janeiro de 2023 não têm uma pilha do CloudFormation criada automaticamente para elas.

Procedimento

Amazon ECS console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Task definitions (Definições de tarefa).
3. No menu Criar definição de tarefa, escolha Criar definição de tarefa.
4. Em Task definition family (Família de definição de tarefa), especifique um nome exclusivo para a definição de tarefa.
5. Em Tipo de inicialização, escolha o ambiente da aplicação. O padrão do console é AWS Fargate (tecnologia sem servidor). O Amazon ECS usa esse valor para realizar a validação e garantir que os parâmetros de definição de tarefa sejam válidos para o tipo de infraestrutura.

6. Em Operating system/Architecture (Sistema operacional/arquitetura), escolha o sistema operacional e a arquitetura da CPU para a tarefa.

   Para executar a tarefa em uma arquitetura ARM de 64 bits, selecione Linux/ARM64. Para obter mais informações, consulte the section called “Plataforma de runtime”.

   Para executar suas tarefas do AWS Fargate em contêineres do Windows, escolha um sistema operacional Windows compatível. Para obter mais informações, consulte the section called “Arquiteturas e sistemas operacionais”.

7. Em Task size (Tamanho da tarefa), escolha os valores de CPU e memória a serem reservados para a tarefa. O valor da CPU é especificado como vCPUs e a memória é especificada como GB.

   Para tarefas hospedadas no Fargate, a tabela a seguir mostra as combinações válidas de CPU e memória.

Valor de CPU	Valor de memória	Sistemas operacionais com suporte para o AWS Fargate
256 (0,25 vCPU)	512 MiB, 1 GB, 2 GB	Linux
512 (0,5 vCPU)	1 GB, 2 GB, 3 GB, 4 GB	Linux
1024 (1 vCPU)	2 GB, 3 GB, 4 GB, 5 GB, 6 GB, 7 GB, 8 GB	Linux, Windows
2048 (2 vCPU)	Entre 4 GB e 16 GB em incrementos de 1 GB	Linux, Windows
4096 (4 vCPU)	Entre 8 GB e 30 GB em incrementos de 1 GB	Linux, Windows
8192 (8 vCPU)	Entre 16 GB e 60 GB em incrementos de 4 GB	Linux

Valor de CPU	Valor de memória	Sistemas operacionais com suporte para o AWS Fargate
Note Essa opção requer a plataforma Linux `1.4.0` ou posterior.		
16384 (16 vCPU) Note Essa opção requer a plataforma Linux `1.4.0` ou posterior.	Entre 32 GB e 120 GB em incrementos de 8 GB	Linux

Para tarefas que usam o tipo de inicialização do EC2 ou externa, os valores compatíveis de CPU da tarefa são entre 128 unidades de CPU (0,125 vCPUs) EC2 196.608 unidades de CPU (192 vCPUs).

Para especificar o valor da memória em GB, insira GB após o valor. Por exemplo, para definir o Valor da memória como 3 GB, insira 3 GB.

> **Note**
>
> Os parâmetros de CPU e memória em nível de tarefa são ignorados para contêineres do Windows.

8. Em Network mode (Modo de rede), escolha o modo de rede a ser usado. O padrão é o modo awsvpc. Para obter mais informações, consulte Redes de tarefas no Amazon ECS.

   Se escolher bridge, em Mapeamentos de portas, em Porta do host, insira o número da porta na instância de contêiner a ser reservada para o contêiner.

9. (Opcional) Expanda a seção Perfis de tarefa para configurar os perfis do AWS Identity and Access Management (IAM) para a tarefa:

    a. Em Task role (Perfil da tarefa), escolha um perfil do IAM para atribuir à tarefa. Um perfil do IAM de tarefa fornece permissões aos contêineres em uma tarefa para chamar operações de API da AWS.

    b. Em Perfil de execução de tarefas, escolha o perfil.

    Para obter mais informações sobre quando usar um perfil de execução de tarefas, consulte the section called “Função do IAM de execução de tarefas”. Se você não precisar do perfil, escolha Nenhum.

10. (Opcional) Expanda a seção Posicionamento de tarefas para adicionar restrições de posicionamento. Restrições de posicionamento de tarefas permitem filtrar instâncias de contêiner utilizadas para o posicionamento das suas tarefas usando atributos integrados ou personalizados.

11. (Opcional) Expanda a seção Injeção de falhas para habilitar a injeção de falhas. A injeção de falhas permite testar como sua aplicação responde a certos cenários de comprometimento.

12. Para cada contêiner ser definido em sua definição de tarefa, conclua as seguintes etapas.

    a. Em Name (Nome), insira um nome para o contêiner.

    b. Em Image URI (URI da imagem), especifique a imagem a ser usada para iniciar um contêiner. As imagens no registro da Galeria pública do Amazon ECR podem ser especificadas usando apenas o nome do registro do Amazon ECR Public. Por exemplo, se `public.ecr.aws/ecs/amazon-ecs-agent:latest` for especificado, o contêiner do Amazon Linux hospedado na Galeria pública do Amazon ECR será usado. Para todos os outros repositórios, especifique o repositório usando os formatos `repository-url/image:tag` ou `repository-url/image@digest`.

    c. Se sua imagem estiver em um registro privado fora do Amazon ECR, em Registro privado, ative a Autenticação de registro privado. Em seguida, em ARN ou nome do Secrets Manager, insira o nome do recurso da Amazon (ARN) do segredo.

    d. Em Contêiner essencial, se a definição de tarefa tiver dois ou mais contêineres definidos, é possível especificar se o contêiner deve ser considerado essencial. Se um contêiner estiver marcado como Essencial, caso esse contêiner pare, a tarefa será interrompida. Cada definição de tarefa deve conter pelo menos um contêiner essencial.

e. Um mapeamento de porta permite que o contêiner acesse portas no host para enviar ou receber tráfego. Em Port mappings (Mapeamentos de porta), siga um destes procedimentos:

   - Quando você usar o modo de rede awsvpc, em Container port (Porta do contêiner) e Protocol (Protocolo), escolha o mapeamento de porta a ser usado para o contêiner.
   - Quando você usar o modo de rede bridge, em Container port (Porta do contêiner) e Protocol (Protocolo), especifique o mapeamento de porta a ser usado para o contêiner.

   Escolha Add more port mappings (Adicionar mais mapeamentos de portas) para especificar mapeamentos de porta de contêiner adicionais.

f. Para dar ao contêiner acesso somente de leitura ao sistema de arquivos raiz, em Sistema de arquivos raiz somente leitura, selecione Somente leitura.

g. (Opcional) Para definir os limites de CPU, GPU e memória em nível de contêiner que sejam diferentes dos valores em nível de tarefa, em Limites de alocação de recursos, faça o seguinte:

   - Em CPU, insira o número de unidades de CPU que o agente de contêiner do Amazon ECS reserva para o contêiner.
   - Em GPU, insira o número de unidades de GPU para a instância de contêiner.

     Uma instância do Amazon EC2 compatível com GPU tem 1 unidade de GPU para cada GPU. Para obter mais informações, consulte the section called “Definições de tarefa para workloads de GPU”.
   - Em Limite rígido de memória, insira a quantidade de memória, em GB, para fornecer ao contêiner. Se o contêiner tentar exceder o limite rígido, o contêiner será interrompido.
   - O daemon 20.10.0 ou posterior do Docker reserva um mínimo de 6 mebibytes (MiB) de memória para um contêiner, portanto, não especifique menos de 6 MiB de memória para os contêineres.

     O daemon 19.03.13-ce ou anterior do Docker reserva um mínimo de 4 MiB de memória para um contêiner, portanto, não especifique menos de 4 MiB de memória para os contêineres.

- Em Limite flexível de memória, insira o limite flexível (em GB) de memória a ser reservada para o contêiner.

  Quando a memória do sistema estiver em contenção, o Docker tentará manter a memória do contêiner dentro desse limite flexível. Se não especificar memória no nível de tarefa, será necessário especificar um número inteiro diferente de zero para um ou ambos os Limite rígido de memória e limite flexível de memória. Se você especificar ambos, o Limite rígido de memória deverá ser maior que o Limite flexível de memória.

  Não há suporte para esse recurso em contêineres do Windows.

h. (Opcional) Expanda a seção Variáveis de ambiente para especificar variáveis de ambiente a serem injetadas no contêiner. É possível especificar variáveis de ambiente individualmente usando pares de chave-valor ou em massa, especificando um arquivo de variável de ambiente hospedado em um bucket do Amazon S3. Para obter informações sobre como formatar um arquivo de variáveis de ambiente, consulte Transferência de uma variável de ambiente individual para um contêiner do Amazon ECS.

   Ao especificar uma variável de ambiente para o armazenamento de segredos, em Key, insira o nome do segredo. Em seguida, para ValueFrom, insira o ARN completo do segredo do Systems Manager Parameter Store ou do segredo do Secrets Manager.

i. (Opcional) Selecione a opção Use log collection (Usar coleção de logs) para especificar uma configuração de log. Para cada driver de log disponível, há opções de driver de log a serem especificadas. A opção padrão envia os logs do contêiner ao Amazon CloudWatch Logs. As outras opções de driver de log são configuradas usando o AWS FireLens. Para obter mais informações, consulte Envio de logs do Amazon ECS para um serviço da AWS ou para uma AWS Partner.

   Veja a seguir a descrição de cada destino de log de contêiner mais detalhadamente.

   - Amazon CloudWatch: configura a tarefa para enviar logs de contêiner ao CloudWatch Logs. São fornecidas as opções de driver de log padrão, que criam um grupo de logs do CloudWatch em seu nome. Para especificar um nome de grupo de logs diferente, altere os valores da opção de driver.
   - Exportar logs para o Splunk: configure a tarefa para enviar os log do contêiner ao driver do Splunk que envia os logs para a serviço remoto. Você deve inserir o URL do

serviço Web Splunk. O token do Splunk é especificado como uma opção secreta, pois ele pode ser tratado como dados confidenciais.

- Exportar logs para o Amazon Data Firehose: configure a tarefa para enviar logs de contêiner ao Firehose. São fornecidas as opções de driver de log padrão, que envia logs para um fluxo de entrega do Firehose. Para especificar um nome de fluxo de entrega diferente, altere os valores da opção de driver.
- Exportar logs para o Amazon Kinesis Data Streams: configure a tarefa para enviar logs de contêiner ao Kinesis Data Streams. São fornecidas as opções de driver de log padrão, que envia logs a um fluxo do Kinesis Data Streams. Para especificar um nome de transmissão diferente, altere os valores da opção de driver.
- Exportar logs para o Amazon OpenSearch Service: configure a tarefa para enviar logs de contêiner a um domínio do OpenSearch Service. As opções de driver de log devem ser fornecidas.
- Exportar logs para o Amazon S3: configure a tarefa para enviar logs de contêiner a um bucket do Amazon S3. As opções de driver de log padrão são fornecidas, mas você deve especificar um nome de bucket válido do Amazon S3.

j. (Opcional) Configure parâmetros adicionais do contêiner.

Para configurar essa opção	Faça o seguinte	
Política de reinicialização Essas opções definem uma política de reinicialização para reiniciar um contêiner quando ele sair.	Expanda Restart policy e configure os seguintes itens: • Para habilitar uma política de reinicialização para o contêiner, ative Enable Restart policy. • Para Ignored exit codes, especifique uma lista separada por vírgulas de códigos de saída de contêineres inteiros. Se o contêiner sair com qualquer um dos códigos de saída especificados, o Amazon ECS não tentará reiniciá-lo. Se nada for especificado, o Amazon ECS não ignorará nenhum código de saída. • Em Attempt reset period, especifique um período inteiro, em segundos, pelo qual o contêiner deve ser executado antes que	

Para configurar essa opção	Faça o seguinte	
	uma reinicialização possa ser tentada no caso de uma saída. O Amazon ECS pode tentar reiniciar um contêiner somente uma vez a cada período de redefinição de tentativa em segundos. Se nada for especificado, o contêiner deverá ser executado por 300 segundos antes que uma reinicialização possa ser tentada.	

Para configurar essa opção	Faça o seguinte	
HealthCheck Esses comandos determinam se um contêiner está íntegro. Para obter mais informaçõ es, consulte Como determinar a integridade das tarefas do Amazon ECS usando verificaç ões de integridade de contêineres.	Expanda Verificação de integridade e configure os itens a seguir: • Em Command (Comando), insira uma lista de comandos separados por vírgulas. Os comandos podem ser iniciados por `CMD` para executar os argumentos de comando diretamen te ou por `CMD-SHELL` para executar o comando com o shell padrão do contêiner. Se nenhuma for especific ado, `CMD` será usada. • Em Interval (Interval o), insira o número de segundos entre cada verificação de integrida de. Os valores válidos estão entre 5 e 30. • Em Timeout (Tempo limite), insira o período de espera (em segundos) para que uma verificação de integridade seja bem-	

Para configurar essa opção	Faça o seguinte	
	sucedida antes de ser considerada uma falha. Os valores válidos estão entre 2 e 60. • Em Start period (Período de início), insira o período de espera (em segundos) para que um contêiner seja inicializado antes da execução dos comandos de verificação de integridade. Os valores válidos estão entre 0 e 300. • Em Retries (Repetições), insira o número de vezes para repetir os comandos de verificação de integridade quando houver uma falha. Os valores válidos estão entre 1 e 10.	

Para configurar essa opção	Faça o seguinte	
Ordem das dependências de startup Essa opção define dependências para startup e desligamento do contêiner. Um contêiner pode conter várias dependências.	Expanda a Ordem das dependências de startup e, em seguida, configure o seguinte: a. Escolha Adicionar dependência de contêiner. b. Em Contêiner, escolha o contêiner. c. Em Condição, escolha a condição de dependência de startup. Para acrescentar uma dependência adicional, escolha Adicionar dependência de contêiner.	

Para configurar essa opção	Faça o seguinte	
Tempos limite de contêineres Essas opções determinam quando iniciar e parar um contêiner.	Expanda Tempos limite de contêineres e, em seguida, configure o seguinte: • Para configurar o tempo de espera antes de desistir de resolver dependências de um contêiner, em Tempo limite de início, insira o número de segundos. • Para configurar o tempo de espera antes de o contêiner ser interrompido se não for encerrado normalmente por conta própria, em Tempo limite de interrupção, insira o número de segundos.	

Para configurar essa opção	Faça o seguinte	
Configurações de rede de contêineres Essas opções determinam se a rede deve ser usada em um contêiner.	Expanda Configurações de rede de contêineres e, em seguida, configure o seguinte: • Para desativar a rede de contêineres, selecione Desativar a rede. • Para configurar os endereços IP do servidor DNS que são apresentados ao contêiner, nos Servidores de DNS, insira o endereço IP de cada servidor em uma linha separada. • Para configurar domínios de DNS para pesquisar nomes de host que não são totalmente qualificados e que sejam apresentados ao contêiner, em Domínios de pesquisa de DNS, insira cada domínio em uma linha separada.	

Para configurar essa opção	Faça o seguinte	
	O padrão é **`^[a-zA-Z0-9-.]{0,253}[a-zA-Z0-9]$`** . • Para configurar o nome do host do contêiner , em Nome do host, insira o nome do host do contêiner. • Para adicionar nomes de host e mapeamentos de endereço IP que são anexados ao arquivo `/etc/hosts` no contêiner, escolha Adicionar host extra e, em seguida, em Nome do host e Endereço IP, insira o nome do host e o endereço IP.	

Para configurar essa opção	Faça o seguinte	
Configuração da Docker Essas configurações substituem os valores no Dockerfile.	Expanda Configuração do Docker e configure os seguintes itens: • Em Comando, insira um comando executável para um contêiner. Esse parâmetro é mapeado para `Cmd` na seção Criar um contêiner da API remota do Docker e a opção `COMMAND` para `docker run`. Esse parâmetro substitui a instrução `CMD` em um Dockerfile. • Em Ponto de entrada, insira o ENTRYPOINT do Docker que é transferido para o contêiner. Esse parâmetro é mapeado para `Entrypoint` na seção Criar um contêiner da API remota do Docker e a opção `--entrypoint` para `docker`	

Para configurar essa opção	Faça o seguinte	
	`run`. Esse parâmetro substitui a instrução `ENTRYPOINT` em um Dockerfile. • Em Diretório de trabalho, insira o diretório no qual o contêiner executará qualquer ponto de entrada e instruções de comando fornecidas. Esse parâmetro é mapeado para `WorkingDir` na seção Criar um contêiner da API remota do Docker e a opção `--workdir` para `docker run`. Esse parâmetro substitui a instrução `WORKDIR` em um Dockerfile.	

Para configurar essa opção	Faça o seguinte
Limites de recurso (Ulimits) Esses valores substitue m a configuração de cota de recursos padrão do sistema operacional. Esse parâmetro é mapeado para `Ulimits` na seção Criar um Contêiner da API remota do Docker e a opção `--ulimit` para docker run.	Expanda Limites de recursos (ulimits) e escolha Adicionar ulimit. Em Nome do limite, escolha o limite. Em seguida, para Limite flexível e Limite rígido, insira os valores. Para adicionar ulimits, escolha Adicionar ulimit.
Rótulos do Docker Essa opção adiciona metadados ao seu contêiner. Esse parâmetro é mapeado para `Labels` na seção Criar um Contêiner da API remota do Docker e a opção `--label` para docker run.	Expanda Rótulos do Docker, escolha Adicionar par de chave-valor e insira a Chave e o Valor. Para adicionar outros rótulos do Docker, escolha Adicionar par de chave-valor.

k. (Opcional) Escolha Add more containers (Adicionar mais contêineres) para incluir contêineres adicionais à definição de tarefa.

13. (Opcional) A seção Armazenamento é usada para expandir a quantidade de armazenamento temporário para tarefas hospedadas no Fargate. Também é possível usar essa seção para adicionar uma configuração de volume de dados à tarefa.

- Para expandir o armazenamento temporário disponível além do valor padrão de 20 gibibytes (GiB) para as tarefas do Fargate, em Amount (Quantidade), insira um valor até 200 GiB.

14. (Opcional) Para adicionar uma configuração de volume de dados à definição da tarefa, escolha Adicionar volume e siga as etapas a seguir.

    a. Em Volume name (Nome do volume), insira um nome para o volume de dados. O nome do volume de dados é usado na ocasião da criação de um ponto de montagem de contêiner.

    b. Em Configuração de volume, selecione se deseja configurar o volume ao criar a definição de tarefa ou durante a implantação.

    > **Note**
    >
    > Os volumes que podem ser configurados ao criar uma definição de tarefa incluem montagem vinculada, Docker, Amazon EFS e Amazon FSx para Windows File Server. Os volumes que podem ser configurados na implantação ao executar uma tarefa ou ao criar ou atualizar um serviço incluem o Amazon EBS.

    c. Em Tipo de volume, selecione um tipo de volume compatível com o tipo de configuração selecionado e configure o tipo de volume.

Tipo de volume	Etapas	
Montagem vinculada	a. Escolha Add mount point (Adicionar ponto de montagem) e configure o seguinte: • Em Conteiner (Contêiner), escolha o contêiner para o ponto de montagem.	

Tipo de volume	Etapas	
	• Em Source volume (Volume de origem), escolha o volume de dados a ser montado no contêiner. • Em Conteiner path (Caminho do contêiner), insira o caminho no contêiner no qual montar o volume. • Em Somente leitura, selecione se o contêiner tem acesso somente leitura ao volume. b. Para pontos de montagem adicionais, escolha Add mount point (Adicionar ponto de montagem).	

Tipo de volume	Etapas	
EFS	a. Em File system ID (ID do sistema de arquivos), escolha o ID do sistema de arquivos do Amazon EFS. b. (Opcional) Em Root directory (Diretório raiz), insira o diretório dentro do sistema de arquivos do Amazon EFS que deve ser montado como o diretório raiz dentro do host. Caso esse parâmetro seja omitido, a raiz do volume Amazon EFS será usada. Caso planeje usar um ponto de acesso do EFS, deixe este campo em branco. c. (Opcional) Em Access point (Ponto de acesso), escolha o ID do ponto de acesso a ser usado. d. (Opcional) Para criptografar os dados entre o sistema de arquivos do Amazon EFS e o host do Amazon ECS ou para usar o perfil de	

Tipo de volume	Etapas	
	execução de tarefas quando montar o volume, escolha Advanced configurations (Configurações avançadas) e configure o seguinte: • Para criptografar os dados entre o sistema de arquivos do Amazon EFS e o host do Amazon ECS, selecione Transit encryption (Criptografia de trânsito) e, em seguida, em Port (Porta), insira a porta a ser usada quando forem enviados dados criptografados entre o host do Amazon ECS e o servidor do Amazon EFS. Se você não especificar uma porta de criptografia em trânsito, será usada a estratégia de seleção de porta usada pelo assistente de montagem do Amazon EFS. Para mais informações, consulte Auxiliar de Montagem EFS no Guia de Usuário	

Tipo de volume	Etapas	
	Amazon Elastic File System. • Para usar o perfil do IAM da tarefa do Amazon ECS definido em uma definição de tarefa quando montar o sistema de arquivos do Amazon EFS, selecione IAM authorization (Autorização do IAM). e. Escolha Add mount point (Adicionar ponto de montagem) e configure o seguinte: • Em Conteiner (Contêiner), escolha o contêiner para o ponto de montagem. • Em Source volume (Volume de origem), escolha o volume de dados a ser montado no contêiner. • Em Conteiner path (Caminho do contêiner), insira o caminho no contêiner no qual montar o volume.	

Tipo de volume	Etapas	
	• Em Somente leitura, selecione se o contêiner tem acesso somente leitura ao volume. f. Para pontos de montagem adicionais, escolha Add mount point (Adicionar ponto de montagem).	

Tipo de volume	Etapas	
Docker	a. Em Driver, insira a configuração de volume do Docker. Os contêineres do Windows só são compatíveis com o uso do driver local. Para usar montagens bind, especifique um host. b. Em Scope (Escopo), escolha o ciclo de vida do volume. • Para que o ciclo de vida dure quando a tarefa for iniciada e interrompida, escolha Task (Tarefa). • Para que o volume persista após a interrupção da tarefa, escolha Shared (Compartilhado). c. Escolha Add mount point (Adicionar ponto de montagem) e configure o seguinte: • Em Conteiner (Contêiner), escolha o contêiner para o ponto de montagem.	

Tipo de volume	Etapas	
	• Em Source volume (Volume de origem), escolha o volume de dados a ser montado no contêiner. • Em Conteiner path (Caminho do contêiner), insira o caminho no contêiner no qual montar o volume. • Em Somente leitura, selecione se o contêiner tem acesso somente leitura ao volume. d. Para pontos de montagem adicionais, escolha Add mount point (Adicionar ponto de montagem).	

Tipo de volume	Etapas	
FSx para Windows File Server	a. Em ID do sistema de arquivos, escolha o ID do sistema de arquivos do FSx para Windows File Server. b. Em Diretório raiz, insira o diretório no sistema de arquivos do FSx para Windows File Server que deve ser montado como o diretório raiz no host. c. Em Parâmetro de credencial, escolha como as credenciais são armazenadas. • Para usar o AWS Secrets Manager, insira o nome do recurso da Amazon (ARN) de um segredo do Secrets Manager. • Para usar o AWS Systems Manager, insira o nome do recurso da Amazon (ARN) de um parâmetro do Systems Manager. d.	

Tipo de volume	Etapas	
	Em Domínio, insira o nome de domínio totalmente qualificado hospedado por um diretório do AWS Directory Service for Microsoft Active Directory (AWS Managed Microsoft AD) ou de um Active Directory do EC2 auto-hospedado. e. Escolha Add mount point (Adicionar ponto de montagem) e configure o seguinte: • Em Conteiner (Contêiner), escolha o contêiner para o ponto de montagem. • Em Source volume (Volume de origem), escolha o volume de dados a ser montado no contêiner. • Em Conteiner path (Caminho do contêiner), insira o caminho no contêiner no qual montar o volume. •	

Tipo de volume	Etapas	
	Em Somente leitura, selecione se o contêiner tem acesso somente leitura ao volume. f. Para pontos de montagem adicionais, escolha Add mount point (Adicionar ponto de montagem).	

Tipo de volume	Etapas	
Amazon EBS	a. Escolha Add mount point (Adicionar ponto de montagem) e configure o seguinte: • Em Conteiner (Contêiner), escolha o contêiner para o ponto de montagem. • Em Source volume (Volume de origem), escolha o volume de dados a ser montado no contêiner. • Em Conteiner path (Caminho do contêiner), insira o caminho no contêiner no qual montar o volume. • Em Somente leitura, selecione se o contêiner tem acesso somente leitura ao volume. b. Para pontos de montagem adicionais, escolha Add mount point (Adicionar ponto de montagem).	

Tipo de volume	Etapas	

15. Para adicionar um volume de outro contêiner, escolha Adicionar volume de e configure o seguinte:

    - Em Contêiner, escolha o contêiner.
    - Em Origem, escolha o contêiner que tem o volume que você deseja montar.
    - Em Somente leitura, selecione se o contêiner tem acesso somente leitura ao volume.

16. (Opcional) Para definir as configurações de rastreamento e coleção de métricas de aplicações usando a integração AWS Distro for OpenTelemetry, expanda Monitoramento e selecione Usar coleção de métricas para coletar e enviar métricas das tarefas ao Amazon CloudWatch ou ao Amazon Managed Service for Prometheus. Quando essa opção é selecionada, o Amazon ECS cria um arquivo associado de contêiner do AWS Distro for OpenTelemetry pré-configurado para enviar as métricas da aplicação. Para obter mais informações, consulte Correlação do desempenho da aplicação do Amazon ECS usando métricas de aplicações.

    a. Quando Amazon CloudWatch está selecionado, suas métricas de aplicação personalizadas são roteadas para o CloudWatch como métricas personalizadas. Para obter mais informações, consulte Exportação de métricas de aplicações para o Amazon CloudWatch.

    > **Important**
    >
    > Ao exportar métricas de aplicações para o Amazon CloudWatch, sua definição de tarefa requer uma função do IAM de tarefa com as permissões necessárias. Para obter mais informações, consulte Permissões obrigatórias do IAM para integração da distribuição da AWS do OpenTelemetry com o Amazon CloudWatch.

    b. Quando Amazon Managed Service for Prometheus (Prometheus libraries instrumentation) (Amazon Managed Service for Prometheus [instrumentação das bibliotecas Prometheus]) está selecionado, as métricas de CPU, memória, rede e armazenamento no nível da tarefa e as métricas de aplicação personalizadas são roteadas para o Amazon Managed Service for Prometheus. Em Endpoint de gravação

remoto do espaço de trabalho, insira o URL do endpoint de gravação remoto para o espaço de trabalho do Prometheus. Em Destino de rascunho, insira o host e a porta que o coletor Distro for OpenTelemetry da AWS pode usar para extrair dados de métricas. Para obter mais informações, consulte Exportação de métricas de aplicações para o Amazon Managed Service for Prometheus.

> **Important**
>
> Ao exportar métricas de aplicações para o Amazon Managed Service for Prometheus, sua definição de tarefa requer uma função do IAM de tarefa com as permissões necessárias. Para obter mais informações, consulte Permissões obrigatórias do IAM para integração da distribuição da AWS do OpenTelemetry com o Amazon Managed Service for Prometheus.

c. Ao selecionar Amazon Managed Service for Prometheus (instrumentação do OpenTelemetry), as métricas de CPU, memória, rede e armazenamento em nível de tarefa e as métricas de aplicação personalizadas são encaminhadas ao Amazon Managed Service for Prometheus. Em Endpoint de gravação remoto do espaço de trabalho, insira o URL do endpoint de gravação remoto para o espaço de trabalho do Prometheus. Para obter mais informações, consulte Exportação de métricas de aplicações para o Amazon Managed Service for Prometheus.

> **Important**
>
> Ao exportar métricas de aplicações para o Amazon Managed Service for Prometheus, sua definição de tarefa requer uma função do IAM de tarefa com as permissões necessárias. Para obter mais informações, consulte Permissões obrigatórias do IAM para integração da distribuição da AWS do OpenTelemetry com o Amazon Managed Service for Prometheus.

17. (Opcional) Expanda a seção Tags para adicionar tags à definição da tarefa, como pares de chave-valor.

    - [Adicionar uma etiqueta] Escolha Add tag (Adicionar etiqueta) e faça o seguinte:
      - Em Chave, insira o nome da chave.
      - Em Valor insira o valor da chave.
    - [Remover uma tag] Ao lado da tag, escolha Remove tag (Remover tag).

18. Escolha Criar para registrar a definição de tarefa.

Amazon ECS console JSON editor

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Task definitions (Definições de tarefa).
3. No menu Criar definição de tarefa, escolha Criar definição de tarefa com JSON.
4. Na caixa do editor de JSON, edite o arquivo JSON.

   O JSON deve ser aprovado nas verificações de validação especificadas em the section called “Validação de JSON”.
5. Escolha Criar.

Atualizar uma definição de tarefa do Amazon ECS usando o console

Uma revisão de definição de tarefa é uma cópia da definição de tarefa atual com os novos valores de parâmetro substituindo os existentes. Todos os parâmetros que você não modificar estão na nova revisão.

Para atualizar uma definição de tarefa, crie uma revisão de definição de tarefa. Se a definição de tarefa for usada em um serviço, será necessário atualizá-lo para usar a definição de tarefa atualizada.

Ao criar uma revisão, você pode modificar as propriedades do contêiner e as propriedades do ambiente a seguir.

- URI da imagem do contêiner
- Mapeamentos de porta
- Variáveis de ambiente
- Tamanho da tarefa
- Tamanho do contêiner
- Função da tarefa
- Função de execução de tarefas

- Pontos de montagem de volumes e contêineres
- Registro privado

Validação de JSON

O editor de JSON do console do Amazon ECS valida o seguinte no arquivo JSON:

- O arquivo é um arquivo JSON válido
- O arquivo não contém nenhuma chave estranha
- O arquivo contém o parâmetro `familyName`
- Há pelo menos uma entrada em `containerDefinitions`

Procedimento

Amazon ECS console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, selecione a região que contém a definição de tarefa.
3. No painel de navegação, escolha Task definitions (Definições de tarefa).
4. Escolha a definição da tarefa.
5. Selecione a revisão da definição de tarefa e escolha Criar nova revisão, Criar nova revisão.
6. Na página Create new task definition revision (Criar nova revisão da definição de tarefa), faça as alterações. Por exemplo, para alterar as definições do contêiner existentes (como a imagem do contêiner, limites de memória ou mapeamentos de porta), selecione o contêiner e faça as alterações.
7. Verifique as informações e escolha Criar.
8. Se a sua definição de tarefa for usada em um serviço, atualize o serviço com a definição de tarefa atualizada. Para obter mais informações, consulte Atualizar um serviço do Amazon ECS.

Amazon ECS console JSON editor

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Task definitions (Definições de tarefa).

3. Escolha Create new revision (Criar nova revisão), Create new revision with JSON (Criar nova revisão com JSON).
4. Na caixa do editor de JSON, edite o arquivo JSON.

   O JSON deve ser aprovado nas verificações de validação especificadas em the section called “Validação de JSON”.
5. Escolha Criar.

Cancelar registro de uma revisão de definição de tarefa do Amazon ECS usando o console

É possível cancelar o registro da revisão de definição de tarefa, de modo que ela não seja mais exibida nas chamadas de API `ListTaskDefinition` ou no console quando quiser executar uma tarefa ou atualizar um serviço.

Quando você cancela o registro de uma revisão de definição de tarefa, ela é marcada imediatamente como `INACTIVE`. Tarefas e serviços existentes que fazem referência a uma revisão de definição de tarefa `INACTIVE` continuam a ser executados sem interrupções. Serviços existentes que fazem referência a uma revisão de definição de tarefa `INACTIVE` ainda podem aumentar ou diminuir na escala vertical modificando-se a contagem desejada do serviço.

Não é possível usar uma revisão de definição de tarefa `INACTIVE` para executar novas tarefas ou criar novos serviços. Você também não pode atualizar um serviço existente para fazer referência a uma revisão de definição de tarefa `INACTIVE` (embora possa haver uma janela de até 10 minutos após o cancelamento do registro no qual essas restrições ainda não tenham entrado em vigor).

> **Note**
>
> Quando você cancela o registro de todas as revisões em uma família de tarefas, a família de definição de tarefa é movida para a lista `INACTIVE`. Adicionar uma nova revisão de uma definição de tarefa de `INACTIVE` fará com que a família de definição de tarefa retorne para a lista `ACTIVE`.
> Neste momento, as revisões de definição de tarefa `INACTIVE` permanecem detectáveis na sua conta indefinidamente. No entanto, esse comportamento está sujeito a alterações no futuro. Portanto, você não deve confiar em revisões de definição de tarefa `INACTIVE` que persistem além do ciclo de vida de quaisquer tarefas e serviços associados.

Pilhas do AWS CloudFormation

O comportamento a seguir é aplicável às definições de tarefa criadas no novo console do Amazon ECS antes de 12 de janeiro de 2023.

Ao criar uma definição de tarefa, o console do Amazon ECS cria automaticamente uma pilha do CloudFormation com um nome que começa com `ECS-Console-V2-TaskDefinition-`. Se tiver usado a AWS CLI ou um AWS SDK para cancelar o registro da definição de tarefa, você deverá excluir manualmente a pilha de definições de tarefa. Para obter mais informações, consulte Excluir uma pilha no Guia do usuário do AWS CloudFormation.

As definições de tarefa criadas após 12 de janeiro de 2023 não têm uma pilha do CloudFormation criada automaticamente para elas.

Procedimento

Para cancelar o registro de uma nova definição de tarefa (console do Amazon ECS)

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, selecione a região que contém a definição de tarefa.
3. No painel de navegação, escolha Task definitions (Definições de tarefa).
4. Na página Task definitions (Definições de tarefa), escolha a família de definições de tarefa que contém uma ou mais revisões cujos registros você quer cancelar.
5. Na página Nome da definição da tarefa, selecione as revisões a serem excluídas e escolha Ações, Cancelar registro.
6. Verifique as informações na janela Deregister (Cancelar registro) e escolha Deregister (Cancelar registro) para terminar.

Exclusão de uma revisão de definição de tarefa do Amazon ECS usando o console

Quando não precisar mais de uma revisão específica da definição de tarefa no Amazon ECS, você pode excluir a revisão da definição da tarefa.

Quando você exclui uma revisão de definição de tarefa, ela passa imediatamente de `INACTIVE` para `DELETE_IN_PROGRESS`. Tarefas e serviços existentes que fazem referência a uma revisão de definição de tarefa `DELETE_IN_PROGRESS` continuam a ser executados sem interrupções.

Não é possível usar uma revisão de definição de tarefa DELETE_IN_PROGRESS para executar novas tarefas ou criar novos serviços. Também não é possível atualizar um serviço existente para fazer referência a uma revisão de definição de tarefa DELETE_IN_PROGRESS.

Quando você exclui todas as revisões de definição da tarefa INACTIVE, o nome da definição da tarefa não é exibido no console e não é retornado na API. Se uma revisão de definição de tarefa estiver no estado DELETE_IN_PROGRESS, o nome da definição de tarefa é exibido no console e retornado na API. O nome da definição da tarefa é mantido pelo Amazon ECS e a revisão será incrementada na próxima vez que você criar uma definição de tarefa com esse nome.

Recursos do Amazon ECS que podem bloquear uma exclusão

Uma solicitação de exclusão da definição de tarefa não será concluída se houver algum recurso do Amazon ECS que dependa da revisão da definição de tarefa. Os recursos a seguir podem impedir que uma definição de tarefa seja excluída:

- Tarefas autônomas do Amazon ECS: a definição da tarefa é necessária para que a tarefa permaneça íntegra.
- Tarefas de serviço do Amazon ECS: a definição da tarefa é necessária para que a tarefa permaneça íntegra.
- Implantações e conjuntos de tarefas do serviço do Amazon ECS: a definição da tarefa é necessária quando um evento de escalabilidade é iniciado para uma implantação ou conjunto de tarefas do Amazon ECS.

Se sua definição de tarefa permanecer no estado DELETE_IN_PROGRESS, será possível usar o console ou a AWS CLI para identificar e interromper os recursos que bloqueiem a exclusão da definição de tarefa.

Exclusão da definição de tarefa após a remoção do recurso bloqueado

As regras a seguir se aplicam depois que você remove os recursos que bloqueiam a exclusão da definição da tarefa:

- Tarefas do Amazon ECS: a exclusão da definição da tarefa pode levar até uma hora para ser concluída após a interrupção da tarefa.
- Implantações e conjuntos de tarefas do serviço do Amazon ECS: a exclusão da definição da tarefa pode levar até 24 horas para ser concluída após a exclusão da implantação ou do conjunto de tarefas.

Procedimento

Para excluir definições de tarefa (console do Amazon ECS)

É necessário cancelar o registro de uma revisão da definição de tarefa antes de excluí-la. Para obter mais informações, consulte the section called “Cancelamento de registro de uma revisão de definição de tarefa usando o console”.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, selecione a região que contém a definição de tarefa.
3. No painel de navegação, escolha Task definitions (Definições de tarefa).
4. Na página Definições de tarefa, escolha a família de definições de tarefa que contém uma ou mais revisões cujos registros você quer excluir.
5. Na página Nome da definição de tarefa, selecione as revisões a serem excluídas e escolha Ações, Excluir.

   Se a opção Excluir não estiver disponível, você deve cancelar o registro da definição da tarefa.
6. Verifique as informações na caixa de confirmação Excluir e escolha Excluir para finalizar.

Casos de uso de definição de tarefa do Amazon ECS

Saiba mais sobre como escrever definições de tarefas para vários serviços e recursos da AWS.

Dependendo da workload, há certos parâmetros de definição de tarefa que precisam ser definidos. Além disso, para o tipo de execução do EC2, você precisa escolher instâncias específicas projetadas para a workload.

Tópicos

- Definições de tarefa do Amazon ECS para workloads de GPU
- Definições de tarefas do Amazon ECS para workloads de transcodificação de vídeo
- Definições de tarefa do Amazon ECS para workloads de machine learning do AWS Neuron
- Definições de tarefa do Amazon ECS para instâncias de aprendizado profundo
- Definições de tarefa do Amazon ECS para workloads do ARM de 64 bits
- Envio de logs do Amazon ECS para o CloudWatch
- Envio de logs do Amazon ECS para um serviço da AWS ou para uma AWS Partner

- Uso de imagens de contêiner que não são da AWS no Amazon ECS
- Reiniciar contêineres individuais em tarefas do Amazon ECS com políticas de reinicialização de contêineres
- Transferência de dados confidenciais para um contêiner do Amazon ECS

Definições de tarefa do Amazon ECS para workloads de GPU

O Amazon ECS é compatível com workloads que utilizam GPUs quando você cria clusters com instâncias de contêiner compatíveis com GPU. As instâncias de contêiner baseadas em GPU do Amazon EC2 que usam os tipos de instância p2, p3, p5, g3, g4 e g5 fornecem acesso a GPUs NVIDIA. Para obter mais informações, consulte Instâncias com computação acelerada do Linux no Guia de tipos de instância do Amazon EC2.

O Amazon ECS fornece uma AMI otimizada para GPU que é fornecida com drivers de kernel NVIDIA pré-configurados e um runtime de GPU do Docker. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.

É possível designar várias GPUs na sua definição de tarefa para consideração do posicionamento de tarefas em um nível de contêiner. O Amazon ECS programa instâncias de contêineres disponíveis compatíveis com GPU, atribuindo GPUs físicas a contêineres adequados para proporcionar a performance ideal.

Os seguintes tipos de instância do Amazon EC2 baseados em GPU são compatíveis. Para obter mais informações, consulte Instâncias P2 do Amazon EC2, Instâncias P3 do Amazon EC2, Instâncias P4d do Amazon EC2, Instâncias P5 do Amazon EC2, Instâncias G3 do Amazon EC2, Instâncias G4 do Amazon EC2, Instâncias G5 do Amazon EC2, Instâncias G6 do Amazon EC2 e Instâncias G6e do Amazon EC2.

Tipo de instância	GPUs	Memória da GPU (GiB)	vCPUs	Memória (GiB)
p3.2xlarge	1	16	8	61
p3.8xlarge	4	64	32	244
p3.16xlarge	8	128	64	488
p3dn.24xlarge	8	256	96	768

Tipo de instância	GPUs	Memória da GPU (GiB)	vCPUs	Memória (GiB)
p4d.24xlarge	8	320	96	1152
p5.48xlarge	8	640	192	2048
g3s.xlarge	1	8	4	30.5
g3.4xlarge	1	8	16	122
g3.8xlarge	2	16	32	244
g3.16xlarge	4	32	64	488
g4dn.xlarge	1	16	4	16
g4dn.2xlarge	1	16	8	32
g4dn.4xlarge	1	16	16	64
g4dn.8xlarge	1	16	32	128
g4dn.12xlarge	4	64	48	192
g4dn.16xlarge	1	16	64	256
g5.xlarge	1	24	4	16
g5.2xlarge	1	24	8	32
g5.4xlarge	1	24	16	64
g5.8xlarge	1	24	32	128
g5.16xlarge	1	24	64	256
g5.12xlarge	4	96	48	192
g5.24xlarge	4	96	96	384
g5.48xlarge	8	192	192	768

Tipo de instância	GPUs	Memória da GPU (GiB)	vCPUs	Memória (GiB)
g6.xlarge	1	24	4	16
g6.2xlarge	1	24	8	32
g6.4xlarge	1	24	16	64
g6.8xlarge	1	24	32	128
g6.16.xlarge	1	24	64	256
g6.12xlarge	4	96	48	192
g6.24xlarge	4	96	96	384
g6.48xlarge	8	192	192	768
g6.metal	8	192	192	768
gr6.4xlarge	1	24	16	128
g6e.xlarge	1	48	4	32
g6e.2xlarge	1	48	8	64
g6e.4xlarge	1	48	16	128
g6e.8xlarge	1	48	32	256
g6e16.xlarge	1	48	64	512
g6e12.xlarge	4	192	48	384
g6e24.xlarge	4	192	96	768
g6e48.xlarge	8	384	192	1536
gr6.8xlarge	1	24	32	256

É possível recuperar o ID da imagem de máquina da Amazon (AMI) para AMIs otimizadas para o Amazon ECS consultando a API Parameter Store do AWS Systems Manager. Ao usar esse parâmetro, não será necessário pesquisar manualmente IDs de AMIs otimizadas para o Amazon ECS. Para obter mais informações sobre a API Systems Manager Parameter Store, consulte GetParameter. O usuário que você usou deve ter a permissão `ssm:GetParameter` do IAM para recuperar os metadados da AMI otimizada para o Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended --region us-east-1
```

Considerações

> **Note**
>
> O suporte para o tipo de família de instâncias g2 foi descontinuado.
> Há suporte para o tipo da família de instâncias p2 somente em versões anteriores à `20230912` da AMI otimizada para GPU do Amazon ECS. Se você precisar continuar usando instâncias p2, consulte O que fazer se você precisar de uma instância P2.
> As atualizações locais dos drivers NVIDIA/CUDA nesses dois tipos de família de instâncias causarão possíveis falhas na workload da GPU.

Convém considerar as observações a seguir antes de iniciar o trabalho com GPUs no Amazon ECS.

- Seus clusters podem conter uma combinação de instâncias de contêiner de GPU e não GPU.
- É possível executar workloads de GPU em instâncias externas. Ao registrar uma instância externa no cluster, certifique-se de que o sinalizador `--enable-gpu` esteja incluído no script de instalação. Para obter mais informações, consulte Registro de uma instância externa para um cluster do Amazon ECS.
- É necessário definir `ECS_ENABLE_GPU_SUPPORT` como `true` no seu arquivo de configuração do agente. Para obter mais informações, consulte the section called “Configuração do agente de contêiner”.
- Ao executar uma tarefa ou criar um serviço, você pode usar atributos de tipo de instância ao configurar restrições de posicionamento de tarefas para determinar em quais instâncias de contêiner a tarefa deve ser inicada. Ao fazer isso, você pode usar seus recursos com mais eficiência. Para obter mais informações, consulte Como o Amazon ECS posiciona tarefas em instâncias de contêineres.

O exemplo a seguir executa uma tarefa em uma instância de contêiner `g4dn.xlarge` em seu cluster padrão.

```
aws ecs run-task --cluster default --task-definition ecs-gpu-task-def \
     --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
  g4dn.xlarge" --region us-east-2
```

- Para cada contêiner que tenha um requisito de recurso de GPU especificado na definição do contêiner, o Amazon ECS define o runtime do contêiner como o runtime do contêiner NVIDIA.
- O runtime do contêiner NVIDIA requer que algumas variáveis de ambiente sejam definidas no contêiner para funcionar corretamente. Para obter uma lista dessas variáveis de ambiente, consulte Specialized Configurations with Docker. O Amazon ECS define o valor da variável de ambiente `NVIDIA_VISIBLE_DEVICES` como uma lista dos IDs de dispositivo de GPU que o Amazon ECS atribui ao contêiner. O Amazon ECS não define as outras variáveis de ambiente necessárias. Por isso, certifique-se de que a imagem de contêiner as defina ou que elas sejam especificadas na definição de contêiner.
- A família de tipos de instância p5 tem suporte na versão `20230929` e posterior da AMI otimizada para GPU do Amazon ECS.
- A família de tipos de instância g4 tem suporte na versão `20230913` e posterior da AMI otimizada para GPU do Amazon ECS. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS. Ela é compatível com o fluxo de trabalho Create Cluster (Criar cluster) no console do Amazon ECS. Para usar esses tipos de instância, é necessário usar o console do Amazon EC2, a AWS CLI ou a API e registrar manualmente as instâncias no cluster.
- O tipo de instância p4d.24xlarge só funciona com CUDA 11 ou posterior.
- A AMI otimizada para GPU do Amazon ECS tem IPv6 habilitado, o que causa problemas ao usar `yum`. Isso pode ser resolvido configurando `yum` para usar o IPv4 com o seguinte comando.

```
echo "ip_resolve=4" >> /etc/yum.conf
```

- Quando você criar uma imagem de contêiner que não usa as imagens de base NVIDIA/CUDA, deverá definir a variável do runtime do contêiner `NVIDIA_DRIVER_CAPABILITIES` em um dos seguintes valores:
  - `utility,compute`
  - `all`

Para obter informações sobre como definir a variável, consulte Controlar o runtime do contêiner NVIDIA no site da NVIDIA.

- Não há suporte para GPUs em contêineres do Windows.

Iniciar uma instância de contêiner de GPU para o Amazon ECS

Para usar uma instância de GPU no Amazon ECS, você precisa criar um modelo de execução, um arquivo de dados do usuário e executar a instância.

Em seguida, você pode executar uma tarefa que usa uma definição de tarefa configurada para GPU.

Usar um modelo de execução

Você pode criar um modelo de execução.

- Crie um modelo de execução que use o ID da AMI de GPU otimizada para o Amazon ECS para a AMI. Para obter informações sobre como criar um modelo de inicialização, consulte Criar um novo modelo de inicialização usando parâmetros definidos por você no Manual do usuário do Amazon EC2.

  Use o ID da AMI da etapa anterior na Imagem da máquina da Amazon. Para obter informações sobre como especificar o ID da AMI com o parâmetro do Systems Manager, consulte Especificar um parâmetro do Systems Manager em um modelo de inicialização no Manual do usuário do Amazon EC2.

  Adicione os itens a seguir aos Dados do usuário no modelo de execução. Substitua *cluster-name* pelo nome do seu cluster.

  ```
  #!/bin/bash
  echo ECS_CLUSTER=cluster-name >> /etc/ecs/ecs.config;
  echo ECS_ENABLE_GPU_SUPPORT=true >> /etc/ecs/ecs.config
  ```

Usar a AWS CLI

É possível executar uma instância de contêiner usando a AWS CLI.

1. Crie um arquivo chamado `userdata.toml`. Esse arquivo será usado para dados do usuário da instância. Substitua *cluster-name* pelo nome do seu cluster.

```
#!/bin/bash
echo ECS_CLUSTER=cluster-name >> /etc/ecs/ecs.config;
echo ECS_ENABLE_GPU_SUPPORT=true >> /etc/ecs/ecs.config
```

2. Execute o comando a seguir para obter o ID da AMI de GPU. Você usará isso na etapa a seguir.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended --region us-east-1
```

3. Execute o comando a seguir para executar a instância de GPU. Lembre-se de substituir os parâmetros a seguir:

   - Substitua *sub-rede* pelo ID da sub-rede pública ou privada na qual sua instância será iniciada.
   - Substitua *gpu_ami* pelo ID da AMI da etapa anterior.
   - Substitua *t3.large* pelo tipo de instância que você deseja usar.
   - Substitua *região* pelo código da região.

```
aws ec2 run-instances --key-name ecs-gpu-example \
   --subnet-id subnet \
   --image-id gpu_ami \
   --instance-type t3.large \
   --region region \
   --tag-specifications 'ResourceType=instance,Tags=[{Key=GPU,Value=example}]' \
   --user-data file://userdata.toml \
   --iam-instance-profile Name=ecsInstanceRole
```

4. Execute o comando a seguir para verificar se a instância de contêiner está registrada no cluster. Ao executar esse comando, lembre-se de substituir os parâmetros a seguir:

   - Substitua *cluster* pelo nome do seu cluster.
   - Substitua *região* pelo código da sua região.

```
aws ecs list-container-instances --cluster cluster-name --region region
```

Especificar GPUs em uma definição de tarefa do Amazon ECS

Para usar as GPUs em uma instância de contêiner e o runtime da GPU do Docker, certifique-se de designar, na definição de tarefa, o número de GPUs que o seu contêiner requer. O agente de contêiner do Amazon ECS atribui o número desejado de GPUs físicas ao contêiner adequado conforme os contêineres compatíveis com GPU são posicionados. O número de GPUs reservadas para todos os contêineres em uma tarefa não deve exceder o número de GPUs disponíveis na instância de contêiner na qual a tarefa é executada. Para obter mais informações, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

> **Important**
>
> Se seus requisitos de GPU não forem especificados na definição de tarefa, a tarefa usará o runtime padrão do Docker.

Veja a seguir o formato JSON para os requisitos de GPU em uma definição de tarefa.

```
{
  "containerDefinitions": [
     {
        ...
        "resourceRequirements" : [
            {
               "type" : "GPU",
               "value" : "2"
            }
        ],
     },
...
}
```

O exemplo a seguir demonstra a sintaxe para um contêiner do Docker que especifica um requisito de GPU. Esse contêiner usa duas GPUs, executa o utilitário `nvidia-smi` e, em seguida, é encerrado.

```
{
  "containerDefinitions": [
    {
      "memory": 80,
      "essential": true,
```

```
      "name": "gpu",
      "image": "nvidia/cuda:11.0.3-base",
      "resourceRequirements": [
         {
           "type":"GPU",
           "value": "2"
         }
      ],
      "command": [
        "sh",
        "-c",
        "nvidia-smi"
      ],
      "cpu": 100
    }
  ],
  "family": "example-ecs-gpu"
}
```

Compartilhar GPUs

Para compartilhar GPUs, você precisará configurar as opções a seguir.

1. Remova os requisitos de recursos de GPU das suas definições de tarefas para que o Amazon ECS não reserve nenhuma GPU que deveria ser compartilhada.
2. Adicione os dados de usuário a seguir às suas instâncias quando desejar compartilhar GPUs. Isso tornará nvidia o runtime padrão do contêiner do Docker na instância do contêiner para que todos os contêineres do Amazon ECS possam usar as GPUs. Para obter mais informações, consulte Executar comandos ao executar uma instância do EC2 com entrada de dados do usuário no Guia do usuário do Amazon EC2.

   ```
   const userData = ec2.UserData.forLinux();
    userData.addCommands(
    'sudo rm /etc/sysconfig/docker',
    'echo DAEMON_MAXFILES=1048576 | sudo tee -a /etc/sysconfig/docker',
    'echo OPTIONS="--default-ulimit nofile=32768:65536 --default-runtime nvidia" |
    sudo tee -a /etc/sysconfig/docker',
    'echo DAEMON_PIDFILE_TIMEOUT=10 | sudo tee -a /etc/sysconfig/docker',
    'sudo systemctl restart docker',
   );
   ```

3. Defina a variável de ambiente `NVIDIA_VISIBLE_DEVICES` em seu contêiner. Você pode fazer isso especificando a variável de ambiente na definição da sua tarefa. Para obter informações sobre os valores válidos, consulte Enumeração de GPUs no site de documentação da NVIDIA.

O que fazer se você precisar de uma instância P2

Se você precisar usar a instância P2, poderá usar uma das opções a seguir para continuar usando as instâncias.

Será preciso modificar os dados do usuário da instância para ambas as opções. Para obter mais informações, consulte Executar comandos ao executar uma instância do EC2 com entrada de dados do usuário no Guia do usuário do Amazon EC2.

Use a última AMI otimizada para GPU com suporte

É possível usar a versão `20230906` da AMI otimizada para GPU e adicionar o seguinte aos dados do usuário da instância.

Substitua cluster-name pelo nome do seu cluster.

```
#!/bin/bash
echo "exclude=*nvidia* *cuda*" >> /etc/yum.conf
echo "ECS_CLUSTER=cluster-name" >> /etc/ecs/ecs.config
```

Use a AMI otimizada para GPU mais recente e atualize os dados do usuário

É possível adicionar o seguinte aos dados do usuário da instância. Isso desinstala os drivers Nvidia 535/Cuda12.2 e, em seguida, instala os drivers Nvidia 470/Cuda11.4 e corrige a versão.

```
#!/bin/bash
yum remove -y cuda-toolkit* nvidia-driver-latest-dkms*
tmpfile=$(mktemp)
cat >$tmpfile <<EOF
[amzn2-nvidia]
name=Amazon Linux 2 Nvidia repository
mirrorlist=\$awsproto://\$amazonlinux.\$awsregion.\$awsdomain/\$releasever/amzn2-
nvidia/latest/\$basearch/mirror.list
priority=20
gpgcheck=1
gpgkey=https://developer.download.nvidia.com/compute/cuda/repos/rhel7/
x86_64/7fa2af80.pub
```

```
enabled=1
exclude=libglvnd-*
EOF

mv $tmpfile /etc/yum.repos.d/amzn2-nvidia-tmp.repo
yum install -y system-release-nvidia cuda-toolkit-11-4 nvidia-driver-latest-
dkms-470.182.03
yum install -y libnvidia-container-1.4.0 libnvidia-container-tools-1.4.0 nvidia-
container-runtime-hook-1.4.0 docker-runtime-nvidia-1

echo "exclude=*nvidia* *cuda*" >> /etc/yum.conf
nvidia-smi
```

Criar sua própria AMI otimizada para GPU compatível com P2

É possível criar sua própria AMI personalizada otimizada para GPU do Amazon ECS que seja compatível com instâncias P2 e, em seguida, executar instâncias P2 usando a AMI.

1. Execute o comando a seguir para clonar a `amazon-ecs-ami repo`.

   ```
   git clone https://github.com/aws/amazon-ecs-ami
   ```

2. Defina o agente do Amazon ECS necessário e as versões de origem do Amazon Linux AMI em `release.auto.pkrvars.hcl` ou `overrides.auto.pkrvars.hcl`.
3. Execute o comando a seguir para criar uma AMI do EC2 privada compatível com P2.

   Substitua região pela Região com a instância Região.

   ```
   REGION=region make al2keplergpu
   ```

4. Use a AMI com os dados de usuário da instância a seguir para se conectar ao cluster do Amazon ECS.

   Substitua cluster-name pelo nome do seu cluster.

   ```
   #!/bin/bash
   echo "ECS_CLUSTER=cluster-name" >> /etc/ecs/ecs.config
   ```

Definições de tarefas do Amazon ECS para workloads de transcodificação de vídeo

Para usar workloads de transcodificação de vídeo no Amazon ECS, registre instâncias VT1 do Amazon EC2. Após registrar essas instâncias, será possível executar workloads de transcodificação de vídeo em tempo real e pré-renderizadas como tarefas no Amazon ECS. Instâncias VT1 do Amazon EC2 utilizam placas de transcodificação de mídia Xilinx U30 para acelerar workloads de transcodificação de vídeo em tempo real e pré-renderizadas.

> **Note**
>
> Para obter instruções sobre como executar workloads de transcodificação de vídeo em outros contêineres que não os do Amazon ECS, consulte a documentação da Xilinx.

Considerações

Antes de começar a implantar a VT1 no Amazon ECS, considere o seguinte:

- Seus clusters podem conter uma combinação de instâncias VT1 e não VT1.
- Você precisa de uma aplicação Linux que utilize placas de transcodificação de mídia Xilinx U30 com codecs AVC (H.264) e HEVC (H.265) acelerados.

  > **Important**
  >
  > Aplicações que utilizam outros codecs podem não ter performance aprimorada em instâncias VT1.

- Apenas uma tarefa de transcodificação pode ser executada em uma placa U30. Cada placa tem dois dispositivos associados. É possível executar tarefas de transcodificação na medida em que houver placas para cada instância VT1.
- Ao criar um serviço ou executar uma tarefa autônoma, você pode usar atributos de tipo de instância ao configurar as restrições de posicionamento de tarefas. Assim, você se certifica de que a tarefa será iniciada na instância de contêiner que você especificar. Isso ajuda a garantir que você use seus recursos de maneira eficaz e que suas tarefas para workloads de transcodificação de vídeo estejam em suas instâncias VT1. Para obter mais informações, consulte Como o Amazon ECS posiciona tarefas em instâncias de contêineres.

No exemplo a seguir, uma tarefa é executada em uma instância `vt1.3xlarge` do cluster `default`.

```
aws ecs run-task \
     --cluster default \
     --task-definition vt1-3xlarge-xffmpeg-processor \
     --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
 vt1.3xlarge"
```

- Você configura um contêiner para usar a placa U30 específica disponível na instância de contêiner host. É possível fazer isso utilizando o parâmetro `linuxParameters` e especificando os detalhes do dispositivo. Para obter mais informações, consulte Requisitos de definição de tarefa.

Usar uma AMI VT1

Existem duas opções para executar uma AMI no Amazon EC2 para instâncias de contêiner do Amazon ECS. A primeira é usar a AMI oficial da Xilinx no AWS Marketplace. A segunda é criar a sua própria AMI do repositório de amostra.

- A Xilinx oferece AMIs no AWS Marketplace.
- O Amazon ECS fornece um repositório de amostra que pode ser usado para criar uma AMI para workloads de transcodificação de vídeo. Essa AMI acompanha drivers Xilinx U30. O repositório que contém os scripts do Packer pode ser encontrado no GitHub. Para obter mais informações sobre o Packer, consulte a documentação do Packer.

Requisitos de definição de tarefa

Para executar contêiners de transcodificação de vídeo no Amazon ECS, sua definição de tarefa precisa conter uma aplicação de transcodificação de vídeo que utilize codecs H.264/AVC e H.265/HEVC acelerados. É possível criar uma imagem de contêiner seguindo as etapas no repositório da Xilinx no GitHub.

A definição de tarefa deve ser específica do tipo de instância. Os tipos de instância são: 3xlarge, 6xlarge e 24xlarge. É necessário configurar um contêiner para usar dispositivos Xilinx U30 específicos disponíveis na instância de contêiner host. Para isso, use o parâmetro `linuxParameters`. A tabela a seguir fornece detalhes sobre as placas e SoCs de dispositivo específicos de cada tipo de instância.

Tipo de instância	vCPUs	RAM (GiB)	Placas aceleradoras U30	Dispositivos SoC XCU30 endereçáveis	Caminhos de dispositivos
vt1.3xlarge	12	24	1	2	`/dev/dri/renderD128 ,/dev/dri/renderD129`
vt1.6xlarge	24	48	2	4	`/dev/dri/renderD128 ,/dev/dri/renderD129 ,/dev/dri/renderD130 ,/dev/dri/renderD131`
vt1.24xlarge	96	182	8	16	`/dev/dri/renderD128 ,/dev/dri/renderD129 ,/dev/dri/renderD130 ,/dev/dri/renderD131 ,/dev/dri/`

Tipo de instância	vCPUs	RAM (GiB)	Placas aceleradoras U30	Dispositivos SoC XCU30 endereçáveis	Caminhos de dispositivos
					renderD132 ,/dev/dri/renderD133 ,/dev/dri/renderD134 ,/dev/dri/renderD135 ,/dev/dri/renderD136 ,/dev/dri/renderD137 ,/dev/dri/renderD138 ,/dev/dri/renderD139 ,/dev/dri/renderD140 ,/dev/dri/renderD141 ,/dev/dri/renderD142 ,/dev/

Tipo de instância	vCPUs	RAM (GiB)	Placas aceleradoras U30	Dispositivos SoC XCU30 endereçáveis	Caminhos de dispositivos
					`dri/renderD143`

> **Important**
>
> Se a definição de tarefa indicar dispositivos ausentes na instância do EC2, a tarefa não será executada. Quando a tarefa falhar, a mensagem de erro a seguir aparecerá no `stoppedReason`: `CannotStartContainerError: Error response from daemon: error gathering device information while adding custom device "/dev/dri/renderD130": no such file or directory`.

Especificar transcodificação de vídeo em uma definição de tarefa do Amazon ECS

No exemplo abaixo, é fornecida a sintaxe utilizada para uma definição de tarefa de um contêiner Linux no Amazon EC2. Essa definição de tarefa refere-se a imagens de contêiner criadas de acordo com o procedimento fornecido na documentação da Xilinx. Se você utilizar esse exemplo, substitua `image` por sua própria imagem e copie seus arquivos de vídeo para a instância no diretório `/home/ec2-user`.

vt1.3xlarge

1. Crie um arquivo de texto denominado `vt1-3xlarge-ffmpeg-linux.json`, com o seguinte conteúdo.

```
{
    "family": "vt1-3xlarge-xffmpeg-processor",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
```

```
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == vt1.3xlarge"
        }
    ],
    "containerDefinitions": [
        {
            "entryPoint": [
                "/bin/bash",
                "-c"
            ],
            "command": ["/video/ecs_ffmpeg_wrapper.sh"],
            "linuxParameters": {
                "devices": [
                    {
                        "containerPath": "/dev/dri/renderD128",
                        "hostPath": "/dev/dri/renderD128",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD129",
                        "hostPath": "/dev/dri/renderD129",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ]
            },
            "mountPoints": [
                {
                    "containerPath": "/video",
                    "sourceVolume": "video_file"
                }
            ],
            "cpu": 0,
            "memory": 12000,
            "image": "0123456789012.dkr.ecr.us-west-2.amazonaws.com/aws/xilinx-
xffmpeg",
            "essential": true,
            "name": "xilinix-xffmpeg"
        }
```

```
    ],
    "volumes": [
        {
            "name": "video_file",
            "host": {"sourcePath": "/home/ec2-user"}
        }
    ]
}
```

2. Registre a definição de tarefa.

```
aws ecs register-task-definition --family vt1-3xlarge-xffmpeg-processor --cli-
input-json file://vt1-3xlarge-xffmpeg-linux.json --region us-east-1
```

vt1.6xlarge

1. Crie um arquivo de texto denominado `vt1-6xlarge-ffmpeg-linux.json`, com o seguinte conteúdo.

```
{
    "family": "vt1-6xlarge-xffmpeg-processor",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == vt1.6xlarge"
        }
    ],
    "containerDefinitions": [
        {
            "entryPoint": [
                "/bin/bash",
                "-c"
            ],
            "command": ["/video/ecs_ffmpeg_wrapper.sh"],
            "linuxParameters": {
                "devices": [
                    {
```

```
                        "containerPath": "/dev/dri/renderD128",
                        "hostPath": "/dev/dri/renderD128",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD129",
                        "hostPath": "/dev/dri/renderD129",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD130",
                        "hostPath": "/dev/dri/renderD130",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD131",
                        "hostPath": "/dev/dri/renderD131",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ]
            },
            "mountPoints": [
                {
                    "containerPath": "/video",
                    "sourceVolume": "video_file"
                }
            ],
            "cpu": 0,
            "memory": 12000,
            "image": "0123456789012.dkr.ecr.us-west-2.amazonaws.com/aws/xilinx-
xffmpeg",
            "essential": true,
```

```
            "name": "xilinix-xffmpeg"
        }
    ],
    "volumes": [
        {
            "name": "video_file",
            "host": {"sourcePath": "/home/ec2-user"}
        }
    ]
}
```

2. Registre a definição de tarefa.

```
aws ecs register-task-definition --family vt1-6xlarge-xffmpeg-processor --cli-
input-json file://vt1-6xlarge-xffmpeg-linux.json --region us-east-1
```

vt1.24xlarge

1. Crie um arquivo de texto denominado `vt1-24xlarge-ffmpeg-linux.json`, com o seguinte conteúdo.

```
{
    "family": "vt1-24xlarge-xffmpeg-processor",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == vt1.24xlarge"
        }
    ],
    "containerDefinitions": [
        {
            "entryPoint": [
                "/bin/bash",
                "-c"
            ],
            "command": ["/video/ecs_ffmpeg_wrapper.sh"],
            "linuxParameters": {
```

```
                "devices": [
                    {
                        "containerPath": "/dev/dri/renderD128",
                        "hostPath": "/dev/dri/renderD128",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD129",
                        "hostPath": "/dev/dri/renderD129",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD130",
                        "hostPath": "/dev/dri/renderD130",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD131",
                        "hostPath": "/dev/dri/renderD131",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD132",
                        "hostPath": "/dev/dri/renderD132",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD133",
                        "hostPath": "/dev/dri/renderD133",
```

```
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD134",
                        "hostPath": "/dev/dri/renderD134",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD135",
                        "hostPath": "/dev/dri/renderD135",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD136",
                        "hostPath": "/dev/dri/renderD136",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD137",
                        "hostPath": "/dev/dri/renderD137",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD138",
                        "hostPath": "/dev/dri/renderD138",
                        "permissions": [
                            "read",
                            "write"
                        ]
```

```
                    },
                    {
                        "containerPath": "/dev/dri/renderD139",
                        "hostPath": "/dev/dri/renderD139",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD140",
                        "hostPath": "/dev/dri/renderD140",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD141",
                        "hostPath": "/dev/dri/renderD141",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD142",
                        "hostPath": "/dev/dri/renderD142",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD143",
                        "hostPath": "/dev/dri/renderD143",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ]
            },
            "mountPoints": [
```

```
                {
                    "containerPath": "/video",
                    "sourceVolume": "video_file"
                }
            ],
            "cpu": 0,
            "memory": 12000,
            "image": "0123456789012.dkr.ecr.us-west-2.amazonaws.com/aws/xilinx-
xffmpeg",
            "essential": true,
            "name": "xilinix-xffmpeg"
        }
    ],
    "volumes": [
        {
            "name": "video_file",
            "host": {"sourcePath": "/home/ec2-user"}
        }
    ]
}
```

2. Registre a definição de tarefa.

```
aws ecs register-task-definition --family vt1-24xlarge-xffmpeg-processor --cli-
input-json file://vt1-24xlarge-xffmpeg-linux.json --region us-east-1
```

Definições de tarefa do Amazon ECS para workloads de machine learning do AWS Neuron

É possível registar as instâncias Amazon EC2 Trn1, Amazon EC2 Inf1 e Amazon EC2 Inf2 nos seus clusters para workloads de machine learning.

As instâncias Trn1 do Amazon EC2 são executadas em chips AWS Trainium. Essas instâncias oferecem treinamento de alta performance e baixo custo para machine learning na nuvem. É possível treinar um modelo de inferência de machine learning usando um framework de machine learning com o AWS Neuron em uma instância Trn1. Em seguida, é possível executar o modelo em uma instância Inf1 ou uma instância Inf2 para usar a aceleração dos chips AWS Inferentia.

As instâncias Inf1 e as instâncias Inf2 do Amazon EC2 são executadas em chips AWS Inferentia. Eles fornecem alta performance e inferência de menor custo na nuvem.

Modelos de machine learning são implantados em contêineres com o uso do AWS Neuron, um kit de desenvolvimento de software (SDK)especializado. O SDK consiste em um compilador, runtime e ferramentas de criação de perfil que otimizam a performance de machine learning dos chips de machine learning da AWS. AWS O Neuron é compatível com frameworks de machine learning conhecidos, como TensorFlow, PyTorch e Apache MXNet.

Considerações

Antes de começar a implantar o Neuron no Amazon ECS, considere o seguinte:

- Seus clusters podem conter uma combinação de instâncias Trn1, Inf1, Inf2 e outras.
- Você precisa de uma aplicação do Linux em um contêiner que use um framework de machine learning compatível com o AWS Neuron.

> **Important**
>
> Aplicações que usem outros frameworks podem não apresentar uma melhoria de performance nas instâncias Trn1, Inf1 e Inf2.

- Apenas uma tarefa de inferência ou de treinamento de inferência pode ser executada em cada chip AWS Trainium ou AWS Inferentia. Para a Inf1, cada chip possui 4 NeuronCores. Para a Trn1 e a Inf2, cada chip possui 2 NeuronCores. É possível executar tantas tarefas quantos forem os chips para cada uma de suas instâncias Trn1, Inf1 e Inf2.
- Ao criar um serviço ou executar uma tarefa autônoma, você pode usar atributos de tipo de instância ao configurar as restrições de posicionamento de tarefas. Assim, você se certifica de que a tarefa será iniciada na instância de contêiner que você especificar. Isso pode ajudar você a otimizar o uso dos recursos em geral e garantir que as tarefas para workloads de inferência ocorram nas instâncias Trn, Inf1 e Inf2. Para obter mais informações, consulte Como o Amazon ECS posiciona tarefas em instâncias de contêineres.

  No exemplo a seguir, uma tarefa é executada em uma instância `Inf1.xlarge` do cluster `default`.

```
aws ecs run-task \
     --cluster default \
     --task-definition ecs-inference-task-def \
     --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
 Inf1.xlarge"
```

- Os requisitos de recursos do Neuron não podem ser definidos em uma definição de tarefa. Em vez disso, você configura um contêiner para usar os chips AWS Trainium ou AWS Inferentia específicos, disponíveis na instância de contêiner host. É possível fazer isso usando o parâmetro `linuxParameters` e especificando os detalhes do dispositivo. Para obter mais informações, consulte Requisitos de definição de tarefa.

Usar a AMI do Amazon Linux 2023 (Neuron) otimizada para o Amazon ECS

O Amazon ECS fornece uma AMI otimizada para o Amazon ECS que se baseia no Amazon Linux 2023 para workloads do AWS Trainium e do AWS Inferentia. Ela vem com drivers do AWS Neuron e runtime para Docker. Essa AMI facilita a execução de workloads de inferência de machine learning no Amazon ECS.

Recomendamos usar a AMI do Amazon Linux 2023 (Neuron) otimizada para o Amazon ECS ao iniciar as instâncias Trn1, Inf1 e Inf2 do Amazon EC2.

É possível recuperar a AMI atual do Amazon Linux 2023 (Neuron) otimizada para o Amazon ECS ao usar a AWS CLI com o comando a seguir.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/neuron/recommended
```

Há suporte para a AMI do Amazon Linux 2023 (Neuron) otimizada para o Amazon ECS nas seguintes regiões:

- Leste dos EUA (Norte da Virgínia)
- Leste dos EUA (Ohio)
- Oeste dos EUA (Norte da Califórnia)
- Oeste dos EUA (Oregon)
- Ásia-Pacífico (Mumbai)
- Ásia-Pacífico (Osaka)
- Ásia-Pacífico (Seul)
- Ásia-Pacífico (Tóquio)
- Ásia-Pacífico (Singapura)
- Ásia-Pacífico (Sydney)
- Canadá (Central)

- Europa (Frankfurt)
- Europa (Irlanda)
- Europa (Londres)
- Europa (Paris)
- Europa (Estocolmo)
- América do Sul (São Paulo)

Usar a AMI do Amazon Linux 2 (Neuron) otimizada para o Amazon ECS

O Amazon ECS oferece uma AMI otimizada para o Amazon ECS que é baseada no Amazon Linux 2 para workloads do AWS Trainium e do AWS Inferentia. Ela vem com drivers do AWS Neuron e runtime para Docker. Essa AMI facilita a execução de workloads de inferência de machine learning no Amazon ECS.

Recomendamos o uso da AMI do Amazon Linux 2 (Neuron) otimizada para o Amazon ECS ao iniciar as instâncias Trn, Inf1 e Inf2 do Amazon EC2.

É possível recuperar a AMI atual do Amazon Linux 2 (Neuron) otimizada para o Amazon ECS usando a AWS CLI com o comando a seguir.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/inf/
recommended
```

A AMI do Amazon Linux 2 (Neuron) otimizada para o Amazon ECS é compatível com as seguintes regiões:

- Leste dos EUA (Norte da Virgínia)
- Leste dos EUA (Ohio)
- Oeste dos EUA (Norte da Califórnia)
- Oeste dos EUA (Oregon)
- Ásia-Pacífico (Mumbai)
- Ásia-Pacífico (Osaka)
- Ásia-Pacífico (Seul)
- Ásia-Pacífico (Tóquio)
- Ásia-Pacífico (Singapura)

- Ásia-Pacífico (Sydney)
- Canadá (Central)
- Europa (Frankfurt)
- Europa (Irlanda)
- Europa (Londres)
- Europa (Paris)
- Europa (Estocolmo)
- América do Sul (São Paulo)

Requisitos de definição de tarefa

Para implantar o Neuron no Amazon ECS, sua definição de tarefa deve conter a definição do contêiner para um contêiner pré-criado servindo o modelo de inferência do TensorFlow. Além disso, é fornecido por contêiners do AWS Deep Learning. O contêiner possui o runtime do AWS Neuron e a aplicação TensorFlow Serving. No startup, esse container busca o modelo no Amazon S3, inicia o serviço do Neuron TensorFlow com o modelo salvo e aguarda as solicitações de previsão. No exemplo a seguir, a imagem de contêiner tem o TensorFlow 1.15 e o Ubuntu 18.04. Uma lista completa de Deep Learning Containers pré-criados otimizados para Neuron é mantida no GitHub. Para obter mais informações, consulte Uso do AWS Neuron TensorFlow Serving.

```
763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-inference-neuron:1.15.4-neuron-py37-ubuntu18.04
```

Como alternativa, é possível criar sua própria imagem de contêiner de arquivo associado do Neuron. Para obter mais informações, consulte Tutorial: Neuron TensorFlow Serving no Guia do desenvolvedor do AMIs de deep learning da AWS.

A definição de tarefa deve ser específica de um único tipo de instância. É necessário configurar um contêiner para usar os dispositivos AWS Trainium ou AWS Inferentia específicos, disponíveis na instância de contêiner host. Para isso, use o parâmetro `linuxParameters`. A tabela a seguir detalha os chips específicos de cada tipo de instância.

Tipo de instância	vCPUs	RAM (GiB)	Chips aceleradores de ML AWS	Caminhos de dispositivos
trn1.2xlarge	8	32	1	`/dev/neuron0`
trn1.32xlarge	128	512	16	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5 ,/dev/neuron6 ,/dev/neuron7 ,/dev/neuron8 ,/dev/neuron9 ,/dev/neuron10 ,/dev/neuron11 ,/dev/neuron12 ,/dev/neuron13 ,/dev/neuron14 ,/dev/neuron15`
inf1.xlarge	4	8	1	`/dev/neuron0`
inf1.2xlarge	8	16	1	`/dev/neuron0`

Tipo de instância	vCPUs	RAM (GiB)	Chips aceleradores de ML AWS	Caminhos de dispositivos
inf1.6xlarge	24	48	4	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3`
inf1.24xlarge	96	192	16	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5 ,/dev/neuron6 ,/dev/neuron7 ,/dev/neuron8 ,/dev/neuron9 ,/dev/neuron10 ,/dev/neuron11 ,/dev/neuron12 ,/dev/neuron13 ,/dev/neuron14 ,/dev/neuron15`
inf2.xlarge	8	16	1	`/dev/neuron0`

Tipo de instância	vCPUs	RAM (GiB)	Chips aceleradores de ML AWS	Caminhos de dispositivos
inf2.8xlarge	32	64	1	`/dev/neuron0`
inf2.24xlarge	96	384	6	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5 ,`
inf2.48xlarge	192	768	12	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5 ,/dev/neuron6 ,/dev/neuron7 ,/dev/neuron8 ,/dev/neuron9 ,/dev/neuron10 ,/dev/neuron11`

Especificar machine learning do AWS Neuron em uma definição de tarefa do Amazon ECS

Veja a seguir um exemplo de definição de tarefa do Linux para `inf1.xlarge`, mostrando a sintaxe a ser usada.

```
{
    "family": "ecs-neuron",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == inf1.xlarge"
        }
    ],
    "executionRoleArn": "${YOUR_EXECUTION_ROLE}",
    "containerDefinitions": [
        {
            "entryPoint": [
                "/usr/local/bin/entrypoint.sh",
                "--port=8500",
                "--rest_api_port=9000",
                "--model_name=resnet50_neuron",
                "--model_base_path=s3://amzn-s3-demo-bucket/resnet50_neuron/"
            ],
            "portMappings": [
                {
                    "hostPort": 8500,
                    "protocol": "tcp",
                    "containerPort": 8500
                },
                {
                    "hostPort": 8501,
                    "protocol": "tcp",
                    "containerPort": 8501
                },
                {
                    "hostPort": 0,
                    "protocol": "tcp",
```

```
                    "containerPort": 80
                }
            ],
            "linuxParameters": {
                "devices": [
                    {
                        "containerPath": "/dev/neuron0",
                        "hostPath": "/dev/neuron0",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ],
                "capabilities": {
                    "add": [
                        "IPC_LOCK"
                    ]
                }
            },
            "cpu": 0,
            "memoryReservation": 1000,
            "image": "763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-
inference-neuron:1.15.4-neuron-py37-ubuntu18.04",
            "essential": true,
            "name": "resnet50"
        }
    ]
}
```

Definições de tarefa do Amazon ECS para instâncias de aprendizado profundo

Para utilizar workloads de aprendizado profundo no Amazon ECS, registre instâncias DL1 do Amazon EC2 nos seus clusters. Instâncias DL1 do Amazon EC2 contam com a tecnologia dos aceleradores Gaudi da Habana Labs (uma empresa Intel). Utilize o Habana SynapseAI SDK para se conectar aos aceleradores Habana Gaudi. O SDK é compatível com frameworks de machine learning populares, como TensorFlow e PyTorch.

Considerações

Antes de começar a implantar a DL1 no Amazon ECS, considere o seguinte:

- Seus clusters podem conter uma combinação de instâncias DL1 e não DL1.
- Ao criar um serviço ou executar uma tarefa autônoma, você pode usar atributos de tipo de instância especificamente ao configurar restrições de posicionamento de tarefas para garantir que sua tarefa seja iniciada na instância de contêiner que você especificar. Isso garante que seus recursos sejam usados com eficiência e que suas tarefas para workloads de aprendizado profundo estejam em suas instâncias DL1. Para obter mais informações, consulte Como o Amazon ECS posiciona tarefas em instâncias de contêineres.

  O exemplo a seguir executa uma tarefa em uma instância `dl1.24xlarge` do cluster `default`.

```
aws ecs run-task \
     --cluster default \
     --task-definition ecs-dl1-task-def \
     --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
 dl1.24xlarge"
```

Utilizar uma AMI DL1

Existem três opções para executar uma AMI em instâncias DL1 do Amazon EC2 para o Amazon ECS:

- AMIs do AWS Marketplace que são fornecidas pela Habana aqui.
- AMIs do Habana Deep Learning que são fornecidas pela Amazon Web Services. Como isso não está incluído, é necessário instalar o agente de contêiner do Amazon ECS separadamente.
- Use o Packer para criar uma AMI personalizada que é fornecida pelo repositório do GitHub. Para obter mais informações, consulte a documentação do Packer.

Especificar aprendizado profundo em uma definição de tarefa do Amazon ECS

Para executar contêineres acelerados de aprendizado profundo do Habana Gaudi no Amazon ECS, sua definição de tarefa precisa conter a definição de um contêiner pré-criado que serve o modelo de aprendizado profundo para o TensorFlow ou PyTorch utilizando o Habana SynapseAI fornecido por contêiners do AWS Deep Learning.

A imagem de contêiner a seguir tem o TensorFlow 2.7.0 e o Ubuntu 20.04. Uma lista completa de contêineres de aprendizado profundo pré-construídos otimizados para os aceleradores Habana Gaudi é mantida no GitHub. Para saber mais, consulte Contêiners de treinamento da Habana.

```
763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-training-habana:2.7.0-hpu-py38-
synapseai1.2.0-ubuntu20.04
```

O exemplo a seguir é uma definição de tarefa para contêineres do Linux no Amazon EC2, exibindo a sintaxe a ser usada. Este exemplo utiliza uma imagem que contém a Habana Labs System Management Interface Tool (HL-SMI), disponível aqui: `vault.habana.ai/gaudi-docker/1.1.0/ubuntu20.04/habanalabs/tensorflow-installer-tf-cpu-2.6.0:1.1.0-614`

```
{
    "family": "dl-test",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == dl1.24xlarge"
        }
    ],
    "networkMode": "host",
    "cpu": "10240",
    "memory": "1024",
    "containerDefinitions": [
        {
            "entryPoint": [
                "sh",
                "-c"
            ],
            "command": ["hl-smi"],
            "cpu": 8192,
            "environment": [
                {
                    "name": "HABANA_VISIBLE_DEVICES",
                    "value": "all"
                }
            ],
            "image": "vault.habana.ai/gaudi-docker/1.1.0/ubuntu20.04/habanalabs/
tensorflow-installer-tf-cpu-2.6.0:1.1.0-614",
            "essential": true,
```

```
            "name": "tensorflow-installer-tf-hpu"
        }
    ]
}
```

Definições de tarefa do Amazon ECS para workloads do ARM de 64 bits

O Amazon ECS suporta o uso de aplicações ARM de 64 bits. É possível executar aplicações na plataforma baseada em processadores AWS Graviton. Ela é adequada para uma ampla variedade de workloads. Isso inclui workloads como servidores de aplicações, microsserviços, computação de alta performance, inferência de machine learning com base em CPU, codificação de vídeo, automação de design eletrônico, jogos, bancos de dados de código aberto e caches na memória.

Considerações

Antes de começar a implantar definições de tarefa que usem a arquitetura ARM de 64 bits, considere o seguinte:

- As aplicações podem usar os tipos de inicialização do Fargate ou do EC2.
- As aplicações só podem usar o sistema operacional Linux.
- Para o tipo Fargate, as aplicações devem usar a versão da plataforma Fargate `1.4.0` ou posterior.
- As aplicações podem usar Fluent Bit ou CloudWatch para monitoramento.
- Para o tipo de lançamento do Fargate, as seguintes Regiões da AWS não oferecem suporte a workloads ARM de 64 bits:
  - Leste dos EUA (Norte da Virgínia), a zona de disponibilidade `use1-az3`
- Para o tipo de inicialização do Amazon EC2, consulte o seguinte para verificar se a região em que você está oferece suporte ao tipo de instância desejado:
  - Instâncias M6g do Amazon EC2
  - Instâncias T4g do Amazon EC2
  - Instâncias C6g do Amazon EC2
  - Instâncias R6gd do Amazon EC2
  - Instâncias X2gd do Amazon EC2

  Também é possível usar o comando `describe-instance-type-offerings` do Amazon EC2 com um filtro para exibir a oferta de instâncias para sua região.

```
aws ec2 describe-instance-type-offerings --filters Name=instance-
type,Values=instance-type --region region
```

O exemplo a seguir verifica a disponibilidade do tipo de instância M6 na região Leste dos EUA (N. da Virgínia) (us-east-1).

```
aws ec2 describe-instance-type-offerings --filters "Name=instance-type,Values=m6*" --
region us-east-1
```

Para obter mais informações, consulte describe-instance-type-offerings na Referência de linha de comando do Amazon EC2.

Especificar a arquitetura ARM em uma definição de tarefa do Amazon ECS

Para usar a arquitetura ARM, especifique `ARM64` para o parâmetro de definição de tarefa `cpuArchitecture`.

No exemplo abaixo, a arquitetura do ARM é especificada em uma definição de tarefa. Ele é no formato JSON.

```
{
    "runtimePlatform": {
        "operatingSystemFamily": "LINUX",
        "cpuArchitecture": "ARM64"
    },
...
}
```

No exemplo a seguir, uma definição de tarefa para a arquitetura ARM exibe “hello world”.

```
{
 "family": "arm64-testapp",
 "networkMode": "awsvpc",
 "containerDefinitions": [
    {
        "name": "arm-container",
        "image": "public.ecr.aws/docker/library/busybox:latest",
        "cpu": 100,
        "memory": 100,
```

```
        "essential": true,
        "command": [ "echo hello world" ],
        "entryPoint": [ "sh", "-c" ]
    }
 ],
 "requiresCompatibilities": [ "EC2" ],
 "cpu": "256",
 "memory": "512",
 "runtimePlatform": {
        "operatingSystemFamily": "LINUX",
        "cpuArchitecture": "ARM64"
  },
 "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole"
}
```

Envio de logs do Amazon ECS para o CloudWatch

É possível configurar os contêineres das tarefas para enviar informações de log ao CloudWatch Logs. Se estiver usando o tipo de inicialização do Fargate para suas tarefas, será possível visualizar os logs a partir dos seus contêineres. Se você estiver usando o tipo de inicialização do EC2, poderá visualizar logs diferentes dos contêineres em um local conveniente e evitar que os logs de contêiner ocupem espaço em disco nas instâncias do seu contêiner.

> **Note**
>
> O tipo de informações registradas em log pelos contêineres em sua tarefa depende principalmente do comando `ENTRYPOINT`. Por padrão, os logs capturados mostram a saída do comando que você normalmente veria em um terminal interativo, se executasse o contêiner localmente, que são os fluxos de E/S `STDOUT` e `STDERR`. O driver de log `awslogs` simplesmente envia esses logs do Docker para o CloudWatch Logs. Para obter mais informações sobre como os logs do Docker são processados, incluindo maneiras alternativas de capturar fluxos ou dados de arquivos diferentes, consulte Visualizar logs de um contêiner ou serviço na documentação do Docker.

Para enviar logs do sistema de instâncias de contêiner do Amazon ECS para o CloudWatch Logs, consulte Monitoração de arquivos de log e Cotas de logs do CloudWatch no Guia do usuário do Amazon CloudWatch Logs.

Tipo de inicialização do Fargate

Se você estiver usando o tipo de inicialização do Fargate para suas tarefas, precisará adicionar os parâmetros `awslogs` necessários à sua definição de tarefa para ativar o driver de log `logConfiguration`. Para obter mais informações, consulte Exemplo de definição de tarefa do Amazon ECS: rotear logs para o CloudWatch.

No contêiner do Windows no Fargate, execute uma das seguintes opções quando qualquer um dos parâmetros de definição de tarefa tiver caracteres especiais, como `& \ < > ^ |`:

- Adicione um escape (\) com aspas duplas em torno de toda a string do parâmetro

  Exemplo

  ```
  "awslogs-multiline-pattern": "\"^[|DEBUG|INFO|WARNING|ERROR\"",
  ```

- Adicione um caractere de escape (^) em torno de cada caractere especial

  Exemplo

  ```
  "awslogs-multiline-pattern": "^^[^|DEBUG^|INFO^|WARNING^|ERROR",
  ```

Tipo de inicialização do EC2

Se você estiver usando o tipo de inicialização do EC2 para suas tarefas e quer ativar o driver de log `awslogs`, suas instâncias de contêiner do Amazon ECS precisarão pelo menos da versão 1.9.0 do agente de contêiner. Para obter informações sobre como verificar a versão do agente e atualizar para a versão mais recente, consulte Atualizar o agente de contêiner do Amazon ECS.

> **Note**
>
> Você deve usar uma AMI otimizada para o Amazon ECS ou uma AMI personalizada com pelo menos a versão `1.9.0-1` do pacote `ecs-init`. Ao usar uma AMI personalizada, você deve especificar que o driver de registro em log `awslogs` está disponível na instância do Amazon EC2 ao iniciar o agente, usando a variável de ambiente a seguir na instrução docker run ou arquivo de variável de ambiente.
>
> ```
> ECS_AVAILABLE_LOGGING_DRIVERS=["json-file","awslogs"]
> ```

As instâncias de contêiner do Amazon ECS também exigem permissão de `logs:CreateLogStream` e `logs:PutLogEvents` na função do IAM com a qual você inicia as instâncias de contêiner. Caso você tenha criado a função de instância de contêiner do Amazon ECS antes que o suporte ao driver de log `awslogs` tenha sido habilitado no Amazon ECS, talvez seja necessário adicionar essa permissão. A `ecsTaskExecutionRole` é usada quando é atribuída à tarefa e provavelmente contém as permissões corretas. Para obter informações sobre o perfil de execução de tarefas, consulte Função do IAM de execução de tarefas do Amazon ECS. Se as instâncias de contêiner usarem a política do IAM gerenciada para instâncias de contêiner, as instâncias de contêiner terão provavelmente as permissões corretas. Para obter informações sobre a política do IAM gerenciada para instâncias de contêiner, consulte Função do IAM de instância de contêiner do Amazon ECS.

Exemplo de definição de tarefa do Amazon ECS: rotear logs para o CloudWatch

Para os contêineres enviarem logs ao CloudWatch, você deve especificar o driver de log `awslogs` para contêineres na definição de tarefa. Para obter mais informações sobre os parâmetros de log, consulte Armazenamento e registro.

A definição de tarefa JSON a seguir tem um objeto `logConfiguration` especificado para cada contêiner. Um deles é para o contêiner do WordPress que envia logs a um grupo de logs denominado `awslogs-wordpress`. O outro é para um contêiner MySQL que envia logs a um grupo de logs denominado `awslogs-mysql`. Ambos os contêineres usam o prefixo de fluxo de log `awslogs-example`.

```
{
    "containerDefinitions": [
        {
            "name": "wordpress",
            "links": [
                "mysql"
            ],
            "image": "public.ecr.aws/docker/library/wordpress:latest",
            "essential": true,
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80
                }
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
```

```
                "options": {
                    "awslogs-create-group": "true",
                    "awslogs-group": "awslogs-wordpress",
                    "awslogs-region": "us-west-2",
                    "awslogs-stream-prefix": "awslogs-example"
                }
            },
            "memory": 500,
            "cpu": 10
        },
        {
            "environment": [
                {
                    "name": "MYSQL_ROOT_PASSWORD",
                    "value": "password"
                }
            ],
            "name": "mysql",
            "image": "public.ecr.aws/docker/library/mysql:latest",
            "cpu": 10,
            "memory": 500,
            "essential": true,
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-create-group": "true",
                    "awslogs-group": "awslogs-mysql",
                    "awslogs-region": "us-west-2",
                    "awslogs-stream-prefix": "awslogs-example",
                    "mode": "non-blocking",
                    "max-buffer-size": "25m"
                }
            }
        }
    ],
    "family": "awslogs-example"
}
```

Próximas etapas

- Você também pode definir uma política de retenção para o grupo de logs usando a AWS CLI ou a API do CloudWatch. Para obter mais informações, consulte put-retention-policy na Referência da AWS Command Line Interface.

- Depois que tiver registrado uma definição de tarefa com o driver de log `awslogs` em uma configuração de log de definição de contêiner, será possível executar uma tarefa ou criar um serviço com essa definição de tarefa para começar a enviar logs ao CloudWatch Logs. Para obter mais informações, consulte Execução de uma aplicação como uma tarefa do Amazon ECS e Criação de uma implantação de atualização contínua do Amazon ECS.

Envio de logs do Amazon ECS para um serviço da AWS ou para uma AWS Partner

É possível usar o FireLens para Amazon ECS para usar parâmetros de definição de tarefa para encaminhar logs para um serviço da AWS ou um destino da AWS Partner Network (APN) para o armazenamento e analytics de logs. O AWS Partner Network é uma comunidade global de parceiros que utiliza programas, experiência e recursos para criar, comercializar e vender ofertas aos clientes. Para obter mais informações, consulte AWS Partner. O FireLens funciona com o Fluentd e o Fluent Bit. Fornecemos a imagem da AWS for Fluent Bit, ou é possível usar sua própria imagem do Fluentd ou Fluent Bit.

Por padrão, o Amazon ECS configura a dependência do contêiner para que o contêiner do Firelens comece antes de qualquer contêiner que o utiliza. O contêiner do Firelens também é interrompido depois que todos os contêineres que o usam são interrompidos.

Considere o seguinte ao usar o FireLens para Amazon ECS:

- Recomendamos adicionar `my_service_` ao nome do contêiner de log para distinguir facilmente os nomes dos contêineres no console.
- Por padrão, o Amazon ECS adiciona uma dependência de ordem inicial de contêiner entre os contêineres da aplicação e o contêiner do FireLens. Quando você especifica uma ordem de contêiner entre os contêineres da aplicação e o contêiner do FireLens, a ordem inicial padrão do contêiner é substituída.
- O FireLens para Amazon ECS é compatível com tarefas hospedadas no AWS Fargate no Linux e no Amazon EC2 no Linux. Os contêineres do Windows não são compatíveis com o FireLens.

  Para obter informações sobre como configurar o log centralizado para contêineres do Windows, consulte Centralized logging for Windows containers on Amazon ECS using Fluent Bit (Logs centralizados para contêineres do Windows no Amazon ECS usando o FluentBit).

- É possível usar modelos do AWS CloudFormation para configurar o FireLens para o Amazon ECS. Para obter mais informações, consulte AWS::ECS::TaskDefinition FirelensConfiguration no Guia do usuário do AWS CloudFormation
- O FireLens escuta na porta `24224`. Portanto, para garantir que o roteador de log do FireLens não seja acessível fora da tarefa, você não deve permitir o tráfego de entrada na porta `24224` no grupo de segurança que a tarefa usa. Para as tarefas que usam o modo de rede `awsvpc` esse é o grupo de segurança associado à tarefa. Para as tarefas que usam o modo de rede `host` esse é o grupo de segurança associado à instância do Amazon EC2 que hospeda a tarefa. Para as tarefas que usam o modo de rede `bridge`, não crie qualquer mapeamento de porta que use porta `24224`.
- Para tarefas que usam o modo de rede `bridge`, o contêiner com a configuração do FireLens deve ser iniciado antes que um contêiner de aplicativo que dependa dele seja iniciado. Para controlar a ordem inicial dos contêineres, use as condições de dependência na definição de tarefa. Para obter mais informações, consulte Dependência de contêiner.

> **Note**
>
> Se você usar parâmetros de condição de dependência em definições de contêiner com uma configuração do FireLens, verifique se cada contêiner tem um requisito de condição `START` ou `HEALTHY`.

- Por padrão, o FireLens adiciona o nome de definição de cluster e o nome do recurso da Amazon (ARN) do cluster como chaves de metadados aos seus logs de contêiner stdout/stderr. O exemplo a seguir é do formato de metadados.

```
"ecs_cluster": "cluster-name",
"ecs_task_arn": "arn:aws:ecs:region:111122223333:task/cluster-
name/f2ad7dba413f45ddb4EXAMPLE",
"ecs_task_definition": "task-def-name:revision",
```

Se não quiser os metadados nos seus logs, defina `enable-ecs-log-metadata` como `false`na seção `firelensConfiguration` da definição de tarefa.

```
"firelensConfiguration":{
   "type":"fluentbit",
   "options":{
      "enable-ecs-log-metadata":"false",
      "config-file-type":"file",
      "config-file-value":"/extra.conf"
```

```
}
```

Para usar esse recurso, é necessário criar uma função do IAM para suas tarefas que forneça as permissões necessárias para usar todos os serviços da AWS necessários para as tarefas. Por exemplo, se um contêiner estiver encaminhando logs para o Firehose, a tarefa exigirá permissão para chamar a API `firehose:PutRecordBatch`. Para obter mais informações, consulte Adicionar e remover permissões de identidade do IAM no Guia do usuário do IAM.

Além disso, a tarefa pode requerer o perfil de execução de tarefas do Amazon ECS nas condições apresentadas a seguir. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.

- Se a tarefa for hospedada no Fargate e você estiver extraindo imagens de contêiner do Amazon ECR ou fazendo referência a dados sigilosos do AWS Secrets Manager na sua configuração de log, deverá incluir a função do IAM de execução de tarefas.
- Ao usar um arquivo de configuração personalizado hospedado no Amazon S3, o perfil do IAM de execução de tarefas deve incluir a permissão `s3:GetObject`.

Para obter informações sobre como usar vários arquivos de configuração com o Amazon ECS, incluindo arquivos hospedados por você ou arquivos no Amazon S3, consulte Init process for Fluent Bit on ECS, multi-config support.

Para obter informações sobre configurações de exemplo, consulte Exemplo de definição de tarefa do Amazon ECS: rotear logs para o FireLens.

Configuração de logs do Amazon ECS para obtenção de alto throughput

Ao criar uma definição de tarefa, você pode especificar o número de linhas de log que são armazenadas em buffer na memória especificando o valor em `log-driver-buffer-limit`. Para obter mais informações, consulte Driver de registro do Fluentd na documentação do Docker.

Use essa opção quando o throughput for alto, porque o Docker pode ficar sem memória de buffer e descartar as mensagens em buffer para poder adicionar novas mensagens.

Considere o seguinte ao usar o FireLens para Amazon ECS com a opção de limite de buffer:

- Essa opção é compatível com o tipo de inicialização do Amazon EC2 e o tipo de inicialização do Fargate com a versão da plataforma `1.4.0` ou posterior.

- A opção só é válida quando `logDriver` estiver definido como `awsfirelens`.
- O limite de buffer padrão é de `1048576` linhas de log.
- O limite de buffer deve ser maior ou igual a `0` ou menor que as linhas de log `536870912`.
- A quantidade máxima de memória usada para esse buffer é o produto do tamanho de cada linha de log pelo tamanho do buffer. Por exemplo, se as linhas de log da aplicação tivessem em média 2 KiB, um limite de buffer de 4096 usaria no máximo 8 MiB. A quantidade total de memória alocada no nível da tarefa deve ser maior que a quantidade de memória alocada para todos os contêineres, mais o limite de buffer da memória.

Quando o driver de log `awsfirelens` for especificado em uma definição de tarefa, o agente do Amazon ECS injetará as seguintes variáveis de ambiente no contêiner:

`FLUENT_HOST`

O endereço IP atribuído ao contêiner do FireLens.

> **Note**
>
> Se você estiver usando o tipo de inicialização do EC2 com o modo de rede `bridge`, a variável de ambiente `FLUENT_HOST` no contêiner da aplicação pode se tornar imprecisa após a reinicialização do contêiner do roteador de log do FireLens (o contêiner com o objeto `firelensConfiguration` em sua definição de contêiner). Isso ocorre porque o `FLUENT_HOST` é um endereço IP dinâmico e pode mudar após uma reinicialização. O registro em log diretamente do contêiner da aplicação para o endereço IP do `FLUENT_HOST` pode começar a falhar após a alteração do endereço. Para obter mais informações sobre como reiniciar contêineres individuais, consulte Reiniciar contêineres individuais em tarefas do Amazon ECS com políticas de reinicialização de contêineres.

`FLUENT_PORT`

A porta em que o protocolo Fluent Forward está escutando.

É possível usar as variáveis de ambiente `FLUENT_HOST` e `FLUENT_PORT` para fazer login diretamente no roteador de log do código em vez de passar por `stdout`. Para obter mais informações, consulte fluent-logger-golang no GitHub.

Veja a seguir a sintaxe para especificar o `log-driver-buffer-limit`. Substitua `my_service_` pelo nome do seu serviço:

```
{
    "containerDefinitions": [
        {
            "name": "my_service_log_router",
            "image": "public.ecr.aws/aws-observability/aws-for-fluent-bit:stable",
            "cpu": 0,
            "memoryReservation": 51,
            "portMappings": [],
            "essential": true,
            "environment": [],
            "mountPoints": [],
            "volumesFrom": [],
            "user": "0",
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/ecs-aws-firelens-sidecar-container",
                    "mode": "non-blocking",
                    "awslogs-create-group": "true",
                    "max-buffer-size": "25m",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "firelens"
                },
                "secretOptions": []
            },
            "systemControls": [],
            "firelensConfiguration": {
                "type": "fluentbit"
            }
        },
        {
            "essential": true,
            "image": "public.ecr.aws/docker/library/httpd:latest",
            "name": "app",
            "logConfiguration": {
                "logDriver": "awsfirelens",
                "options": {
                    "Name": "firehose",
                    "region": "us-west-2",
                    "delivery_stream": "my-stream",
```

```
                    "log-driver-buffer-limit": "51200"
                }
            },
            "dependsOn": [
                {
                    "containerName": "log_router",
                    "condition": "START"
                }
            ],
            "memoryReservation": 100
        }
    ]
}
```

Repositórios de imagens da AWS para Fluent Bit para Amazon ECS

A AWS fornece uma imagem do Fluent Bit com plug-ins para o CloudWatch Logs e o Firehose. Recomendamos usar o Fluent Bit como seu roteador de log porque ele tem uma taxa de utilização de recursos mais baixa do que o Fluentd. Para obter mais informações, consulte CloudWatch Logs for Fluent Bit e Amazon Kinesis Firehose for Fluent Bit.

A imagem do AWS para o Fluent Bit está disponível no Amazon ECR na Galeria Pública do Amazon ECR e em um repositório do Amazon ECR na maioria das Regiões da AWS para alta disponibilidade.

Galeria pública do Amazon ECR

A imagem do AWS for Fluent Bit está disponível na Galeria Pública do Amazon ECR. Esse é o local recomendado para baixar a imagem da AWS for Fluent Bit, uma vez que é um repositório público e está disponível para ser usado em todas as Regiões da AWS. Para obter mais informações, consulte aws-for-fluent-bit na Galeria Pública do Amazon ECR.

Linux

A imagem do Fluent Bit para AWS na Galeria pública do Amazon ECR é compatível com o sistema operacional Amazon Linux com a arquitetura `ARM 64` ou `x86-64`.

É possível extrair a imagem do AWS for Fluent Bit da Galeria Pública do Amazon ECR especificando o URL do repositório com a etiqueta de imagem desejada. As etiquetas de imagem disponíveis podem ser encontradas na guia Image tags (Etiquetas de imagem) na Galeria Pública do Amazon ECR.

Veja a seguir a sintaxe a ser usada para a CLI do Docker.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:tag
```

Por exemplo, você pode extrair a imagem mais recente do AWS for Fluent Bit usando este comando da CLI do Docker.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:stable
```

> **Note**
>
> Extrações não autenticadas são permitidas, mas têm um limite de taxa mais baixo do que as extrações autenticadas. Para autenticar usando a conta da AWS antes da extração, use o seguinte comando.
>
> ```
> aws ecr-public get-login-password --region us-east-1 | docker login --username
>  AWS --password-stdin public.ecr.aws
> ```

Windows

A imagem do Fluent Bit para AWS na Galeria pública do Amazon ECR é compatível com a arquitetura `AMD64` dos seguintes sistemas operacionais:

- Windows Server 2022 Full
- Windows Server 2022 Core
- Windows Server 2019 Full
- Windows Server 2019 Core

Contêineres do Windows que estão na AWS não oferecem suporte ao FireLens.

É possível extrair a imagem do AWS for Fluent Bit da Galeria Pública do Amazon ECR especificando o URL do repositório com a etiqueta de imagem desejada. As etiquetas de imagem disponíveis podem ser encontradas na guia Image tags (Etiquetas de imagem) na Galeria Pública do Amazon ECR.

Veja a seguir a sintaxe a ser usada para a CLI do Docker.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:tag
```

Por exemplo, você pode extrair a imagem estável mais recente do AWS para Fluent Bit usando este comando da CLI do Docker.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:windowsservercore-stable
```

> **Note**
>
> Extrações não autenticadas são permitidas, mas têm um limite de taxa mais baixo do que as extrações autenticadas. Para autenticar usando a conta da AWS antes da extração, use o seguinte comando.
>
> ```
> aws ecr-public get-login-password --region us-east-1 | docker login --username AWS --password-stdin public.ecr.aws
> ```

Amazon ECR

A imagem da AWS for Fluent Bit está disponível no Amazon ECR para a obtenção de alta disponibilidade. Essas imagens estão disponíveis na maioria das Regiões da AWS, incluindo AWS GovCloud (US).

Linux

O URI da imagem estável mais recente do AWS for Fluent Bit pode ser recuperado por meio do seguinte comando.

```
aws ssm get-parameters \
      --names /aws/service/aws-for-fluent-bit/stable \
      --region us-east-1
```

Todas as versões da imagem da AWS for Fluent Bit podem ser listadas por meio do seguinte comando para consultar o parâmetro do Systems Manager Parameter Store.

```
aws ssm get-parameters-by-path \
      --path /aws/service/aws-for-fluent-bit \
      --region us-east-1
```

A imagem estável mais recente do AWS para Fluent Bit pode ser especificada em um modelo do AWS CloudFormation mediante a menção do nome de armazenamento de parâmetros do Systems Manager. Veja um exemplo a seguir:

```
Parameters:
  FireLensImage:
    Description: Fluent Bit image for the FireLens Container
    Type: AWS::SSM::Parameter::Value<String>
    Default: /aws/service/aws-for-fluent-bit/stable
```

Windows

O URI da imagem estável mais recente do AWS for Fluent Bit pode ser recuperado por meio do seguinte comando.

```
aws ssm get-parameters \
      --names /aws/service/aws-for-fluent-bit/windowsservercore-stable \
      --region us-east-1
```

Todas as versões da imagem da AWS for Fluent Bit podem ser listadas por meio do seguinte comando para consultar o parâmetro do Systems Manager Parameter Store.

```
aws ssm get-parameters-by-path \
      --path /aws/service/aws-for-fluent-bit/windowsservercore \
      --region us-east-1
```

A imagem estável mais recente do AWS for Fluent Bit pode ser especificada em um modelo do AWS CloudFormation ao fazer referência ao nome de armazenamento de parâmetros do Systems Manager. Veja um exemplo a seguir.

```
Parameters:
  FireLensImage:
    Description: Fluent Bit image for the FireLens Container
    Type: AWS::SSM::Parameter::Value<String>
    Default: /aws/service/aws-for-fluent-bit/windowsservercore-stable
```

Exemplo de definição de tarefa do Amazon ECS: rotear logs para o FireLens

Para usar o roteamento de log personalizado com o FireLens, é necessário especificar o seguinte em sua definição de tarefa:

- Um contêiner de roteador de log com uma configuração do FireLens. Recomendamos que o contêiner seja marcado como `essential`.
- Um ou mais contêineres de aplicativo que contêm uma configuração de log especificando o driver de log `awsfirelens`.
- Um nome do recurso da Amazon (ARN) de perfil do IAM que contém as permissões necessárias para que a tarefa roteie os logs.

Ao criar uma nova definição de tarefa usando o AWS Management Console, há uma seção de integração do FireLens que facilita a adição de um contêiner de roteador de log. Para obter mais informações, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

O Amazon ECS converte a configuração de log e gera a configuração de saída do Fluentd ou Fluent Bit. A configuração de saída é montada no contêiner de roteamento de log em `/fluent-bit/etc/fluent-bit.conf` para o Fluent Bit e `/fluentd/etc/fluent.conf` para o Fluentd.

> **Important**
>
> O FireLens escuta na porta `24224`. Portanto, para garantir que o roteador de log do FireLens não seja acessível fora da tarefa, você não deve permitir tráfego de entrada na porta `24224` no grupo de segurança usado por essa tarefa. Para tarefas que usam o modo de rede `awsvpc`, esse é o grupo de segurança associado à tarefa. Para as tarefas que usam o modo de rede `host` esse é o grupo de segurança associado à instância do Amazon EC2 que hospeda a tarefa. Para as tarefas que usam o modo de rede `bridge`, não crie qualquer mapeamento de porta que use porta `24224`.

Por padrão, o Amazon ECS adiciona campos às entradas de log que ajudam a identificar a fonte dos logs.

- `ecs_cluster`: o nome do cluster do qual a tarefa faz parte.
- `ecs_task_arn`: o nome do recurso da Amazon (ARN) da tarefa da qual o contêiner faz parte.
- `ecs_task_definition`: o nome da definição de tarefa e a revisão que a tarefa está usando.
- `ec2_instance_id`: o ID da instância do Amazon EC2 na qual o contêiner está hospedado. Esse campo só é válido para tarefas que usam o tipo de inicialização do EC2.

Você pode definir os `enable-ecs-log-metadata` como `false` se não quiser os metadados.

O exemplo de definição de tarefa a seguir define um contêiner de roteador de log que usa o Fluent para encaminhar seus logs para o CloudWatch Logs. Ele também define um contêiner de aplicação que usa uma configuração de log para encaminhar logs ao Amazon Data Firehose e definir a memória usada para eventos de buffer como 2 MiB.

> **Note**
>
> Para obter exemplos de definições de tarefas, consulte Exemplos do FireLens do Amazon ECS no GitHub.

```
{
  "family": "firelens-example-firehose",
  "taskRoleArn": "arn:aws:iam::123456789012:role/ecs_task_iam_role",
  "containerDefinitions": [
    {
            "name": "log_router",
            "image": "public.ecr.aws/aws-observability/aws-for-fluent-bit:stable",
            "cpu": 0,
            "memoryReservation": 51,
            "portMappings": [],
            "essential": true,
            "environment": [],
            "mountPoints": [],
            "volumesFrom": [],
            "user": "0",
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/ecs-aws-firelens-sidecar-container",
                    "mode": "non-blocking",
                    "awslogs-create-group": "true",
                    "max-buffer-size": "25m",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "firelens"
                },
                "secretOptions": []
            },
            "systemControls": [],
            "firelensConfiguration": {
                "type": "fluentbit"
            }
```

```
        },
    {
      "essential": true,
      "image": "public.ecr.aws/docker/library/httpd:latest",
      "name": "app",
      "logConfiguration": {
        "logDriver": "awsfirelens",
        "options": {
          "Name": "firehose",
          "region": "us-west-2",
          "delivery_stream": "my-stream",
          "log-driver-buffer-limit": "2097152"
        }
      },
      "memoryReservation": 100
    }
  ]
}
```

Os pares de chave/valor especificados como opções no objeto `logConfiguration` são usados para gerar a configuração de saída do Fluentd ou do Fluent Bit. Veja a seguir um exemplo de código de uma definição de saída do Fluent Bit.

```
[OUTPUT]
    Name   firehose
    Match  app-firelens*
    region us-west-2
    delivery_stream my-stream
```

> **Note**
>
> O FireLens gerencia a configuração `match`. Você não especifica a configuração `match` na definição de tarefa.

Uso de um arquivo de configuração personalizado

É possível especificar um arquivo de configuração personalizado. O formato do arquivo de configuração é o formato nativo do roteador de log que você está usando. Para obter mais informações, consulte Sintaxe do arquivo de configuração do Fluentd e Configuração YAML.

Em seu arquivo de configuração personalizado, para tarefas que usam o modo de rede `bridge` ou `awsvpc`, não defina uma entrada de encaminhamento do Fluentd ou Fluent Bit por TCP porque o FireLens a adiciona à configuração de entrada.

Sua configuração do FireLens deve conter as seguintes opções para especificar um arquivo de configuração personalizado:

`config-file-type`

O local de origem do arquivo de configuração personalizado. As opções disponíveis são `s3` ou `file`.

> **Note**
>
> Tarefas hospedadas no AWS Fargate só são compatíveis com o tipo de arquivo de configuração `file`.

`config-file-value`

A origem do arquivo de configuração personalizado. Se for usado o tipo de arquivo de configuração `s3`, o valor do arquivo de configuração será o ARN completo do arquivo e do bucket do Amazon S3. Se o tipo de arquivo de configuração `file` for usado, o valor do arquivo de configuração será o caminho completo do arquivo de configuração que existe na imagem do contêiner ou em um volume montado no contêiner.

> **Important**
>
> Quando você usa um arquivo de configuração personalizado, precisa especificar um caminho diferente do que o FireLens usa. O Amazon ECS reserva o caminho de arquivo `/fluent-bit/etc/fluent-bit.conf` para o Fluent Bit e `/fluentd/etc/fluent.conf` para o Fluentd.

O exemplo a seguir mostra a sintaxe necessária ao especificar uma configuração personalizada.

> **Important**
>
> Para especificar um arquivo de configuração personalizado hospedado no Amazon S3, verifique se criou uma função do IAM de execução de tarefas com as permissões apropriadas.

Veja a seguir a sintaxe necessária ao especificar uma configuração personalizada.

```
{
  "containerDefinitions": [
    {
      "essential": true,
      "image": "906394416424.dkr.ecr.us-west-2.amazonaws.com/aws-for-fluent-
bit:stable",
      "name": "log_router",
      "firelensConfiguration": {
        "type": "fluentbit",
        "options": {
          "config-file-type": "s3 | file",
          "config-file-value": "arn:aws:s3:::amzn-s3-demo-bucket/fluent.conf
 | filepath"
        }
      }
    }
  ]
}
```

> **Note**
>
> Tarefas hospedadas no AWS Fargate só são compatíveis com o tipo de arquivo de configuração `file`.

Uso de imagens de contêiner que não são da AWS no Amazon ECS

Use o registro privado para armazenar suas credenciais no AWS Secrets Manager e referenciá-las na definição de tarefa. Isso fornece uma maneira de fazer referência a imagens de contêiner que existem em registros privados fora da AWS que exijam autenticação em suas definições de tarefa.

Há suporte para esse recurso em tarefas hospedadas no Fargate, instâncias do Amazon EC2 e instâncias externas usando o Amazon ECS Anywhere.

> **Important**
>
> Se a definição de tarefa faz referência a uma imagem armazenada no Amazon ECR, esse tópico não se aplica. Para obter mais informações, consulte Usar imagens do Amazon ECR com o Amazon ECS no Guia do usuário do Amazon Elastic Container Registry.

Para tarefas hospedadas em instâncias do Amazon EC2, sesse recurso requer a versão `1.19.0` ou posterior do agente de contêiner. Recomendamos usar a versão mais recente do atendente de contêiner. Para obter informações sobre como verificar a versão do agente e atualizar para a versão mais recente, consulte Atualizar o agente de contêiner do Amazon ECS.

Para tarefas hospedadas no Fargate, esse recurso requer a versão da plataforma `1.2.0` ou posterior. Para mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.

Em sua definição do contêiner, especifique o objeto `repositoryCredentials` com os detalhes do segredo que você criou. O segredo referenciado pode estar em uma Região da AWS diferente ou em uma conta distinta daquela que o utiliza para a tarefa.

> **Note**
>
> Quando você usar a API do Amazon ECS, a AWS CLI ou o AWS SDK, se o segredo existir na mesma Região da AWS da tarefa que estiver sendo inicializada, será possível usar o ARN completo ou o nome do segredo. Se o segredo existir em outra conta, o ARN completo do segredo deve ser especificado. Ao usar o AWS Management Console, o ARN completo do segredo deve ser sempre especificado.

Veja a seguir um trecho de uma definição de tarefa que mostra os parâmetros necessários.

Substitua os seguintes parâmetros:

- *`private-repo`* pelo nome do host do repositório privado
- *`private-image`* pelo nome da imagem
- *`arn:aws:secretsmanager:region:aws_account_id:secret:secret_name`* com o nome do recurso da Amazon (ARN) secreto

```
"containerDefinitions": [
    {
        "image": "private-repo/private-image",
        "repositoryCredentials": {
            "credentialsParameter":
 "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
        }
    }
]
```

> **Note**
>
> Outro método para habilitar a autenticação de registro privado usa variáveis de ambiente do agente de contêiner do Amazon ECS para autenticar registros privados. Só há suporte para esse método para tarefas hospedadas em instâncias do Amazon EC2. Para obter mais informações, consulte Configuração de instâncias de contêiner do Amazon ECS para imagens do Docker privadas .

Para usar o registro privado

1. A definição de tarefa deve ter um perfil de execução de tarefas. Isso permite que o agente de contêiner obtenha a imagem do contêiner. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.

   A autenticação de registro privado permite que suas tarefas do Amazon ECS extraiam imagens de contêiner de registros privados fora da AWS (como Docker Hub, Quay.io ou seu próprio registro privado) que exijam credenciais de autenticação. Esse recurso usa o Secrets Manager para armazenar com segurança suas credenciais de registro, que serão mencionadas na definição da tarefa usando o parâmetro `repositoryCredentials`.

   Para obter mais informações sobre como configurar a autenticação de registro privado, consulte Uso de imagens de contêiner que não são da AWS no Amazon ECS.

   Para fornecer acesso aos segredos que contêm suas credenciais de registro privado, adicione as permissões a seguir como uma política em linha à função de execução da tarefa. Para obter mais informações, consulte Adicionar e remover políticas do IAM.

- `secretsmanager:GetSecretValue`: obrigatório para recuperar as credenciais do registro privado do Secrets Manager.
- `kms:Decrypt`–obrigatório somente se o segredo usar uma chave personalizada KMS e não a chave padrão. O nome do recurso da Amazon (ARN) da chave personalizada deve ser adicionado como um recurso.

Veja a seguir um exemplo de política em linha que adiciona as permissões.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [

 "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:secret_name",
                "arn:aws:kms:<region>:<aws_account_id>:key/key_id"
            ]
        }
    ]
}
```

2. Use o AWS Secrets Manager para criar um segredo para suas credenciais de registro privado. Para obter informações sobre como criar segredos, consulte Create an AWS Secrets Manager secret no Guia do usuário do AWS Secrets Manager.

   Insira suas credenciais de registro privado usando o seguinte formato:

   ```
   {
     "username" : "privateRegistryUsername",
     "password" : "privateRegistryPassword"
   }
   ```

3. Registre uma definição de tarefa. Para obter mais informações, consulte the section called “Criação de uma definição de tarefa usando o console”.

Reiniciar contêineres individuais em tarefas do Amazon ECS com políticas de reinicialização de contêineres

Você pode habilitar uma política de reinicialização para cada contêiner essencial e não essencial definido na sua definição de tarefa, para superar falhas transitórias mais rapidamente e manter a disponibilidade da tarefa. Quando você habilita uma política de reinicialização para um contêiner, o Amazon ECS pode reiniciar o contêiner se ele sair, sem precisar substituir a tarefa.

Por padrão, as políticas de reinicialização não estão habilitadas para contêineres. Ao habilitar uma política de reinicialização para um contêiner, você pode especificar códigos de saída nos quais o contêiner não será reiniciado. Eles podem ser códigos de saída que indicam sucesso, como o código de saída `0`, que não exige reinicialização. Você também pode especificar por quanto tempo um contêiner deve ser executado com sucesso antes que uma reinicialização possa ser tentada. Para mais informações sobre esses parâmetros, consulte Política de reinicialização. Para obter um exemplo de definição de tarefa que especifica esses valores, consulte Especificar a política de reinicialização de contêiner em uma definição de tarefa do Amazon ECS.

Você pode usar o endpoint de metadados de tarefas do Amazon ECS ou o CloudWatch Container Insights para monitorar o número de vezes que um contêiner foi reiniciado. Para obter mais informações sobre o endpoint de metadados de tarefa, consulte Endpoint de metadados de tarefas do Amazon ECS versão 4 e Endpoint de metadados de tarefas do Amazon ECS versão 4 para tarefas no Fargate. Para obter informações sobre métricas do Container Insights para o Amazon ECS, consulte Métricas Amazon ECS Container Insights no Guia de Usuário do Amazon CloudWatch.

Há suporte para políticas de reinicialização de contêiner em tarefas hospedadas no Fargate, instâncias do Amazon EC2 e instâncias externas usando o Amazon ECS Anywhere.

Considerações

Considere o seguinte antes de habilitar uma política de reinicialização para seu contêiner:

- As políticas de reinicialização não são compatíveis com contêineres do Windows no Fargate.
- Para tarefas hospedadas em instâncias do Amazon EC2, esse recurso requer a versão `1.86.0` ou posterior do agente de contêiner. Recomendamos usar a versão mais recente do atendente de contêiner. Para obter informações sobre como verificar a versão do agente e atualizar para a versão mais recente, consulte Atualizar o agente de contêiner do Amazon ECS.

- Para tarefas hospedadas no Fargate, esse recurso requer a versão da plataforma `1.4.0` ou posterior. Para mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.
- Se você estiver usando o tipo de inicialização do EC2 com o modo de rede `bridge`, a variável de ambiente `FLUENT_HOST` no contêiner da aplicação poderá se tornar imprecisa após a reinicialização do contêiner do roteador de log do FireLens (o contêiner com o objeto `firelensConfiguration` em sua definição de contêiner). Isso ocorre porque o `FLUENT_HOST` é um endereço IP dinâmico e pode mudar após uma reinicialização. O registro em log diretamente do contêiner da aplicação para o endereço IP do `FLUENT_HOST` pode começar a falhar após a alteração do endereço. Para obter mais informações sobre o `FLUENT_HOST`, consulte Configuração de logs do Amazon ECS para obtenção de alto throughput.
- O agente do Amazon ECS gerencia as políticas de reinicialização do contêiner. Se, por algum motivo inesperado, o agente do Amazon ECS falhar ou não estiver mais em execução, o contêiner não será reiniciado.
- O período de tentativa de reinicialização definido em sua política determina o período de tempo (em segundos) durante o qual o contêiner deve funcionar antes que o Amazon ECS reinicie um contêiner.

Especificar a política de reinicialização de contêiner em uma definição de tarefa do Amazon ECS

Para especificar uma política de reinicialização para um contêiner em uma definição de tarefa, na definição do contêiner, especifique o objeto `restartPolicy`. Para obter mais informações sobre o objeto `restartPolicy`, consulte Política de reinicialização.

Veja a seguir uma definição de tarefa usando o tipo de inicialização de contêineres de Linux do Fargate que configura um servidor da Web: A definição do contêiner inclui o objeto `restartPolicy`, com `enabled` definido como true para permitir uma política de reinicialização para o contêiner. O contêiner deve ser executado por 180 segundos antes de poder ser reiniciado e não será reiniciado se sair com o código de saída `0`, o que indica sucesso.

```
{
  "containerDefinitions": [
    {
      "command": [
        "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</title>
 <style>body {margin-top: 40px; background-color: #333;} </style> </head><body>
```

```
 <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in Amazon
 ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html && httpd-
foreground\""
      ],
      "entryPoint": ["sh", "-c"],
      "essential": true,
      "image": "public.ecr.aws/docker/library/httpd:2.4",
      "logConfiguration": {
        "logDriver": "awslogs",
        "options": {
          "awslogs-group": "/ecs/fargate-task-definition",
          "awslogs-region": "us-east-1",
          "awslogs-stream-prefix": "ecs"
        }
      },
      "name": "sample-fargate-app",
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 80,
          "protocol": "tcp"
        }
      ],
      "restartPolicy": {
        "enabled": true,
        "ignoredExitCodes": [0],
        "restartAttemptPeriod": 180
      }
    }
  ],
  "cpu": "256",
  "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
  "family": "fargate-task-definition",
  "memory": "512",
  "networkMode": "awsvpc",
  "runtimePlatform": {
    "operatingSystemFamily": "LINUX"
  },
  "requiresCompatibilities": ["FARGATE"]
}
```

Depois que tiver registrado uma definição de tarefa com o objeto `restartPolicy` em uma definição de contêiner, você poderá executar uma tarefa ou criar um serviço com a definição de tarefa. Para obter mais informações, consulte Execução de uma aplicação como uma tarefa do Amazon ECS e Criação de uma implantação de atualização contínua do Amazon ECS.

Transferência de dados confidenciais para um contêiner do Amazon ECS

É possível transmitir dados confidenciais (p. ex., credenciais de um banco de dados) para seu contêiner com segurança.

Os segredos, como as chaves de API e as credenciais de banco de dados, são frequentemente usados por aplicações para obter acesso a outros sistemas. Eles geralmente consistem em um nome de usuário e senha, um certificado ou uma chave de API. O acesso a esses segredos deve ser restrito a entidades principais específicas do IAM que estejam usando o IAM, e injetados em contêineres no runtime.

Os segredos podem ser injetados perfeitamente nos contêineres a partir do AWS Secrets Manager e do Systems Manager Parameter Store do Amazon ECS. Esses segredos podem ser referenciados em sua tarefa como qualquer um dos seguintes.

1. Eles são referenciadas como variáveis de ambiente que usam o parâmetro de definição de contêiner `secrets`.
2. Eles são referenciados como `secretOptions` se sua plataforma de registro em log exigir autenticação. Para obter mais informações, consulte opções de configuração de logs.
3. Eles são referenciados como segredos extraídos por imagens que usam o parâmetro de definição do contêiner `repositoryCredentials` se o registro de onde o contêiner está sendo retirado exigir autenticação. Use esse método ao extrair imagens da Galeria Pública do Amazon ECR. Para obter mais informações, consulte Autenticação de registro privado para tarefas.

Recomendamos fazer o seguinte ao configurar o gerenciamento de segredos.

Usar o AWS Secrets Manager ou o AWS Systems Manager Parameter Store para armazenar materiais secretos

Você deve armazenar com segurança chaves de API, credenciais de banco de dados e outros materiais secretos no Secrets Manager ou como um parâmetro criptografado no Systems Manager Parameter Store. Esses serviços são semelhantes porque ambos são armazenamentos gerenciados

de chave-valor que usam AWS KMS para criptografar dados sigilosos. O Secrets Manager, entretanto, também inclui a capacidade de alternar segredos automaticamente, gerar segredos aleatórios e compartilhar segredos entre contas. Para utilizar esses recursos, use o Secrets Manager. Caso contrário, use parâmetros criptografados no Systems Manager Parameter Store.

> **Important**
>
> Se seu segredo mudar, você deverá forçar uma nova implantação ou iniciar uma nova tarefa para recuperar o valor secreto mais recente. Para obter mais informações, consulte os tópicos a seguir.
>
> - Tarefas: pare a tarefa e, em seguida, inicie-a. Para obter mais informações, consulte Interrupção de uma tarefa do Amazon ECS e Execução de uma aplicação como uma tarefa do Amazon ECS.
> - Serviço: atualize o serviço e use a opção forçar nova implantação. Para obter mais informações, consulte Atualizar um serviço do Amazon ECS.

Recuperação de dados de um bucket criptografado do Amazon S3

Você deve armazenar os segredos em um bucket criptografado do Amazon S3 e usar perfis de tarefa para restringir o acesso a esses segredos. Isso evita que os valores das variáveis de ambiente vazem inadvertidamente nos logs e sejam revelados durante a execução do `docker inspect`. Ao fazer isso, sua aplicação deve ser gravada para ler o segredo do bucket do Amazon S3. Para obter instruções, consulte Definição do comportamento padrão da criptografia do lado do servidor para buckets do Amazon S3.

Monte o segredo em um volume usando um contêiner auxiliar

Como há um risco elevado de vazamento de dados com variáveis de ambiente, você deve usar um contêiner auxiliar que leia seus segredos do AWS Secrets Manager e os grave em um volume compartilhado. Esse contêiner pode ser executado e sair antes do contêiner da aplicação usando o pedido de contêineres do Amazon ECS. Quando você faz isso, o contêiner da aplicação monta posteriormente o volume em que o segredo foi gravado. Assim como o método de bucket do Amazon S3, sua aplicação deve ser gravada para ler o segredo do volume compartilhado. Como o volume tem como escopo a tarefa, o volume será excluído automaticamente após a interrupção da tarefa. Para obter um exemplo, consulte o projeto task-def.json.

No Amazon EC2, o volume no qual o segredo é gravado pode ser criptografado com uma chave gerenciada pelo cliente AWS KMS. No AWS Fargate, o armazenamento em volume é criptografado automaticamente usando uma chave gerenciada pelo serviço.

Transferência de uma variável de ambiente individual para um contêiner do Amazon ECS

> **Important**
>
> Recomendamos armazenar seus dados sigilosos em segredos do AWS Secrets Manager ou parâmetros do AWS Systems Manager Parameter Store. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.
> As variáveis de ambiente especificadas na definição de tarefa podem ser legíveis por todos os usuários e perfis a quem são permitidos a ação `DescribeTaskDefinition` para a definição de tarefa.

É possível transmitir variáveis de ambiente aos seus contêineres das maneiras a seguir:

- Individualmente usando o parâmetro de definição de contêiner `environment`. Isso é mapeado para a opção `--env` para docker container run.
- Em massa, usando o parâmetro de definição de contêiner `environmentFiles` para listar um ou mais arquivos que contêm as variáveis de ambiente. O arquivo deve estar hospedado no Amazon S3. Isso é mapeado para a opção `--env-file` para docker run.

Veja a seguir um trecho de uma definição de tarefa que mostra como especificar variáveis de ambiente individuais.

```
{
    "family": "",
    "containerDefinitions": [
        {
            "name": "",
            "image": "",
            ...
            "environment": [
                {
                    "name": "variable",
                    "value": "value"
```

```
                }
            ],
            ...
        }
    ],
    ...
}
```

Transferência de variáveis de ambiente para um contêiner do Amazon ECS

> **Important**
>
> Recomendamos armazenar seus dados sigilosos em segredos do AWS Secrets Manager ou parâmetros do AWS Systems Manager Parameter Store. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.
> Arquivos de variáveis de ambiente são objetos no Amazon S3 e todas as considerações de segurança do Amazon S3 se aplicam.
> Não é possível usar o parâmetro `environmentFiles` em contêineres do Windows e em contêineres do Windows no Fargate.

Você pode criar um arquivo de variável de ambiente e armazená-lo no Amazon S3 para passar variáveis de ambiente ao contêiner.

Ao especificar variáveis de ambiente em um arquivo, é possível injetar variáveis de ambiente em massa. Na definição do contêiner, especifique o objeto `environmentFiles` com uma lista de buckets do Amazon S3 que contêm seus arquivos de variáveis de ambiente.

O Amazon ECS não impõe um limite de tamanho às variáveis de ambiente, mas um arquivo de variáveis de ambiente grande pode ocupar o espaço em disco. Cada tarefa que usa um arquivo de variáveis de ambiente faz com que uma cópia do arquivo seja baixada no disco. O Amazon ECS remove o arquivo como parte da limpeza da tarefa.

Para obter informações sobre as variáveis de ambiente com suporte, consulte Parâmetros avançados de definição de contêiner - Ambiente.

Considere o seguinte ao especificar um arquivo de variáveis de ambiente em uma definição de contêiner.

- Para tarefas do Amazon ECS no Amazon EC2, suas instâncias de contêiner exigem que o agente seja da versão `1.39.0` ou posterior para usar esse recurso. Para obter informações sobre como verificar a versão do agente e atualizar para a versão mais recente, consulte Atualizar o agente de contêiner do Amazon ECS.
- Para tarefas do Amazon ECS no AWS Fargate, as tarefas devem usar a versão `1.4.0` ou posterior (Linux) da plataforma para usar este recurso. Para obter mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.

  Verifique se há suporte para a variável para a plataforma do sistema operacional. Para obter mais informações, consulte the section called “Definições de contêiner” e the section called “Outros parâmetros de definição de tarefa”.
- O arquivo deve usar a extensão `.env` e a codificação UTF-8.
- O perfil de execução da tarefa é necessário para usar esse recurso com as permissões adicionais para o Amazon S3. Isso permite que o agente de contêiner extraia o arquivo de variáveis de ambiente do Amazon S3. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.
- Há um limite de dez arquivos por definição de tarefa.
- Cada linha em um arquivo de ambiente deve conter uma variável de ambiente no formato `VARIABLE=VALUE`. Espaços ou aspas são incluídos como parte dos valores para arquivos do Amazon ECS. As linhas que começam com # são tratadas como comentários e são ignoradas. Para obter mais informações sobre a sintaxe de arquivos de variáveis de ambiente, consulte Definir variáveis de ambiente (-e, --env, --env-file) na documentação do Docker.

  Confira a sintaxe apropriada a seguir.

  ```
  #This is a comment and will be ignored
  VARIABLE=VALUE
  ENVIRONMENT=PRODUCTION
  ```
- Se houver variáveis de ambiente especificadas usando o parâmetro `environment` em uma definição de contêiner, elas terão precedência sobre as variáveis contidas em um arquivo de ambiente.
- Se forem especificados vários arquivos de ambiente que contenham a mesma variável, elas serão processadas na ordem de entrada. Isso significa que o primeiro valor da variável é usado e os valores subsequentes de variáveis duplicadas são ignorados. Recomendamos usar nomes de variáveis exclusivos.

- Se um arquivo de ambiente for especificado como substituição de contêiner, ele será utilizado. Além disso, quaisquer outros arquivos de ambiente especificados na definição de contêiner são ignorados.
- As regras a seguir se aplicam ao tipo de inicialização do Fargate:
  - O arquivo é tratado de forma semelhante a um arquivo env nativo do Docker.
  - As definições de contêiner que fazem referência a variáveis de ambiente que estão em branco e armazenadas no Amazon S3 não aparecem no contêiner.
  - Não há suporte para o tratamento de escape de shell.
  - O ponto de entrada do contêiner interpreta os valores `VARIABLE`.

Exemplo

Veja a seguir um trecho de uma definição de tarefa que mostra como especificar um arquivo de variável de ambiente.

```
{
    "family": "",
    "containerDefinitions": [
        {
            "name": "",
            "image": "",
            ...
            "environmentFiles": [
                {
                    "value": "arn:aws:s3:::amzn-s3-demo-
bucket/envfile_object_name.env",
                    "type": "s3"
                }
            ],
            ...
        }
    ],
    ...
}
```

Transmissão de segredos do Secrets Manager de forma programática no Amazon ECS

Em vez de codificar informações confidenciais em texto simples na aplicação, você pode usar o Secrets Manager para armazenar os dados confidenciais.

Recomendamos esse método de recuperação de dados confidenciais porque a aplicação recuperará automaticamente a versão mais recente do segredo se o segredo do Secrets Manager for atualizado posteriormente.

Crie um segredo no Secrets Manager. Após criar um segredo do Secrets Manager, atualize o código da aplicação para recuperá-lo.

Antes de proteger dados confidenciais no Secrets Manager, analise os seguintes fatores.

- Somente segredos que armazenem dados de texto, criados com parâmetro `SecretString` API CreateSecret, são compatíveis. Segredos que armazenam dados binários, criados com o parâmetro `SecretBinary` da API CreateSecret, não são compatíveis.
- Use os endpoints da VPC de interface para aprimorar os controles de segurança. É necessário criar endpoints da VPC de interface para o Secrets Manager. Para obter informações o endpoint da VPC, consulte Criar endpoints da VPC no Guia do usuário do AWS Secrets Manager.
- A VPC usada por sua tarefa deve usar resolução de DNS.
- A definição de tarefa deve usar um perfil de tarefa com as permissões adicionais para o Secrets Manager. Para obter mais informações, consulte Perfil do IAM para tarefas do Amazon ECS.

Criar o segredo do Secrets Manager

É possível usar o console do Secrets Manager para criar um segredo para seus dados sigilosos. Para obter informações sobre como criar segredos, consulte Criar um segredo do AWS Secrets Manager no Guia do usuário do AWS Secrets Manager.

Atualizar sua aplicação para recuperar programaticamente segredos do Secrets Manager

É possível recuperar segredos com uma chamada diretamente da sua aplicação para as APIs do Secrets Manager. Para obter informações, consulte Retrieve secrets from AWS Secrets Manager no Guia do usuário do AWS Secrets Manager.

Para recuperar os dados confidenciais armazenados no AWS Secrets Manager, consulte Exemplos de código para AWS Secrets Manager usando AWS SDKs na Biblioteca de códigos de exemplos de códigos do AWS SDK.

Transmissão de segredos do Systems Manager Parameter Store de forma programática no Amazon ECS

O Systems Manager Parameter Store fornece armazenamento e gerenciamento seguros de segredos. É possível armazenar dados como senhas, strings de banco de dados, IDs da instância do EC2 e IDs da AMI, e códigos de licença como valores de parâmetros, em vez de codificar essa informação em sua aplicação. É possível armazenar valores como texto sem formatação ou dados criptografados.

Recomendamos esse método de recuperação de dados confidenciais porque a aplicação recuperará automaticamente a versão mais recente se o parâmetro do Systems Manager Parameter Store for atualizado posteriormente.

Antes de proteger dados confidenciais no Systems Manager Parameter Store, analise os seguintes fatores.

- Só há compatibilidade com segredos que armazenam dados de texto. Não há compatibilidade com segredos que armazenam dados binários.
- Use os endpoints da VPC de interface para aprimorar os controles de segurança.
- A VPC usada por sua tarefa deve usar resolução de DNS.
- Para tarefas que usam o tipo de inicialização do EC2, você deve usar a variável de configuração `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE=true` do agente do Amazon ECS para usar esse recurso. É possível adicioná-lo ao arquivo `/etc/ecs/ecs.config` durante a criação da instância de contêiner ou adicioná-lo a uma instância existente e reiniciar o agente do ECS. Para obter mais informações, consulte Configuração do agente de contêiner do Amazon ECS.
- A definição da tarefa deve usar um perfil de tarefa com as permissões adicionais para o Systems Manager Parameter Store. Para obter mais informações, consulte Perfil do IAM para tarefas do Amazon ECS.

Criar o parâmetro

É possível usar o console do Systems Manager para criar um parâmetro do Systems Manager Parameter Store para seus dados confidenciais. Para obter mais informações, consulte Criar um

parâmetro do Systems Manager (console) ou Criar um parâmetro do Systems Manager (AWS CLI) no Guia do usuário do AWS Systems Manager.

Atualizar sua aplicação para recuperar programaticamente segredos do Systems Manager Parameter Store

Para recuperar os dados confidenciais armazenados no parâmetro do Systems Manager Parameter Store, consulte Exemplos de código para Systems Manager usando AWS SDKs na Biblioteca de códigos de exemplos de códigos do AWS SDK.

Transmissão de segredos do Secrets Manager por meio de variáveis de ambiente do Amazon ECS

Ao injetar um segredo como uma variável de ambiente, você pode especificar o conteúdo completo de um segredo, uma chave JSON específica em um segredo. Isso ajuda você a controlar os dados sigilosos expostos ao seu contêiner. Para obter mais informações sobre versionamento de segredos, consulte Qual é o conteúdo de um segredo do Secrets Manager? no Guia do usuário do AWS Secrets Manager.

As informações a seguir devem ser consideradas ao usar uma variável de ambiente para injetar um segredo do Secrets Manager em um contêiner.

- Os dados sigilosos são injetados no contêiner inicialmente quando o contêiner é iniciado. Se o segredo for posteriormente atualizado ou modificado, o contêiner não receberá o valor atualizado automaticamente. Você deve executar uma nova tarefa ou se a tarefa for parte de um serviço, será possível atualizar o serviço e usar a opção Force new deployment (Forçar nova implantação) para forçar o serviço a iniciar uma nova tarefa.
- Os aplicativos executados no contêiner, os logs do contêiner e as ferramentas de depuração têm acesso às variáveis de ambiente.
- Para tarefas do Amazon ECS no AWS Fargate, considere o seguinte:
  - Para injetar o conteúdo completo de um segredo como uma variável de ambiente ou em uma configuração de log, você deve usar a versão `1.3.0` ou posterior da plataforma. Para mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.
  - Para injetar uma chave JSON ou uma versão específica de um segredo como uma variável de ambiente ou em uma configuração de log, você deve usar a versão `1.4.0` ou posterior (Linux) ou `1.0.0` (windows) da plataforma. Para mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.
- Para tarefas do Amazon ECS no EC2, é necessário considerar o seguinte:

- Para injetar um segredo usando uma chave JSON ou uma versão específica de um segredo, sua instância de contêiner deve ter a versão `1.37.0` ou posterior do agente de contêiner. Recomendamos usar a versão mais recente do atendente de contêiner. Para obter informações sobre como verificar a versão do agente e atualizar para a versão mais recente, consulte Atualizar o agente de contêiner do Amazon ECS.

  Para injetar o conteúdo completo de um segredo como uma variável de ambiente ou injetar um segredo em uma configuração de log, sua instância de contêiner deve ter a versão `1.22.0` ou posterior do agente de contêiner.

- Use os endpoints da VPC de interface para aprimorar os controles de segurança e se conectar ao Secrets Manager por meio de uma sub-rede privada. É necessário criar endpoints da VPC de interface para o Secrets Manager. Para obter informações o endpoint da VPC, consulte Criar endpoints da VPC no Guia do usuário do AWS Secrets Manager. Para obter mais informações sobre como usar o Secrets Manager e a Amazon VPC, consulte How to connect to Secrets Manager service within a Amazon VPC.
- Para tarefas do Windows configuradas para usar o driver de log `awslogs`, também é necessário definir a variável de ambiente `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE` na instância de contêiner. Use a seguinte sintaxe:

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE",
 $TRUE, "Machine")
Initialize-ECSAgent -Cluster <cluster name> -EnableTaskIAMRole -LoggingDrivers
 '["json-file","awslogs"]'
</powershell>
```

- A definição de tarefa deve usar um perfil de execução de tarefa com as permissões adicionais para o Secrets Manager. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.

Criar o segredo do AWS Secrets Manager

É possível usar o console do Secrets Manager para criar um segredo para seus dados sigilosos. Para obter mais informações, consulte Criar um segredo do AWS Secrets Manager no Guia do usuário do AWS Secrets Manager.

Adicionar a variável de ambiente à definição de contêiner

Na definição de contêiner, você pode especificar o seguinte:

- O objeto `secrets` contendo o nome da variável de ambiente a ser definida no contêiner
- O nome do recurso da Amazon (ARN) do segredo do Secrets Manager
- Parâmetros adicionais que contêm os dados confidenciais a serem apresentados ao contêiner

O exemplo a seguir mostra a sintaxe completa que deve ser especificada para o segredo do Secrets Manager.

```
arn:aws:secretsmanager:region:aws_account_id:secret:secret-name:json-key:version-stage:version-id
```

A seção a seguir descreve os parâmetros adicionais. Esses parâmetros são opcionais, mas, se não usá-los, você deverá incluir os dois-pontos `:` para usar os valores padrão. Abaixo, exemplos provendo mais contexto.

`json-key`

Especifica o nome da chave em um par de chave/ valor, com o valor que deseje definir como o valor da variável de ambiente. Somente valores formato JSON são compatíveis. Se você não especificar uma chave JSON, será usado o conteúdo completo do segredo.

`version-stage`

Especifique o rótulo de preparação da versão do segredo que deseja usar. Se um rótulo de preparação de versão for especificado, você não poderá especificar um ID de versão. Se nenhum estágio de versão for especificado, o comportamento padrão será recuperar o segredo com o rótulo de preparação `AWSCURRENT`.

Rótulos de preparação são usados para monitoramento de diferentes versões de um segredo quando eles forem atualizados ou alternados. Cada versão de um segredo tem um ou mais rótulos de preparação e uma ID.

`version-id`

Especifica o identificador exclusivo da versão do segredo que você deseja usar. Se um ID de versão for especificado, você não poderá especificar um rótulo de preparação de versão. Se nenhuma ID de versão for especificada, o comportamento padrão será recuperar o segredo com o rótulo de preparação `AWSCURRENT`.

IDs da versão são usadas para monitoramento de diferentes versões de um segredo quando atualizados ou alternados. Cada versão de um segredo tem uma ID. Para obter mais

informações, consulte Termos e conceitos principais do AWS Secrets Manager no Guia do usuário do AWS Secrets Manager.

Exemplo de definições de contêiner

Os exemplos a seguir mostram maneiras como você pode fazer referência a segredos do Secrets Manager nas suas definições de contêiner.

Example Referenciando um segredo completo

O seguinte é um trecho de definição de tarefa mostrando formato, ao fazer referência ao texto completo de um segredo Secrets Manager.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
    }]
  }]
}
```

Para acessar o valor desse segredo diretamente do contêiner, você precisaria chamar `$environment_variable_name`.

Example referenciando segredos completos

Veja a seguir um trecho de uma definição de tarefa mostrando o formato ao fazer referência ao texto completo de vários segredos do Secrets Manager.

```
{
  "containerDefinitions": [{
     "secrets": [
      {
        "name": "environment_variable_name1",
         "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
      },
      {
        "name": "environment_variable_name2",
```

```
          "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
abcdef"
      },
      {
        "name": "environment_variable_name3",
        "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
ABCDEF"
      }
    ]
  }]
}
```

Para acessar o valor desse segredo diretamente do contêiner, você precisaria chamar `$environment_variable_name1`, `$environment_variable_name2` e `$environment_variable_name3`.

Example Referenciando uma chave específica dentro de um segredo

O seguinte é um exemplo de saída de comando get-secret-value exibe o conteúdo de um segredo junto ao rótulo de estágio da versão e ID da versão associada a ele.

```
{
    "ARN": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf",
    "Name": "appauthexample",
    "VersionId": "871d9eca-18aa-46a9-8785-981ddEXAMPLE",
    "SecretString": "{\"username1\":\"password1\",\"username2\":\"password2\",
\"username3\":\"password3\"}",
    "VersionStages": [
        "AWSCURRENT"
    ],
    "CreatedDate": 1581968848.921
}
```

Faça referência a uma chave específica de saída anterior em uma definição de contêiner especificando o nome da chave no fim do ARN.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1::"
```

```
    }]
  }]
}
```

Example Referenciando uma versão secreta específica

O seguinte é um exemplo de saída de comando describe-secret exibindo o conteúdo não criptografado de um segredo junto aos metadados de todas as versões do segredo.

```
{
    "ARN": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf",
    "Name": "appauthexample",
    "Description": "Example of a secret containing application authorization data.",
    "RotationEnabled": false,
    "LastChangedDate": 1581968848.926,
    "LastAccessedDate": 1581897600.0,
    "Tags": [],
    "VersionIdsToStages": {
        "871d9eca-18aa-46a9-8785-981ddEXAMPLE": [
            "AWSCURRENT"
        ],
        "9d4cb84b-ad69-40c0-a0ab-cead3EXAMPLE": [
            "AWSPREVIOUS"
        ]
    }
}
```

Faça referência a um rótulo específico de preparação de versão de saída anterior em uma definição de contêiner especificando o nome da chave no fim do ARN.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf::AWSPREVIOUS:"
    }]
  }]
}
```

Faça referência a uma ID da versão específica da saída anterior em uma definição de contêiner especificando o nome da chave no final do ARN.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:::9d4cb84b-ad69-40c0-a0ab-cead3EXAMPLE"
    }]
  }]
}
```

Example Referenciando uma chave específica e um rótulo de estágio de versão de um segredo

O seguinte é um exemplo de como fazer referência a uma chave específica dentro de um segredo e rótulo de estágio de versão específico.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1:AWSPREVIOUS:"
    }]
  }]
}
```

Para uma chave específica e ID da versão, use a sintaxe a seguir.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1::9d4cb84b-ad69-40c0-a0ab-cead3EXAMPLE"
    }]
  }]
}
```

Para obter informações sobre como criar uma definição de tarefa com o segredo especificado em uma variável de ambiente, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

Transmissão de parâmetros do Systems Manager por meio de variáveis de ambiente do Amazon ECS

O Amazon ECS permite que você introduza dados confidenciais em contêineres, ao armazená-los nos parâmetros do AWS Systems Manager Parameter Store e, em seguida, referenciá-los na definição do contêiner.

Considere as informações apresentadas a seguir ao usar uma variável de ambiente para introduzir um segredo do Systems Manager em um contêiner.

- Os dados sigilosos são injetados no contêiner inicialmente quando o contêiner é iniciado. Se o segredo for posteriormente atualizado ou modificado, o contêiner não receberá o valor atualizado automaticamente. Você deve executar uma nova tarefa ou se a tarefa for parte de um serviço, será possível atualizar o serviço e usar a opção Force new deployment (Forçar nova implantação) para forçar o serviço a iniciar uma nova tarefa.
- Para tarefas do Amazon ECS no AWS Fargate, o seguinte deve ser considerado:
  - Para injetar o conteúdo completo de um segredo como uma variável de ambiente ou em uma configuração de log, você deve usar a versão `1.3.0` ou posterior da plataforma. Para mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.
  - Para injetar uma chave JSON ou uma versão específica de um segredo como uma variável de ambiente ou em uma configuração de log, você deve usar a versão `1.4.0` ou posterior (Linux) ou `1.0.0` (windows) da plataforma. Para mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.
- Para tarefas do Amazon ECS no EC2, é necessário considerar o seguinte:
  - Para injetar um segredo usando uma chave JSON ou uma versão específica de um segredo, sua instância de contêiner deve ter a versão `1.37.0` ou posterior do agente de contêiner. Recomendamos usar a versão mais recente do atendente de contêiner. Para obter informações sobre como verificar a versão do agente e atualizar para a versão mais recente, consulte Atualizar o agente de contêiner do Amazon ECS.

    Para injetar o conteúdo completo de um segredo como uma variável de ambiente ou injetar um segredo em uma configuração de log, sua instância de contêiner deve ter a versão `1.22.0` ou posterior do agente de contêiner.
- Use os endpoints da VPC de interface para aprimorar os controles de segurança. Você deve criar os endpoints da VPC de interface para o Systems Manager. Para obter mais informações sobre o

endpoint da VPC, consulte Melhorar a segurança das instâncias do EC2 usando endpoints da VPC para o Systems Manager no Guia do usuário do AWS Systems Manager.

- A definição da tarefa deve usar um perfil de execução de tarefa com as permissões adicionais para o Systems Manager Parameter Store. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.
- Para tarefas do Windows configuradas para usar o driver de log `awslogs`, também é necessário definir a variável de ambiente `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE` na instância de contêiner. Use a seguinte sintaxe:

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE",
 $TRUE, "Machine")
Initialize-ECSAgent -Cluster <cluster name> -EnableTaskIAMRole -LoggingDrivers
 '["json-file","awslogs"]'
</powershell>
```

Criação do parâmetro do Systems Manager

É possível usar o console do Systems Manager para criar um parâmetro do Systems Manager Parameter Store para seus dados confidenciais. Para obter mais informações, consulte Criar um parâmetro do Systems Manager (console) ou Criar um parâmetro do Systems Manager (AWS CLI) no Guia do usuário do AWS Systems Manager.

Adicionar a variável de ambiente à definição de contêiner

Na definição do contêiner na definição da tarefa, especifique `secrets` com o nome da variável de ambiente a ser definida no contêiner e o ARN completo do parâmetro Systems Manager Parameter Store contendo os dados sigilosos a serem apresentados. Para obter mais informações, consulte `secrets`.

Veja a seguir um trecho de uma definição de tarefa mostrando o formato ao fazer referência a um parâmetro do Systems Manager Parameter Store. Se o parâmetro do Systems Manager Parameter Store existir na mesma região da tarefa que está sendo iniciada, será possível usar o ARN completo ou o nome do parâmetro. Se o parâmetro existir em uma região diferente, o ARN completo deverá ser especificado.

```
{
  "containerDefinitions": [{
```

```
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }]
  }]
}
```

Para obter informações sobre como criar uma definição de tarefa com o segredo especificado em uma variável de ambiente, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

Atualizar sua aplicação para recuperar programaticamente segredos do Systems Manager Parameter Store

Para recuperar os dados confidenciais armazenados no parâmetro do Systems Manager Parameter Store, consulte Exemplos de código para Systems Manager usando AWS SDKs na Biblioteca de códigos de exemplos de códigos do AWS SDK.

Transmissão de segredos para a configuração do registro em log do Amazon ECS

Você pode usar o parâmetro `secretOptions` em `logConfiguration` para passar dados confidenciais usados no registro em log.

É possível armazenar o segredo no Secrets Manager ou no Systems Manager.

Uso do Secrets Manager

Na sua definição de contêiner, ao especificar uma `logConfiguration`, você pode especificar `secretOptions` com o nome da opção de driver de log a ser definida no contêiner e o ARN completo do segredo do Secrets Manager que contém os dados sigilosos a serem apresentados ao contêiner. Para obter mais informações sobre a criação de segredos, consulte Criar um AWS Secrets Manager.

Veja a seguir um trecho de uma definição de tarefa mostrando o formato ao fazer referência a um segredo do Secrets Manager.

```
{
  "containerDefinitions": [{
    "logConfiguration": [{
      "logDriver": "splunk",
      "options": {
```

```
        "splunk-url": "https://your_splunk_instance:8088"
      },
      "secretOptions": [{
        "name": "splunk-token",
        "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
      }]
    }]
  }]
}
```

Adicionar a variável de ambiente à definição de contêiner

Em sua definição de contêiner, especifique `secrets` com o nome da variável de ambiente a ser definida no contêiner e o ARN completo do parâmetro Systems Manager Parameter Store contendo os dados sigilosos a serem apresentados. Para obter mais informações, consulte `secrets`.

Veja a seguir um trecho de uma definição de tarefa mostrando o formato ao fazer referência a um parâmetro do Systems Manager Parameter Store. Se o parâmetro do Systems Manager Parameter Store existir na mesma região da tarefa que está sendo iniciada, será possível usar o ARN completo ou o nome do parâmetro. Se o parâmetro existir em uma região diferente, o ARN completo deverá ser especificado.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }]
  }]
}
```

Para obter informações sobre como criar uma definição de tarefa com o segredo especificado em uma variável de ambiente, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

Usar o Systems Manager

É possível injetar dados confidenciais em uma configuração de log. Em sua definição de contêiner, ao especificar `logConfiguration`, será possível especificar `secretOptions` com o nome da opção de driver de log a ser definida no contêiner e o ARN completo do parâmetro do Systems Manager Parameter Store que contém os dados sigilosos a serem apresentados ao contêiner.

> **Important**
>
> Se o parâmetro do Systems Manager Parameter Store existir na mesma região da tarefa que está sendo iniciada, será possível usar o ARN completo ou o nome do parâmetro. Se o parâmetro existir em uma região diferente, o ARN completo deverá ser especificado.

Veja a seguir um trecho de uma definição de tarefa mostrando o formato ao fazer referência a um parâmetro do Systems Manager Parameter Store.

```
{
  "containerDefinitions": [{
    "logConfiguration": [{
      "logDriver": "fluentd",
      "options": {
        "tag": "fluentd demo"
      },
      "secretOptions": [{
        "name": "fluentd-address",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter:/parameter_name"
      }]
    }]
  }]
}
```

Especificar dados confidenciais usando segredos do Secrets Manager no Amazon ECS

O Amazon ECS permite a injeção de dados confidenciais nos contêineres armazenando esses dados confidenciais em segredos do AWS Secrets Manager e depois referenciando-os na definição do contêiner. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.

Saiba como criar um segredo do Secrets Manager, fazer referência ao segredo em uma definição de tarefa do Amazon ECS e, em seguida, verificar se isso funcionou ao consultar a variável de ambiente dentro de um contêiner que mostra o conteúdo do segredo.

Pré-requisitos

Este tutorial pressupõe que os seguintes pré-requisitos foram concluídos:

- As etapas em Configuração para usar o Amazon ECS foram concluídas.
- Seu usuário tem as permissões do IAM necessárias para criar os recursos do Secrets Manager e do Amazon ECS.

Etapa 1: Criar um segredo do Secrets Manager

É possível usar o console do Secrets Manager para criar um segredo para seus dados sigilosos. Neste tutorial, vamos criar um segredo básico para armazenar um nome de usuário e uma senha para consulta posterior em um contêiner. Para obter mais informações, consulte Criar um AWS Secrets Manager segredo no Guia do usuário AWS Secrets Manager.

Os pares de chave/valor a serem armazenados nesse segredo constituem o valor da variável de ambiente no contêiner no final do tutorial.

Salve o Secret ARN (ARN do segredo) para ser referenciado na política do IAM de execução de tarefa e a definição de tarefa em etapas posteriores.

Etapa 2: adicionar as permissões de segredos ao perfil de execução da tarefa

Para que o Amazon ECS recupere os dados sigilosos do seu segredo do Secrets Manager, é necessário ter as permissões de segredo para o perfil de execução da tarefa. Para obter mais informações, consulte Permissões do Secrets Manager ou do Systems Manager.

Etapa 3: criar uma definição de tarefa

Você pode usar o console do Amazon ECS para criar uma definição de tarefa que faça referência a um segredo do Secrets Manager.

Para criar uma definição de tarefa que especifica um segredo

Use o console do IAM para atualizar a função de execução de tarefa com as permissões necessárias.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Task definitions (Definições de tarefa).
3. Escolha Create new task definition (Criar nova definição de tarefa), Create new task definition with JSON (Criar nova definição de tarefa com JSON).
4. Na caixa do editor JSON, e insira o texto JSON de definição de tarefa a seguir, garantindo que você especifique o ARN completo do segredo do Secrets Manager criado na etapa 1 e o perfil de execução da tarefa atualizado na etapa 2. Escolha Salvar.

5.

```
{
    "executionRoleArn": "arn:aws:iam::aws_account_id:role/ecsTaskExecutionRole",
    "containerDefinitions": [
        {
            "entryPoint": [
                "sh",
                "-c"
            ],
            "portMappings": [
                {
                    "hostPort": 80,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "command": [
                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
            ],
            "cpu": 10,
            "secrets": [
                {
                    "valueFrom":
 "arn:aws:secretsmanager:region:aws_account_id:secret:username_value",
                    "name": "username_value"
                }
            ],
            "memory": 300,
            "image": "public.ecr.aws/docker/library/httpd:2.4",
            "essential": true,
            "name": "ecs-secrets-container"
        }
    ],
    "family": "ecs-secrets-tutorial"
}
```

6. Escolha Criar.

Etapa 4: criar um cluster

É possível usar o console do Amazon ECS para criar um cluster que contém uma instância de contêiner na qual a tarefa será executada. Se você tiver um cluster existente com pelo menos uma instância de contêiner registrada nele com os recursos disponíveis para executar uma instância da definição de tarefa criada para este tutorial, será possível pular para a próxima etapa.

Para este tutorial, vamos criar um cluster com uma instância de contêiner `t2.micro` usando a AMI do Amazon Linux 2 otimizada para o Amazon ECS.

Para obter informações sobre como criar um cluster para o tipo de inicialização do EC2, consulte the section called “Criação de um cluster do Amazon ECS para o tipo de inicialização do Amazon EC2”.

Etapa 5: executar uma tarefa

É possível usar o console do Amazon ECS para executar uma tarefa usando a definição de tarefa que você criou. Neste tutorial, executaremos uma tarefa usando o tipo de inicialização do EC2, usando o cluster que criamos na etapa anterior.

Para obter informações sobre como executar uma tarefa, consulte the section called “Execução de uma aplicação como uma tarefa”.

Etapa 6: verificar

É possível verificar se todas as etapas foram concluídas com êxito e a variável de ambiente foi criada corretamente em seu contêiner usando as etapas a seguir.

Para verificar se a variável de ambiente foi criada

1. Encontre o endereço IP público ou DNS para sua instância de contêiner.
    a. Abra o console em https://console.aws.amazon.com/ecs/v2.
    b. No painel de navegação, escolha Clusters e selecione o cluster que você criou.
    c. Escolha Infraestrutura e, em seguida, escolha a instância de contêiner.
    d. Registre o IP público ou o DNS público para sua instância.
2. Se você estiver usando um computador Linux ou macOS, conecte-se à sua instância com o seguinte comando, substituindo o caminho para sua chave privada e o endereço público para sua instância:

```
$ ssh -i /path/to/my-key-pair.pem ec2-user@ec2-198-51-100-1.compute-1.amazonaws.com
```

Para obter mais informações sobre como usar um computador com Windows, consulte Conectar à instância do Linux usando PuTTY no Guia do usuário do Amazon EC2.

> **Important**
>
> Para obter mais informações sobre problemas ao se conectar à instância, consulte Solução de problemas para conectar-se à sua instância no Manual do usuário do Amazon EC2.

3. Liste os contêineres em execução na instância. Anote o ID do contêiner `ecs-secrets-tutorial`.

```
docker ps
```

4. Conecte-se ao contêiner `ecs-secrets-tutorial` usando o ID do contêiner da saída da etapa anterior.

```
docker exec -it container_ID /bin/bash
```

5. Use o comando `echo` para imprimir o valor da variável de ambiente.

```
echo $username_value
```

Se o tutorial foi bem-sucedido, você deverá ver a saída a seguir.

```
password_value
```

> **Note**
>
> Como alternativa, você pode listar todas as variáveis de ambiente em seu contêiner usando o comando `env` (ou `printenv`).

Etapa 7: limpar

Ao concluir este tutorial, você deve limpar os recursos associados para evitar cobranças por recursos não utilizados.

Limpeza dos recursos

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Clusters.
3. Na página Clusters, escolha o cluster.
4. Escolha Delete Cluster.
5. Na caixa de confirmação, insira excluir ***nome do cluster*** e, em seguida, escolha Excluir.
6. Abra o console do IAM em https://console.aws.amazon.com/iam/.
7. No painel de navegação, escolha Perfis.
8. Procure por `ecsTaskExecutionRole` na lista de funções e selecione-a.
9. Escolha Permissões e escolha o X ao lado de ECSSecretsTutorial. Escolha Remover.
10. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
11. Selecione o segredo username_value que você criou e escolha Actions (Ações), Delete secret (Excluir segredo).

Parâmetros de definição de tarefa do Amazon ECS para o tipo de inicialização do Fargate

As definições de tarefa se dividem em partes distintas: a família da tarefa, o perfil da tarefa do AWS Identity and Access Management (IAM), o modo de rede, definições de contêiner, volumes e tipos de inicialização. As definições de família e de contêiner são obrigatórias em uma definição de tarefa. Por outro lado, perfil de tarefa, modo da rede, volumes e tipo de inicialização são opcionais.

É possível usar esses parâmetros em um arquivo JSON para configurar sua definição de tarefa.

Veja a seguir descrições mais detalhadas de cada parâmetro de definição de tarefa para o tipo de inicialização do Fargate.

Família

`family`

Tipo: string

Obrigatório: Sim

Ao registrar uma definição de tarefa, você dá a ela uma família, semelhante a um nome para várias versões da definição de tarefa, especificada com um número de revisão. A primeira definição de tarefa registrada em uma determinada família recebe uma revisão 1, e todas as definições de tarefa registradas depois receberão um número de revisão sequencial.

Tipos de inicialização

Ao registrar uma definição de tarefa, você pode especificar um tipo de inicialização em relação à qual o Amazon ECS deve validar a definição de tarefa. Se a definição de tarefa não for validada em relação às compatibilidades especificadas, uma exceção de cliente será retornada. Para obter mais informações, consulte Tipos de inicialização do Amazon ECS.

O seguinte parâmetro é permitido em uma definição de tarefa.

`requiresCompatibilities`

Tipo: Matriz de strings

Obrigatório: Não

Valores Válidos: `FARGATE`

O tipo de inicialização em relação à qual a definição de tarefa é validada. Isso inicia uma verificação para garantir que todos os parâmetros usados na definição de tarefa atendam aos requisitos do tipo de inicialização.

Função da tarefa

`taskRoleArn`

Tipo: String

Obrigatório: Não

Ao registrar uma definição de tarefa, você pode fornecer uma função de tarefa para uma função do IAM que permita que os contêineres da permissão de tarefa chame as APIs da AWS especificadas nas políticas associadas em seu nome. Para obter mais informações, consulte Perfil do IAM para tarefas do Amazon ECS.

Função de execução de tarefas

`executionRoleArn`

Tipo: string

Obrigatório: Condicional

O nome do recurso da Amazon (ARN) da função de execução de tarefas que concede ao agente de contêiner do Amazon ECS permissão para fazer chamadas de API da AWS em seu nome.

> **Note**
>
> A função do IAM para execução da tarefa é necessária dependendo dos requisitos da sua tarefa. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.

Modo de rede

`networkMode`

Tipo: string

Obrigatório: Sim

O modo de rede do Docker a ser usado para os contêineres na tarefa. Para tarefas do Amazon ECS hospedadas no Fargate, é necessário o modo de rede `awsvpc`.

Plataforma de runtime

`operatingSystemFamily`

Tipo: string

Obrigatório: Condicional

Padrão: LINUX

Esse parâmetro é necessário para tarefas do Amazon ECS hospedadas no Fargate.

Ao registrar uma definição de tarefa, você especifica a família do sistema operacional.

Os valores válidos são `LINUX`, `WINDOWS_SERVER_2025_FULL`, `WINDOWS_SERVER_2025_CORE`, `WINDOWS_SERVER_2022_FULL`, `WINDOWS_SERVER_2022_CORE`, `WINDOWS_SERVER_2019_FULL` e `WINDOWS_SERVER_2019_CORE`.

Todas as definições de tarefa usadas em um serviço devem ter o mesmo valor para esse parâmetro.

Quando uma definição de tarefa faz parte de um serviço, esse valor deve corresponder ao valor `platformFamily` do serviço.

`cpuArchitecture`

Tipo: string

Obrigatório: Condicional

Padrão: X86_64

Se o parâmetro for deixado como `null`, o valor padrão é atribuído automaticamente ao iniciar uma tarefa hospedada no Fargate.

Ao registrar uma definição de tarefa, você especifica a arquitetura da CPU. Os valores válidos são `X86_64` e `ARM64`.

Todas as definições de tarefa usadas em um serviço devem ter o mesmo valor para esse parâmetro.

Quando você tem tarefas do Linux, pode definir o valor como `ARM64`. Para obter mais informações, consulte the section called “Definições de tarefa para workloads do ARM de 64 bits”.

Tamanho da tarefa

Ao registrar uma definição de tarefa, você pode especificar o total de CPU e memória usados para a tarefa. Isso é separado dos valores de `cpu` e `memory` no nível de definição de contêiner. Para tarefas hospedadas no Fargate (tanto de Linux quanto de Windows), esses campos são obrigatórios e existem valores específicos para `cpu` e `memory` com suporte.

O seguinte parâmetro é permitido em uma definição de tarefa:

`cpu`

Tipo: string

Obrigatório: Sim

> **Note**
>
> É necessário informar os parâmetros de CPU e memória no nível da tarefa, que são usados para determinar o tipo e o tamanho da instância na qual as tarefas são executadas. Para tarefas do Windows, esses valores não são aplicados em runtime, porque o Windows não tem um mecanismo nativo que possa impor facilmente limites coletivos de recursos em um grupo de contêineres. Se você quiser impor limites de recursos, recomendamos usar os recursos no nível de contêiner para contêineres do Windows.

O limite rígido de unidades de CPU a ser apresentado para a tarefa. Você pode especificar valores de CPU no arquivo JSON como uma string em unidades de CPU ou em CPUs virtuais (vCPUs). Por exemplo, você pode especificar um valor de CPU como `1024` em unidades de CPU ou `1 vCPU` em vCPUs. Quando a definição de tarefa for registrada, um valor de vCPU será convertido em um inteiro indicando as unidades de CPU.

Esse campo é obrigatório e você deve usar um dos valores a seguir, o que determina o intervalo de valores compatíveis com o parâmetro `memory`. A tabela a seguir mostra as combinações válidas para CPU e memória em nível de tarefa.

Valor de CPU	Valor de memória	Sistemas operacionais com suporte para o AWS Fargate
256 (0,25 vCPU)	512 MiB, 1 GB, 2 GB	Linux
512 (0,5 vCPU)	1 GB, 2 GB, 3 GB, 4 GB	Linux
1024 (1 vCPU)	2 GB, 3 GB, 4 GB, 5 GB, 6 GB, 7 GB, 8 GB	Linux, Windows
2048 (2 vCPU)	Entre 4 GB e 16 GB em incrementos de 1 GB	Linux, Windows

Valor de CPU	Valor de memória	Sistemas operacionais com suporte para o AWS Fargate
4096 (4 vCPU)	Entre 8 GB e 30 GB em incrementos de 1 GB	Linux, Windows
8192 (8 vCPU) **Note** Essa opção requer a plataforma Linux `1.4.0` ou posterior.	Entre 16 GB e 60 GB em incrementos de 4 GB	Linux
16384 (16 vCPU) **Note** Essa opção requer a plataforma Linux `1.4.0` ou posterior.	Entre 32 GB e 120 GB em incrementos de 8 GB	Linux

`memory`

Tipo: string

Obrigatório: Sim

Note

É necessário informar os parâmetros de CPU e memória no nível da tarefa, que são usados para determinar o tipo e o tamanho da instância na qual as tarefas são executadas. Para tarefas do Windows, esses valores não são aplicados em runtime, porque o Windows não tem um mecanismo nativo que possa impor facilmente limites coletivos de recursos em um grupo de contêineres. Se você quiser impor limites de

recursos, recomendamos usar os recursos no nível de contêiner para contêineres do Windows.

O limite fixo de memória a ser apresentada para a tarefa. É possível especificar valores de memória na definição de tarefa como uma string em mebibytes (MiB) ou gigabytes (GB). Por exemplo, você pode especificar um valor de memória como `3072` em MiB ou `3 GB` em GB. Quando a definição de tarefa for registrada, um valor em GB será convertido em um inteiro indicando o MiB.

Esse campo é obrigatório e você deve usar um dos valores a seguir, o que determina o intervalo de valores compatíveis com o parâmetro `cpu`:

Valor da memória (em MiB, com valor equivalente aproximado em GB)	Valor de CPU	Sistemas operacionais com suporte para o Fargate
512 (0,5 GB), 1024 (1 GB), 2048 (2 GB)	256 (0,25 vCPU)	Linux
1024 (1 GB), 2048 (2 GB), 3072 (3 GB), 4096 (4 GB)	512 (0,5 vCPU)	Linux
2048 (2 GB), 3072 (3 GB), 4096 (4 GB), 5120 (5 GB), 6144 (6 GB), 7168 (7 GB), 8192 (8 GB)	1024 (1 vCPU)	Linux, Windows
Entre 4096 (4 GB) e 16384 (16 GB) em incrementos de 1024 (1 GB)	2048 (2 vCPU)	Linux, Windows
Entre 8192 (8 GB) e 30720 (30 GB) em incrementos de 1024 (1 GB)	4096 (4 vCPU)	Linux, Windows
Entre 16 GB e 60 GB em incrementos de 4 GB	8192 (8 vCPU)	Linux

Valor da memória (em MiB, com valor equivalente aproximado em GB)	Valor de CPU	Sistemas operacionais com suporte para o Fargate
Note Essa opção requer a plataforma Linux `1.4.0` ou posterior.		
Entre 32 GB e 120 GB em incrementos de 8 GB **Note** Essa opção requer a plataforma Linux `1.4.0` ou posterior.	16384 (16 vCPU)	Linux

Definições de contêiner

Ao registrar uma definição de tarefa, você deve especificar uma lista de definições de contêiner passadas para o daemon do Docker em uma instância de contêiner. Os parâmetros a seguir são permitidos em uma definição de contêiner.

Tópicos

- Parâmetros padrão de definição de contêiner
- Parâmetros avançados de definição de contêiner
- Outros parâmetros de definição de contêiner

Parâmetros padrão de definição de contêiner

Os parâmetros de definição de tarefa a seguir são obrigatórios ou usados na maioria das definições de contêiner.

Tópicos

- Name
- Imagem
- Memória
- Mapeamentos de porta
- Credenciais de repositório privado

Name

`name`

Tipo: string

Obrigatório: Sim

O nome de um contêiner. São permitidos até 255 letras (caixa alta e baixa), números, hífens e sublinhados. Se estiver vinculando vários contêineres em uma definição de tarefa, o `name` de um contêiner poderá ser informado no `links` de outro contêiner. Isso deve ser feito para conectar os contêineres.

Imagem

`image`

Tipo: string

Obrigatório: Sim

A imagem usada para iniciar um contêiner. Essa string é passada diretamente para o daemon do Docker. Por padrão, as imagens no registro do Docker Hub estão disponíveis. Você também pode especificar outros repositórios com *`repository-url`*`/`*`image`*`:`*`tag`* ou *`repository-url`*`/`*`image`*`@`*`digest`*. São permitidos até 255 letras (caixa alta e baixa), números, hifens, sublinhados, dois pontos, ponto, barras e sinais numéricos. Esse parâmetro é mapeado para `Image` no comando create-container do docker e no parâmetro `IMAGE` do comando run do docker.

- Quando uma nova tarefa é iniciada, o agente do contêiner do Amazon ECS obtém a versão mais recente da imagem especificada e a tag do contêiner a ser usado. No entanto, as atualizações subsequentes feitas em um repositório de imagens não são propagadas para tarefas já em execução.

- Quando não é especificada uma tag ou um resumo no caminho de imagem na definição da tarefa, o agente de contêiner do Amazon ECS extrai a versão mais recente da imagem especificada.
- No entanto, as atualizações subsequentes feitas em um repositório de imagens não são propagadas para tarefas já em execução.
- As imagens em registros privados são suportadas. Para obter mais informações, consulte Uso de imagens de contêiner que não são da AWS no Amazon ECS.
- As imagens nos repositórios do Amazon ECR podem ser especificadas usando a convenção de nomenclatura completa `registry/repository:tag` ou `registry/repository@digest` (por exemplo, *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`* ou *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`@`*`sha256:94afd1f2e64d908bc90dbca0035a5b567EXAMPLE`*).
- As imagens em repositórios oficiais no Docker Hub usam um único nome (por exemplo, `ubuntu` ou `mongo`).
- Imagens em outros repositórios Docker Hub são qualificadas com um nome de organização (por exemplo, `amazon/amazon-ecs-agent`).
- Imagens em outros repositórios online também são qualificadas por um nome de domínio (por exemplo, `quay.io/assemblyline/ubuntu`).

`versionConsistency`

Tipo: String

Valores válidos: `enabled|disabled`

Obrigatório: Não

Especifica se o Amazon ECS resolverá a tag de imagem do contêiner fornecida na definição do contêiner em um resumo de imagem. Por padrão, esse comportamento é `enabled`. Se você definir o valor de um contêiner como `disabled`, o Amazon ECS não resolverá a tag da imagem do contêiner como um resumo e usará o URI da imagem original especificado na definição do contêiner para implantação. Para obter mais informações sobre resolução de imagem do contêiner, consulte Resolução da imagem do contêiner.

Memória

`memory`

Tipo: inteiro

Obrigatório: Não

A quantidade (em MiB) de memória a ser apresentada ao contêiner. Caso tente exceder a memória especificada aqui, o contêiner será excluído. A quantidade total de memória reservada para todos os contêineres dentro da tarefa deve ser menor que o valor da tarefa `memory`, se estiver especificado. Esse parâmetro é mapeado para `Memory` no comando create-container do docker na opção `--memory` de execução do docker.

O daemon para Docker 20.10.0 ou versão posterior reserva no mínimo 6 MiB de memória para um contêiner. Por isso, não especifique menos de 6 MiB de memória para os seus contêineres.

O daemon para Docker 19.03.13-ce ou versão anterior reserva no mínimo 4 MiB de memória para um contêiner. Por isso, não especifique menos de 4 MiB de memória para os seus contêineres.

> **Note**
>
> Se você deseja maximizar a utilização de recursos fornecendo às tarefas o máximo de memória possível para um determinado tipo de instância, consulte Reserva de memória da instância de contêiner do Linux no Amazon ECS.

`memoryReservation`

Tipo: inteiro

Obrigatório: Não

O limite flexível (em MiB) de memória a ser reservado para o contêiner. Quando a memória do sistema estiver em contenção, o Docker tentará manter a memória do contêiner dentro desse limite flexível. Entrentanto, seu contêiner pode usar mais memória quando necessário. O contêiner pode consumir até o limite rígido especificado pelo parâmetro `memory` (se aplicável), ou toda a memória disponível na instância de contêiner, o que ocorrer primeiro. Esse parâmetro é mapeado para `MemoryReservation` no comando create-container do docker na opção `--memory-reservation` de execução do docker.

Se um valor de memória no nível de tarefa não for especificado, será necessário especificar um número inteiro diferente de zero para um ou ambos `memory` ou `memoryReservation` em uma definição de contêiner. Caso você especifique ambos, `memory` deve ser maior que `memoryReservation`. Caso você especifique `memoryReservation`, o valor é subtraído dos recursos de memória disponíveis para a instância de contêiner na qual o contêiner está colocado. Caso contrário, o valor de `memory` é usado.

Por exemplo, suponha que seu contêiner normalmente use 128 MiB de memória, mas às vezes chegue a 256 MiB de memória por períodos curtos. É possível definir um `memoryReservation` de 128 MiB e um limite rígido de `memory` de 300 MiB. Essa configuração permite que contêiner reservasse apenas 128 MiB de memória dos recursos restantes na instância de contêiner. Ao mesmo tempo, essa configuração também possibilita que o contêiner utilize mais recursos de memória quando necessário.

> **Note**
>
> Este parâmetro não é compatível com contêineres do Windows.

O daemon para Docker 20.10.0 ou versão posterior reserva no mínimo 6 MiB de memória para um contêiner. Por isso, não especifique menos de 6 MiB de memória para os seus contêineres.

O daemon para Docker 19.03.13-ce ou versão anterior reserva no mínimo 4 MiB de memória para um contêiner. Por isso, não especifique menos de 4 MiB de memória para os seus contêineres.

> **Note**
>
> Se você deseja maximizar a utilização de recursos fornecendo às tarefas o máximo de memória possível para um determinado tipo de instância, consulte Reserva de memória da instância de contêiner do Linux no Amazon ECS.

Mapeamentos de porta

`portMappings`

Tipo: Matriz de objeto

Obrigatório: Não

Os mapeamentos de portas expõem as portas de rede do seu contêiner ao mundo externo. Isso permite que os clientes acessem sua aplicação. Também é usado para comunicação entre contêineres dentro da mesma tarefa.

Para definições de tarefa que usam o modo de rede `awsvpc`, apenas especifique `containerPort`. A `hostPort` é sempre ignorada e a porta do contêiner é automaticamente mapeada para uma porta aleatória de alto número no host.

Os mapeamentos de porta no Windows usam o endereço de gateway `NetNAT`, e não o `localhost`. Não há loopback para mapeamentos de porta no Windows, de modo que você não pode acessar a porta mapeada de um contêiner no próprio host.

A maioria dos campos desse parâmetro (incluindo `containerPort`, `hostPort`, `protocol`) é mapeada para `PortBindings` no comando create-container do docker e a opção `--publish` de execução do docker. Caso o modo de rede de uma definição de tarefa seja definido como `host`, as portas host devem ser indefinidas ou corresponder à porta de contêiner no mapeamento de porta.

> **Note**
>
> Quando a tarefa obter o status `RUNNING`, as atribuições automáticas e manuais da porta do contêiner e do host estarão visíveis nos seguintes locais:
>
> - Console: a seção Network Bindings (Associações de rede) de uma descrição de contêiner para uma tarefa selecionada.
> - AWS CLI: a seção `networkBindings` do resultado do comando describe-tasks.
> - API: a resposta `DescribeTasks`.
> - Metadados: o endpoint de metadados da tarefa.

`appProtocol`

Tipo: String

Obrigatório: Não

O protocolo da aplicação usado no mapeamento da porta. Esse parâmetro só se aplica ao Service Connect. Recomendamos que você defina esse parâmetro para que seja consistente com o protocolo usado por sua aplicação. Se você definir esse parâmetro, o Amazon ECS

adicionará o tratamento de conexão específico do protocolo ao proxy do Service Connect. Se você definir esse parâmetro, o Amazon ECS adicionará telemetria específica de protocolo no console do Amazon ECS e no CloudWatch.

Se você não definir um valor para esse parâmetro, o TCP será usado. No entanto, o Amazon ECS não adiciona telemetria específica de protocolo para TCP.

Para obter mais informações, consulte the section called “ Service Connect”.

Valores de protocolo válidos: `"http" | "http2" | "grpc"`

`containerPort`

Tipo: inteiro

Exigido: Sim, quando `portMappings` são usados

O número da porta no contêiner vinculado à porta host atribuída automaticamente ou especificada pelo usuário.

Para as tarefas que usam o modo de rede `awsvpc`, você usa `containerPort` para especificar as portas expostas.

Para contêineres Windows no Fargate, não é possível usar a porta 3150 para `containerPort`. Isso acontece porque ela está reservada.

`containerPortRange`

Tipo: String

Obrigatório: Não

O intervalo de números de porta no contêiner vinculado ao intervalo de portas host mapeado dinamicamente.

Você só pode definir esse parâmetro usando a API `register-task-definition`. A opção está disponível no parâmetro `portMappings`. Para obter mais informações, consulte register-task-definition, na Referência da AWS Command Line Interface.

As regras a seguir se aplicam quando você especifica um `containerPortRange`:

- Você deve usar o modo de rede `awsvpc`.
- Esse parâmetro está disponível para os sistemas operacionais Windows e Linux.

- A instância de contêiner deve ter pelo menos a versão 1.67.0 do agente do contêiner e pelo menos a versão 1.67.0-1 do pacote `ecs-init`
- É possível especificar no máximo 100 intervalos de portas para cada contêiner.
- Você não especifica um `hostPortRange`. O valor do `hostPortRange` é definido da seguinte forma:
  - Para contêineres em uma tarefa com o modo de rede `awsvpc`, o `hostPort` é definido com o mesmo valor que `containerPort`. Essa é uma estratégia de mapeamento estático.
- Os valores válidos de `containerPortRange` estão entre 1 e 65535.
- Uma porta só pode ser incluída em um mapeamento de portas para cada contêiner.
- Não é possível especificar intervalos de portas sobrepostos.
- A primeira porta no intervalo deve ser menor que a última porta no intervalo.
- O Docker recomenda que você desative o docker-proxy no arquivo de configuração do daemon do Docker quando tiver um grande número de portas.

  Para obter mais informações, consulte Issue #11185 no GitHub.

  Para obter informações sobre como desativar o docker-proxy no arquivo de configuração do daemon do Docker, consulte Daemon do Docker no Guia do desenvolvedor do Amazon ECS.

É possível chamar DescribeTasks para visualizar o `hostPortRange`, que são as portas do host vinculadas às portas do contêiner.

Os intervalos de portas não estão incluídos nos eventos de tarefas do Amazon ECS que são enviados ao EventBridge. Para obter mais informações, consulte the section called "Automatização de respostas a erros do Amazon ECS usando o EventBridge".

`hostPortRange`

Tipo: Sequência

Obrigatório: Não

O intervalo de números de portas no host que é usado com a vinculação de rede. Isso é atribuído pelo Docker e fornecido pelo agente do Amazon ECS.

`hostPort`

Tipo: inteiro

Obrigatório: Não

O número da porta na instância de contêiner a ser reservado para o contêiner.

A `hostPort` pode ser deixada em branco ou ter um valor igual a `containerPort`.

O intervalo de portas temporárias padrão para a versão 1.6.0 e posterior do Docker é listado na instância em `/proc/sys/net/ipv4/ip_local_port_range`. Se esse parâmetro do kernel não estiver disponível, será usado o intervalo de portas temporárias padrão de `49153–65535`. Não tente especificar uma porta de host no intervalo de portas transitório. Isso ocorre porque essas portas são reservadas para atribuição automática. Em geral, as portas abaixo de `32768` estão fora do intervalo de portas temporárias.

As portas reservadas padrão são `22` para SSH, as portas do Docker `2375` e`2376` e as portas do agente de contêiner do Amazon ECS `51678-51680`. Qualquer porta de host que tenha sido especificada pelo usuário anteriormente para uma tarefa em execução também é reservada enquanto a tarefa está em execução. Depois que uma tarefa é interrompida, a porta do host é liberada. As portas reservadas atuais são exibidas nos `remainingResources` de describe-container-instances de saída. Uma instância de contêiner pode ter até 100 portas reservadas por vez, incluindo as portas reservadas padrão. As portas atribuídas automaticamente não contam em relação à cota de 100 portas reservadas.

`name`

Tipo: String

Obrigatório: não, obrigatório para que o Service Connect e o VPC Lattice sejam configurados em um serviço

Nome que é usado para o mapeamento da porta. Esse parâmetro só se aplica ao Service Connect e ao VPC Lattice. Esse parâmetro é o nome que você usa na configuração do Service Connect e do VPC Lattice de um serviço.

Para obter mais informações, consulte Uso do Service Connect para conectar serviços do Amazon ECS com nomes abreviados.

No exemplo a seguir, ambos os campos obrigatórios para o Service Connect e o VPC Lattice são usados.

```
"portMappings": [
    {
        "name": string,
        "containerPort": integer
    }
]
```

`protocol`

Tipo: String

Obrigatório: Não

O protocolo usado no mapeamento da porta. Os valores válidos são `tcp` e `udp`. O padrão é `tcp`.

> **Important**
>
> Só `tcp` é compatível com o Service Connect. Lembre-se de que `tcp` estará implícito se esse campo não estiver definido.

Caso você esteja especificando uma porta host, use a seguinte sintaxe:

```
"portMappings": [
    {
        "containerPort": integer,
        "hostPort": integer
    }
    ...
]
```

Caso você queira uma porta host atribuída automaticamente, use a seguinte sintaxe:

```
"portMappings": [
    {
        "containerPort": integer
    }
    ...
]
```

Credenciais de repositório privado

`repositoryCredentials`

Tipo: objeto RepositoryCredentials

Obrigatório: Não

As credenciais do repositório para autenticação de registro privado.

Para obter mais informações, consulte Uso de imagens de contêiner que não são da AWS no Amazon ECS.

`credentialsParameter`

Tipo: String

Exigido: Sim, quando `repositoryCredentials` são usados

O nome de recurso da Amazon (ARN) do segredo que contém as credenciais do repositório privado.

Para obter mais informações, consulte Uso de imagens de contêiner que não são da AWS no Amazon ECS.

> **Note**
>
> Quando você usa a API do Amazon ECS, a AWS CLI ou os AWS SDKs, se o segredo existir na mesma região da tarefa que estiver sendo executada, será possível usar o ARN completo ou o nome do segredo. Ao usar o AWS Management Console, é preciso especificar o ARN completo do segredo.

Veja a seguir um trecho de uma definição de tarefa que mostra os parâmetros necessários.

```
"containerDefinitions": [
    {
        "image": "private-repo/private-image",
        "repositoryCredentials": {
            "credentialsParameter":
 "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
```

```
        }
    }
]
```

Parâmetros avançados de definição de contêiner

Os parâmetros avançados de definição de contêiner a seguir fornecem recursos estendidos ao comando run do docker usado para iniciar contêineres nas instâncias de contêiner do Amazon ECS.

Tópicos

- Política de reinicialização
- Verificação de integridade
- Environment
- Configurações de rede
- Armazenamento e registro
- Segurança
- Limites de recurso
- Rótulos do Docker

Política de reinicialização

`restartPolicy`

A política de reinicialização do contêiner e os parâmetros de configuração associados. Quando você configura uma política de reinicialização para um contêiner, o Amazon ECS pode reiniciar o contêiner sem precisar substituir a tarefa. Para obter mais informações, consulte Reiniciar contêineres individuais em tarefas do Amazon ECS com políticas de reinicialização de contêineres.

`enabled`

Tipo: booliano

Obrigatório: Sim

Especifica se uma política de reinicialização está habilitada para o contêiner.

`ignoredExitCodes`

Tipo: matriz de inteiros

Obrigatório: Não

Uma lista de códigos de saída que o Amazon ECS ignorará e não tentará reiniciar. Você pode especificar um máximo de 50 códigos de saída de contêiner. Por padrão, o Amazon ECS não ignora nenhum código de saída.

`restartAttemptPeriod`

Tipo: inteiro

Obrigatório: Não

A quantidade de tempo (em segundos) que o contêiner deve ser executado antes de uma tentativa de reinicialização. Um contêiner só pode ser reiniciado uma vez a cada `restartAttemptPeriod` segundos. Se um contêiner não puder ser executado nesse período e sair mais cedo, ele não será reiniciado. Você pode especificar um `restartAttemptPeriod` mínimo de 60 segundos e um `restartAttemptPeriod` máximo de 1800 segundos. Por padrão, um contêiner deve ser executado por 300 segundos antes de ser reiniciado.

Verificação de integridade

`healthCheck`

O comando da verificação de integridade de contêiner e os parâmetros de configuração associados para o contêiner. Para obter mais informações, consulte Como determinar a integridade das tarefas do Amazon ECS usando verificações de integridade de contêineres.

`command`

Uma matriz de strings que representa o comando executado pelo contêiner para determinar se ela está íntegra. A matriz de strings pode começar com `CMD` para executar os argumentos de comando diretamente ou `CMD-SHELL` para executar o comando com o shell padrão do contêiner. Se nenhuma for especificado, `CMD` será usada.

Ao registrar uma definição de tarefa no AWS Management Console, use uma lista de comandos separados por vírgulas. Esses comandos serão convertidos em uma string após a

criação da definição de tarefa. Veja a seguir um exemplo de entrada para uma verificação de integridade.

```
CMD-SHELL, curl -f http://localhost/ || exit 1
```

Ao registrar uma definição de tarefa usando o painel JSON do AWS Management Console, a AWS CLI ou as APIs, delimite a lista de comandos por colchetes. Veja a seguir um exemplo de entrada para uma verificação de integridade.

```
[ "CMD-SHELL", "curl -f http://localhost/ || exit 1" ]
```

Um código de saída 0, sem saída de `stderr`, indica sucesso, e um código de saída diferente de zero indica falha.

`interval`

O período em segundos entre cada verificação de integridade. É possível especificar entre 5 e 300 segundos. O valor padrão de é 30 segundos.

`timeout`

O período de espera (em segundos) para que uma verificação de integridade seja bem-sucedida antes de ser considerada uma falha. É possível especificar entre 2 e 60 segundos. O valor de padrão é de 5 segundos.

`retries`

O número de novas tentativas de uma verificação de integridade com falha até o contêiner ser considerado não íntegro. É possível especificar entre 1 e 10 novas tentativas. O valor padrão é três novas tentativas.

`startPeriod`

O período de carência opcional para que os contêineres possam inicializar antes de as verificações de integridade com falha serem contabilizadas no número máximo de novas tentativas. Você pode especificar um valor entre 0 e 300 segundos. Por padrão, o `startPeriod` está desabilitado.

Se uma verificação de integridade for bem-sucedida dentro do `startPeriod`, então o contêiner é considerado saudável e quaisquer falhas subsequentes contam para o número máximo de novas tentativas.

Environment

`cpu`

Tipo: inteiro

Obrigatório: Não

O número de unidades de `cpu` que o agente de contêiner do Amazon ECS reservará para o contêiner. No Linux, esse parâmetro é mapeado como `CpuShares` na seção Criar um contêiner.

Este campo é opcional para tarefas que usam o tipo de inicialização do Fargate. A quantidade total de CPU reservada para todos os contêineres dentro da tarefa deve ser menor que o valor `cpu` de nível da tarefa.

Os contêineres de Linux compartilham unidades de CPU não alocadas com outros contêineres na instância de contêiner que tem a mesma proporção que a respectiva quantidade alocada. Por exemplo, suponha que você execute uma tarefa de contêiner único em um tipo de instância de núcleo único com 512 unidades de CPU especificadas para esse contêiner. Além disso, essa tarefa é a única em execução na instância de contêiner. Nesse exemplo, o contêiner pode utilizar o compartilhamento total de 1.024 unidades de CPU a qualquer momento. Porém, suponha que você tenha iniciado outra cópia da mesma tarefa nessa instância de contêiner. Cada tarefa tem garantido um mínimo de 512 unidades de CPU quando necessário. Da mesma forma, se o outro contêiner não estiver usando a CPU restante, cada contêiner poderá flutuar para o uso maior da CPU. Porém, se ambas as tarefas estiverem 100% ativas o tempo todo, elas ficarão limitadas a 512 unidades de CPU.

Nas instâncias de contêiner de Linux, o daemon do Docker na instância de contêiner usa o valor de CPU para calcular as proporções de compartilhamento de CPU para contêineres em execução. O valor de compartilhamento de CPU mínimo válido permitido pelo kernel do Linux é 2 e o valor de compartilhamento de CPU máximo válido permitido pelo kernel do Linux é 262144. No entanto, o parâmetro de CPU não é necessário, e é possível usar valores de CPU abaixo de dois e superior a 262144 em suas definições de contêiner. Para valores de CPU abaixo de dois (inclusive nulo) e acima de 262144, o comportamento varia com base na versão do agente de contêiner do Amazon ECS:

Nas instâncias de contêiner de Windows, a cota de CPU é imposta como uma cota absoluta. Os contêineres de Windows só têm acesso à quantidade de CPU especificada na definição de tarefa. Um valor de CPU nulo ou zero é transmitido para Docker como `0`. Em seguida, Windows interpreta esse valor como 1% de uma CPU.

Para obter exemplos adicionais, consulte Como o Amazon ECS gerencia recursos de CPU e memória.

`gpu`

Esse parâmetro não é compatível com contêineres hospedados no Fargate.

Tipo: objeto ResourceRequirement

Obrigatório: Não

O número de GPUs físicas que o agente de contêiner do Amazon ECS reservará para o contêiner. O número de GPUs reservadas para todos os contêineres em uma tarefa não deve exceder o número de GPUs disponíveis na instância de contêiner na qual a tarefa é executada. Para obter mais informações, consulte Definições de tarefa do Amazon ECS para workloads de GPU.

`Elastic Inference accelerator`

Esse parâmetro não é compatível com contêineres hospedados no Fargate.

Tipo: objeto ResourceRequirement

Obrigatório: Não

Para o tipo `InferenceAccelerator`, o `value` corresponde ao `deviceName` para um `InferenceAccelerator` especificado em uma definição de tarefa. Para obter mais informações, consulte the section called “Nome do acelerador de inferência elástica”.

`essential`

Tipo: booliano

Obrigatório: Não

Suponha que o parâmetro `essential` de um contêiner esteja marcado como `true`, e que esse contêiner falhe ou seja interrompido por algum motivo. Em seguida, todos os outros contêineres que fazem parte da tarefa são interrompidos. Caso o parâmetro `essential` de um contêiner esteja marcado como `false`, a falha não afeta o restante dos contêineres em uma tarefa. Caso esse parâmetro seja omitido, um contêiner pressupõe-se que um contêiner seja essencial.

Todas as tarefas devem ter pelo menos um contêiner essencial. Suponha que você tenha uma aplicação composta por vários contêineres. Em seguida, agrupe os contêineres usados para uma

finalidade em comum em componentes e separe os componentes diferentes em várias definições de tarefa. Para obter mais informações, consulte Projetar sua aplicação para o Amazon ECS.

```
"essential": true|false
```

`entryPoint`

> **Important**
>
> As versões anteriores do agente de contêiner do Amazon ECS não lidam corretamente com parâmetros `entryPoint`. Caso você enfrente problemas para usar `entryPoint`, atualize o agente de contêiner ou informe os comandos e os argumentos como itens de matriz `command` em vez disso.

Tipo: Matriz de strings

Obrigatório: Não

O ponto de entrada passado para o contêiner.

```
"entryPoint": ["string", ...]
```

`command`

Tipo: Matriz de strings

Obrigatório: Não

O comando que é passado para o contêiner. Esse parâmetro é mapeado para `Cmd` no comando create-container e o parâmetro `COMMAND` de execução do docker. Se houver vários argumentos, certifique-se de que cada um seja uma string separada na matriz.

```
"command": ["string", ...]
```

`workingDirectory`

Tipo: String

Obrigatório: Não

O diretório de trabalho no qual executar comandos no contêiner. Esse parâmetro é mapeado para `WorkingDir` na seção Criar um Contêiner da API remota do Docker e a opção `--workdir` para docker run.

```
"workingDirectory": "string"
```

`environmentFiles`

Isso não está disponível para contêineres do Windows no Fargate.

Tipo: Matriz de objeto

Obrigatório: Não

Uma lista de arquivos que contêm as variáveis de ambiente a serem passadas para um contêiner. Esse parâmetro é mapeado para a opção `--env-file` para o comando run do docker.

É possível especificar até 10 arquivos de ambiente. O arquivo deve ter uma extensão `.env`. Cada linha em um arquivo de ambiente contém uma variável de ambiente no formato `VARIABLE=VALUE`. As linhas que começam com # são tratadas como comentários e são ignoradas.

Se houver variáveis de ambiente individuais especificadas na definição de contêiner, elas terão precedência sobre as variáveis contidas em um arquivo de ambiente. Se forem especificados vários arquivos de ambiente que contenham a mesma variável, elas serão processadas de cima para baixo. Recomendamos usar nomes de variáveis exclusivos. Para obter mais informações, consulte Transferência de uma variável de ambiente individual para um contêiner do Amazon ECS.

`value`

Tipo: string

Obrigatório: Sim

O nome de recurso da Amazon (ARN) do objeto do Amazon S3 que contém o arquivo de variáveis de ambiente.

`type`

Tipo: string

Obrigatório: Sim

Tipo de arquivo a ser usado O único valor aceito é `s3`.

`environment`

Tipo: Matriz de objeto

Obrigatório: Não

As variáveis de ambiente a serem passadas para um contêiner. Esse parâmetro é mapeado para `Env` no comando create-container do docker na opção `--env` para o comando run do docker.

> **Important**
>
> Não recomendamos o uso de variáveis de ambiente de texto simples para informações confidenciais, tais como dados de credenciais.

`name`

Tipo: String

Exigido: sim, quando `environment` for usado

O nome da variável de ambiente.

`value`

Tipo: String

Exigido: sim, quando `environment` for usado

O valor da variável de ambiente.

```
"environment" : [
    { "name" : "string", "value" : "string" },
    { "name" : "string", "value" : "string" }
]
```

`secrets`

Tipo: Matriz de objeto

Obrigatório: Não

Um objeto que representa o segredo a ser exposto ao seu contêiner. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.

`name`

Tipo: string

Obrigatório: Sim

O valor a ser definido como a variável de ambiente no contêiner.

`valueFrom`

Tipo: string

Obrigatório: Sim

O segredo a ser exposto ao contêiner. Os valores com suporte são o nome do recurso da Amazon (ARN) completo do segredo do AWS Secrets Manager ou o ARN completo do parâmetro no Parameter Store do AWS Systems Manager.

> **Note**
>
> Se o parâmetro do Systems Manager Parameter Store ou do Secrets Manager existir na mesma Região da AWS que a tarefa sendo iniciada, você poderá usar o ARN completo ou o nome do segredo. Se o parâmetro existir em uma Região diferente, o ARN completo deverá ser especificado.

```
"secrets": [
    {
        "name": "environment_variable_name",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }
]
```

Configurações de rede

`disableNetworking`

Esse parâmetro não é compatível com tarefas que são executadas no Fargate.

Tipo: booliano

Obrigatório: Não

Quando esse parâmetro é verdadeiro, a rede é desativada dentro do contêiner.

O padrão é `false`.

```
"disableNetworking": true|false
```

`links`

Esse parâmetro não é compatível com tarefas que usam o modo de rede `awsvpc`.

Tipo: Matriz de strings

Obrigatório: Não

O parâmetro `link` permite que os contêineres se comuniquem entre si sem a necessidade de mapeamentos de porta. Só há suporte para esse parâmetro se o modo de rede de uma definição de tarefa estiver definido como `bridge`. O construto `name:internalName` é igual ao `name:alias` em links do Docker. São permitidos até 255 letras (caixa alta e baixa), números, hífens e sublinhados.

> **Important**
>
> Os contêineres colocados na mesma instância de contêiner podem se comunicar entre si sem exigir links ou mapeamentos de porta de host. O isolamento de rede em uma instância de contêiner é controlado por grupos de segurança e configurações de VPC.

```
"links": ["name:internalName", ...]
```

`hostname`

Tipo: String

Obrigatório: Não

O nome de host a ser usado para o contêiner. Esse parâmetro é mapeado para `Hostname` em create-container do docker na opção `--hostname` de execução do docker.

> **Note**
>
> Se você estiver usando o modo de rede `awsvpc`, o parâmetro `hostname` não terá suporte.

```
"hostname": "string"
```

`dnsServers`

Esse parâmetro não é compatível com tarefas sendo executadas no Fargate.

Tipo: Matriz de strings

Obrigatório: Não

Uma lista de servidores DNS apresentados ao contêiner.

```
"dnsServers": ["string", ...]
```

`extraHosts`

Esse parâmetro não é compatível com tarefas que usam o modo de rede `awsvpc`.

Tipo: Matriz de objeto

Obrigatório: Não

Uma lista de nomes de host e mapeamentos de endereço IP a ser acrescentada ao arquivo `/etc/hosts` no contêiner.

Esse parâmetro é mapeado para `ExtraHosts` no comando create-container do docker na opção `--add-host` de execução do docker.

```
"extraHosts": [
      {
        "hostname": "string",
        "ipAddress": "string"
      }
      ...
```

```
    ]
```

`hostname`

Tipo: String

Exigido: Sim, quando `extraHosts` são usados

O nome do host a ser usado na entrada `/etc/hosts`.

`ipAddress`

Tipo: String

Exigido: Sim, quando `extraHosts` são usados

O endereço IP a ser usado na entrada `/etc/hosts`.

Armazenamento e registro

`readonlyRootFilesystem`

Tipo: booliano

Obrigatório: Não

Quando esse parâmetro é verdadeiro, o contêiner recebe acesso somente leitura ao sistema de arquivos raiz. Esse parâmetro é mapeado para `ReadonlyRootfs` no comando create-container do docker na opção `--read-only` de execução do docker.

> **Note**
>
> Este parâmetro não é compatível com contêineres do Windows.

O padrão é `false`.

```
"readonlyRootFilesystem": true|false
```

`mountPoints`

Tipo: Matriz de objeto

Obrigatório: Não

Os pontos de montagem dos volumes de dados no contêiner. Esse parâmetro é mapeado em `Volumes` na API create-container do Docker na opção `--volume` de execução do docker.

Os contêineres do Windows podem montar diretórios inteiros na mesma unidade como `$env:ProgramData`. Os contêineres do Windows não podem montar diretórios em uma unidade diferente, e os pontos de montagem não podem ser usados entre unidades. Você deve especificar pontos de montagem para anexar um volume do Amazon EBS diretamente a uma tarefa do Amazon ECS.

`sourceVolume`

Tipo: String

Exigido: Sim, quando `mountPoints` são usados

O nome do volume a ser montado.

`containerPath`

Tipo: String

Exigido: Sim, quando `mountPoints` são usados

O caminho no contêiner onde o volume será montado.

`readOnly`

Tipo: booliano

Obrigatório: Não

Caso o valor seja `true`, o contêiner tem acesso somente leitura ao volume. Caso esse valor seja `false`, o contêiner pode gravar no volume. O valor padrão é `false`.

Para tarefas executadas em instâncias do EC2 que executam o sistema operacional Windows, deixe o valor como o padrão de `false`.

`volumesFrom`

Tipo: Matriz de objeto

Obrigatório: Não

Volumes de dados a serem montados de outro contêiner. Esse parâmetro é mapeado para `VolumesFrom` no comando create-container do docker na opção `--volumes-from` de execução do docker.

`sourceContainer`

Tipo: String

Exigido: sim, quando `volumesFrom` for usado

O nome do contêiner com base no qual montar volumes.

`readOnly`

Tipo: booliano

Obrigatório: Não

Caso o valor seja `true`, o contêiner tem acesso somente leitura ao volume. Caso esse valor seja `false`, o contêiner pode gravar no volume. O valor padrão é `false`.

```
"volumesFrom": [
                {
                  "sourceContainer": "string",
                  "readOnly": true|false
                }
              ]
```

`logConfiguration`

Tipo: objeto LogConfiguration

Obrigatório: Não

A especificação de configuração do log para o contêiner.

Para obter definições de tarefa de exemplo que usam uma configuração de log, consulte Exemplos de definições de tarefa do Amazon ECS.

Esse parâmetro é mapeado para `LogConfig` no comando create-container do docker na opção `--log-driver` de execução do docker. Por padrão, os contêineres usam o mesmo driver de registro em log que o daemon do Docker usa. No entanto, o contêiner pode usar um driver

de registro em log diferente do daemon do Docker especificando um driver de log com esse parâmetro na definição de contêiner. Para usar um driver de registro em log diferente para um contêiner, o sistema de log deve ser configurado corretamente na instância de contêiner (ou em um servidor de log diferente para opções de registro em log remotas).

Considere o seguinte ao especificar uma configuração de log para seus contêineres:

- O Amazon ECS oferece suporte a um subconjunto dos drivers de registro em log disponíveis para o daemon Docker. Os drivers de log adicionais podem estar disponíveis em versões futuras do agente de contêiner do Amazon ECS.
- Esse parâmetro exige a versão 1.18 ou posterior da API remota do Docker na sua instância de contêiner.
- Qualquer software adicional deve ser instalado fora da tarefa. Por exemplo, os agregadores de saída Fluentd ou um host remoto executando o Logstash para o qual enviar logs Gelf.

```
"logConfiguration": {
      "logDriver": "awslogs","fluentd","gelf","json-
file","journald","splunk","syslog","awsfirelens",
      "options": {"string": "string"
        ...},
 "secretOptions": [{
  "name": "string",
  "valueFrom": "string"
 }]
}
```

`logDriver`

Tipo: String

Valores válidos: `"awslogs","fluentd","gelf","json-file","journald","splunk","syslog","awsfirelens"`

Exigido: sim, quando `logConfiguration` for usado

O driver de log a ser usado para o contêiner. Por padrão, os valores válidos listados anteriormente são drivers de log com os quais o agente de contêiner do Amazon ECS pode se comunicar.

Os drivers de log compatíveis são `awslogs`, `splunk` e `awsfirelens`.

Para obter mais informações sobre como usar o driver de log `awslogs` nas definições de tarefa para enviar os logs de contêiner ao CloudWatch Logs, consulte Envio de logs do Amazon ECS para o CloudWatch .

Para obter mais informações sobre como usar o driver de log `awsfirelens`, consulte Roteamento de logs personalizados.

> **Note**
>
> Se tiver um driver personalizado que não esteja listado acima, será possível bifurcar o projeto do agente de contêiner do Amazon ECS que está disponível no GitHub e personalizá-lo para funcionar com esse driver. Incentivamos você a enviar solicitações pull para alterações que você gostaria de ter incluído. Porém, no momento, não oferecemos suporte à execução de cópias modificadas desse software.

Este parâmetro requer a versão 1.18 da Docker Remote API ou posterior em sua instância de contêiner.

`options`

Tipo: mapa de string para string

Obrigatório: Não

O mapa chave/valor de opções de configuração a serem enviadas para o driver de log.

As opções que você pode especificar dependem do driver de log. Algumas das opções que você pode especificar ao usar o roteador `awslogs` para rotear os logs para o Amazon CloudWatch incluem o seguinte:

`awslogs-create-group`

Obrigatório: Não

Especifique se você deseja que o grupo de logs seja criado automaticamente. Se essa opção não for especificada, o padrão será `false`.

> **Note**
>
> Sua política do IAM deve incluir a permissão `logs:CreateLogGroup` antes que você tente usar `awslogs-create-group`.

`awslogs-region`

Obrigatório: Sim

Especifique a Região da AWS para a qual o driver de log do `awslogs` deve enviar seus logs do Docker. É possível optar por enviar todos os logs de clusters em regiões distintas para uma única região no CloudWatch Logs. Isso tem o objetivo de fazer com que todos fiquem visíveis em um único local. Caso contrário, você pode separá-los por região para mais granularidade. Certifique-se de que o grupo de logs especificado exista na região especificada com essa opção.

`awslogs-group`

Obrigatório: Sim

Certifique-se de especificar um grupo de logs para o qual o driver de log do `awslogs` envia seus fluxos de logs.

`awslogs-stream-prefix`

Obrigatório: Sim

Use a opção `awslogs-stream-prefix` para associar um fluxo de log ao prefixo especificado, ao nome do contêiner e ao ID da tarefa do Amazon ECS à qual o contêiner pertence. Caso você especifique um prefixo com essa opção, o fluxo de log utiliza o seguinte formato.

```
prefix-name/container-name/ecs-task-id
```

Se você não especificar um prefixo com essa opção, o fluxo de log será nomeado depois que o ID do contêiner for atribuído pelo daemon do Docker na instância de contêiner. Como é difícil rastrear logs até o contêiner que os enviou apenas com o ID do contêiner do Docker (disponível apenas na instância de contêiner), recomendamos especificar um prefixo com essa opção.

Para serviços do Amazon ECS, é possível utilizar o nome do serviço como prefixo. Ao fazer isso, será possível rastrear fluxos de logs até o serviço ao qual o contêiner pertence, o nome do contêiner que os enviou e o ID da tarefa à qual o contêiner pertence.

Você deve especificar um stream-prefix para seus logs para fazer com que eles apareçam no painel Logs quando o console do Amazon ECS for usado.

`awslogs-datetime-format`

Obrigatório: Não

Essa opção define um padrão de início de várias linhas no formato `strftime` em Python. Uma mensagem de log é formada por uma linha em conformidade com o padrão e as linhas seguintes que não correspondem ao padrão. A linha em conformidade é o delimitador entre as mensagens de log.

Um exemplo de um caso de uso para esse formato é a análise da saída, como um despejo de pilha, que poderia ser registrado em várias entradas. O padrão correto permite que ele seja capturado em uma única entrada.

Para obter mais informações, consulte awslogs-datetime-format.

Você não pode configurar as opções `awslogs-datetime-format` e `awslogs-multiline-pattern`.

> **Note**
>
> O registro em várias linhas executa a análise da expressão regular e a correspondência de todas as mensagens de log. Isso pode ter um impacto negativo na performance do registro em log.

`awslogs-multiline-pattern`

Obrigatório: Não

Essa opção define um padrão inicial de várias linhas que usa uma expressão regular. Uma mensagem de log é formada por uma linha em conformidade com o padrão e as linhas seguintes que não correspondem ao padrão. A linha em conformidade é o delimitador entre as mensagens de log.

Para obter mais informações, consulte awslogs-multiline-pattern.

Essa opção será ignorada se `awslogs-datetime-format` também estiver configurado.

Você não pode configurar as opções `awslogs-datetime-format` e `awslogs-multiline-pattern`.

> **Note**
>
> O registro em várias linhas executa a análise da expressão regular e a correspondência de todas as mensagens de log. Isso pode ter um impacto negativo na performance do registro em log.

As opções a seguir se aplicam a todos os drivers de logs compatíveis.

`mode`

Obrigatório: Não

Valores válidos: `non-blocking` | `blocking`

Essa opção define o modo de entrega de mensagens de log do contêiner para o driver de logs especificado usando o `logDriver`. O modo de entrega escolhido afeta a disponibilidade da aplicação quando o fluxo de logs do contêiner é interrompido.

Se você usar o modo `blocking` e o fluxo de logs for interrompido, as chamadas do código do contêiner para gravar nos fluxos `stdout` e `stderr` serão bloqueadas. Como resultado, o encadeamento de logs da aplicação será bloqueado. Isso pode fazer com que a aplicação pare de responder e causar uma falha na verificação de integridade do contêiner.

Se você usar o modo `non-blocking`, os registros do contêiner serão armazenados em um buffer intermediário na memória configurado com a opção `max-buffer-size`. Isso evita que a aplicação pare de responder quando não for possível enviar os logs. Recomendamos usar esse modo se você quiser garantir a disponibilidade do serviço e aceitar alguma perda de logs. Para obter mais informações, consulte Preventing log loss with non-blocking mode in the `awslogs` container log driver.

Você pode definir um `mode` padrão para todos os contêineres em uma determinada Região da AWS usando a configuração de conta `defaultLogDriverMode`. Se você não especificar a opção de `mode` na `logConfiguration` nem definir a configuração de conta, o Amazon ECS usará o modo `non-blocking` como padrão. Para obter mais informações sobre a configuração de conta, consulte Modo de driver de logs padrão.

Quando o modo `non-blocking` é usado, a opção de log `max-buffer-size` controla o tamanho do buffer de anel usado para armazenamento de mensagens intermediárias.

Certifique-se de especificar um tamanho de buffer adequado com base na sua aplicação. A quantidade total de memória alocada no nível da tarefa deve ser maior que a quantidade de memória alocada para todos os contêineres, mais o limite de buffer da memória.

> **Note**
>
> Em 25 de junho de 2025, o Amazon ECS alterou o modo padrão do driver de log de `blocking` para `non-blocking` a fim de priorizar a disponibilidade das tarefas em relação ao registro em log. Para continuar usando o modo `blocking` após essa alteração, siga uma destas opções:
>
> - Defina a opção `mode` na definição `logConfiguration` do seu contêiner como `blocking`.
> - Defina a configuração da conta `defaultLogDriverMode` como `blocking`.

`max-buffer-size`

Obrigatório: Não

Valor padrão: `10m`

Quando o modo `non-blocking` é usado, a opção de log `max-buffer-size` controla o tamanho do buffer de anel usado para armazenamento de mensagens intermediárias. Certifique-se de especificar um tamanho de buffer adequado com base na sua aplicação. Quando o buffer ficar cheio, logs adicionais não poderão ser armazenados. Os logs que não puderem ser armazenados serão perdidos.

Para rotear logs usando o roteador de logs do `splunk`, você precisa especificar um `splunk-token` e um `splunk-url`.

Ao usar o roteador de log `awsfirelens` para rotear logs para um destino do AWS service (Serviço da AWS) ou para armazenamento e analytics de logs do AWS Partner Network, você pode definir a opção `log-driver-buffer-limit` para limitar o número de eventos que são armazenados em buffer na memória antes de serem enviados ao contêiner do roteador de logs. Isso pode ajudar a resolver possíveis problemas de perda de logs, pois o alto throughput pode resultar na falta de memória para o buffer dentro do Docker. Para obter mais informações, consulte the section called “Configuração de logs para obtenção de alto throughput”.

Outras opções que você pode especificar ao usar o `awsfirelens` para rotear logs dependem do destino. Ao exportar logs para o Amazon Data Firehose, você pode especificar a Região da AWS com a `region` e um nome para o fluxo de logs com `delivery_stream`.

Ao exportar logs para o Amazon Kinesis Data Streams, você pode especificar um Região da AWS com a `region` um com e um nome de fluxo de dados com `stream`.

Ao exportar logs para o Amazon OpenSearch Service, você pode especificar opções como `Name`, `Host` (endpoint do OpenSearch Service sem protocolo) `Port`, `Index`, `Type`, `Aws_auth`, `Aws_region`, `Suppress_Type_Name` e `tls`.

Ao exportar logs para o Amazon S3, você pode especificar o bucket usando a opção `bucket`. Você também pode especificar `region`, `total_file_size`, `upload_timeout` e `use_put_object` como opções.

Este parâmetro requer a versão 1.19 da Docker Remote API ou posterior em sua instância de contêiner.

`secretOptions`

Tipo: Matriz de objeto

Obrigatório: Não

Um objeto que representa o segredo a ser passado para a configuração de log. Os segredos usados na configuração de log podem incluir tokens de autenticação, certificados ou chaves de criptografia. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.

`name`

Tipo: string

Obrigatório: Sim

O valor a ser definido como a variável de ambiente no contêiner.

`valueFrom`

Tipo: string

Obrigatório: Sim

O segredo a ser exposto à configuração de log do contêiner.

```
"logConfiguration": {
 "logDriver": "splunk",
 "options": {
  "splunk-url": "https://cloud.splunk.com:8080",
  "splunk-token": "...",
  "tag": "...",
  ...
 },
 "secretOptions": [{
  "name": "splunk-token",
  "valueFrom": "/ecs/logconfig/splunkcred"
 }]
}
```

`firelensConfiguration`

Tipe: objeto FirelensConfiguration

Obrigatório: Não

A configuração do FireLens para o contêiner. Isso é usado para especificar e configurar um roteador de log para logs de contêiner. Para obter mais informações, consulte Envio de logs do Amazon ECS para um serviço da AWS ou para uma AWS Partner.

```
{
    "firelensConfiguration": {
        "type": "fluentd",
        "options": {
            "KeyName": ""
        }
    }
}
```

`options`

Tipo: Mapa de string para string

Obrigatório: Não

O mapa chave/valor de opções a serem usadas ao configurar o roteador de log. Esse campo é opcional e pode ser usado para especificar um arquivo de configuração personalizado

ou para adicionar outros metadados, como a tarefa, a definição de tarefa, o cluster e detalhes da instância de contêiner ao evento de log. Se especificado, a sintaxe a ser usada é `"options":{"enable-ecs-log-metadata":"true|false","config-file-type:"s3|file","config-file-value":"arn:aws:s3:::amzn-s3-demo-bucket/fluent.conf|filepath"}`. Para obter mais informações, consulte Exemplo de definição de tarefa do Amazon ECS: rotear logs para o FireLens.

`type`

Tipo: string

Obrigatório: Sim

O roteador de log a ser usado. Os valores válidos são `fluentd` ou `fluentbit`.

Segurança

Para obter mais informações sobre a segurança de contêineres, consulte Práticas recomendadas de segurança de tarefas e contêineres no Amazon ECS.

`credentialSpecs`

Tipo: Matriz de strings

Obrigatório: Não

Uma lista de ARNs no SSM ou no Amazon S3 para um arquivo de especificação de credencial (`CredSpec`) que configura o contêiner para autenticação do Active Directory. Recomendamos o uso deste parâmetro, em vez de `dockerSecurityOptions`. O número máximo de ARNs é 1.

Há dois formatos para cada ARN.

credentialspecdomainless:MyARN

Você usa `credentialspecdomainless:MyARN` para fornecer uma `CredSpec` com uma seção adicional para um segredo no Secrets Manager. Você fornece as credenciais de login para o domínio no segredo.

Cada tarefa executada em qualquer instância de contêiner pode se associar a domínios diferentes.

É possível usar esse formato sem associar a instância de contêiner a um domínio.

credentialspec:MyARN

Você usa `credentialspec:MyARN` para fornecer um `CredSpec` para um único domínio.

Você deve associar a instância de contêiner ao domínio antes de iniciar qualquer tarefa que use essa definição de tarefa.

Em ambos os formatos, substitua `MyARN` pelo ARN no SSM ou no Amazon S3.

A `credspec` deve fornecer um ARN no Secrets Manager para um segredo contendo o nome de usuário, a senha e o domínio ao qual se conectar. Para maior segurança, a instância não é associada ao domínio para autenticação sem domínio. Outras aplicações na instância não podem usar as credenciais sem domínio. É possível usar esse parâmetro para executar tarefas na mesma instância, mesmo que as tarefas precisem ser associadas a domínios diferentes. Para obter mais informações, consulte Uso de gMSAs para contêineres de Windows e Uso de gMSAs para contêineres de Linux.

`user`

Tipo: String

Obrigatório: Não

O usuário a ser utilizado dentro do contêiner. Esse parâmetro é mapeado para `User` no comando create-container do docker na opção `--user` de execução do docker.

> **Important**
>
> Ao executar tarefas que usam o modo de rede `host`, não execute contêineres usando o usuário raiz (UID 0). Como uma prática recomendada de segurança, sempre utilize um usuário não raiz.

É possível especificar o `user` usando os formatos a seguir. Se especificar um UID ou um GID, você deverá especificá-lo como um número inteiro positivo.

- `user`
- `user:group`
- `uid`
- `uid:gid`

- `user:gid`
- `uid:group`

> **Note**
>
> Este parâmetro não é compatível com contêineres do Windows.

```
"user": "string"
```

Limites de recurso

`ulimits`

Tipo: Matriz de objeto

Obrigatório: Não

Lista de valores `ulimit` a serem definidos para um contêiner. Esse valor substitui a configuração de cota de recursos padrão do sistema operacional. Esse parâmetro é mapeado para `Ulimits` no comando create-container do docker na opção `--ulimit` de execução do docker.

As tarefas do Amazon ECS hospedadas no Fargate usam os valores de limite de recursos padrão limitados pelo sistema operacional, com exceção do parâmetro de limite de recursos `nofile`. O limite de recursos `nofile` define uma restrição sobre o número de arquivos abertos que um contêiner pode usar. No Fargate, o limite flexível padrão de `nofile` é `65535`, e o limite rígido é `65535`. É possível definir os valores de ambos os limites até `1048576`. Para obter mais informações, consulte Limites de recursos de tarefa.

Este parâmetro requer a versão 1.18 da Docker Remote API ou posterior em sua instância de contêiner.

> **Note**
>
> Este parâmetro não é compatível com contêineres do Windows.

```
"ulimits": [
```

```
      {
        "name":
 "core"|"cpu"|"data"|"fsize"|"locks"|"memlock"|"msgqueue"|"nice"|"nofile"|"nproc"|"rss"|"rtp
        "softLimit": integer,
        "hardLimit": integer
      }
      ...
    ]
```

`name`

Tipo: String

Valores válidos: `"core" | "cpu" | "data" | "fsize" | "locks" | "memlock" | "msgqueue" | "nice" | "nofile" | "nproc" | "rss" | "rtprio" | "rttime" | "sigpending" | "stack"`

Exigido: Sim, quando `ulimits` são usados

O `type` do `ulimit`.

`hardLimit`

Tipo: inteiro

Exigido: Sim, quando `ulimits` são usados

O limite rígido do tipo `ulimit`. O valor pode ser especificado em bytes, segundos ou como uma contagem, dependendo `type` do `ulimit`.

`softLimit`

Tipo: inteiro

Exigido: Sim, quando `ulimits` são usados

O limite flexível do tipo `ulimit`. O valor pode ser especificado em bytes, segundos ou como uma contagem, dependendo `type` do `ulimit`.

Rótulos do Docker

`dockerLabels`

Tipo: mapa de string para string

Obrigatório: Não

Um mapa de chave/valor de rótulos a ser adicionado ao contêiner. Esse parâmetro é mapeado para `Labels` no comando create-container do docker na opção `--label` de execução do docker.

Este parâmetro requer a versão 1.18 da Docker Remote API ou posterior em sua instância de contêiner.

```
"dockerLabels": {"string": "string"
      ...}
```

Outros parâmetros de definição de contêiner

Os parâmetros de definição de contêiner a seguir podem ser usados durante o registro das definições de tarefa no console do Amazon ECS usando a opção Configure via JSON (Configurar via JSON). Para obter mais informações, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

Tópicos

- Parâmetros do Linux
- Dependência de contêiner
- Tempos limite de contêiner
- Controles do sistema
- Interativo
- Pseudoterminal

Parâmetros do Linux

`linuxParameters`

Tipo: Objeto LinuxParameters

Obrigatório: Não

Opções específicas do Linux que são aplicadas ao contêiner, como KernelCapabilities.

> **Note**
>
> Não há suporte para este parâmetro em contêineres do Windows.

```
"linuxParameters": {
      "capabilities": {
        "add": ["string", ...],
        "drop": ["string", ...]
        }
      }
```

`capabilities`

Tipo: objeto KernelCapabilities

Obrigatório: Não

Os recursos do Linux para o contêiner que são descartados da configuração padrão fornecida pelo Docker. Para obter mais informações sobre esses recursos do Linux, consulte a página do manual do Linux recursos(7).

`add`

Tipo: matriz de strings

Valores válidos: `"SYS_PTRACE"`

Obrigatório: Não

Os recursos do Linux para o contêiner que são adicionados à configuração padrão que é fornecida pelo Docker. Esse parâmetro é mapeado para `CapAdd` no comando create-container do docker na opção `--cap-add` de execução do docker.

`drop`

Tipo: matriz de strings

Valores válidos: `"ALL" | "AUDIT_CONTROL" | "AUDIT_WRITE" | "BLOCK_SUSPEND" | "CHOWN" | "DAC_OVERRIDE" | "DAC_READ_SEARCH" | "FOWNER" | "FSETID" | "IPC_LOCK" | "IPC_OWNER" | "KILL" | "LEASE" | "LINUX_IMMUTABLE" | "MAC_ADMIN" | "MAC_OVERRIDE" | "MKNOD" |`

```
"NET_ADMIN" | "NET_BIND_SERVICE" | "NET_BROADCAST" | "NET_RAW"
| "SETFCAP" | "SETGID" | "SETPCAP" | "SETUID" | "SYS_ADMIN" |
"SYS_BOOT" | "SYS_CHROOT" | "SYS_MODULE" | "SYS_NICE" | "SYS_PACCT"
| "SYS_PTRACE" | "SYS_RAWIO" | "SYS_RESOURCE" | "SYS_TIME" |
"SYS_TTY_CONFIG" | "SYSLOG" | "WAKE_ALARM"
```

Obrigatório: Não

Os recursos do Linux para o contêiner que são removidos da configuração padrão fornecida pelo Docker. Esse parâmetro é mapeado para `CapDrop` no comando create-container do docker na opção `--cap-drop` de execução do docker.

`devices`

Todos os dispositivos do host a serem expostos ao contêiner. Esse parâmetro é mapeado para `Devices` no comando create-container do docker na opção `--device` de execução do docker.

> **Note**
>
> O parâmetro `devices` não é compatível com o tipo de inicialização do Fargate.

Tipo: Matriz de objetos Device

Obrigatório: Não

`hostPath`

O caminho para o dispositivo na instância de contêiner host.

Tipo: string

Obrigatório: Sim

`containerPath`

O caminho dentro do contêiner no qual expor o dispositivo do host.

Tipo: String

Obrigatório: Não

`permissions`

As permissões explícitas a serem fornecidas ao contêiner para o dispositivo. Por padrão, o contêiner tem permissões para `read`, `write` e `mknod` no dispositivo.

Tipo: matriz de strings

Valores válidos: `read` | `write` | `mknod`

`initProcessEnabled`

Execute um processo `init` dentro do contêiner que encaminha sinais e colhe processos. Esse parâmetro é mapeado para a opção `--init` de execução do docker.

Este parâmetro requer a versão 1.25 ou posterior da Docker Remote API ou posterior em sua instância de contêiner.

`maxSwap`

Esse parâmetro não é compatível com tarefas sendo executadas no Fargate.

A quantidade total de memória de troca (em MiB) que um contêiner pode usar. Esse parâmetro será convertido na opção `--memory-swap` para execução do docker em que o valor é a soma da memória do contêiner mais o valor de `maxSwap`.

Se um valor `maxSwap` de `0` for especificado, o contêiner não usará a troca. Os valores aceitos são `0` ou qualquer número inteiro positivo. Se o parâmetro `maxSwap` for omitido, o contêiner usará a configuração de troca para a instância de contêiner na qual ele está sendo executado. Um valor `maxSwap` deve ser definido para que o parâmetro `swappiness` seja usado.

`sharedMemorySize`

O valor do tamanho (em MiB) do volume `/dev/shm`. Esse parâmetro é mapeado para a opção `--shm-size` de execução do docker.

> **Note**
>
> Se você estiver usando tarefas que utilizem o tipo de inicialização Fargate, `sharedMemorySize` não haverá suporte para o parâmetro .

Tipo: inteiro

`tmpfs`

O caminho do contêiner, as opções de montagem e o tamanho (em MiB) da montagem do tmpfs. Esse parâmetro é mapeado para a opção `--tmpfs` de execução do docker.

> **Note**
>
> Se você estiver usando tarefas que utilizem o tipo de inicialização Fargate, `tmpfs` não haverá suporte para o parâmetro .

Tipo: Matriz de objetos Tmpfs

Obrigatório: Não

`containerPath`

O caminho absoluto do arquivo no qual o volume tmpfs deve ser montado.

Tipo: string

Obrigatório: Sim

`mountOptions`

A lista de opções de montagem do volume tmpfs.

Tipo: matriz de strings

Obrigatório: Não

Valores Válidos: `"defaults" | "ro" | "rw" | "suid" | "nosuid" | "dev" | "nodev" | "exec" | "noexec" | "sync" | "async" | "dirsync" | "remount" | "mand" | "nomand" | "atime" | "noatime" | "diratime" | "nodiratime" | "bind" | "rbind" | "unbindable" | "runbindable" | "private" | "rprivate" | "shared" | "rshared" | "slave" | "rslave" | "relatime" | "norelatime" | "strictatime" | "nostrictatime" | "mode" | "uid" | "gid" | "nr_inodes" | "nr_blocks" | "mpol"`

`size`

O tamanho máximo (em MiB) do volume tmpfs.

Tipo: inteiro

Obrigatório: Sim

Dependência de contêiner

`dependsOn`

Tipo: matriz de objetos ContainerDependency

Obrigatório: Não

As dependências definidas para startup e desligamento do contêiner. Um contêiner pode conter várias dependências. Quando uma dependência é definida para o startup do contêiner, ela é revertida para o desligamento do contêiner. Para obter um exemplo, consulte Dependência de contêiner.

> **Note**
>
> Se um contêiner não atender a uma restrição de dependência ou expirar antes de cumprir a restrição, o Amazon ECS não evoluirá contêineres dependentes para seu próximo estado.

Este parâmetro exige que a tarefa ou o serviço usem a plataforma versão `1.3.0` ou posterior (Linux), ou `1.0.0` (Windows).

```
"dependsOn": [
    {
        "containerName": "string",
        "condition": "string"
    }
]
```

`containerName`

Tipo: string

Obrigatório: Sim

O nome do contêiner que deve atender à condição especificada.

`condition`

Tipo: string

Obrigatório: Sim

A condição de dependência do contêiner. A seguir estão as condições disponíveis e seus comportamentos.

- `START`: essa condição emula o comportamento de links e volumes atuais. A condição valida que um contêiner dependente seja iniciado antes de permitir que outros contêineres sejam iniciados.
- `COMPLETE`: essa condição valida que um contêiner dependente seja executado até a conclusão (encerramento) antes de permitir que outros contêineres sejam iniciados. Isso pode ser útil para os contêineres não essenciais que executam um script e depois são encerrados. Não é possível definir essa condição em um contêiner essencial.
- `SUCCESS`: essa condição é igual a `COMPLETE`, mas também exige que o contêiner seja encerrado com um status `zero`. Não é possível definir essa condição em um contêiner essencial.
- `HEALTHY`: essa condição valida que o contêiner dependente foi aprovado na verificação de integridade do Docker antes de permitir que outros contêineres sejam iniciados. Isso requer que o contêiner dependente tenha verificações de integridade configuradas na definição de tarefa. Essa condição é confirmada apenas no startup da tarefa.

Tempos limite de contêiner

`startTimeout`

Tipo: inteiro

Obrigatório: Não

Valores de exemplo: `120`

Tempo a ser aguardado (em segundos) antes de desistir de resolver dependências para um contêiner.

Por exemplo, você especifica dois contêineres em uma definição de tarefa com o `containerA` tendo uma dependência do `containerB` atingir um status `COMPLETE`, `SUCCESS` ou `HEALTHY`.

Se um valor de `startTimeout` for especificado para o `containerB` e ele não atingir o status desejado nesse tempo, o `containerA` não será iniciado.

> **Note**
>
> Se um contêiner não atender a uma restrição de dependência ou expirar antes de cumprir a restrição, o Amazon ECS não evoluirá contêineres dependentes para seu próximo estado.

Este parâmetro exige que a tarefa ou o serviço usem a plataforma versão `1.3.0` ou posterior (Linux). O valor máximo é 120 segundos.

`stopTimeout`

Tipo: inteiro

Obrigatório: Não

Valores de exemplo: `120`

Período (em segundos) a ser aguardado antes de o contêiner ser eliminado de maneira forçada se não for encerrado normalmente por conta própria.

Este parâmetro exige que a tarefa ou o serviço usem a plataforma versão `1.3.0` ou posterior (Linux). Se o parâmetro não for especificado, então o valor padrão de 30 segundos será usado. O valor máximo é 120 segundos.

Controles do sistema

`systemControls`

Tipo: objeto SystemControl

Obrigatório: Não

Uma lista de parâmetros de kernel com namespace a ser definida no contêiner. Esse parâmetro é mapeado para `Sysctls` no comando create-container do docker e na opção `--sysctl` de execução do docker. Por exemplo, é possível definir a configuração `net.ipv4.tcp_keepalive_time` para manter conexões de longa duração.

Não recomendamos que você especifique parâmetros `systemControls` relacionados à rede para vários contêineres em uma única tarefa que também use o modo de rede `awsvpc` ou `host`. Fazer isso apresenta as desvantagens a seguir:

- Se você definir `systemControls` para algum contêiner, isso se aplicará a todos os contêineres na tarefa. Se você definir um `systemControls` diferente para vários contêineres em uma única tarefa, o contêiner iniciado por último determinará qual `systemControls` entra em vigor.

Se você estiver configurando um namespace de recurso IPC para usar nos contêineres da tarefa, as condições a seguir serão aplicadas aos controles do sistema. Para obter mais informações, consulte Modo IPC.

- Para tarefas que usam o modo IPC `host`, o namespace IPC `systemControls` não é compatível.
- Para tarefas que usam o modo IPC de `task`, os valores de `systemControls` do namespace IPC serão aplicados a todos os contêineres em uma tarefa.

> **Note**
>
> Este parâmetro não é compatível com contêineres do Windows.

> **Note**
>
> Só há suporte para o parâmetro para tarefas hospedadas no AWS Fargate que usem a versão da plataforma `1.4.0` ou posterior (Linux). Não há suporte para isso nos contêineres do Windows no Fargate.

```
"systemControls": [
    {
         "namespace":"string",
         "value":"string"
    }
]
```

`namespace`

Tipo: String

Obrigatório: Não

O parâmetro de kernel com namespace para o qual definir um `value`.

Valores de namespace IPC válidos: `"kernel.msgmax" | "kernel.msgmnb" | "kernel.msgmni" | "kernel.sem" | "kernel.shmall" | "kernel.shmmax" | "kernel.shmmni" | "kernel.shm_rmid_forced"` e `Sysctls`, que comecem com `"fs.mqueue.*"`

Valores de namespace de rede válidos: `Sysctls`, que comece com `"net.*"`. No Fargate, apenas os `Sysctls` com namespace existentes dentro do contêiner são aceitos.

Há suporte para todos esses valores no Fargate.

`value`

Tipo: String

Obrigatório: Não

O valor do parâmetro de kernel com namespace especificado em `namespace`.

Interativo

`interactive`

Tipo: booliano

Obrigatório: Não

Quando esse parâmetro é `true`, você pode implantar aplicações em contêineres que exigem a alocação de `stdin` ou `tty`. Esse parâmetro é mapeado para `OpenStdin` no comando create-container do docker na opção `--interactive` de execução do docker.

O padrão é `false`.

Pseudoterminal

`pseudoTerminal`

Tipo: booliano

Obrigatório: Não

Quando esse parâmetro é `true`, um TTY é alocado. Esse parâmetro é mapeado para `Tty` no comando create-container do docker na opção `--tty` de execução do docker.

O padrão é `false`.

Nome do acelerador de inferência elástica

Requisito do recurso acelerador Elastic Inference para a definição de tarefa.

> **Note**
>
> O Amazon Elastic Inference (EI) não está mais disponível para clientes.

Os seguintes parâmetros são permitidos em uma definição de tarefa:

`deviceName`

Tipo: string

Obrigatório: Sim

O nome do dispositivo do acelerador de inferência elástica. O `deviceName` também deve ser referenciado em uma definição de contêiner, consulte `Elastic Inference accelerator`.

`deviceType`

Tipo: string

Obrigatório: Sim

O acelerador de inferência elástica a ser usado.

Configuração do proxy

`proxyConfiguration`

Tipo: objeto ProxyConfiguration

Obrigatório: Não

Os detalhes de configuração do proxy do App Mesh.

> **Note**
>
> Este parâmetro não é compatível com contêineres do Windows.

```
"proxyConfiguration": {
    "type": "APPMESH",
    "containerName": "string",
    "properties": [
        {
           "name": "string",
           "value": "string"
        }
    ]
}
```

`type`

Tipo: String

Valor: `APPMESH`

Obrigatório: Não

O tipo de proxy. O único valor aceito é `APPMESH`.

`containerName`

Tipo: string

Obrigatório: Sim

O nome do contêiner que serve como proxy do App Mesh.

`properties`

Tipo: matriz de objetos KeyValuePair

Obrigatório: Não

O conjunto de parâmetros de configuração de rede para fornecer o plugin Container Network Interface (CNI), especificado como pares de chave/valor.

- `IgnoredUID`: (obrigatório) o ID de usuário (UID) do contêiner de proxy conforme definido pelo parâmetro `user` em uma definição de contêiner. Isso é usado para garantir que o

proxy ignore seu próprio tráfego. Se `IgnoredGID` for especificado, esse campo poderá estar vazio.

- `IgnoredGID`: (obrigatório) o ID de grupo (GID) do contêiner de proxy conforme definido pelo parâmetro `user` em uma definição de contêiner. Isso é usado para garantir que o proxy ignore seu próprio tráfego. Se `IgnoredUID` for especificado, esse campo poderá estar vazio.
- `AppPorts`: (obrigatório) a lista de portas usadas pelo aplicativo. O tráfego de rede para essas portas é encaminhado para `ProxyIngressPort` e `ProxyEgressPort`.
- `ProxyIngressPort`: (obrigatório) especifica a porta para a qual o tráfego de entrada para `AppPorts` é direcionado.
- `ProxyEgressPort`: (obrigatório) especifica a porta para a qual o tráfego de saída de `AppPorts` é direcionado.
- `EgressIgnoredPorts`: (obrigatório) o tráfego de saída para essas portas especificadas é ignorado e não é redirecionado para `ProxyEgressPort`. Ele pode ser uma lista vazia.
- `EgressIgnoredIPs`: (obrigatório) o tráfego de saída para esses endereços IP especificados é ignorado e não é redirecionado para `ProxyEgressPort`. Ele pode ser uma lista vazia.

`name`

Tipo: String

Obrigatório: Não

O nome do par de chave/valor.

`value`

Tipo: String

Obrigatório: Não

O valor do par de chave/valor.

Volumes

Se preferir, ao registrar uma definição de tarefa, é possível especificar uma lista de volumes a serem passados para o daemon do Docker em uma instância de contêiner, que se tornará disponível para o acesso de outros contêineres da mesma instância de contêiner.

A seguir estão os tipos de volumes de dados que podem ser usados.

- Volumes do Amazon EBS: fornece armazenamento em blocos econômico, durável e de alto desempenho para workloads em contêineres com uso intenso de dados. Você pode anexar um volume do Amazon EBS por tarefa do Amazon ECS ao executar uma tarefa autônoma ou ao criar ou atualizar um serviço. Os volumes do Amazon EBS são compatíveis com as tarefas do Linux hospedadas no Fargate. Para obter mais informações, consulte Uso de volumes do Amazon EBS com o Amazon ECS.
- Volumes do Amazon EFS: oferece armazenamento de arquivos simples, escalável e persistente para uso com tarefas do Amazon ECS. Com o Amazon EFS, a capacidade de armazenamento é elástica. Ela aumenta e diminui automaticamente à medida que arquivos são adicionados e removidos. As aplicações podem ter o armazenamento de que precisam, quando precisam. Os volumes do Amazon EFS são compatíveis com as tarefas hospedadas no Fargate. Para obter mais informações, consulte Uso de volumes do Amazon EFS com o Amazon ECS.
- Volumes do FSx para Windows File Server: fornece servidores de arquivos Microsoft Windows totalmente gerenciados. Esses servidores têm o suporte de um sistema de arquivos do Windows. Ao usar o FSx for Windows File Server junto com o Amazon ECS, você pode provisionar tarefas do Windows com armazenamento de arquivos persistente, distribuído, compartilhado e estático. Para obter mais informações, consulte Uso de volumes do FSx para Windows File Server com Amazon ECS.

  Não há suporte para essa opção nos contêineres do Windows no Fargate.
- Montagens vinculadas: um arquivo ou diretório do computador host é montado em um contêiner. Volumes de host de montagem vinculada são compatíveis ao executar tarefas. Para usar os volumes de host de montagens bind, especifique um `host` e um valor `sourcePath` opcional em sua definição de tarefa.

Para obter mais informações, consulte Opções de armazenamento para tarefas do Amazon ECS.

Os parâmetros a seguir são permitidos em uma definição de contêiner.

`name`

Tipo: String

Obrigatório: Não

O nome do volume. São permitidos até 255 letras (caixa alta e baixa), números, hífens (`-`) e sublinhados (`_`). Esse nome é referenciado no parâmetro `sourceVolume` do objeto `mountPoints` de definição do contêiner.

`host`

Obrigatório: Não

O parâmetro `host` é usado para vincular o ciclo de vida da montagem bind à instância host do Amazon EC2, em vez da tarefa, e onde ela está armazenada. Caso o parâmetro `host` esteja vazio, o daemon do Docker atribui um caminho host para o volume de dados, mas os dados não têm garantia de persistir depois que os contêineres associados deixarem de ser executados.

Os contêineres do Windows podem montar diretórios inteiros na mesma unidade como `$env:ProgramData`.

> **Note**
>
> O parâmetro `sourcePath` só é compatível ao usar tarefas hospedadas em instâncias do Amazon EC2.

`sourcePath`

Tipo: String

Obrigatório: Não

Quando o parâmetro `host` é usado, especifique um `sourcePath` para declarar o caminho na instância host do Amazon EC2 que é apresentada ao contêiner. Caso esse parâmetro esteja vazio, o daemon do Docker atribui um caminho host para você. Caso o parâmetro `host` contenha um local de arquivo `sourcePath`, o volume de dados persistirá no local especificado na instância host do Amazon EC2 até você excluí-lo manualmente. Caso o valor `sourcePath` não exista na instância host do Amazon EC2, o daemon do Docker o criará. Caso o local exista, o conteúdo da pasta do caminho de origem é exportado.

`configuredAtLaunch`

Tipo: booliano

Obrigatório: Não

Especifica se um volume é configurável na execução. Quando definido como `true`, você pode configurar o volume ao executar uma tarefa autônoma ou ao criar ou atualizar um serviço. Quando definido como `true`, não será possível fornecer outra configuração de volume na definição da tarefa. Esse parâmetro deve ser definido como `true` para configurar um volume do Amazon EBS para anexação a uma tarefa. Definir `configuredAtLaunch` como `true` e adiar a configuração do volume para a fase de execução permite criar definições de tarefas que não estão restritas a um tipo de volume ou a configurações de volume específicas. Isso torna a definição de tarefa reutilizável em diferentes ambientes de execução. Para obter mais informações, consulte Volumes do Amazon EBS.

`dockerVolumeConfiguration`

Tipo: objeto DockerVolumeConfiguration

Obrigatório: Não

Esse parâmetro é especificado ao usar volumes do Docker. Os volumes do Docker só são compatíveis ao executar tarefas em instâncias do EC2. Os contêineres do Windows só são compatíveis com o uso do driver `local`. Para usar montagens bind, em vez disso, especifique um `host`.

`scope`

Tipo: String

Valores válidos: `task` | `shared`

Obrigatório: Não

O escopo para o volume do Docker, que determina o ciclo de vida. Os volumes do Docker que são delimitados para uma `task` são provisionados automaticamente quando a tarefa é iniciada e destruídos quando a tarefa é interrompida. Volumes do Docker delimitados como `shared` são mantidos após a interrupção da tarefa.

`autoprovision`

Tipo: booliano

Valor padrão: `false`

Obrigatório: Não

Se o valor for `true`, o volume de Docker será criado se ele ainda não existir. Esse campo só será usado se `scope` for `shared`. Se `scope` for `task`, esse parâmetro deverá ser omitido.

`driver`

Tipo: String

Obrigatório: Não

O driver do volume do Docker a ser usado. O valor do driver deve corresponder ao nome do driver fornecido pelo Docker porque esse nome é usado no posicionamento de tarefas. Se o driver foi instalado usando a CLI de plug-in do Docker, use `docker plugin ls` para recuperar o nome do driver na instância de contêiner. Se o driver foi instalado usando outro método, use a descoberta de plug-in do Docker para recuperar o nome do driver.

`driverOpts`

Tipo: String

Obrigatório: Não

Um mapa de opções específicas do driver do Docker pelas quais passar. Esse parâmetro é mapeado para `DriverOpts` na seção Criar um volume da API remota do Docker.

`labels`

Tipo: String

Obrigatório: Não

Metadados personalizados para adicionar ao volume do Docker.

`efsVolumeConfiguration`

Tipo: objeto EFSVolumeConfiguration

Obrigatório: Não

Esse parâmetro é especificado quando volumes Amazon EFS são usados.

`fileSystemId`

Tipo: string

Obrigatório: Sim

A ID do sistema de arquivamento Amazon EFS a ser usada.

`rootDirectory`

Tipo: string

Obrigatório: Não

O diretório dentro do sistema de arquivamento Amazon EFS a ser montado como diretório raiz dentro do host. Se esse parâmetro for omitido, a raiz do volume do Amazon EFS será usada. Especificar / tem o mesmo efeito que omitir esse parâmetro.

> **Important**
>
> Se um ponto de acesso do EFS for especificado em `authorizationConfig`, o parâmetro de diretório raiz deverá ser omitido ou definido como `/`, o que imporá o caminho definido no ponto de acesso do EFS.

`transitEncryption`

Tipo: String

Valores válidos: `ENABLED` | `DISABLED`

Obrigatório: Não

Especifica se a criptografia deve ou não ser habilitada para dados do Amazon EFS em trânsito entre o host do Amazon ECS e o servidor do Amazon EFS. A criptografia de trânsito deverá ser habilitada se a autorização do IAM do Amazon EFS for usada. Se o parâmetro for omitido, o valor padrão `DISABLED` será usado. Para obter mais informações, consulte Criptografar dados em trânsito no Guia do usuário do Amazon Elastic File System.

`transitEncryptionPort`

Tipo: inteiro

Obrigatório: Não

A porta a ser usada ao enviar dados criptografados entre o host do Amazon ECS e o servidor do Amazon EFS. Caso não especifique uma porta de criptografia em trânsito, a tarefa usará a estratégia de seleção de porta usada pelo assistente de montagem do Amazon EFS. Para mais informações, consulte Auxiliar de Montagem EFS no Guia de Usuário Amazon Elastic File System.

`authorizationConfig`

Tipo: objeto EFSAuthorizationConfig

Obrigatório: Não

Detalhes de configuração de autorização do sistema de arquivamento Amazon EFS.

`accessPointId`

Tipo: Sequência

Obrigatório: Não

ID do ponto de acesso a ser usado. Se um ponto de acesso for especificado, o valor do diretório raiz em `efsVolumeConfiguration` deve ser omitido ou definido como `/`, o que imporá o caminho definido no ponto de acesso do EFS. Se um ponto de acesso for usado, a criptografia em trânsito deverá ser habilitada em `EFSVolumeConfiguration`. Para mais informações, consulte Trabalhando com Pontos de Acesso Amazon EFS no Guia de Usuário Amazon Elastic File System.

`iam`

Tipo: String

Valores válidos: `ENABLED` | `DISABLED`

Obrigatório: Não

Especifica se é necessário ou não usar o perfil do IAM para a tarefa do Amazon ECS estabelecida em uma definição de tarefa ao montar o sistema de arquivos do Amazon EFS. Em caso positivo, a criptografia de trânsito deve estar habilitada em `EFSVolumeConfiguration`. Se o parâmetro for omitido, o valor padrão `DISABLED` será usado. Para obter mais informações, consulte Funções do IAM para tarefas.

`FSxWindowsFileServerVolumeConfiguration`

Tipo: objeto FSxWindowsFileServerVolumeConfiguration

Obrigatório: Sim

Esse parâmetro é especificado quando você está usando um sistema de arquivos do Amazon FSx para Windows File Server para armazenamento de tarefas.

`fileSystemId`

Tipo: string

Obrigatório: Sim

O ID do sistema de arquivos do FSx for Windows File Server a ser usado.

`rootDirectory`

Tipo: string

Obrigatório: Sim

O diretório no sistema de arquivos do FSx for Windows File Server que deve ser montado como o diretório raiz dentro do host.

`authorizationConfig`

`credentialsParameter`

Tipo: string

Obrigatório: Sim

As opções de credencial de autorização.

opções:

- O nome do recurso da Amazon (ARN) do segredo do AWS Secrets Manager.
- ARN de um parâmetro do AWS Systems Manager.

`domain`

Tipo: string

Obrigatório: Sim

Um nome de domínio totalmente qualificado hospedado por um diretório do AWS Directory Service for Microsoft Active Directory (AWS Managed Microsoft AD) ou por um Active Directory do EC2 auto-hospedado.

Tags

Quando você registra uma definição de tarefa, pode especificar opcionalmente etiquetas de metadados aplicadas à definição de tarefa. As etiquetas ajudam você a categorizar e organizar sua definição de tarefa. Cada tag consiste em uma chave e um valor opcional. Defina ambos. Para obter mais informações, consulte Marcação de recursos do Amazon ECS.

> **Important**
>
> Não adicione informações de identificação pessoal nem outras informações confidenciais ou sigilosas em tags. As tags são acessíveis a muitos serviços da AWS, incluindo faturamento. As tags não devem ser usadas para dados privados ou confidenciais.

Os parâmetros a seguir são permitidos em um objeto de etiqueta.

`key`

Tipo: String

Obrigatório: Não

Uma parte de um par de chave/valor que compõe uma tag. Uma chave é um rótulo geral que age como uma categoria para valores de tag mais específicos.

`value`

Tipo: String

Obrigatório: Não

A parte opcional de um par de chave/valor que compõe uma tag. Um valor atua como um descritor dentro de uma categoria de tag (chave).

Outros parâmetros de definição de tarefa

Os parâmetros de definição de tarefa a seguir podem ser usados durante o registro das definições de tarefa no console do Amazon ECS usando a opção Configure via JSON (Configurar via JSON). Para obter mais informações, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

Tópicos

- Armazenamento temporário
- Modo IPC
- Modo PID
- Injeção de falhas

Armazenamento temporário

`ephemeralStorage`

Tipo: objeto EphemeralStorage

Obrigatório: Não

A quantidade de armazenamento temporário (em GB) a ser alocado para a tarefa. Esse parâmetro é usado para expandir a quantidade total de armazenamento temporário, além da quantidade padrão, para tarefas hospedadas no AWS Fargate. Para obter mais informações, consulte the section called “Montagens bind”.

> **Note**
>
> Este parâmetro é compatível apenas com a plataforma versão `1.4.0` ou posterior (Linux), ou `1.0.0` ou posterior (Windows).

Modo IPC

`ipcMode`

Esse parâmetro não é compatível com tarefas sendo executadas no Fargate.

Tipo: String

Obrigatório: Não

O namespace de recurso IPC a ser usado para os contêineres na tarefa. Os valores válidos são `host`, `task` ou `none`. Se o `host` for especificado, todos os contêineres dentro das tarefas que especificaram o modo IPC `host` na mesma instância de contêiner compartilham os mesmos recursos IPC com a instância do Amazon EC2 do host. Se `task` for especificado, todos os contêineres dentro da tarefa especificada compartilharão os mesmos recursos IPC. Se `none` for especificado, os recursos IPC nos contêineres de uma tarefa serão privados e não serão compartilhados com outros contêineres em uma tarefa ou na instância de contêiner. Se nenhum valor for especificado, o compartilhamento do namespace de recurso IPC dependerá da configuração do daemon do Docker na instância de contêiner.

Se o modo IPC `host` for usado, haverá um risco elevado de exposição indesejada do namespace IPC.

Se você estiver definindo parâmetros de kernel com namespace que usa `systemControls` para os contêineres na tarefa, os itens a seguir serão aplicados ao seu namespace de recurso IPC.

- Para tarefas que usam o modo IPC `host`, o namespace IPC relacionado `systemControls` não é compatível.
- Para tarefas que usam o modo IPC `task`, `systemControls` relacionados ao namespace IPC serão aplicados a todos os contêineres em uma tarefa.

> **Note**
>
> Esse parâmetro não é compatível com os contêineres do Windows ou tarefas que usam o tipo de inicialização Fargate.

Modo PID

`pidMode`

Tipo: String

Valores válidos: `host` | `task`

Obrigatório: Não

O namespace do processo a ser usado para os contêineres na tarefa. Os valores válidos são `host` ou `task`. Em contêineres do Linux, o único valor válido é `task`. Por exemplo, o monitoramento de arquivos associados pode precisar que `pidMode` acesse informações sobre outros contêineres em execução na mesma tarefa.

Se a `task` for especificada, todos os contêineres da tarefa especificada compartilharão o mesmo namespace de processo.

Se nenhum valor for especificado, o padrão será um namespace privado para cada contêiner.

> **Note**
>
> Só há suporte para o parâmetro para tarefas hospedadas no AWS Fargate que usem a versão da plataforma `1.4.0` ou posterior (Linux). Não há suporte para isso nos contêineres do Windows no Fargate.

Injeção de falhas

`enableFaultInjection`

Tipo: booliano

Valores válidos: `true` | `false`

Obrigatório: Não

Se esse parâmetro estiver definido como `true`, na carga útil de uma tarefa, o Amazon ECS e o Fargate aceitam solicitações de injeção de falhas dos contêineres da tarefa. Esse parâmetro é definido como `false` por padrão.

Parâmetros de definição de tarefa do Amazon ECS para o tipo de inicialização do EC2

As definições de tarefa são divididas em partes separadas: família de tarefas, perfil de tarefas do AWS Identity and Access Management (IAM), modo de rede, definições de contêiner, volumes, restrições de posicionamento de tarefas e tipos de execução. As definições de família e de contêiner são obrigatórias em uma definição de tarefa. Por outro lado, a função de tarefa, o modo da rede, volumes, restrições de posicionamento de tarefa e o tipo de inicialização são opcionais.

É possível usar esses parâmetros em um arquivo JSON para configurar sua definição de tarefa.

Veja a seguir descrições mais detalhadas de cada parâmetro de definição de tarefa para o tipo de inicialização do EC2.

Família

`family`

Tipo: string

Obrigatório: Sim

Ao registrar uma definição de tarefa, você dá a ela uma família, semelhante a um nome para várias versões da definição de tarefa, especificada com um número de revisão. A primeira

definição de tarefa registrada em uma determinada família recebe uma revisão 1, e todas as definições de tarefa registradas depois receberão um número de revisão sequencial.

Tipos de inicialização

Ao registrar uma definição de tarefa, você pode especificar um tipo de inicialização em relação à qual o Amazon ECS deve validar a definição de tarefa. Se a definição de tarefa não for validada em relação às compatibilidades especificadas, uma exceção de cliente será retornada. Para obter mais informações, consulte Tipos de inicialização do Amazon ECS.

O seguinte parâmetro é permitido em uma definição de tarefa.

`requiresCompatibilities`

Tipo: Matriz de strings

Obrigatório: Não

Valores Válidos: `EC2`

O tipo de inicialização em relação à qual a definição de tarefa é validada. Isso inicia uma verificação para garantir que todos os parâmetros usados na definição de tarefa atendam aos requisitos do tipo de inicialização.

Função da tarefa

`taskRoleArn`

Tipo: String

Obrigatório: Não

Ao registrar uma definição de tarefa, você pode fornecer uma função de tarefa para uma função do IAM que permita que os contêineres da permissão de tarefa chame as APIs da AWS especificadas nas políticas associadas em seu nome. Para obter mais informações, consulte Perfil do IAM para tarefas do Amazon ECS.

Quando você inicia a AMI do Windows Server otimizada para o Amazon ECS, os perfis do IAM para tarefas no Windows exigem que a opção `-EnableTaskIAMRole` seja definida. Seus contêineres também devem executar algum código de configuração para usar o recurso. Para

obter mais informações, consulte Configuração adicional de instância do Windows no Amazon EC2.

Função de execução de tarefas

`executionRoleArn`

Tipo: string

Obrigatório: Condicional

O nome do recurso da Amazon (ARN) da função de execução de tarefas que concede ao agente de contêiner do Amazon ECS permissão para fazer chamadas de API da AWS em seu nome.

> **Note**
>
> A função do IAM para execução da tarefa é necessária dependendo dos requisitos da sua tarefa. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.

Modo de rede

`networkMode`

Tipo: String

Obrigatório: Não

O modo de rede do Docker a ser usado para os contêineres na tarefa. Para tarefas do Amazon ECS hospedadas em instâncias do Linux do Amazon EC2, os valores válidos são `none`, `bridge`, `awsvpc` e `host`. Se nenhum modo de rede for especificado, o modo de rede padrão será `bridge`. Para tarefas do Amazon ECS hospedadas em instâncias do Windows do Amazon EC2, os valores válidos são `default` e `awsvpc`. Se nenhum modo de rede for especificado, o modo de rede `default` será usado.

Caso o modo de rede estiver definido como `none`, os contêineres da tarefa não terão conectividade externa e os mapeamentos de porta não poderão ser especificados na definição de contêiner.

Se o modo da rede for `bridge`, a tarefa usará a rede virtual integrada do Docker no Linux, que é executada em cada instância do Amazon EC2 que hospeda a tarefa. A rede virtual integrada no Linux usa o driver de rede `bridge` do Docker.

Se o modo de rede for `host`, a tarefa usará a rede do host que ignora a rede virtual integrada do Docker mapeando as portas de contêiner diretamente para a ENI da instância do Amazon EC2 que hospeda a tarefa. Os mapeamentos dinâmicos de portas não podem ser usados nesse modo de rede. Um contêiner em uma definição de tarefa que use esse modo deve especificar um número de `hostPort` específico. Um número de porta em um host não pode ser usado por várias tarefas. Como resultado, não é possível executar várias tarefas da mesma definição de tarefa em uma única instância do Amazon EC2.

> **Important**
>
> Ao executar tarefas que usam o modo de rede `host`, não execute contêineres usando o usuário raiz (UID 0) para melhor segurança. Como uma prática recomendada de segurança, sempre utilize um usuário não raiz.

Se o modo de rede for `awsvpc`, será alocada uma interface de rede elástica para a tarefa e você deverá especificar uma `NetworkConfiguration` ao criar um serviço ou executar uma tarefa com a definição de tarefa. Para obter mais informações, consulte Opções de redes de tarefas do Amazon ECS para o tipo de inicialização do EC2.

Se o modo da rede for `default`, a tarefa usará a rede virtual integrada do Docker no Windows, que é executada em cada instância do Amazon EC2 que hospeda a tarefa. A rede virtual integrada no Windows usa o driver de rede `nat` do Docker.

Os modos de rede `host` e `awsvpc` oferecem a melhor performance de rede para contêineres, pois usam a pilha de rede do Amazon EC2. Com os modos de rede `host` e `awsvpc`, as portas do contêiner expostas são mapeadas diretamente para a porta do host correspondente (para o modo de rede do `host`) ou para a porta de interface de rede elástica anexada (para o modo de rede `awsvpc`). Por causa disso, não é possível usar mapeamentos dinâmicos de portas de host.

O modo de rede permitido depende do sistema operacional da instância do EC2 subjacente. Se for o Linux, qualquer modo de rede pode ser usado. Se for o Windows, podem ser usados os modos `default` e `awsvpc`.

Plataforma de runtime

`operatingSystemFamily`

Tipo: string

Obrigatório: Condicional

Padrão: LINUX

Ao registrar uma definição de tarefa, você especifica a família do sistema operacional.

Os valores válidos são `LINUX`, `WINDOWS_SERVER_2025_FULL`, `WINDOWS_SERVER_2025_CORE`, `WINDOWS_SERVER_2022_CORE`, `WINDOWS_SERVER_2022_FULL`, `WINDOWS_SERVER_2019_FULL` e `WINDOWS_SERVER_2019_CORE`, `WINDOWS_SERVER_2016_FULL`, `WINDOWS_SERVER_2004_CORE` e `WINDOWS_SERVER_20H2_CORE`.

Todas as definições de tarefa usadas em um serviço devem ter o mesmo valor para esse parâmetro.

Quando uma definição de tarefa faz parte de um serviço, esse valor deve corresponder ao valor `platformFamily` do serviço.

`cpuArchitecture`

Tipo: string

Obrigatório: Condicional

Padrão: X86_64

Ao registrar uma definição de tarefa, você especifica a arquitetura da CPU. Os valores válidos são `X86_64` e `ARM64`.

Todas as definições de tarefa usadas em um serviço devem ter o mesmo valor para esse parâmetro.

Quando você tem tarefas do Linux, pode definir o valor como `ARM64`. Para obter mais informações, consulte the section called “Definições de tarefa para workloads do ARM de 64 bits”.

Tamanho da tarefa

Ao registrar uma definição de tarefa, você pode especificar o total de CPU e memória usados para a tarefa. Isso é separado dos valores de `cpu` e `memory` no nível de definição de contêiner. Para tarefas hospedadas em instâncias do Amazon EC2, esses campos são opcionais.

> **Note**
>
> Os parâmetros de CPU e memória em nível de tarefa são ignorados para contêineres do Windows. É recomendável especificar recursos em nível de contêiner para contêineres do Windows.

`cpu`

Tipo: string

Obrigatório: Condicional

> **Note**
>
> Este parâmetro não é compatível com contêineres do Windows.

O limite rígido de unidades de CPU a ser apresentado para a tarefa. Você pode especificar valores de CPU no arquivo JSON como uma string em unidades de CPU ou em CPUs virtuais (vCPUs). Por exemplo, você pode especificar um valor de CPU como `1024` em unidades de CPU ou `1 vCPU` em vCPUs. Quando a definição de tarefa for registrada, um valor de vCPU será convertido em um inteiro indicando as unidades de CPU.

Esse campo é opcional. Se o seu cluster não tiver registrado instâncias de contêiner com as unidades de CPU solicitadas disponíveis, ocorrerá uma falha na tarefa. Os valores compatível são de `0.125` vCPUs a `192` vCPUs.

`memory`

Tipo: string

Obrigatório: Condicional

> **Note**
>
> Este parâmetro não é compatível com contêineres do Windows.

O limite fixo de memória a ser apresentada para a tarefa. É possível especificar valores de memória na definição de tarefa como uma string em mebibytes (MiB) ou gigabytes (GB). Por exemplo, você pode especificar um valor de memória como `3072` em MiB ou `3 GB` em GB. Quando a definição de tarefa for registrada, um valor em GB será convertido em um inteiro indicando o MiB.

Esse campo é opcional e qualquer valor pode ser usado. Se um valor de memória no nível de tarefa for especificado, o valor de memória no nível de contêiner será opcional. Se o seu cluster não tiver registrado instâncias de contêiner com a memória solicitada disponível, ocorrerá uma falha na tarefa. É possível maximizar a utilização de recursos fornecendo às tarefas o máximo de memória possível para um determinado tipo de instância. Para obter mais informações, consulte Reserva de memória da instância de contêiner do Linux no Amazon ECS.

Definições de contêiner

Ao registrar uma definição de tarefa, você deve especificar uma lista de definições de contêiner passadas para o daemon do Docker em uma instância de contêiner. Os parâmetros a seguir são permitidos em uma definição de contêiner.

Tópicos

- Parâmetros padrão de definição de contêiner
- Parâmetros avançados de definição de contêiner
- Outros parâmetros de definição de contêiner

Parâmetros padrão de definição de contêiner

Os parâmetros de definição de tarefa a seguir são obrigatórios ou usados na maioria das definições de contêiner.

Tópicos

- Name

- Imagem
- Memória
- Mapeamentos de porta
- Credenciais de repositório privado

Name

`name`

Tipo: string

Obrigatório: Sim

O nome de um contêiner. São permitidos até 255 letras (caixa alta e baixa), números, hífens e sublinhados. Se estiver vinculando vários contêineres em uma definição de tarefa, o `name` de um contêiner poderá ser informado no `links` de outro contêiner. Isso deve ser feito para conectar os contêineres.

Imagem

`image`

Tipo: string

Obrigatório: Sim

A imagem usada para iniciar um contêiner. Essa string é passada diretamente para o daemon do Docker. Por padrão, as imagens no registro do Docker Hub estão disponíveis. Você também pode especificar outros repositórios com *`repository-url`*`/`*`image`*`:`*`tag`* ou *`repository-url`*`/`*`image`*`@`*`digest`*. São permitidos até 255 letras (caixa alta e baixa), números, hifens, sublinhados, dois pontos, ponto, barras e sinais numéricos. Esse parâmetro é mapeado para `Image` no comando create-container do docker e no parâmetro `IMAGE` do comando run do docker.

- Quando uma nova tarefa é iniciada, o agente do contêiner do Amazon ECS obtém a versão mais recente da imagem especificada e a tag do contêiner a ser usado. No entanto, as atualizações subsequentes feitas em um repositório de imagens não são propagadas para tarefas já em execução.

- Quando não é especificada uma tag ou um resumo no caminho de imagem na definição da tarefa, o agente de contêiner do Amazon ECS extrai a versão mais recente da imagem especificada.
- No entanto, as atualizações subsequentes feitas em um repositório de imagens não são propagadas para tarefas já em execução.
- As imagens em registros privados são suportadas. Para obter mais informações, consulte Uso de imagens de contêiner que não são da AWS no Amazon ECS.
- As imagens nos repositórios do Amazon ECR podem ser especificadas usando a convenção de nomenclatura completa `registry/repository:tag` ou `registry/repository@digest` (por exemplo, *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`* ou *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`@`*`sha256:94afd1f2e64d908bc90dbca0035a5b567EXAMPLE`*).
- As imagens em repositórios oficiais no Docker Hub usam um único nome (por exemplo, `ubuntu` ou `mongo`).
- Imagens em outros repositórios Docker Hub são qualificadas com um nome de organização (por exemplo, `amazon/amazon-ecs-agent`).
- Imagens em outros repositórios online também são qualificadas por um nome de domínio (por exemplo, `quay.io/assemblyline/ubuntu`).

`versionConsistency`

Tipo: String

Valores válidos: `enabled|disabled`

Obrigatório: Não

Especifica se o Amazon ECS resolverá a tag de imagem do contêiner fornecida na definição do contêiner em um resumo de imagem. Por padrão, esse comportamento é `enabled`. Se você definir o valor de um contêiner como `disabled`, o Amazon ECS não resolverá a tag da imagem do contêiner como um resumo e usará o URI da imagem original especificado na definição do contêiner para implantação. Para obter mais informações sobre resolução de imagem do contêiner, consulte Resolução da imagem do contêiner.

Memória

`memory`

Tipo: inteiro

Obrigatório: Não

A quantidade (em MiB) de memória a ser apresentada ao contêiner. Caso tente exceder a memória especificada aqui, o contêiner será excluído. A quantidade total de memória reservada para todos os contêineres dentro da tarefa deve ser menor que o valor da tarefa `memory`, se estiver especificado. Esse parâmetro é mapeado para `Memory` no comando create-container do docker na opção `--memory` de execução do docker.

Você deve especificar um valor de memória ao nível de tarefa ou um valor de memória ao nível de contêiner. Se você especificar um nível de contêiner `memory` e um valor de `memoryReservation`, o valor de `memory` deverá ser maior que o valor de `memoryReservation`. Caso você especifique `memoryReservation`, o valor é subtraído dos recursos de memória disponíveis para a instância de contêiner na qual o contêiner está colocado. Caso contrário, o valor de `memory` é usado.

O daemon para Docker 20.10.0 ou versão posterior reserva no mínimo 6 MiB de memória para um contêiner. Por isso, não especifique menos de 6 MiB de memória para os seus contêineres.

O daemon para Docker 19.03.13-ce ou versão anterior reserva no mínimo 4 MiB de memória para um contêiner. Por isso, não especifique menos de 4 MiB de memória para os seus contêineres.

> **Note**
>
> Se você deseja maximizar a utilização de recursos fornecendo às tarefas o máximo de memória possível para um determinado tipo de instância, consulte Reserva de memória da instância de contêiner do Linux no Amazon ECS.

`memoryReservation`

Tipo: inteiro

Obrigatório: Não

O limite flexível (em MiB) de memória a ser reservado para o contêiner. Quando a memória do sistema estiver em contenção, o Docker tentará manter a memória do contêiner dentro desse limite flexível. Entretanto, seu contêiner pode usar mais memória quando necessário. O contêiner pode consumir até o limite rígido especificado pelo parâmetro `memory` (se aplicável), ou toda a memória disponível na instância de contêiner, o que ocorrer primeiro. Esse parâmetro é mapeado para `MemoryReservation` no comando create-container do docker na opção `--memory-reservation` de execução do docker.

Se um valor de memória no nível de tarefa não for especificado, será necessário especificar um número inteiro diferente de zero para um ou ambos `memory` ou `memoryReservation` em uma definição de contêiner. Caso você especifique ambos, `memory` deve ser maior que `memoryReservation`. Caso você especifique `memoryReservation`, o valor é subtraído dos recursos de memória disponíveis para a instância de contêiner na qual o contêiner está colocado. Caso contrário, o valor de `memory` é usado.

Por exemplo, suponha que seu contêiner normalmente use 128 MiB de memória, mas às vezes chegue a 256 MiB de memória por períodos curtos. É possível definir um `memoryReservation` de 128 MiB e um limite rígido de `memory` de 300 MiB. Essa configuração permite que contêiner reservasse apenas 128 MiB de memória dos recursos restantes na instância de contêiner. Ao mesmo tempo, essa configuração também possibilita que o contêiner utilize mais recursos de memória quando necessário.

> **Note**
>
> Este parâmetro não é compatível com contêineres do Windows.

O daemon para Docker 20.10.0 ou versão posterior reserva no mínimo 6 MiB de memória para um contêiner. Por isso, não especifique menos de 6 MiB de memória para os seus contêineres.

O daemon para Docker 19.03.13-ce ou versão anterior reserva no mínimo 4 MiB de memória para um contêiner. Por isso, não especifique menos de 4 MiB de memória para os seus contêineres.

> **Note**
>
> Se você deseja maximizar a utilização de recursos fornecendo às tarefas o máximo de memória possível para um determinado tipo de instância, consulte Reserva de memória da instância de contêiner do Linux no Amazon ECS.

Mapeamentos de porta

`portMappings`

Tipo: Matriz de objeto

Obrigatório: Não

Os mapeamentos de portas expõem as portas de rede do seu contêiner ao mundo externo. Isso permite que os clientes acessem sua aplicação. Também é usado para comunicação entre contêineres dentro da mesma tarefa.

Para definições de tarefa que usam o modo de rede `awsvpc`, apenas especifique `containerPort`. A `hostPort` é sempre ignorada e a porta do contêiner é automaticamente mapeada para uma porta aleatória de alto número no host.

Os mapeamentos de porta no Windows usam o endereço de gateway `NetNAT`, e não o `localhost`. Não há loopback para mapeamentos de porta no Windows, de modo que você não pode acessar a porta mapeada de um contêiner no próprio host.

A maioria dos campos desse parâmetro (incluindo `containerPort`, `hostPort`, `protocol`) é mapeada para `PortBindings` no comando create-container do docker e a opção `--publish` de execução do docker. Caso o modo de rede de uma definição de tarefa seja definido como `host`, as portas host devem ser indefinidas ou corresponder à porta de contêiner no mapeamento de porta.

> **Note**
>
> Quando a tarefa obter o status `RUNNING`, as atribuições automáticas e manuais da porta do contêiner e do host estarão visíveis nos seguintes locais:
>
> - Console: a seção Network Bindings (Associações de rede) de uma descrição de contêiner para uma tarefa selecionada.
> - AWS CLI: a seção `networkBindings` do resultado do comando describe-tasks.
> - API: a resposta `DescribeTasks`.
> - Metadados: o endpoint de metadados da tarefa.

`appProtocol`

Tipo: String

Obrigatório: Não

O protocolo da aplicação usado no mapeamento da porta. Esse parâmetro só se aplica ao Service Connect. Recomendamos que você defina esse parâmetro para que seja consistente com o protocolo usado por sua aplicação. Se você definir esse parâmetro, o Amazon ECS adicionará o tratamento de conexão específico do protocolo ao proxy do Service Connect. Se você definir esse parâmetro, o Amazon ECS adicionará telemetria específica de protocolo no console do Amazon ECS e no CloudWatch.

Se você não definir um valor para esse parâmetro, o TCP será usado. No entanto, o Amazon ECS não adiciona telemetria específica de protocolo para TCP.

Para obter mais informações, consulte the section called “ Service Connect”.

Valores de protocolo válidos: `"http" | "http2" | "grpc"`

`containerPort`

Tipo: inteiro

Exigido: Sim, quando `portMappings` são usados

O número da porta no contêiner vinculado à porta host atribuída automaticamente ou especificada pelo usuário.

Para as tarefas que usam o modo de rede `awsvpc`, você usa `containerPort` para especificar as portas expostas.

Suponha que você esteja usando contêineres em uma tarefa com o tipo de inicialização do EC2 e especifique uma porta de contêiner, e não uma porta de host. Em seguida, seu contêiner receberá automaticamente uma porta de host no intervalo de portas temporárias. Para obter mais informações, consulte `hostPort`. Os mapeamentos de porta atribuídos automaticamente dessa maneira não contam para a cota de 100 portas reservadas de uma instância de contêiner.

`containerPortRange`

Tipo: String

Obrigatório: Não

O intervalo de números de porta no contêiner vinculado ao intervalo de portas host mapeado dinamicamente.

Você só pode definir esse parâmetro usando a API `register-task-definition`. A opção está disponível no parâmetro `portMappings`. Para obter mais informações, consulte register-task-definition, na Referência da AWS Command Line Interface.

As regras a seguir se aplicam quando você especifica um `containerPortRange`:

- Você deve usar o modo de rede `bridge` ou o modo de rede `awsvpc`.
- Esse parâmetro está disponível para os sistemas operacionais Windows e Linux.
- A instância de contêiner deve ter pelo menos a versão 1.67.0 do agente do contêiner e pelo menos a versão 1.67.0-1 do pacote `ecs-init`
- É possível especificar no máximo 100 intervalos de portas para cada contêiner.
- Você não especifica um `hostPortRange`. O valor do `hostPortRange` é definido da seguinte forma:
  - Para contêineres em uma tarefa com o modo de rede `awsvpc`, o `hostPort` é definido com o mesmo valor que `containerPort`. Essa é uma estratégia de mapeamento estático.
  - Para contêineres em uma tarefa com o modo de rede `bridge`, o agente do Amazon ECS encontra portas de host abertas no intervalo temporário padrão e as passa ao Docker para vinculá-las às portas do contêiner.
- Os valores válidos de `containerPortRange` estão entre 1 e 65535.
- Uma porta só pode ser incluída em um mapeamento de portas para cada contêiner.
- Não é possível especificar intervalos de portas sobrepostos.
- A primeira porta no intervalo deve ser menor que a última porta no intervalo.
- O Docker recomenda que você desative o docker-proxy no arquivo de configuração do daemon do Docker quando tiver um grande número de portas.

  Para obter mais informações, consulte Issue #11185 no GitHub.

  Para obter informações sobre como desativar o docker-proxy no arquivo de configuração do daemon do Docker, consulte Daemon do Docker no Guia do desenvolvedor do Amazon ECS.

É possível chamar DescribeTasks para visualizar o `hostPortRange`, que são as portas do host vinculadas às portas do contêiner.

Os intervalos de portas não estão incluídos nos eventos de tarefas do Amazon ECS que são enviados ao EventBridge. Para obter mais informações, consulte the section called "Automatização de respostas a erros do Amazon ECS usando o EventBridge".

`hostPortRange`

Tipo: Sequência

Obrigatório: Não

O intervalo de números de portas no host que é usado com a vinculação de rede. Isso é atribuído pelo Docker e fornecido pelo agente do Amazon ECS.

`hostPort`

Tipo: inteiro

Obrigatório: Não

O número da porta na instância de contêiner a ser reservado para o contêiner.

É possível especificar uma porta de host não reservada para o mapeamento da porta do contêiner. Isso é conhecido como mapeamento estático da porta do host. Ou é possível omitir a `hostPort` (ou configurá-la como `0`) ao especificar uma `containerPort`. Seu contêiner recebe automaticamente uma porta no intervalo de portas temporárias para o sistema operacional e a versão Docker da instância de contêiner. Isso é conhecido como mapeamento dinâmico da porta do host.

O intervalo de portas temporárias padrão para a versão 1.6.0 e posterior do Docker é listado na instância em `/proc/sys/net/ipv4/ip_local_port_range`. Se esse parâmetro do kernel não estiver disponível, será usado o intervalo de portas temporárias padrão de `49153-65535`. Não tente especificar uma porta de host no intervalo de portas transitório. Isso ocorre porque essas portas são reservadas para atribuição automática. Em geral, as portas abaixo de `32768` estão fora do intervalo de portas temporárias.

As portas reservadas padrão são `22` para SSH, as portas do Docker `2375` e`2376` e as portas do agente de contêiner do Amazon ECS `51678-51680`. Qualquer porta de host que tenha sido especificada pelo usuário anteriormente para uma tarefa em execução também é reservada enquanto a tarefa está em execução. Depois que uma tarefa é interrompida, a porta do host é liberada. As portas reservadas atuais são exibidas em `remainingResources` da saída describe-container-instances. Uma instância de contêiner pode ter até 100

portas reservadas por vez, incluindo as portas reservadas padrão. As portas atribuídas automaticamente não contam em relação à cota de 100 portas reservadas.

`name`

Tipo: String

Obrigatório: não, obrigatório para que o Service Connect e o VPC Lattice sejam configurados em um serviço

Nome que é usado para o mapeamento da porta. Esse parâmetro só se aplica ao Service Connect e ao VPC Lattice. Esse parâmetro é o nome que você usa na configuração do Service Connect e do VPC Lattice de um serviço.

Para obter mais informações, consulte Uso do Service Connect para conectar serviços do Amazon ECS com nomes abreviados.

No exemplo a seguir, ambos os campos obrigatórios para o Service Connect e o VPC Lattice são usados.

```
"portMappings": [
    {
        "name": string,
        "containerPort": integer
    }
]
```

`protocol`

Tipo: String

Obrigatório: Não

O protocolo usado no mapeamento da porta. Os valores válidos são `tcp` e `udp`. O padrão é `tcp`.

> **Important**
>
> Só `tcp` é compatível com o Service Connect. Lembre-se de que `tcp` estará implícito se esse campo não estiver definido.

> **Important**
>
> O suporte a UDP só está disponível em instâncias de contêiner que foram iniciadas com a versão 1.2.0 ou posterior do agente de contêiner do Amazon ECS (como a AMI `amzn-ami-2015.03.c-amazon-ecs-optimized`) ou com agentes de contêiner que tenham sido atualizados para a versão 1.3.0 ou posterior. Para atualizar o agente de contêiner para a versão mais recente, consulte Atualizar o agente de contêiner do Amazon ECS.

Caso você esteja especificando uma porta host, use a seguinte sintaxe:

```
"portMappings": [
    {
        "containerPort": integer,
        "hostPort": integer
    }
    ...
]
```

Caso você queira uma porta host atribuída automaticamente, use a seguinte sintaxe:

```
"portMappings": [
    {
        "containerPort": integer
    }
    ...
]
```

Credenciais de repositório privado

`repositoryCredentials`

Tipo: objeto RepositoryCredentials

Obrigatório: Não

As credenciais do repositório para autenticação de registro privado.

Para obter mais informações, consulte Uso de imagens de contêiner que não são da AWS no Amazon ECS.

`credentialsParameter`

Tipo: String

Exigido: Sim, quando `repositoryCredentials` são usados

O nome de recurso da Amazon (ARN) do segredo que contém as credenciais do repositório privado.

Para obter mais informações, consulte Uso de imagens de contêiner que não são da AWS no Amazon ECS.

> **Note**
>
> Quando você usa a API do Amazon ECS, a AWS CLI ou os AWS SDKs, se o segredo existir na mesma região da tarefa que estiver sendo executada, será possível usar o ARN completo ou o nome do segredo. Ao usar o AWS Management Console, é preciso especificar o ARN completo do segredo.

Veja a seguir um trecho de uma definição de tarefa que mostra os parâmetros necessários.

```
"containerDefinitions": [
    {
        "image": "private-repo/private-image",
        "repositoryCredentials": {
            "credentialsParameter":
 "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
        }
    }
]
```

Parâmetros avançados de definição de contêiner

Os parâmetros avançados de definição de contêiner a seguir fornecem recursos estendidos ao comando run do docker usado para iniciar contêineres nas instâncias de contêiner do Amazon ECS.

Tópicos

- Política de reinicialização
- Verificação de integridade
- Environment
- Configurações de rede
- Armazenamento e registro
- Segurança
- Limites de recurso
- Rótulos do Docker

Política de reinicialização

`restartPolicy`

A política de reinicialização do contêiner e os parâmetros de configuração associados. Quando você configura uma política de reinicialização para um contêiner, o Amazon ECS pode reiniciar o contêiner sem precisar substituir a tarefa. Para obter mais informações, consulte Reiniciar contêineres individuais em tarefas do Amazon ECS com políticas de reinicialização de contêineres.

`enabled`

Tipo: booliano

Obrigatório: Sim

Especifica se uma política de reinicialização está habilitada para o contêiner.

`ignoredExitCodes`

Tipo: matriz de inteiros

Obrigatório: Não

Uma lista de códigos de saída que o Amazon ECS ignorará e não tentará reiniciar. Você pode especificar um máximo de 50 códigos de saída de contêiner. Por padrão, o Amazon ECS não ignora nenhum código de saída.

`restartAttemptPeriod`

Tipo: inteiro

Obrigatório: Não

A quantidade de tempo (em segundos) que o contêiner deve ser executado antes de uma tentativa de reinicialização. Um contêiner só pode ser reiniciado uma vez a cada `restartAttemptPeriod` segundos. Se um contêiner não puder ser executado nesse período e sair mais cedo, ele não será reiniciado. Você pode especificar um `restartAttemptPeriod` mínimo de 60 segundos e um `restartAttemptPeriod` máximo de 1800 segundos. Por padrão, um contêiner deve ser executado por 300 segundos antes de ser reiniciado.

Verificação de integridade

`healthCheck`

O comando da verificação de integridade de contêiner e os parâmetros de configuração associados para o contêiner. Para obter mais informações, consulte Como determinar a integridade das tarefas do Amazon ECS usando verificações de integridade de contêineres.

`command`

Uma matriz de strings que representa o comando executado pelo contêiner para determinar se ela está íntegra. A matriz de strings pode começar com `CMD` para executar os argumentos de comando diretamente ou `CMD-SHELL` para executar o comando com o shell padrão do contêiner. Se nenhuma for especificado, `CMD` será usada.

Ao registrar uma definição de tarefa no AWS Management Console, use uma lista de comandos separados por vírgulas. Esses comandos serão convertidos em uma string após a criação da definição de tarefa. Veja a seguir um exemplo de entrada para uma verificação de integridade.

```
CMD-SHELL, curl -f http://localhost/ || exit 1
```

Ao registrar uma definição de tarefa usando o painel JSON do AWS Management Console, a AWS CLI ou as APIs, delimite a lista de comandos por colchetes. Veja a seguir um exemplo de entrada para uma verificação de integridade.

```
[ "CMD-SHELL", "curl -f http://localhost/ || exit 1" ]
```

Um código de saída 0, sem saída de `stderr`, indica sucesso, e um código de saída diferente de zero indica falha.

`interval`

O período em segundos entre cada verificação de integridade. É possível especificar entre 5 e 300 segundos. O valor padrão de é 30 segundos.

`timeout`

O período de espera (em segundos) para que uma verificação de integridade seja bem-sucedida antes de ser considerada uma falha. É possível especificar entre 2 e 60 segundos. O valor de padrão é de 5 segundos.

`retries`

O número de novas tentativas de uma verificação de integridade com falha até o contêiner ser considerado não íntegro. É possível especificar entre 1 e 10 novas tentativas. O valor padrão é três novas tentativas.

`startPeriod`

O período de carência opcional para que os contêineres possam inicializar antes de as verificações de integridade com falha serem contabilizadas no número máximo de novas tentativas. É possível especificar entre 0 e 300 segundos. Por padrão, o `startPeriod` está desabilitado.

Se uma verificação de integridade for bem-sucedida dentro do `startPeriod`, então o contêiner é considerado saudável e quaisquer falhas subsequentes contam para o número máximo de novas tentativas.

Environment

`cpu`

Tipo: inteiro

Obrigatório: Não

O número de unidades de `cpu` que o agente de contêiner do Amazon ECS reservará para o contêiner. No Linux, esse parâmetro é mapeado como `CpuShares` na seção Criar um contêiner.

> **Note**
>
> É possível determinar o número de unidades de CPU disponíveis para cada tipo de instância do Amazon EC2. Para fazer isso, multiplique o número de vCPUs listadas para esse tipo de instância na página de detalhes Instâncias do Amazon EC2 por 1.024.

Os contêineres de Linux compartilham unidades de CPU não alocadas com outros contêineres na instância de contêiner que tem a mesma proporção que a respectiva quantidade alocada. Por exemplo, suponha que você execute uma tarefa de contêiner único em um tipo de instância de núcleo único com 512 unidades de CPU especificadas para esse contêiner. Além disso, essa tarefa é a única em execução na instância de contêiner. Nesse exemplo, o contêiner pode utilizar o compartilhamento total de 1.024 unidades de CPU a qualquer momento. Porém, suponha que você tenha iniciado outra cópia da mesma tarefa nessa instância de contêiner. Cada tarefa tem garantido um mínimo de 512 unidades de CPU quando necessário. Da mesma forma, se o outro contêiner não estiver usando a CPU restante, cada contêiner poderá flutuar para o uso maior da CPU. Porém, se ambas as tarefas estiverem 100% ativas o tempo todo, elas ficarão limitadas a 512 unidades de CPU.

Nas instâncias de contêiner de Linux, o daemon do Docker na instância de contêiner usa o valor de CPU para calcular as proporções de compartilhamento de CPU para contêineres em execução. O valor de compartilhamento de CPU mínimo válido permitido pelo kernel do Linux é 2 e o valor de compartilhamento de CPU máximo válido permitido pelo kernel do Linux é 262144. No entanto, o parâmetro de CPU não é necessário, e é possível usar valores de CPU abaixo de dois e superior a 262144 em suas definições de contêiner. Para valores de CPU abaixo de dois (inclusive nulo) e acima de 262144, o comportamento varia com base na versão do agente de contêiner do Amazon ECS:

- Versões de agente <= 1.1.0: os valores de CPU nulo e zero são passados para o Docker como 0. O Docker, em seguida, converte esse valor para 1.024 compartilhamentos de CPU. Os valores de CPU de um são passados para o Docker como um, que o kernel do Linux converte para dois compartilhamentos de CPU.
- Versões de agente >= 1.2.0: os valores nulo, zero e de CPU de 1 são passados para o Docker como dois compartilhamentos de CPU.
- Versões do agente >= 1.84.0: valores de CPU maiores que 256 vCPU são passados para Docker como 256, o que equivale a 262144 compartilhamentos de CPU.

Nas instâncias de contêiner de Windows, a cota de CPU é imposta como uma cota absoluta. Os contêineres de Windows só têm acesso à quantidade de CPU especificada na definição de tarefa. Um valor de CPU nulo ou zero é transmitido para Docker como `0`. Em seguida, Windows interpreta esse valor como 1% de uma CPU.

Para obter exemplos adicionais, consulte Como o Amazon ECS gerencia recursos de CPU e memória.

`gpu`

Tipo: objeto ResourceRequirement

Obrigatório: Não

O número de GPUs físicas que o agente de contêiner do Amazon ECS reservará para o contêiner. O número de GPUs reservadas para todos os contêineres em uma tarefa não deve exceder o número de GPUs disponíveis na instância de contêiner na qual a tarefa é executada. Para obter mais informações, consulte Definições de tarefa do Amazon ECS para workloads de GPU.

> **Note**
>
> Não há suporte para este parâmetro em contêineres do Windows.

`Elastic Inference accelerator`

Tipo: objeto ResourceRequirement

Obrigatório: Não

Para o tipo `InferenceAccelerator`, o `value` corresponde ao `deviceName` para um `InferenceAccelerator` especificado em uma definição de tarefa. Para obter mais informações, consulte the section called “Nome do acelerador de inferência elástica”.

> **Note**
>
> Não há suporte para este parâmetro em contêineres do Windows.

`essential`

Tipo: booliano

Obrigatório: Não

Suponha que o parâmetro `essential` de um contêiner esteja marcado como `true`, e que esse contêiner falhe ou seja interrompido por algum motivo. Em seguida, todos os outros contêineres que fazem parte da tarefa são interrompidos. Caso o parâmetro `essential` de um contêiner esteja marcado como `false`, a falha não afeta o restante dos contêineres em uma tarefa. Caso esse parâmetro seja omitido, um contêiner pressupõe-se que um contêiner seja essencial.

Todas as tarefas devem ter pelo menos um contêiner essencial. Suponha que você tenha uma aplicação composta por vários contêineres. Em seguida, agrupe os contêineres usados para uma finalidade em comum em componentes e separe os componentes diferentes em várias definições de tarefa. Para obter mais informações, consulte Projetar sua aplicação para o Amazon ECS.

```
"essential": true|false
```

`entryPoint`

> **Important**
>
> As versões anteriores do agente de contêiner do Amazon ECS não lidam corretamente com parâmetros `entryPoint`. Caso você enfrente problemas para usar `entryPoint`, atualize o agente de contêiner ou informe os comandos e os argumentos como itens de matriz `command` em vez disso.

Tipo: Matriz de strings

Obrigatório: Não

O ponto de entrada passado para o contêiner.

```
"entryPoint": ["string", ...]
```

`command`

Tipo: Matriz de strings

Obrigatório: Não

O comando que é passado para o contêiner. Esse parâmetro é mapeado para `Cmd` no comando create-container e o parâmetro `COMMAND` de execução do docker. Se houver vários argumentos, certifique-se de que cada um seja uma string separada na matriz.

```
"command": ["string", ...]
```

`workingDirectory`

Tipo: String

Obrigatório: Não

O diretório de trabalho no qual executar comandos no contêiner. Esse parâmetro é mapeado para `WorkingDir` na seção Criar um contêiner da Docker Remote API e a opção `--workdir` para docker run.

```
"workingDirectory": "string"
```

`environmentFiles`

Tipo: Matriz de objeto

Obrigatório: Não

Uma lista de arquivos que contêm as variáveis de ambiente a serem passadas para um contêiner. Esse parâmetro é mapeado para a opção `--env-file` para o comando run do docker.

Quando o FIPS está habilitado, nomes de bucket que contêm pontos (.) (por exemplo, amzn-s3-demo-bucket1.name.example) não são compatíveis. Ter pontos (.) no nome do bucket impede que a tarefa seja iniciada porque o agente não consegue extrair o arquivo da variável de ambiente do Amazon S3.

Isso não está disponível para contêineres Windows.

É possível especificar até 10 arquivos de ambiente. O arquivo deve ter uma extensão `.env`. Cada linha em um arquivo de ambiente contém uma variável de ambiente no formato `VARIABLE=VALUE`. As linhas que começam com # são tratadas como comentários e são ignoradas.

Se houver variáveis de ambiente individuais especificadas na definição de contêiner, elas terão precedência sobre as variáveis contidas em um arquivo de ambiente. Se forem especificados

vários arquivos de ambiente que contenham a mesma variável, elas serão processadas de cima para baixo. Recomendamos usar nomes de variáveis exclusivos. Para obter mais informações, consulte Transferência de uma variável de ambiente individual para um contêiner do Amazon ECS.

`value`

Tipo: string

Obrigatório: Sim

O nome de recurso da Amazon (ARN) do objeto do Amazon S3 que contém o arquivo de variáveis de ambiente.

`type`

Tipo: string

Obrigatório: Sim

Tipo de arquivo a ser usado O único valor aceito é `s3`.

`environment`

Tipo: Matriz de objeto

Obrigatório: Não

As variáveis de ambiente a serem passadas para um contêiner. Esse parâmetro é mapeado para `Env` no comando create-container do docker na opção `--env` para o comando run do docker.

> **Important**
>
> Não recomendamos o uso de variáveis de ambiente de texto simples para informações confidenciais, tais como dados de credenciais.

`name`

Tipo: String

Exigido: sim, quando `environment` for usado

O nome da variável de ambiente.

`value`

Tipo: String

Exigido: sim, quando `environment` for usado

O valor da variável de ambiente.

```
"environment" : [
    { "name" : "string", "value" : "string" },
    { "name" : "string", "value" : "string" }
]
```

`secrets`

Tipo: Matriz de objeto

Obrigatório: Não

Um objeto que representa o segredo a ser exposto ao seu contêiner. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.

`name`

Tipo: string

Obrigatório: Sim

O valor a ser definido como a variável de ambiente no contêiner.

`valueFrom`

Tipo: string

Obrigatório: Sim

O segredo a ser exposto ao contêiner. Os valores com suporte são o nome do recurso da Amazon (ARN) completo do segredo do AWS Secrets Manager ou o ARN completo do parâmetro no Parameter Store do AWS Systems Manager.

> **Note**
>
> Se o parâmetro do Systems Manager Parameter Store ou do Secrets Manager existir na mesma Região da AWS que a tarefa sendo iniciada, você poderá usar o ARN

completo ou o nome do segredo. Se o parâmetro existir em uma Região diferente, o ARN completo deverá ser especificado.

```
"secrets": [
    {
        "name": "environment_variable_name",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }
]
```

Configurações de rede

`disableNetworking`

Tipo: booliano

Obrigatório: Não

Quando esse parâmetro é verdadeiro, a rede é desativada dentro do contêiner.

> **Note**
>
> Não há suporte para esse parâmetro em contêineres do Windows ou tarefas que usem o modo de rede `awsvpc`.

O padrão é `false`.

```
"disableNetworking": true|false
```

`links`

Tipo: Matriz de strings

Obrigatório: Não

O parâmetro `link` permite que os contêineres se comuniquem entre si sem a necessidade de mapeamentos de porta. Só há suporte para esse parâmetro se o modo de rede de uma definição de tarefa estiver definido como `bridge`. O construto `name:internalName` é igual ao

`name:alias` em links do Docker. São permitidos até 255 letras (caixa alta e baixa), números, hífens e sublinhados.

> **Note**
>
> Não há suporte para esse parâmetro em contêineres do Windows ou tarefas que usem o modo de rede `awsvpc`.

> **Important**
>
> Os contêineres colocados na mesma instância de contêiner podem se comunicar entre si sem exigir links ou mapeamentos de porta de host. O isolamento de rede em uma instância de contêiner é controlado por grupos de segurança e configurações de VPC.

```
"links": ["name:internalName", ...]
```

`hostname`

Tipo: String

Obrigatório: Não

O nome de host a ser usado para o contêiner. Esse parâmetro é mapeado para `Hostname` em create-container do docker na opção `--hostname` de execução do docker.

> **Note**
>
> Se você estiver usando o modo de rede `awsvpc`, o parâmetro `hostname` não terá suporte.

```
"hostname": "string"
```

`dnsServers`

Tipo: Matriz de strings

Obrigatório: Não

Uma lista de servidores DNS apresentados ao contêiner.

> **Note**
>
> Não há suporte para esse parâmetro em contêineres do Windows ou tarefas que usem o modo de rede `awsvpc`.

```
"dnsServers": ["string", ...]
```

`dnsSearchDomains`

Tipo: Matriz de strings

Obrigatório: Não

Padrão: ^[a-zA-Z0-9-.]{0,253}[a-zA-Z0-9]$

Uma lista de domínios de pesquisa DNS apresentados ao contêiner. Esse parâmetro é mapeado para `DnsSearch` no comando create-container do docker na opção `--dns-search` de execução do docker.

> **Note**
>
> Esse parâmetro não tem suporte para contêineres do Windows ou tarefas que usam o modo de rede `awsvpc`.

```
"dnsSearchDomains": ["string", ...]
```

`extraHosts`

Tipo: Matriz de objeto

Obrigatório: Não

Uma lista de nomes de host e mapeamentos de endereço IP a ser acrescentada ao arquivo `/etc/hosts` no contêiner.

Esse parâmetro é mapeado para `ExtraHosts` no comando create-container do docker na opção `--add-host` de execução do docker.

> **Note**
>
> Esse parâmetro não tem suporte para contêineres do Windows ou tarefas que usam o modo de rede `awsvpc`.

```
"extraHosts": [
      {
        "hostname": "string",
        "ipAddress": "string"
      }
      ...
    ]
```

`hostname`

Tipo: String

Exigido: Sim, quando `extraHosts` são usados

O nome do host a ser usado na entrada `/etc/hosts`.

`ipAddress`

Tipo: String

Exigido: Sim, quando `extraHosts` são usados

O endereço IP a ser usado na entrada `/etc/hosts`.

Armazenamento e registro

`readonlyRootFilesystem`

Tipo: booliano

Obrigatório: Não

Quando esse parâmetro é verdadeiro, o contêiner recebe acesso somente leitura ao sistema de arquivos raiz. Esse parâmetro é mapeado para `ReadonlyRootfs` no comando create-container do docker na opção `--read-only` de execução do docker.

> **Note**
>
> Este parâmetro não é compatível com contêineres do Windows.

O padrão é `false`.

```
"readonlyRootFilesystem": true|false
```

`mountPoints`

Tipo: Matriz de objeto

Obrigatório: Não

Os pontos de montagem dos volumes de dados no contêiner. Esse parâmetro é mapeado em `Volumes` na API create-container do Docker na opção `--volume` de execução do docker.

Os contêineres do Windows podem montar diretórios inteiros na mesma unidade como `$env:ProgramData`. Os contêineres do Windows não podem montar diretórios em uma unidade diferente, e os pontos de montagem não podem ser usados entre unidades. Você deve especificar pontos de montagem para anexar um volume do Amazon EBS diretamente a uma tarefa do Amazon ECS.

`sourceVolume`

Tipo: String

Exigido: Sim, quando `mountPoints` são usados

O nome do volume a ser montado.

`containerPath`

Tipo: String

Exigido: Sim, quando `mountPoints` são usados

O caminho no contêiner onde o volume será montado.

`readOnly`

Tipo: booliano

Obrigatório: Não

Caso o valor seja `true`, o contêiner tem acesso somente leitura ao volume. Caso esse valor seja `false`, o contêiner pode gravar no volume. O valor padrão é `false`.

Em tarefas que executam o sistema operacional Windows, não modifique o valor padrão `false`.

`volumesFrom`

Tipo: Matriz de objeto

Obrigatório: Não

Volumes de dados a serem montados de outro contêiner. Esse parâmetro é mapeado para `VolumesFrom` no comando create-container do docker na opção `--volumes-from` de execução do docker.

`sourceContainer`

Tipo: String

Exigido: sim, quando `volumesFrom` for usado

O nome do contêiner com base no qual montar volumes.

`readOnly`

Tipo: booliano

Obrigatório: Não

Caso o valor seja `true`, o contêiner tem acesso somente leitura ao volume. Caso esse valor seja `false`, o contêiner pode gravar no volume. O valor padrão é `false`.

```
"volumesFrom": [
                {
                  "sourceContainer": "string",
                  "readOnly": true|false
                }
              ]
```

`logConfiguration`

Tipo: objeto LogConfiguration

Obrigatório: Não

A especificação de configuração do log para o contêiner.

Para obter definições de tarefa de exemplo que usam uma configuração de log, consulte Exemplos de definições de tarefa do Amazon ECS.

Esse parâmetro é mapeado para `LogConfig` no comando create-container do docker na opção `--log-driver` de execução do docker. Por padrão, os contêineres usam o mesmo driver de registro em log que o daemon do Docker usa. No entanto, o contêiner pode usar um driver de registro em log diferente do daemon do Docker especificando um driver de log com esse parâmetro na definição de contêiner. Para usar um driver de registro em log diferente para um contêiner, o sistema de log deve ser configurado corretamente na instância de contêiner (ou em um servidor de log diferente para opções de registro em log remotas).

Considere o seguinte ao especificar uma configuração de log para seus contêineres:

- O Amazon ECS oferece suporte a um subconjunto dos drivers de registro em log disponíveis para o daemon Docker. Os drivers de log adicionais podem estar disponíveis em versões futuras do agente de contêiner do Amazon ECS.
- Esse parâmetro exige a versão 1.18 ou posterior da API remota do Docker na sua instância de contêiner.
- O agente de contêiner do Amazon ECS executado em uma instância de contêiner deve registrar os drivers de log disponíveis na instância com a variável de ambiente `ECS_AVAILABLE_LOGGING_DRIVERS` para que os contêineres colocados nessa instância possam usar essas opções de configuração de log. Para obter mais informações, consulte Configuração do agente de contêiner do Amazon ECS.

```
"logConfiguration": {
      "logDriver": "awslogs","fluentd","gelf","json-
file","journald","splunk","syslog","awsfirelens",
      "options": {"string": "string"
        ...},
 "secretOptions": [{
  "name": "string",
  "valueFrom": "string"
 }]
```

```
}
```

`logDriver`

Tipo: String

Valores válidos: `"awslogs","fluentd","gelf","json-file","journald","splunk","syslog","awsfirelens"`

Exigido: sim, quando `logConfiguration` for usado

O driver de log a ser usado para o contêiner. Por padrão, os valores válidos listados anteriormente são drivers de log com os quais o agente de contêiner do Amazon ECS pode se comunicar.

Os drivers de log compatíveis são `awslogs`, `fluentd`, `gelf`, `json-file`, `journald`, `syslog`, `splunk` e `awsfirelens`.

Para obter mais informações sobre como usar o driver de log `awslogs` nas definições de tarefa para enviar os logs de contêiner ao CloudWatch Logs, consulte Envio de logs do Amazon ECS para o CloudWatch .

Para obter mais informações sobre como usar o driver de log `awsfirelens`, consulte Roteamento de logs personalizados.

> **Note**
>
> Se tiver um driver personalizado que não esteja listado acima, será possível bifurcar o projeto do agente de contêiner do Amazon ECS que está disponível no GitHub e personalizá-lo para funcionar com esse driver. Incentivamos você a enviar solicitações pull para alterações que você gostaria de ter incluído. Porém, no momento, não oferecemos suporte à execução de cópias modificadas desse software.

Este parâmetro requer a versão 1.18 da Docker Remote API ou posterior em sua instância de contêiner.

`options`

Tipo: mapa de string para string

Obrigatório: Não

O mapa chave/valor de opções de configuração a serem enviadas para o driver de log.

As opções que você pode especificar dependem do driver de log. Algumas das opções que você pode especificar ao usar o roteador `awslogs` para rotear os logs para o Amazon CloudWatch incluem o seguinte:

`awslogs-create-group`

Obrigatório: Não

Especifique se você deseja que o grupo de logs seja criado automaticamente. Se essa opção não for especificada, o padrão será `false`.

> **Note**
>
> Sua política do IAM deve incluir a permissão `logs:CreateLogGroup` antes que você tente usar `awslogs-create-group`.

`awslogs-region`

Obrigatório: Sim

Especifique a Região da AWS para a qual o driver de log do `awslogs` deve enviar seus logs do Docker. É possível optar por enviar todos os logs de clusters em regiões distintas para uma única região no CloudWatch Logs. Isso tem o objetivo de fazer com que todos fiquem visíveis em um único local. Caso contrário, você pode separá-los por região para mais granularidade. Certifique-se de que o grupo de logs especificado exista na região especificada com essa opção.

`awslogs-group`

Obrigatório: Sim

Certifique-se de especificar um grupo de logs para o qual o driver de log do `awslogs` envia seus fluxos de logs.

`awslogs-stream-prefix`

Obrigatório: opcional

Use a opção `awslogs-stream-prefix` para associar um fluxo de log ao prefixo especificado, ao nome do contêiner e ao ID da tarefa do Amazon ECS à qual o contêiner

pertence. Caso você especifique um prefixo com essa opção, o fluxo de log utiliza o seguinte formato.

```
prefix-name/container-name/ecs-task-id
```

Se você não especificar um prefixo com essa opção, o fluxo de log será nomeado depois que o ID do contêiner for atribuído pelo daemon do Docker na instância de contêiner. Como é difícil rastrear logs até o contêiner que os enviou apenas com o ID do contêiner do Docker (disponível apenas na instância de contêiner), recomendamos especificar um prefixo com essa opção.

Para serviços do Amazon ECS, é possível utilizar o nome do serviço como prefixo. Ao fazer isso, será possível rastrear fluxos de logs até o serviço ao qual o contêiner pertence, o nome do contêiner que os enviou e o ID da tarefa à qual o contêiner pertence.

Você deve especificar um stream-prefix para seus logs para fazer com que eles apareçam no painel Logs quando o console do Amazon ECS for usado.

`awslogs-datetime-format`

Obrigatório: Não

Essa opção define um padrão de início de várias linhas no formato `strftime` em Python. Uma mensagem de log é formada por uma linha em conformidade com o padrão e as linhas seguintes que não correspondem ao padrão. A linha em conformidade é o delimitador entre as mensagens de log.

Um exemplo de um caso de uso para esse formato é a análise da saída, como um despejo de pilha, que poderia ser registrado em várias entradas. O padrão correto permite que ele seja capturado em uma única entrada.

Para obter mais informações, consulte awslogs-datetime-format.

Você não pode configurar as opções `awslogs-datetime-format` e `awslogs-multiline-pattern`.

> **Note**
>
> O registro em várias linhas executa a análise da expressão regular e a correspondência de todas as mensagens de log. Isso pode ter um impacto negativo na performance do registro em log.

`awslogs-multiline-pattern`

Obrigatório: Não

Essa opção define um padrão inicial de várias linhas que usa uma expressão regular. Uma mensagem de log é formada por uma linha em conformidade com o padrão e as linhas seguintes que não correspondem ao padrão. A linha em conformidade é o delimitador entre as mensagens de log.

Para obter mais informações, consulte awslogs-multiline-pattern.

Essa opção será ignorada se `awslogs-datetime-format` também estiver configurado.

Você não pode configurar as opções `awslogs-datetime-format` e `awslogs-multiline-pattern`.

> **Note**
>
> O registro em várias linhas executa a análise da expressão regular e a correspondência de todas as mensagens de log. Isso pode ter um impacto negativo na performance do registro em log.

As opções a seguir se aplicam a todos os drivers de logs compatíveis.

`mode`

Obrigatório: Não

Valores válidos: `non-blocking` | `blocking`

Essa opção define o modo de entrega de mensagens de log do contêiner para o driver de logs especificado usando o `logDriver`. O modo de entrega escolhido afeta a disponibilidade da aplicação quando o fluxo de logs do contêiner é interrompido.

Se você usar o modo `blocking` e o fluxo de logs for interrompido, as chamadas do código do contêiner para gravar nos fluxos `stdout` e `stderr` serão bloqueadas. Como resultado, o encadeamento de logs da aplicação será bloqueado. Isso pode fazer com que a aplicação pare de responder e causar uma falha na verificação de integridade do contêiner.

Se você usar o modo `non-blocking`, os registros do contêiner serão armazenados em um buffer intermediário na memória configurado com a opção `max-buffer-size`. Isso evita que a aplicação pare de responder quando não for possível enviar os logs. Recomendamos usar esse modo se você quiser garantir a disponibilidade do serviço e aceitar alguma perda de logs. Para obter mais informações, consulte Preventing log loss with non-blocking mode in the `awslogs` container log driver.

Você pode definir um `mode` padrão para todos os contêineres em uma determinada Região da AWS usando a configuração de conta `defaultLogDriverMode`. Se você não especificar a opção de `mode` na `logConfiguration` nem definir a configuração de conta, o Amazon ECS usará o modo `non-blocking` como padrão. Para obter mais informações sobre a configuração de conta, consulte Modo de driver de logs padrão.

Quando o modo `non-blocking` é usado, a opção de log `max-buffer-size` controla o tamanho do buffer de anel usado para armazenamento de mensagens intermediárias. Certifique-se de especificar um tamanho de buffer adequado com base na sua aplicação. A quantidade total de memória alocada no nível da tarefa deve ser maior que a quantidade de memória alocada para todos os contêineres, mais o limite de buffer da memória.

> **Note**
>
> Em 25 de junho de 2025, o Amazon ECS alterou o modo padrão do driver de log de `blocking` para `non-blocking` a fim de priorizar a disponibilidade das tarefas em relação ao registro em log. Para continuar usando o modo `blocking` após essa alteração, siga uma destas opções:
>
> - Defina a opção `mode` na definição `logConfiguration` do seu contêiner como `blocking`.
> - Defina a configuração da conta `defaultLogDriverMode` como `blocking`.

`max-buffer-size`

Obrigatório: Não

Valor padrão: `10 m`

Quando o modo `non-blocking` é usado, a opção de log `max-buffer-size` controla o tamanho do buffer de anel usado para armazenamento de mensagens intermediárias. Certifique-se de especificar um tamanho de buffer adequado com base na sua aplicação.

Quando o buffer ficar cheio, logs adicionais não poderão ser armazenados. Os logs que não puderem ser armazenados serão perdidos.

Para rotear logs usando o roteador de logs do `splunk`, você precisa especificar um `splunk-token` e um `splunk-url`.

Ao usar o roteador de log `awsfirelens` para rotear logs para um destino do AWS service (Serviço da AWS) ou para armazenamento e analytics de logs do AWS Partner Network, você pode definir a opção `log-driver-buffer-limit` para limitar o número de eventos que são armazenados em buffer na memória antes de serem enviados ao contêiner do roteador de logs. Isso pode ajudar a resolver possíveis problemas de perda de logs, pois o alto throughput pode resultar na falta de memória para o buffer dentro do Docker. Para obter mais informações, consulte the section called “Configuração de logs para obtenção de alto throughput”.

Outras opções que você pode especificar ao usar o `awsfirelens` para rotear logs dependem do destino. Ao exportar logs para o Amazon Data Firehose, você pode especificar a Região da AWS com a `region` e um nome para o fluxo de logs com `delivery_stream`.

Ao exportar logs para o Amazon Kinesis Data Streams, você pode especificar um Região da AWS com a `region` um com e um nome de fluxo de dados com `stream`.

Ao exportar logs para o Amazon OpenSearch Service, você pode especificar opções como `Name`, `Host` (endpoint do OpenSearch Service sem protocolo) `Port`, `Index`, `Type`, `Aws_auth`, `Aws_region`, `Suppress_Type_Name` e `tls`.

Ao exportar logs para o Amazon S3, você pode especificar o bucket usando a opção `bucket`. Você também pode especificar `region`, `total_file_size`, `upload_timeout` e `use_put_object` como opções.

Este parâmetro requer a versão 1.19 da Docker Remote API ou posterior em sua instância de contêiner.

`secretOptions`

Tipo: Matriz de objeto

Obrigatório: Não

Um objeto que representa o segredo a ser passado para a configuração de log. Os segredos usados na configuração de log podem incluir tokens de autenticação, certificados ou chaves

de criptografia. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.

`name`

Tipo: string

Obrigatório: Sim

O valor a ser definido como a variável de ambiente no contêiner.

`valueFrom`

Tipo: string

Obrigatório: Sim

O segredo a ser exposto à configuração de log do contêiner.

```
"logConfiguration": {
 "logDriver": "splunk",
 "options": {
  "splunk-url": "https://cloud.splunk.com:8080",
  "splunk-token": "...",
  "tag": "...",
  ...
 },
 "secretOptions": [{
  "name": "splunk-token",
  "valueFrom": "/ecs/logconfig/splunkcred"
 }]
}
```

`firelensConfiguration`

Tipe: objeto FirelensConfiguration

Obrigatório: Não

A configuração do FireLens para o contêiner. Isso é usado para especificar e configurar um roteador de log para logs de contêiner. Para obter mais informações, consulte Envio de logs do Amazon ECS para um serviço da AWS ou para uma AWS Partner.

```
{
```

```
    "firelensConfiguration": {
        "type": "fluentd",
        "options": {
            "KeyName": ""
        }
    }
}
```

`options`

Tipo: Mapa de string para string

Obrigatório: Não

O mapa chave/valor de opções a serem usadas ao configurar o roteador de log. Esse campo é opcional e pode ser usado para especificar um arquivo de configuração personalizado ou para adicionar outros metadados, como a tarefa, a definição de tarefa, o cluster e detalhes da instância de contêiner ao evento de log. Se especificado, a sintaxe a ser usada é `"options":{"enable-ecs-log-metadata":"true|false","config-file-type:"s3|file","config-file-value":"arn:aws:s3:::amzn-s3-demo-bucket/fluent.conf|filepath"}`. Para obter mais informações, consulte Exemplo de definição de tarefa do Amazon ECS: rotear logs para o FireLens.

`type`

Tipo: string

Obrigatório: Sim

O roteador de log a ser usado. Os valores válidos são `fluentd` ou `fluentbit`.

Segurança

Para obter mais informações sobre a segurança de contêineres, consulte Práticas recomendadas de segurança de tarefas e contêineres no Amazon ECS.

`credentialSpecs`

Tipo: Matriz de strings

Obrigatório: Não

Uma lista de ARNs no SSM ou no Amazon S3 para um arquivo de especificação de credencial (`CredSpec`) que configura o contêiner para autenticação do Active Directory. Recomendamos o uso deste parâmetro, em vez de `dockerSecurityOptions`. O número máximo de ARNs é 1.

Há dois formatos para cada ARN.

credentialspecdomainless:MyARN

Você usa `credentialspecdomainless:MyARN` para fornecer uma `CredSpec` com uma seção adicional para um segredo no Secrets Manager. Você fornece as credenciais de login para o domínio no segredo.

Cada tarefa executada em qualquer instância de contêiner pode se associar a domínios diferentes.

É possível usar esse formato sem associar a instância de contêiner a um domínio.

credentialspec:MyARN

Você usa `credentialspec:MyARN` para fornecer um `CredSpec` para um único domínio.

Você deve associar a instância de contêiner ao domínio antes de iniciar qualquer tarefa que use essa definição de tarefa.

Em ambos os formatos, substitua `MyARN` pelo ARN no SSM ou no Amazon S3.

A `credspec` deve fornecer um ARN no Secrets Manager para um segredo contendo o nome de usuário, a senha e o domínio ao qual se conectar. Para maior segurança, a instância não é associada ao domínio para autenticação sem domínio. Outras aplicações na instância não podem usar as credenciais sem domínio. É possível usar esse parâmetro para executar tarefas na mesma instância, mesmo que as tarefas precisem ser associadas a domínios diferentes. Para obter mais informações, consulte Uso de gMSAs para contêineres de Windows e Uso de gMSAs para contêineres de Linux.

`privileged`

Tipo: booliano

Obrigatório: Não

Quando esse parâmetro é verdadeiro, o contêiner recebe privilégios elevados na instância de contêiner host (semelhante ao usuário `root`). Não recomendamos operar contêineres com `privileged`. Na maioria dos casos, é possível especificar os privilégios exatos necessários usando os parâmetros específicos em vez de `privileged`.

Esse parâmetro é mapeado para `Privileged` no comando create-container do docker na opção `--privileged` de execução do docker.

> **Note**
>
> Esse parâmetro não é compatível com os contêineres do Windows ou tarefas que usam o tipo de inicialização Fargate.

O padrão é `false`.

```
"privileged": true|false
```

`user`

Tipo: String

Obrigatório: Não

O usuário a ser utilizado dentro do contêiner. Esse parâmetro é mapeado para `User` no comando create-container do docker na opção `--user` de execução do docker.

> **Important**
>
> Ao executar tarefas que usam o modo de rede `host`, não execute contêineres usando o usuário raiz (UID 0). Como uma prática recomendada de segurança, sempre utilize um usuário não raiz.

É possível especificar o `user` usando os formatos a seguir. Se especificar um UID ou um GID, você deverá especificá-lo como um número inteiro positivo.

- `user`
- `user:group`
- `uid`
- `uid:gid`
- `user:gid`
- `uid:group`

> **Note**
>
> Este parâmetro não é compatível com contêineres do Windows.

```
"user": "string"
```

`dockerSecurityOptions`

Tipo: matriz de strings

Valores válidos: "no-new-privileges" | "apparmor:PROFILE" | "label:*value*" | "credentialspec:*CredentialSpecFilePath*"

Obrigatório: Não

Uma lista de strings para fornecer configuração personalizada para vários sistemas de segurança.

Em tarefas do Linux, esse parâmetro pode ser usado para referenciar rótulos personalizados para sistemas de segurança de vários níveis SELinux e AppArmor .

Esse parâmetro pode ser usado para referenciar um arquivo de especificação de credenciais que configura um contêiner para autenticação do Active Directory. Para obter mais informações, consulte Saiba como usar gMSAs para contêineres do Windows do Amazon EC2 para o Amazon ECS e Usar gMSA para contêineres do Linux do EC2 no Amazon ECS.

Esse parâmetro é mapeado para `SecurityOpt` no comando create-container do docker na opção `--security-opt` de execução do docker.

```
"dockerSecurityOptions": ["string", ...]
```

> **Note**
>
> O agente de contêiner do Amazon ECS em execução em uma instância de contêiner deve se registrar com as variáveis de ambiente `ECS_SELINUX_CAPABLE=true` ou `ECS_APPARMOR_CAPABLE=true` para contêineres colocados nessa instância possam usar essas opções de segurança. Para obter mais informações, consulte Configuração do agente de contêiner do Amazon ECS.

Limites de recurso

`ulimits`

Tipo: Matriz de objeto

Obrigatório: Não

Lista de valores `ulimit` a serem definidos para um contêiner. Esse valor substitui a configuração de cota de recursos padrão do sistema operacional. Esse parâmetro é mapeado para `Ulimits` no comando create-container do docker na opção `--ulimit` de execução do docker.

Este parâmetro requer a versão 1.18 da Docker Remote API ou posterior em sua instância de contêiner.

> **Note**
>
> Este parâmetro não é compatível com contêineres do Windows.

```
"ulimits": [
      {
        "name":
 "core"|"cpu"|"data"|"fsize"|"locks"|"memlock"|"msgqueue"|"nice"|"nofile"|"nproc"|"rss"|"rtp
        "softLimit": integer,
        "hardLimit": integer
      }
      ...
    ]
```

`name`

Tipo: String

Valores válidos: `"core" | "cpu" | "data" | "fsize" | "locks" | "memlock" | "msgqueue" | "nice" | "nofile" | "nproc" | "rss" | "rtprio" | "rttime" | "sigpending" | "stack"`

Exigido: Sim, quando `ulimits` são usados

O `type` do `ulimit`.

`hardLimit`

Tipo: inteiro

Exigido: Sim, quando `ulimits` são usados

O limite rígido do tipo `ulimit`. O valor pode ser especificado em bytes, segundos ou como uma contagem, dependendo `type` do `ulimit`.

`softLimit`

Tipo: inteiro

Exigido: Sim, quando `ulimits` são usados

O limite flexível do tipo `ulimit`. O valor pode ser especificado em bytes, segundos ou como uma contagem, dependendo `type` do `ulimit`.

Rótulos do Docker

`dockerLabels`

Tipo: mapa de string para string

Obrigatório: Não

Um mapa de chave/valor de rótulos a ser adicionado ao contêiner. Esse parâmetro é mapeado para `Labels` no comando create-container do docker na opção `--label` de execução do docker.

Este parâmetro requer a versão 1.18 da Docker Remote API ou posterior em sua instância de contêiner.

```
"dockerLabels": {"string": "string"
      ...}
```

Outros parâmetros de definição de contêiner

Os parâmetros de definição de contêiner a seguir podem ser usados durante o registro das definições de tarefa no console do Amazon ECS usando a opção Configure via JSON (Configurar via

JSON). Para obter mais informações, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

Tópicos

- Parâmetros do Linux
- Dependência de contêiner
- Tempos limite de contêiner
- Controles do sistema
- Interativo
- Pseudoterminal

Parâmetros do Linux

`linuxParameters`

Tipo: Objeto LinuxParameters

Obrigatório: Não

Opções específicas do Linux que são aplicadas ao contêiner, como KernelCapabilities.

> **Note**
>
> Não há suporte para este parâmetro em contêineres do Windows.

```
"linuxParameters": {
      "capabilities": {
        "add": ["string", ...],
        "drop": ["string", ...]
        }
      }
```

`capabilities`

Tipo: objeto KernelCapabilities

Obrigatório: Não

Os recursos do Linux para o contêiner que são adicionados ou descartados da configuração padrão fornecida pelo Docker. Para obter mais informações sobre esses recursos do Linux, consulte a página do manual do Linux recursos(7).

`add`

Tipo: matriz de strings

Valores válidos: `"ALL" | "AUDIT_CONTROL" | "AUDIT_READ" | "AUDIT_WRITE" | "BLOCK_SUSPEND" | "CHOWN" | "DAC_OVERRIDE" | "DAC_READ_SEARCH" | "FOWNER" | "FSETID" | "IPC_LOCK" | "IPC_OWNER" | "KILL" | "LEASE" | "LINUX_IMMUTABLE" | "MAC_ADMIN" | "MAC_OVERRIDE" | "MKNOD" | "NET_ADMIN" | "NET_BIND_SERVICE" | "NET_BROADCAST" | "NET_RAW" | "SETFCAP" | "SETGID" | "SETPCAP" | "SETUID" | "SYS_ADMIN" | "SYS_BOOT" | "SYS_CHROOT" | "SYS_MODULE" | "SYS_NICE" | "SYS_PACCT" | "SYS_PTRACE" | "SYS_RAWIO" | "SYS_RESOURCE" | "SYS_TIME" | "SYS_TTY_CONFIG" | "SYSLOG" | "WAKE_ALARM"`

Obrigatório: Não

Os recursos do Linux para o contêiner que são adicionados à configuração padrão fornecida pelo Docker. Esse parâmetro é mapeado para `CapAdd` no comando create-container do docker na opção `--cap-add` de execução do docker.

`add`

Tipo: matriz de strings

Valores válidos: `"SYS_PTRACE"`

Obrigatório: Não

Os recursos do Linux para o contêiner que são adicionados à configuração padrão que é fornecida pelo Docker. Esse parâmetro é mapeado para `CapAdd` no comando create-container do docker na opção `--cap-add` de execução do docker.

`drop`

Tipo: matriz de strings

Valores válidos: `"ALL" | "AUDIT_CONTROL" | "AUDIT_WRITE" | "BLOCK_SUSPEND" | "CHOWN" | "DAC_OVERRIDE" | "DAC_READ_SEARCH" | "FOWNER" | "FSETID" | "IPC_LOCK" | "IPC_OWNER" | "KILL" | "LEASE"`

```
| "LINUX_IMMUTABLE" | "MAC_ADMIN" | "MAC_OVERRIDE" | "MKNOD" |
"NET_ADMIN" | "NET_BIND_SERVICE" | "NET_BROADCAST" | "NET_RAW"
| "SETFCAP" | "SETGID" | "SETPCAP" | "SETUID" | "SYS_ADMIN" |
"SYS_BOOT" | "SYS_CHROOT" | "SYS_MODULE" | "SYS_NICE" | "SYS_PACCT"
| "SYS_PTRACE" | "SYS_RAWIO" | "SYS_RESOURCE" | "SYS_TIME" |
"SYS_TTY_CONFIG" | "SYSLOG" | "WAKE_ALARM"
```

Obrigatório: Não

Os recursos do Linux para o contêiner que são removidos da configuração padrão fornecida pelo Docker. Esse parâmetro é mapeado para `CapDrop` no comando create-container do docker na opção `--cap-drop` de execução do docker.

`devices`

Todos os dispositivos do host a serem expostos ao contêiner. Esse parâmetro é mapeado para `Devices` no comando create-container do docker na opção `--device` de execução do docker.

Tipo: Matriz de objetos Device

Obrigatório: Não

`hostPath`

O caminho para o dispositivo na instância de contêiner host.

Tipo: string

Obrigatório: Sim

`containerPath`

O caminho dentro do contêiner no qual expor o dispositivo do host.

Tipo: String

Obrigatório: Não

`permissions`

As permissões explícitas a serem fornecidas ao contêiner para o dispositivo. Por padrão, o contêiner tem permissões para `read`, `write` e `mknod` no dispositivo.

Tipo: matriz de strings

Valores válidos: `read` | `write` | `mknod`

`initProcessEnabled`

Execute um processo `init` dentro do contêiner que encaminha sinais e colhe processos. Esse parâmetro é mapeado para a opção `--init` de execução do docker.

Este parâmetro requer a versão 1.25 ou posterior da Docker Remote API ou posterior em sua instância de contêiner.

`maxSwap`

A quantidade total de memória de troca (em MiB) que um contêiner pode usar. Esse parâmetro será convertido na opção `--memory-swap` para execução do docker em que o valor é a soma da memória do contêiner mais o valor de `maxSwap`.

Se um valor `maxSwap` de `0` for especificado, o contêiner não usará a troca. Os valores aceitos são `0` ou qualquer número inteiro positivo. Se o parâmetro `maxSwap` for omitido, o contêiner usará a configuração de troca para a instância de contêiner na qual ele está sendo executado. Um valor `maxSwap` deve ser definido para que o parâmetro `swappiness` seja usado.

`sharedMemorySize`

O valor do tamanho (em MiB) do volume `/dev/shm`. Esse parâmetro é mapeado para a opção `--shm-size` de execução do docker.

Tipo: inteiro

`swappiness`

É possível usar este parâmetro para ajustar o comportamento de troca de memória de um contêiner. Um valor de `swappiness` igual a `0` evita a troca, a menos que ela seja necessária. Um valor de `swappiness` igual a `100` resulta na troca frequente das páginas. Os valores aceitos são números inteiros entre `0` e `100`. Se você não especificar um valor, será usado o valor padrão de `60`. Além disso, se você não especificar um valor para `maxSwap`, esse parâmetro será ignorado. Esse parâmetro é mapeado para a opção `--memory-swappiness` de execução do docker.

> **Note**
>
> Se você estiver usando tarefas no Amazon Linux 2023, não haverá suporte para o parâmetro `swappiness`.

`tmpfs`

O caminho do contêiner, as opções de montagem e o tamanho (em MiB) da montagem do tmpfs. Esse parâmetro é mapeado para a opção `--tmpfs` de execução do docker.

Tipo: Matriz de objetos Tmpfs

Obrigatório: Não

`containerPath`

O caminho absoluto do arquivo no qual o volume tmpfs deve ser montado.

Tipo: string

Obrigatório: Sim

`mountOptions`

A lista de opções de montagem do volume tmpfs.

Tipo: matriz de strings

Obrigatório: Não

Valores Válidos: `"defaults" | "ro" | "rw" | "suid" | "nosuid" | "dev" | "nodev" | "exec" | "noexec" | "sync" | "async" | "dirsync" | "remount" | "mand" | "nomand" | "atime" | "noatime" | "diratime" | "nodiratime" | "bind" | "rbind" | "unbindable" | "runbindable" | "private" | "rprivate" | "shared" | "rshared" | "slave" | "rslave" | "relatime" | "norelatime" | "strictatime" | "nostrictatime" | "mode" | "uid" | "gid" | "nr_inodes" | "nr_blocks" | "mpol"`

`size`

O tamanho máximo (em MiB) do volume tmpfs.

Tipo: inteiro

Obrigatório: Sim

Dependência de contêiner

`dependsOn`

Tipo: matriz de objetos ContainerDependency

Obrigatório: Não

As dependências definidas para startup e desligamento do contêiner. Um contêiner pode conter várias dependências. Quando uma dependência é definida para o startup do contêiner, ela é revertida para o desligamento do contêiner. Para obter um exemplo, consulte Dependência de contêiner.

> **Note**
>
> Se um contêiner não atender a uma restrição de dependência ou expirar antes de cumprir a restrição, o Amazon ECS não evoluirá contêineres dependentes para seu próximo estado.

As instâncias exigem pelo menos a versão `1.26.0` do agente de contêiner para habilitar as dependências de contêiner. Recomendamos usar a versão mais recente do atendente de contêiner. Para obter informações sobre como verificar a versão do agente e atualizar para a versão mais recente, consulte Atualizar o agente de contêiner do Amazon ECS. Se você estiver usando uma AMI do Amazon Linux otimizada para o Amazon ECS, sua instância precisará pelo menos da versão `1.26.0-1` do pacote `ecs-init`. Se as instâncias de contêiner são executadas na versão `20190301` ou posterior, então elas contêm as versões necessárias do agente de contêiner e do `ecs-init`. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.

```
"dependsOn": [
    {
        "containerName": "string",
        "condition": "string"
    }
]
```

`containerName`

Tipo: string

Obrigatório: Sim

O nome do contêiner que deve atender à condição especificada.

`condition`

Tipo: string

Obrigatório: Sim

A condição de dependência do contêiner. A seguir estão as condições disponíveis e seus comportamentos.

- `START`: essa condição emula o comportamento de links e volumes atuais. A condição valida que um contêiner dependente seja iniciado antes de permitir que outros contêineres sejam iniciados.
- `COMPLETE`: essa condição valida que um contêiner dependente seja executado até a conclusão (encerramento) antes de permitir que outros contêineres sejam iniciados. Isso pode ser útil para os contêineres não essenciais que executam um script e depois são encerrados. Não é possível definir essa condição em um contêiner essencial.
- `SUCCESS`: essa condição é igual a `COMPLETE`, mas também exige que o contêiner seja encerrado com um status `zero`. Não é possível definir essa condição em um contêiner essencial.
- `HEALTHY`: essa condição valida que o contêiner dependente foi aprovado na verificação de integridade do Docker antes de permitir que outros contêineres sejam iniciados. Isso requer que o contêiner dependente tenha verificações de integridade configuradas na definição de tarefa. Essa condição é confirmada apenas no startup da tarefa.

Tempos limite de contêiner

`startTimeout`

Tipo: inteiro

Obrigatório: Não

Valores de exemplo: `120`

Tempo a ser aguardado (em segundos) antes de desistir de resolver dependências para um contêiner.

Por exemplo, você especifica dois contêineres em uma definição de tarefa com o `containerA` tendo uma dependência do `containerB` atingir um status `COMPLETE`, `SUCCESS` ou `HEALTHY`. Se um valor de `startTimeout` for especificado para o `containerB` e ele não atingir o status desejado nesse tempo, o `containerA` não será iniciado.

> **Note**
>
> Se um contêiner não atender a uma restrição de dependência ou expirar antes de cumprir a restrição, o Amazon ECS não evoluirá contêineres dependentes para seu próximo estado.

O valor máximo é 120 segundos.

`stopTimeout`

Tipo: inteiro

Obrigatório: Não

Valores de exemplo: `120`

Período (em segundos) a ser aguardado antes de o contêiner ser eliminado de maneira forçada se não for encerrado normalmente por conta própria.

Se o parâmetro `stopTimeout` não for especificado, o valor definido para a variável de configuração `ECS_CONTAINER_STOP_TIMEOUT` do agente de contêiner do Amazon ECS será usada. Se nem o parâmetro `stopTimeout` nem a variável de configuração do agente `ECS_CONTAINER_STOP_TIMEOUT` forem definidos, serão usados os valores padrão de 30 segundos para contêineres de Linux e 30 segundos para contêineres de Windows. As instâncias de contêiner exigem pelo menos a versão 1.26.0 do agente de contêiner para habilitar um valor de tempo limite de interrupção de contêiner. Recomendamos usar a versão mais recente do atendente de contêiner. Para obter informações sobre como verificar a versão do agente e atualizar para a versão mais recente, consulte Atualizar o agente de contêiner do Amazon ECS. Se você estiver usando a AMI do Amazon Linux otimizada para o Amazon ECS, sua instância precisará pelo menos da versão 1.26.0-1 do pacote `ecs-init`. Se as instâncias de contêiner são executadas na versão `20190301` ou posterior, então elas contêm as versões necessárias do agente de contêiner e do `ecs-init`. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.

Controles do sistema

`systemControls`

Tipo: objeto SystemControl

Obrigatório: Não

Uma lista de parâmetros de kernel com namespace a ser definida no contêiner. Esse parâmetro é mapeado para `Sysctls` no comando create-container do docker e na opção `--sysctl` de execução do docker. Por exemplo, é possível definir a configuração `net.ipv4.tcp_keepalive_time` para manter conexões de longa duração.

Não recomendamos que você especifique parâmetros `systemControls` relacionados à rede para vários contêineres em uma única tarefa que também use o modo de rede `awsvpc` ou `host`. Fazer isso apresenta as desvantagens a seguir:

- Para tarefas que usam o modo de rede `awsvpc`, se você definir `systemControls` para qualquer contêiner, ele se aplicará a todos os contêineres na tarefa. Se você definir um `systemControls` diferente para vários contêineres em uma única tarefa, o contêiner iniciado por último determinará qual `systemControls` entra em vigor.
- Para tarefas que usam o modo de rede `host`, o namespace de rede `systemControls` não é compatível.

Se você estiver configurando um namespace de recurso IPC para usar nos contêineres da tarefa, as condições a seguir serão aplicadas aos controles do sistema. Para obter mais informações, consulte Modo IPC.

- Para tarefas que usam o modo IPC `host`, o namespace IPC `systemControls` não é compatível.
- Para tarefas que usam o modo IPC de `task`, os valores de `systemControls` do namespace IPC serão aplicados a todos os contêineres em uma tarefa.

> **Note**
>
> Este parâmetro não é compatível com contêineres do Windows.

```
"systemControls": [
    {
```

```
        "namespace":"string",
        "value":"string"
    }
]
```

`namespace`

Tipo: String

Obrigatório: Não

O parâmetro de kernel com namespace para o qual definir um `value`.

Valores de namespace IPC válidos: `"kernel.msgmax" | "kernel.msgmnb" | "kernel.msgmni" | "kernel.sem" | "kernel.shmall" | "kernel.shmmax" | "kernel.shmmni" | "kernel.shm_rmid_forced"` e `Sysctls`, que comecem com `"fs.mqueue.*"`

Valores de namespace de rede válidos: `Sysctls`, que comece com `"net.*"`

`value`

Tipo: String

Obrigatório: Não

O valor do parâmetro de kernel com namespace especificado em `namespace`.

Interativo

`interactive`

Tipo: booliano

Obrigatório: Não

Quando esse parâmetro é `true`, você pode implantar aplicações em contêineres que exigem a alocação de `stdin` ou `tty`. Esse parâmetro é mapeado para `OpenStdin` no comando create-container do docker na opção `--interactive` de execução do docker.

O padrão é `false`.

Pseudoterminal

`pseudoTerminal`

Tipo: booliano

Obrigatório: Não

Quando esse parâmetro é `true`, um TTY é alocado. Esse parâmetro é mapeado para `Tty` no comando create-container do docker na opção `--tty` de execução do docker.

O padrão é `false`.

Nome do acelerador de inferência elástica

Requisito do recurso acelerador Elastic Inference para a definição de tarefa.

> **Note**
>
> O Amazon Elastic Inference (EI) não está mais disponível para clientes.

Os seguintes parâmetros são permitidos em uma definição de tarefa:

`deviceName`

Tipo: string

Obrigatório: Sim

O nome do dispositivo do acelerador de inferência elástica. O `deviceName` também deve ser referenciado em uma definição de contêiner, consulte `Elastic Inference accelerator`.

`deviceType`

Tipo: string

Obrigatório: Sim

O acelerador de inferência elástica a ser usado.

Limitações de posicionamento de tarefa

Ao registrar uma definição de tarefa, você pode fornecer restrições de posicionamento de tarefa que personalizam como o Amazon ECS posiciona tarefas.

Você pode usar restrições para posicionar tarefas de acordo com a zona de disponibilidade, o tipo de instância ou os atributos personalizados. Para obter mais informações, consulte Definição de quais instâncias de contêiner o Amazon ECS usa em tarefas.

Os parâmetros a seguir são permitidos em uma definição de contêiner.

`expression`

Tipo: String

Obrigatório: Não

Uma expressão de idioma de consulta de cluster a ser aplicada à restrição. Para obter mais informações, consulte Criação de expressões para definir instâncias de contêiner em tarefas do Amazon ECS.

`type`

Tipo: string

Obrigatório: Sim

O tipo de restrição. Use `memberOf` para restringir a seleção a um grupo de candidatos válidos.

Configuração do proxy

`proxyConfiguration`

Tipo: objeto ProxyConfiguration

Obrigatório: Não

Os detalhes de configuração do proxy do App Mesh.

Para as tarefas que usam o tipo de execução do EC2, as instâncias de contêiner exigem pelo menos a versão 1.26.0 do agente de contêiner e pelo menos a versão 1.26.0-1 do pacote `ecs-`

`init` para habilitar uma configuração de proxy. Se as instâncias de contêiner forem executadas na AMI otimizada para o Amazon ECS versão `20190301` ou posterior, elas conterão as versões necessárias do agente de contêiner e do `ecs-init`. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.

> **Note**
>
> Este parâmetro não é compatível com contêineres do Windows.

```
"proxyConfiguration": {
    "type": "APPMESH",
    "containerName": "string",
    "properties": [
        {
           "name": "string",
           "value": "string"
        }
    ]
}
```

`type`

Tipo: String

Valor: `APPMESH`

Obrigatório: Não

O tipo de proxy. O único valor aceito é `APPMESH`.

`containerName`

Tipo: string

Obrigatório: Sim

O nome do contêiner que serve como proxy do App Mesh.

`properties`

Tipo: matriz de objetos KeyValuePair

Obrigatório: Não

O conjunto de parâmetros de configuração de rede para fornecer o plugin Container Network Interface (CNI), especificado como pares de chave/valor.

- `IgnoredUID`: (obrigatório) o ID de usuário (UID) do contêiner de proxy conforme definido pelo parâmetro `user` em uma definição de contêiner. Isso é usado para garantir que o proxy ignore seu próprio tráfego. Se `IgnoredGID` for especificado, esse campo poderá estar vazio.
- `IgnoredGID`: (obrigatório) o ID de grupo (GID) do contêiner de proxy conforme definido pelo parâmetro `user` em uma definição de contêiner. Isso é usado para garantir que o proxy ignore seu próprio tráfego. Se `IgnoredUID` for especificado, esse campo poderá estar vazio.
- `AppPorts`: (obrigatório) a lista de portas usadas pelo aplicativo. O tráfego de rede para essas portas é encaminhado para `ProxyIngressPort` e `ProxyEgressPort`.
- `ProxyIngressPort`: (obrigatório) especifica a porta para a qual o tráfego de entrada para `AppPorts` é direcionado.
- `ProxyEgressPort`: (obrigatório) especifica a porta para a qual o tráfego de saída de `AppPorts` é direcionado.
- `EgressIgnoredPorts`: (obrigatório) o tráfego de saída para essas portas especificadas é ignorado e não é redirecionado para `ProxyEgressPort`. Ele pode ser uma lista vazia.
- `EgressIgnoredIPs`: (obrigatório) o tráfego de saída para esses endereços IP especificados é ignorado e não é redirecionado para `ProxyEgressPort`. Ele pode ser uma lista vazia.

`name`

Tipo: String

Obrigatório: Não

O nome do par de chave/valor.

`value`

Tipo: String

Obrigatório: Não

O valor do par de chave/valor.

Volumes

Se preferir, ao registrar uma definição de tarefa, é possível especificar uma lista de volumes a serem passados para o daemon do Docker em uma instância de contêiner, que se tornará disponível para o acesso de outros contêineres da mesma instância de contêiner.

A seguir estão os tipos de volumes de dados que podem ser usados.

- Volumes do Amazon EBS: fornece armazenamento em blocos econômico, durável e de alto desempenho para workloads em contêineres com uso intenso de dados. Você pode anexar um volume do Amazon EBS por tarefa do Amazon ECS ao executar uma tarefa autônoma ou ao criar ou atualizar um serviço. Os volumes do Amazon EBS são compatíveis com as tarefas do Linux. Para obter mais informações, consulte Uso de volumes do Amazon EBS com o Amazon ECS.
- Volumes do Amazon EFS: oferece armazenamento de arquivos simples, escalável e persistente para uso com tarefas do Amazon ECS. Com o Amazon EFS, a capacidade de armazenamento é elástica. Ela aumenta e diminui automaticamente à medida que arquivos são adicionados e removidos. As aplicações podem ter o armazenamento de que precisam, quando precisam. Os volumes do Amazon EFS são compatíveis. Para obter mais informações, consulte Uso de volumes do Amazon EFS com o Amazon ECS.
- Volumes do FSx para Windows File Server: fornece servidores de arquivos Microsoft Windows totalmente gerenciados. Esses servidores têm o suporte de um sistema de arquivos do Windows. Ao usar o FSx for Windows File Server junto com o Amazon ECS, você pode provisionar tarefas do Windows com armazenamento de arquivos persistente, distribuído, compartilhado e estático. Para obter mais informações, consulte Uso de volumes do FSx para Windows File Server com Amazon ECS.

  Não há suporte para essa opção nos contêineres do Windows no Fargate.
- Volumes do Docker: um volume gerenciado pelo Docker criado em `/var/lib/docker/volumes` na instância de host do Amazon EC2. Drivers de volume do Docker (também conhecidos como plug-ins) são usados para integrar os volumes com sistemas de armazenamento externos, como o Amazon EBS. O driver de volume `local` integrado ou um driver de volume de terceiros podem ser usados. Os volumes do Docker só são compatíveis com a execução de tarefas em instâncias do Amazon EC2. Os contêineres do Windows só são compatíveis com o uso do driver `local`. Para usar os volumes do Docker, especifique uma `dockerVolumeConfiguration` em sua definição de tarefa.
- Montagens vinculadas: um arquivo ou diretório do computador host é montado em um contêiner. Os volumes de host de montagem vinculada são compatíveis. Para usar os volumes de host de

montagens bind, especifique um `host` e um valor `sourcePath` opcional em sua definição de tarefa.

Para obter mais informações, consulte Opções de armazenamento para tarefas do Amazon ECS.

Os parâmetros a seguir são permitidos em uma definição de contêiner.

`name`

Tipo: String

Obrigatório: Não

O nome do volume. São permitidos até 255 letras (caixa alta e baixa), números, hífens (`-`) e sublinhados (_). Esse nome é referenciado no parâmetro `sourceVolume` do objeto `mountPoints` de definição do contêiner.

`host`

Obrigatório: Não

O parâmetro `host` é usado para vincular o ciclo de vida da montagem bind à instância host do Amazon EC2, em vez da tarefa, e onde ela está armazenada. Caso o parâmetro `host` esteja vazio, o daemon do Docker atribui um caminho host para o volume de dados, mas os dados não têm garantia de persistir depois que os contêineres associados deixarem de ser executados.

Os contêineres do Windows podem montar diretórios inteiros na mesma unidade como `$env:ProgramData`.

> **Note**
>
> O parâmetro `sourcePath` só é compatível ao usar tarefas hospedadas em instâncias do Amazon EC2.

`sourcePath`

Tipo: String

Obrigatório: Não

Quando o parâmetro `host` é usado, especifique um `sourcePath` para declarar o caminho na instância host do Amazon EC2 que é apresentada ao contêiner. Caso esse parâmetro esteja vazio, o daemon do Docker atribui um caminho host para você. Caso o parâmetro `host` contenha um local de arquivo `sourcePath`, o volume de dados persistirá no local especificado na instância host do Amazon EC2 até você excluí-lo manualmente. Caso o valor `sourcePath` não exista na instância host do Amazon EC2, o daemon do Docker o criará. Caso o local exista, o conteúdo da pasta do caminho de origem é exportado.

`configuredAtLaunch`

Tipo: booliano

Obrigatório: Não

Especifica se um volume é configurável na execução. Quando definido como `true`, você pode configurar o volume ao executar uma tarefa autônoma ou ao criar ou atualizar um serviço. Quando definido como `true`, não será possível fornecer outra configuração de volume na definição da tarefa. Esse parâmetro deve ser definido como `true` para configurar um volume do Amazon EBS para anexação a uma tarefa. Definir `configuredAtLaunch` como `true` e adiar a configuração do volume para a fase de execução permite criar definições de tarefas que não estão restritas a um tipo de volume ou a configurações de volume específicas. Isso torna a definição de tarefa reutilizável em diferentes ambientes de execução. Para obter mais informações, consulte Volumes do Amazon EBS.

`dockerVolumeConfiguration`

Tipo: objeto DockerVolumeConfiguration

Obrigatório: Não

Esse parâmetro é especificado ao usar volumes do Docker. Os volumes do Docker só são compatíveis ao executar tarefas em instâncias do EC2. Os contêineres do Windows só são compatíveis com o uso do driver `local`. Para usar montagens bind, em vez disso, especifique um `host`.

`scope`

Tipo: String

Valores válidos: `task` | `shared`

Obrigatório: Não

O escopo para o volume do Docker, que determina o ciclo de vida. Os volumes do Docker que são delimitados para uma `task` são provisionados automaticamente quando a tarefa é iniciada e destruídos quando a tarefa é interrompida. Volumes do Docker delimitados como `shared` são mantidos após a interrupção da tarefa.

`autoprovision`

Tipo: booliano

Valor padrão: `false`

Obrigatório: Não

Se o valor for `true`, o volume de Docker será criado se ele ainda não existir. Esse campo só será usado se `scope` for `shared`. Se `scope` for `task`, esse parâmetro deverá ser omitido.

`driver`

Tipo: String

Obrigatório: Não

O driver do volume do Docker a ser usado. O valor do driver deve corresponder ao nome do driver fornecido pelo Docker porque esse nome é usado no posicionamento de tarefas. Se o driver foi instalado usando a CLI de plug-in do Docker, use `docker plugin ls` para recuperar o nome do driver na instância de contêiner. Se o driver foi instalado usando outro método, use a descoberta de plug-in do Docker para recuperar o nome do driver.

`driverOpts`

Tipo: String

Obrigatório: Não

Um mapa de opções específicas do driver do Docker pelas quais passar. Esse parâmetro é mapeado para `DriverOpts` na seção Criar um volume da API remota do Docker.

`labels`

Tipo: String

Obrigatório: Não

Metadados personalizados para adicionar ao volume do Docker.

`efsVolumeConfiguration`

Tipo: objeto EFSVolumeConfiguration

Obrigatório: Não

Esse parâmetro é especificado quando volumes Amazon EFS são usados.

`fileSystemId`

Tipo: string

Obrigatório: Sim

A ID do sistema de arquivamento Amazon EFS a ser usada.

`rootDirectory`

Tipo: string

Obrigatório: Não

O diretório dentro do sistema de arquivamento Amazon EFS a ser montado como diretório raiz dentro do host. Se esse parâmetro for omitido, a raiz do volume do Amazon EFS será usada. Especificar `/` tem o mesmo efeito que omitir esse parâmetro.

> **Important**
>
> Se um ponto de acesso do EFS for especificado em `authorizationConfig`, o parâmetro de diretório raiz deverá ser omitido ou definido como `/`, o que imporá o caminho definido no ponto de acesso do EFS.

`transitEncryption`

Tipo: String

Valores válidos: `ENABLED` | `DISABLED`

Obrigatório: Não

Especifica se a criptografia deve ou não ser habilitada para dados do Amazon EFS em trânsito entre o host do Amazon ECS e o servidor do Amazon EFS. A criptografia de trânsito deverá ser habilitada se a autorização do IAM do Amazon EFS for usada. Se o parâmetro for omitido,

o valor padrão `DISABLED` será usado. Para obter mais informações, consulte Criptografar dados em trânsito no Guia do usuário do Amazon Elastic File System.

`transitEncryptionPort`

Tipo: inteiro

Obrigatório: Não

A porta a ser usada ao enviar dados criptografados entre o host do Amazon ECS e o servidor do Amazon EFS. Caso não especifique uma porta de criptografia em trânsito, a tarefa usará a estratégia de seleção de porta usada pelo assistente de montagem do Amazon EFS. Para mais informações, consulte Auxiliar de Montagem EFS no Guia de Usuário Amazon Elastic File System.

`authorizationConfig`

Tipo: objeto EFSAuthorizationConfig

Obrigatório: Não

Detalhes de configuração de autorização do sistema de arquivamento Amazon EFS.

`accessPointId`

Tipo: Sequência

Obrigatório: Não

ID do ponto de acesso a ser usado. Se um ponto de acesso for especificado, o valor do diretório raiz em `efsVolumeConfiguration` deve ser omitido ou definido como `/`, o que imporá o caminho definido no ponto de acesso do EFS. Se um ponto de acesso for usado, a criptografia em trânsito deverá ser habilitada em `EFSVolumeConfiguration`. Para mais informações, consulte Trabalhando com Pontos de Acesso Amazon EFS no Guia de Usuário Amazon Elastic File System.

`iam`

Tipo: String

Valores válidos: `ENABLED` | `DISABLED`

Obrigatório: Não

Especifica se é necessário ou não usar o perfil do IAM para a tarefa do Amazon ECS estabelecida em uma definição de tarefa ao montar o sistema de arquivos do

Amazon EFS. Em caso positivo, a criptografia de trânsito deve estar habilitada em `EFSVolumeConfiguration`. Se o parâmetro for omitido, o valor padrão `DISABLED` será usado. Para obter mais informações, consulte Funções do IAM para tarefas.

`FSxWindowsFileServerVolumeConfiguration`

Tipo: objeto FSxWindowsFileServerVolumeConfiguration

Obrigatório: Sim

Esse parâmetro é especificado quando você está usando um sistema de arquivos do Amazon FSx para Windows File Server para armazenamento de tarefas.

`fileSystemId`

Tipo: string

Obrigatório: Sim

O ID do sistema de arquivos do FSx for Windows File Server a ser usado.

`rootDirectory`

Tipo: string

Obrigatório: Sim

O diretório no sistema de arquivos do FSx for Windows File Server que deve ser montado como o diretório raiz dentro do host.

`authorizationConfig`

`credentialsParameter`

Tipo: string

Obrigatório: Sim

As opções de credencial de autorização.

opções:

- O nome do recurso da Amazon (ARN) do segredo do AWS Secrets Manager.
- ARN de um parâmetro do AWS Systems Manager.

`domain`

Tipo: string

Obrigatório: Sim

Um nome de domínio totalmente qualificado hospedado por um diretório do AWS Directory Service for Microsoft Active Directory (AWS Managed Microsoft AD) ou por um Active Directory do EC2 auto-hospedado.

Tags

Quando você registra uma definição de tarefa, pode especificar opcionalmente etiquetas de metadados aplicadas à definição de tarefa. As etiquetas ajudam você a categorizar e organizar sua definição de tarefa. Cada tag consiste em uma chave e um valor opcional. Defina ambos. Para obter mais informações, consulte Marcação de recursos do Amazon ECS.

> **Important**
>
> Não adicione informações de identificação pessoal nem outras informações confidenciais ou sigilosas em tags. As tags são acessíveis a muitos serviços da AWS, incluindo faturamento. As tags não devem ser usadas para dados privados ou confidenciais.

Os parâmetros a seguir são permitidos em um objeto de etiqueta.

`key`

Tipo: String

Obrigatório: Não

Uma parte de um par de chave/valor que compõe uma tag. Uma chave é um rótulo geral que age como uma categoria para valores de tag mais específicos.

`value`

Tipo: String

Obrigatório: Não

A parte opcional de um par de chave/valor que compõe uma tag. Um valor atua como um descritor dentro de uma categoria de tag (chave).

Outros parâmetros de definição de tarefa

Os parâmetros de definição de tarefa a seguir podem ser usados durante o registro das definições de tarefa no console do Amazon ECS usando a opção Configure via JSON (Configurar via JSON). Para obter mais informações, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

Tópicos

- Modo IPC
- Modo PID
- Injeção de falhas

Modo IPC

`ipcMode`

Tipo: String

Obrigatório: Não

O namespace de recurso IPC a ser usado para os contêineres na tarefa. Os valores válidos são `host`, `task` ou `none`. Se o `host` for especificado, todos os contêineres dentro das tarefas que especificaram o modo IPC `host` na mesma instância de contêiner compartilham os mesmos recursos IPC com a instância do Amazon EC2 do host. Se `task` for especificado, todos os contêineres dentro da tarefa especificada compartilharão os mesmos recursos IPC. Se `none` for especificado, os recursos IPC nos contêineres de uma tarefa serão privados e não serão compartilhados com outros contêineres em uma tarefa ou na instância de contêiner. Se nenhum valor for especificado, o compartilhamento do namespace de recurso IPC dependerá da configuração do daemon do Docker na instância de contêiner.

Se o modo IPC `host` for usado, haverá um risco elevado de exposição indesejada do namespace IPC.

Se você estiver definindo parâmetros de kernel com namespace que usa `systemControls` para os contêineres na tarefa, os itens a seguir serão aplicados ao seu namespace de recurso IPC.

- Para tarefas que usam o modo IPC `host`, o namespace IPC relacionado `systemControls` não é compatível.
- Para tarefas que usam o modo IPC `task`, `systemControls` relacionados ao namespace IPC serão aplicados a todos os contêineres em uma tarefa.

Modo PID

`pidMode`

Tipo: String

Valores válidos: `host` | `task`

Obrigatório: Não

O namespace do processo a ser usado para os contêineres na tarefa. Os valores válidos são `host` ou `task`. Por exemplo, o monitoramento de arquivos associados pode precisar que `pidMode` acesse informações sobre outros contêineres em execução na mesma tarefa.

Se o `host` for especificado, todos os contêineres das tarefas que especificaram o modo PID `host` na mesma instância de contêiner compartilharão namespace de processo com a instância do Amazon EC2 do host.

Se a `task` for especificada, todos os contêineres da tarefa especificada compartilharão o mesmo namespace de processo.

Se nenhum valor for especificado, o padrão será um namespace privado para cada contêiner.

Se o modo PID `host` for usado, haverá um risco elevado de exposição indesejada do namespace de processo.

> **Note**
>
> Este parâmetro não é compatível com contêineres do Windows.

Injeção de falhas

`enableFaultInjection`

Tipo: booliano

Valores válidos: `true` | `false`

Obrigatório: Não

Se esse parâmetro estiver definido como `true`, na carga útil de uma tarefa, o Amazon ECS aceita solicitações de injeção de falhas dos contêineres da tarefa. Esse parâmetro é definido como `false` por padrão.

Modelo de definição de tarefa do Amazon ECS

Um modelo de definição de tarefa vazio é mostrado a seguir. É possível usar esse modelo para criar a definição de tarefa que pode acabar sendo colada na área de entrada JSON do console ou salva em um arquivo e usada com a opção da AWS CLI `--cli-input-json`. Para obter mais informações, consulte Parâmetros de definição de tarefa do Amazon ECS para o tipo de inicialização do Fargate.

Modelo do tipo de execução do Amazon EC2

```
{
  "family": "",
  "taskRoleArn": "",
  "executionRoleArn": "",
  "networkMode": "none",
  "containerDefinitions": [
    {
      "name": "",
      "image": "",
      "repositoryCredentials": {
        "credentialsParameter": ""
      },
      "cpu": 0,
      "memory": 0,
      "memoryReservation": 0,
      "links": [""],
      "portMappings": [
        {
          "containerPort": 0,
          "hostPort": 0,
          "protocol": "tcp"
        }
      ],
      "restartPolicy": {
        "enabled": true,
        "ignoredExitCodes": [0],
        "restartAttemptPeriod": 180
```

```
      },
      "essential": true,
      "entryPoint": [""],
      "command": [""],
      "environment": [
        {
          "name": "",
          "value": ""
        }
      ],
      "environmentFiles": [
        {
          "value": "",
          "type": "s3"
        }
      ],
      "mountPoints": [
        {
          "sourceVolume": "",
          "containerPath": "",
          "readOnly": true
        }
      ],
      "volumesFrom": [
        {
          "sourceContainer": "",
          "readOnly": true
        }
      ],
      "linuxParameters": {
        "capabilities": {
          "add": [""],
          "drop": [""]
        },
        "devices": [
          {
            "hostPath": "",
            "containerPath": "",
            "permissions": ["read"]
          }
        ],
        "initProcessEnabled": true,
        "sharedMemorySize": 0,
        "tmpfs": [
```

```
        {
          "containerPath": "",
          "size": 0,
          "mountOptions": [""]
        }
      ],
      "maxSwap": 0,
      "swappiness": 0
    },
    "secrets": [
      {
        "name": "",
        "valueFrom": ""
      }
    ],
    "dependsOn": [
      {
        "containerName": "",
        "condition": "COMPLETE"
      }
    ],
    "startTimeout": 0,
    "stopTimeout": 0,
    "hostname": "",
    "user": "",
    "workingDirectory": "",
    "disableNetworking": true,
    "privileged": true,
    "readonlyRootFilesystem": true,
    "dnsServers": [""],
    "dnsSearchDomains": [""],
    "extraHosts": [
      {
        "hostname": "",
        "ipAddress": ""
      }
    ],
    "dockerSecurityOptions": [""],
    "interactive": true,
    "pseudoTerminal": true,
    "dockerLabels": {
      "KeyName": ""
    },
    "ulimits": [
```

```
        {
          "name": "nofile",
          "softLimit": 0,
          "hardLimit": 0
        }
      ],
      "logConfiguration": {
        "logDriver": "splunk",
        "options": {
          "KeyName": ""
        },
        "secretOptions": [
          {
            "name": "",
            "valueFrom": ""
          }
        ]
      },
      "healthCheck": {
        "command": [""],
        "interval": 0,
        "timeout": 0,
        "retries": 0,
        "startPeriod": 0
      },
      "systemControls": [
        {
          "namespace": "",
          "value": ""
        }
      ],
      "resourceRequirements": [
        {
          "value": "",
          "type": "InferenceAccelerator"
        }
      ],
      "firelensConfiguration": {
        "type": "fluentbit",
        "options": {
          "KeyName": ""
        }
      }
    }
```

```
  ],
  "volumes": [
    {
      "name": "",
      "host": {
        "sourcePath": ""
      },
      "configuredAtLaunch": true,
      "dockerVolumeConfiguration": {
        "scope": "shared",
        "autoprovision": true,
        "driver": "",
        "driverOpts": {
          "KeyName": ""
        },
        "labels": {
          "KeyName": ""
        }
      },
      "efsVolumeConfiguration": {
        "fileSystemId": "",
        "rootDirectory": "",
        "transitEncryption": "DISABLED",
        "transitEncryptionPort": 0,
        "authorizationConfig": {
          "accessPointId": "",
          "iam": "ENABLED"
        }
      },
      "fsxWindowsFileServerVolumeConfiguration": {
        "fileSystemId": "",
        "rootDirectory": "",
        "authorizationConfig": {
          "credentialsParameter": "",
          "domain": ""
        }
      }
    }
  ],
  "placementConstraints": [
    {
      "type": "memberOf",
      "expression": ""
    }
```

```
  ],
  "requiresCompatibilities": ["EC2"],
  "cpu": "",
  "memory": "",
  "tags": [
    {
      "key": "",
      "value": ""
    }
  ],
  "pidMode": "task",
  "ipcMode": "task",
  "proxyConfiguration": {
    "type": "APPMESH",
    "containerName": "",
    "properties": [
      {
        "name": "",
        "value": ""
      }
    ]
  },
  "inferenceAccelerators": [
    {
      "deviceName": "",
      "deviceType": ""
    }
  ],
  "ephemeralStorage": {
    "sizeInGiB": 0
  },
  "runtimePlatform": {
    "cpuArchitecture": "X86_64",
    "operatingSystemFamily": "WINDOWS_SERVER_20H2_CORE"
  }
}
```

Modelo do tipo de execução do Fargate

Important

Para o tipo de execução do Fargate, você deve incluir o parâmetro `operatingSystemFamily` com um dos seguintes valores:

- LINUX

- WINDOWS_SERVER_2019_FULL
- WINDOWS_SERVER_2019_CORE
- WINDOWS_SERVER_2022_FULL
- WINDOWS_SERVER_2022_CORE

```
{
    "family": "",
    "runtimePlatform": {"operatingSystemFamily": ""},
    "taskRoleArn": "",
    "executionRoleArn": "",
    "networkMode": "awsvpc",
    "platformFamily": "",
    "containerDefinitions": [
        {
            "name": "",
            "image": "",
            "repositoryCredentials": {"credentialsParameter": ""},
            "cpu": 0,
            "memory": 0,
            "memoryReservation": 0,
            "links": [""],
            "portMappings": [
                {
                    "containerPort": 0,
                    "hostPort": 0,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "entryPoint": [""],
            "command": [""],
            "environment": [
                {
                    "name": "",
                    "value": ""
                }
            ],
```

```
            "environmentFiles": [
                {
                    "value": "",
                    "type": "s3"
                }
            ],
            "mountPoints": [
                {
                    "sourceVolume": "",
                    "containerPath": "",
                    "readOnly": true
                }
            ],
            "volumesFrom": [
                {
                    "sourceContainer": "",
                    "readOnly": true
                }
            ],
            "linuxParameters": {
                "capabilities": {
                    "add": [""],
                    "drop": [""]
                },
                "devices": [
                    {
                        "hostPath": "",
                        "containerPath": "",
                        "permissions": ["read"]
                    }
                ],
                "initProcessEnabled": true,
                "sharedMemorySize": 0,
                "tmpfs": [
                    {
                        "containerPath": "",
                        "size": 0,
                        "mountOptions": [""]
                    }
                ],
                "maxSwap": 0,
                "swappiness": 0
            },
            "secrets": [
```

```
            {
                "name": "",
                "valueFrom": ""
            }
        ],
        "dependsOn": [
            {
                "containerName": "",
                "condition": "HEALTHY"
            }
        ],
        "startTimeout": 0,
        "stopTimeout": 0,
        "hostname": "",
        "user": "",
        "workingDirectory": "",
        "disableNetworking": true,
        "privileged": true,
        "readonlyRootFilesystem": true,
        "dnsServers": [""],
        "dnsSearchDomains": [""],
        "extraHosts": [
            {
                "hostname": "",
                "ipAddress": ""
            }
        ],
        "dockerSecurityOptions": [""],
        "interactive": true,
        "pseudoTerminal": true,
        "dockerLabels": {"KeyName": ""},
        "ulimits": [
            {
                "name": "msgqueue",
                "softLimit": 0,
                "hardLimit": 0
            }
        ],
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {"KeyName": ""},
            "secretOptions": [
                {
                    "name": "",
```

```
                    "valueFrom": ""
                }
            ]
        },
        "healthCheck": {
            "command": [""],
            "interval": 0,
            "timeout": 0,
            "retries": 0,
            "startPeriod": 0
        },
        "systemControls": [
            {
                "namespace": "",
                "value": ""
            }
        ],
        "resourceRequirements": [
            {
                "value": "",
                "type": "GPU"
            }
        ],
        "firelensConfiguration": {
            "type": "fluentd",
            "options": {"KeyName": ""}
        }
    }
],
"volumes": [
    {
        "name": "",
        "host": {"sourcePath": ""},
        "configuredAtLaunch":true,
        "dockerVolumeConfiguration": {
            "scope": "task",
            "autoprovision": true,
            "driver": "",
            "driverOpts": {"KeyName": ""},
            "labels": {"KeyName": ""}
        },
        "efsVolumeConfiguration": {
            "fileSystemId": "",
            "rootDirectory": "",
```

```
                "transitEncryption": "ENABLED",
                "transitEncryptionPort": 0,
                "authorizationConfig": {
                    "accessPointId": "",
                    "iam": "ENABLED"
                }
            }
        }
    ],
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "",
    "memory": "",
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "ephemeralStorage": {"sizeInGiB": 0},
    "pidMode": "task",
    "ipcMode": "none",
    "proxyConfiguration": {
        "type": "APPMESH",
        "containerName": "",
        "properties": [
            {
                "name": "",
                "value": ""
            }
        ]
    },
    "inferenceAccelerators": [
        {
            "deviceName": "",
            "deviceType": ""
        }
    ]
}
```

É possível gerar esse modelo de definição de tarefa usando o seguinte comando da AWS CLI.

```
aws ecs register-task-definition --generate-cli-skeleton
```

Exemplos de definições de tarefa do Amazon ECS

É possível copiar os exemplos e trechos para começar a criar suas próprias definições de tarefa.

É possível copiar os exemplos e, em seguida, colá-los quando usar a opção Configurar via JSON no console. Certifique-se de personalizar os exemplos, como usar o ID da sua conta. É possível incluir os trechos no JSON de definição de tarefa. Para obter mais informações, consulte Criar uma definição de tarefa do Amazon ECS usando o console e Parâmetros de definição de tarefa do Amazon ECS para o tipo de inicialização do Fargate.

Para obter mais exemplos de definição de tarefa, consulte Exemplos de definição de tarefa da AWS no GitHub.

Tópicos

- Webserver
- Driver de log do splunk
- Driver de log do fluentd
- Driver de log do gelf
- Workloads em instâncias externas
- Perfil do IAM para definição de imagem e tarefa do Amazon ECR
- Ponto de entrada com comando
- Dependência de contêiner
- Volumes nas definições de tarefa
- Definições de tarefa de exemplo do Windows

Webserver

Veja a seguir um exemplo de definição de tarefa usando o tipo de inicialização de contêineres de Linux do Fargate que configura um servidor da Web:

```
{
   "containerDefinitions": [
      {
         "command": [
            "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</
title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
```

```
 <h2>Congratulations!</h2> <p>Your application is now running on a container in Amazon
 ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html && httpd-
foreground\""
         ],
         "entryPoint": [
            "sh",
            "-c"
         ],
         "essential": true,
         "image": "public.ecr.aws/docker/library/httpd:2.4",
         "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
               "awslogs-group" : "/ecs/fargate-task-definition",
               "awslogs-region": "us-east-1",
               "awslogs-stream-prefix": "ecs"
            }
         },
         "name": "sample-fargate-app",
         "portMappings": [
            {
               "containerPort": 80,
               "hostPort": 80,
               "protocol": "tcp"
            }
         ]
      }
   ],
   "cpu": "256",
   "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
   "family": "fargate-task-definition",
   "memory": "512",
   "networkMode": "awsvpc",
   "runtimePlatform": {
        "operatingSystemFamily": "LINUX"
    },
   "requiresCompatibilities": [
       "FARGATE"
    ]
}
```

Veja a seguir um exemplo de definição de tarefa usando o tipo de inicialização de contêineres de Windows do Fargate que configura um servidor da Web:

```
{
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file
 -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>'; C:\\ServiceMonitor.exe
 w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "networkMode": "awsvpc",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["FARGATE"]
}
```

Driver de log do **`splunk`**

O trecho a seguir demonstra como usar o driver de log `splunk` em uma definição de tarefa que envia os logs para um serviço remoto. O parâmetro de token do Splunk é especificado como uma

opção secreta, pois ele pode ser tratado como dados confidenciais. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.

```
"containerDefinitions": [{
  "logConfiguration": {
   "logDriver": "splunk",
   "options": {
    "splunk-url": "https://cloud.splunk.com:8080",
    "tag": "tag_name",
   },
   "secretOptions": [{
    "name": "splunk-token",
    "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:splunk-token-
KnrBkD"
}],
```

Driver de log do `fluentd`

O trecho a seguir demonstra como usar o driver de log `fluentd` em uma definição de tarefa que envia os logs para um serviço remoto. O valor `fluentd-address` é especificado como uma opção secreta, pois ele pode ser tratado como dados confidenciais. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.

```
"containerDefinitions": [{
 "logConfiguration": {
  "logDriver": "fluentd",
  "options": {
   "tag": "fluentd demo"
  },
  "secretOptions": [{
   "name": "fluentd-address",
   "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:fluentd-address-
KnrBkD"
  }]
 },
 "entryPoint": [],
 "portMappings": [{
            "hostPort": 80,
            "protocol": "tcp",
            "containerPort": 80
            },
            {
```

```
  "hostPort": 24224,
  "protocol": "tcp",
  "containerPort": 24224
 }]
}],
```

Driver de log do `gelf`

O trecho a seguir demonstra como usar o driver de log `gelf` em uma definição de tarefa que envia os logs para um host remoto executando o Logstash que leva logs Gelf como uma entrada. Para obter mais informações, consulte `logConfiguration`.

```
"containerDefinitions": [{
 "logConfiguration": {
  "logDriver": "gelf",
  "options": {
   "gelf-address": "udp://logstash-service-address:5000",
   "tag": "gelf task demo"
  }
 },
 "entryPoint": [],
 "portMappings": [{
   "hostPort": 5000,
   "protocol": "udp",
   "containerPort": 5000
  },
  {
   "hostPort": 5000,
   "protocol": "tcp",
   "containerPort": 5000
  }
 ]
}],
```

Workloads em instâncias externas

Ao registrar uma definição de tarefa do Amazon ECS, use o parâmetro `requiresCompatibilities` e especifique `EXTERNAL`, que valida se a definição da tarefa é compatível para usar na execução de workloads do Amazon ECS nas instâncias externas. Se você usar o console para registrar uma definição de tarefa, deverá usar o editor JSON. Para obter mais informações, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

> **Important**
>
> Se as tarefas exigirem uma função do IAM de execução de tarefa, verifique se ela está especificada na definição de tarefa.

Ao implantar a workload, use o tipo de inicialização EXTERNAL quando criar o serviço ou quando executar a tarefa autônoma.

Veja a seguir um exemplo de definição de tarefa.

Linux

```
{
 "requiresCompatibilities": [
  "EXTERNAL"
 ],
 "containerDefinitions": [{
  "name": "nginx",
  "image": "public.ecr.aws/nginx/nginx:latest",
  "memory": 256,
  "cpu": 256,
  "essential": true,
  "portMappings": [{
   "containerPort": 80,
   "hostPort": 8080,
   "protocol": "tcp"
  }]
 }],
 "networkMode": "bridge",
 "family": "nginx"
}
```

Windows

```
{
 "requiresCompatibilities": [
  "EXTERNAL"
 ],
 "containerDefinitions": [{
  "name": "windows-container",
  "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-ltsc2019",
```

```
  "memory": 256,
  "cpu": 512,
  "essential": true,
  "portMappings": [{
   "containerPort": 80,
   "hostPort": 8080,
   "protocol": "tcp"
  }]
 }],
 "networkMode": "bridge",
 "family": "windows-container"
}
```

Perfil do IAM para definição de imagem e tarefa do Amazon ECR

O trecho a seguir usa uma imagem do Amazon ECR denominada `aws-nodejs-sample` com a tag `v1` do registro `123456789012.dkr.ecr.us-west-2.amazonaws.com`. O contêiner dessa tarefa herda permissões do IAM da função `arn:aws:iam::123456789012:role/AmazonECSTaskS3BucketRole`. Para obter mais informações, consulte Perfil do IAM para tarefas do Amazon ECS.

```
{
    "containerDefinitions": [
        {
            "name": "sample-app",
            "image": "123456789012.dkr.ecr.us-west-2.amazonaws.com/aws-nodejs-
sample:v1",
            "memory": 200,
            "cpu": 10,
            "essential": true
        }
    ],
    "family": "example_task_3",
    "taskRoleArn": "arn:aws:iam::123456789012:role/AmazonECSTaskS3BucketRole"
}
```

Ponto de entrada com comando

O trecho a seguir demonstra a sintaxe de um contêiner do Docker que usa um ponto de entrada e um argumento de comando. Este contêiner executa ping no `example.com` quatro vezes e, em seguida, sai.

```
{
    "containerDefinitions": [
        {
            "memory": 32,
            "essential": true,
            "entryPoint": ["ping"],
            "name": "alpine_ping",
            "readonlyRootFilesystem": true,
            "image": "alpine:3.4",
            "command": [
                "-c",
                "4",
                "example.com"
            ],
            "cpu": 16
        }
    ],
    "family": "example_task_2"
}
```

Dependência de contêiner

Este trecho demonstra a sintaxe para uma definição de tarefa com vários contêineres em que a dependência de contêiner é especificada. Na definição de tarefa a seguir, o contêiner `envoy` deve alcançar um status íntegro, determinado pelos parâmetros de verificação de integridade de contêiner necessários, antes que o contêiner `app` seja iniciado. Para obter mais informações, consulte Dependência de contêiner.

```
{
  "family": "appmesh-gateway",
  "runtimePlatform": {
        "operatingSystemFamily": "LINUX"
  },
  "proxyConfiguration":{
      "type": "APPMESH",
```

```
      "containerName": "envoy",
      "properties": [
          {
              "name": "IgnoredUID",
              "value": "1337"
          },
          {
              "name": "ProxyIngressPort",
              "value": "15000"
          },
          {
              "name": "ProxyEgressPort",
              "value": "15001"
          },
          {
              "name": "AppPorts",
              "value": "9080"
          },
          {
              "name": "EgressIgnoredIPs",
              "value": "169.254.170.2,169.254.169.254"
          }
      ]
  },
  "containerDefinitions": [
    {
      "name": "app",
      "image": "application_image",
      "portMappings": [
        {
          "containerPort": 9080,
          "hostPort": 9080,
          "protocol": "tcp"
        }
      ],
      "essential": true,
      "dependsOn": [
        {
          "containerName": "envoy",
          "condition": "HEALTHY"
        }
      ]
    },
    {
```

```
      "name": "envoy",
      "image": "840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-
envoy:v1.15.1.0-prod",
      "essential": true,
      "environment": [
        {
          "name": "APPMESH_VIRTUAL_NODE_NAME",
          "value": "mesh/meshName/virtualNode/virtualNodeName"
        },
        {
          "name": "ENVOY_LOG_LEVEL",
          "value": "info"
        }
      ],
      "healthCheck": {
        "command": [
          "CMD-SHELL",
          "echo hello"
        ],
        "interval": 5,
        "timeout": 2,
        "retries": 3
      }
    }
  ],
  "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
  "networkMode": "awsvpc"
}
```

Volumes nas definições de tarefa

Use as informações a seguir para entender como especificar volumes em tarefas.

- Para obter informações sobre como configurar um volume do Amazon EBS, consulte Especificar a configuração de volumes do Amazon EBS na implantação do Amazon ECS.
- Para obter informações sobre como configurar um volume do Amazon EFS, consulte Configurar os sistemas de arquivos do Amazon EFS para o Amazon ECS usando o console.
- Para obter informações sobre como configurar um volume do FSx para Windows File Server, consulte Saiba como configurar sistemas de arquivos do FSx para Windows File Server para o Amazon ECS.

- Para obter informações sobre como configurar um volume do Docker, consulte Exemplos de volume do Docker para o Amazon ECS.
- Para obter mais informações sobre como configurar uma montagem de associação, consulte Exemplos de montagem de associação para o Amazon ECS.

Definições de tarefa de exemplo do Windows

Veja a seguir um exemplo de definição de tarefa para ajudar você nos conceitos básicos dos contêineres do Windows no Amazon ECS.

Example Exemplo de aplicação de console do Amazon ECS para Windows

A definição de tarefa a seguir é o exemplo de aplicação de console do Amazon ECS produzida no assistente da primeira execução para o Amazon ECS. Ela foi convertida para usar a imagem de contêiner do `microsoft/iis` Windows.

```
{
  "family": "windows-simple-iis",
  "containerDefinitions": [
    {
      "name": "windows_sample_app",
      "image": "mcr.microsoft.com/windows/servercore/iis",
      "cpu": 1024,
      "entryPoint":["powershell", "-Command"],
      "command":["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file -
Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>'; C:\\ServiceMonitor.exe
 w3svc"],
      "portMappings": [
        {
          "protocol": "tcp",
          "containerPort": 80
        }
      ],
      "memory": 1024,
      "essential": true
    }
  ],
  "networkMode": "awsvpc",
```

```
  "memory": "1024",
  "cpu": "1024"
}
```

Clusters do Amazon ECS

Um cluster do Amazon ECS é um agrupamento lógico de tarefas ou serviços. Além de tarefas e serviços, um cluster consiste nos recursos a seguir:

- A capacidade da infraestrutura, que pode ser uma combinação de qualquer uma das seguintes:
  - Instâncias do Amazon EC2 na nuvem da AWS
  - Tecnologia sem servidor (AWS Fargate) na nuvem da AWS
  - Máquinas virtuais (VM) ou servidores on-premises
- A rede (VPC e sub-rede) em que suas tarefas e serviços são executados

  Quando você usa instâncias do Amazon EC2 para a capacidade, a sub-rede pode estar em zonas de disponibilidade, zonas locais, zonas do Wavelength ou AWS Outposts.
- Um namespace opcional

  O namespace é usado para comunicação entre serviços com o Service Connect.
- Uma opção de monitoramento

  O CloudWatch Container Insights tem um custo adicional e é um serviço totalmente gerenciado. Ele coleta, agrega e resume automaticamente métricas e logs do Amazon ECS.

Veja a seguir os conceitos gerais sobre clusters do Amazon ECS.

- Crie clusters adicionais para separar seus recursos.
- Os clusters são específicos da Região da AWS.
- Os clusters podem estar em qualquer um dos estados a seguir.

  ACTIVE

  O cluster está pronto para aceitar tarefas e, se for aplicável, você pode registrar instâncias de contêiner com o cluster.

  PROVISIONING

  O cluster tem provedores de capacidade associados a ele e os recursos necessários para o provedor de capacidade estão sendo criados.

DEPROVISIONING

O cluster tem provedores de capacidade associados a ele e os recursos necessários para o provedor de capacidade estão sendo excluídos.

COM FALHA

O cluster tem provedores de capacidade associados a ele e os recursos necessários para o provedor de capacidade apresentaram falha na criação.

INACTIVE

O cluster foi excluído. Os clusters com um status `INACTIVE` podem permanecer detectáveis em sua conta por um período. Esse comportamento está sujeito a alterações no futuro, portanto, certifique-se de não confiar na persistência de clusters com o estado `INACTIVE`.

- É possível usar diferentes tipos de instância para o tipo de inicialização do EC2 ou provedores de capacidade de grupo do Auto Scaling. Uma instância pode ser registrada somente em um cluster por vez.
- É possível restringir o acesso aos clusters ao criar políticas do IAM personalizadas. Para obter informações, consulte a seção Exemplos de clusters do Amazon ECS em Exemplos de políticas baseadas em identidade do Amazon Elastic Container Service.
- É possível usar o Service Auto Scaling para escalar tarefas do Fargate. Para obter mais informações, consulte Como escalar automaticamente o serviço do Amazon ECS.
- É possível configurar um namespace padrão do Service Connect para um cluster. Depois de definir um namespace padrão do Service Connect, todos os novos serviços criados no cluster podem ser adicionados como serviços de cliente no namespace com a ativação do Service Connect. Não é exigida nenhuma configuração adicional. Para obter mais informações, consulte Uso do Service Connect para conectar serviços do Amazon ECS com nomes abreviados.

Provedores de capacidade

Provedores de capacidade do Amazon ECS gerenciam a escalabilidade da infraestrutura das tarefas dos clusters. Cada cluster pode ter um ou mais provedores de capacidade e uma estratégia opcional de provedor de capacidade. É possível atribuir uma estratégia de provedor de capacidade padrão ao cluster. A estratégia do provedor de capacidade determina como as tarefas são distribuídas entre os provedores de capacidade de clusters. Ao executar uma tarefa autônoma ou criar um serviço, você usa a estratégia de provedor de capacidade padrão do cluster ou uma estratégia de provedor de capacidade que substitua a estratégia padrão do cluster. A estratégia de provedor de

capacidade padrão do cluster só se aplica quando você não especifica um tipo de inicialização ou uma estratégia de provedor de capacidade para sua tarefa ou serviço. Se você fornecer qualquer um desses parâmetros, a estratégia padrão não será usada.

Para o Fargate, você não precisa criar ou gerenciar a capacidade. Para isso, você só precisa associar qualquer um dos seguintes provedores de capacidade definidos previamente ao cluster:

- Fargate
- Fargate Spot

Ao usar instâncias do Amazon EC2 para sua capacidade, você usa grupos do Auto Scaling para gerenciar as instâncias do EC2. O Auto Scaling ajuda a garantir que você tenha o número adequado de instâncias do EC2 disponíveis para lidar com a carga da aplicação.

Um cluster pode conter uma combinação de tarefas hospedadas no AWS Fargate, em instâncias do Amazon EC2 ou em instâncias externas. As tarefas podem ser executadas na infraestrutura do Fargate ou do EC2 como um tipo de inicialização ou uma estratégia de provedor de capacidade. Se você usar o EC2 como um tipo de inicialização, o Amazon ECS não rastreará nem escalará a capacidade dos grupos do Amazon EC2 Auto Scaling. Para obter mais informações sobre tipos de inicialização, consulte Tipos de inicialização do Amazon ECS.

Um cluster pode conter uma combinação de provedores de capacidade de grupo do Auto Scaling e provedores de capacidade do Fargate. Uma estratégia de provedor de capacidade só pode conter provedores de capacidade do grupo do Auto Scaling ou do Fargate.

Clusters do Amazon ECS para Fargate

Provedores de capacidade do Amazon ECS gerenciam a escalabilidade da infraestrutura das tarefas dos clusters. Cada cluster pode ter um ou mais provedores de capacidade e uma estratégia opcional de provedor de capacidade. A estratégia do provedor de capacidade determina como as tarefas são distribuídas entre os provedores de capacidade de clusters. Ao executar uma tarefa autônoma ou criar um serviço, você pode usar a estratégia padrão de provedor de capacidade do cluster ou uma estratégia de provedor de capacidade que substitua a estratégia padrão.

Ao executar suas tarefas no AWS Fargate, não há necessidade de criar ou de gerenciar a capacidade. Para isso, você só precisa associar qualquer um dos seguintes provedores de capacidade definidos previamente ao cluster:

- Fargate

- Fargate Spot

Com o Amazon ECS em provedores de capacidade do AWS Fargate, é possível usar a capacidade do Fargate e do Fargate Spot com as tarefas do Amazon ECS.

Com o Fargate Spot, é possível executar tarefas tolerantes a interrupções do Amazon ECS a uma taxa com desconto em comparação com o preço do Fargate. O Fargate Spot executa tarefas com capacidade adicional de computação. Quando a AWS precisar recuperar a capacidade, as tarefas serão interrompidas com um aviso de dois minutos.

Quando as tarefas que usam os provedores de capacidade do Fargate e do Fargate Spot são interrompidas, o evento de alteração de estado da tarefa é enviado para o Amazon EventBridge. O motivo da interrupção descreve a causa. Para obter mais informações, consulte Eventos de alteração de estado de tarefa do Amazon ECS.

Um cluster pode conter uma combinação de provedores de capacidade do Fargate e do grupo do Auto Scaling. Entretanto, uma estratégia de provedor de capacidade só pode conter provedores de capacidade do Fargate ou do grupo do Auto Scaling, mas não ambos. Para obter mais informações, consulte Auto Scaling Group Capacity Providers.

Leve em consideração o seguinte, quando usar provedores de capacidade:

- Você deve associar um provedor de capacidade a um cluster antes de associá-lo à estratégia do provedor de capacidade.
- É possível especificar, no máximo, 20 provedores de capacidade para uma estratégia do provedor de capacidade.
- Não é possível atualizar um serviço que usa um provedor de capacidade do grupo do Auto Scaling para usar um provedor de capacidade do Fargate. O inverso também é verdadeiro.
- Em uma estratégia de provedor de capacidade, se não houver um valor de `weight` especificado para um provedor de capacidade no console, será usado o valor padrão 1. Se a API ou a AWS CLI estiver sendo usada, será usado o valor padrão `0`.
- Quando vários fornecedores de capacidade são especificados dentro de uma estratégia de provedor de capacidade, pelo menos um dos provedores de capacidade deve ter um valor de peso superior a zero. Quaisquer provedores de capacidade com peso zero não serão usados na atribuição de tarefas. Se você especificar vários provedores de capacidade em uma estratégia em que todos tenham um peso zero, quaisquer ações `RunTask` ou `CreateService` que usarem a estratégia de provedor de capacidade apresentarão falha.

- Somente um provedor de capacidade em uma estratégia de provedor de capacidade pode ter um valor de base definido. Se nenhum valor de base for especificado, será usado o valor padrão de zero.
- Um cluster pode conter uma combinação de provedores de capacidade de grupo do Auto Scaling e provedores de capacidade do Fargate. Entretanto, uma estratégia de provedor de capacidade só pode conter provedores de capacidade do grupo do Auto Scaling ou do Fargate, mas não ambos.
- Um cluster pode conter uma combinação de serviços e tarefas autônomas que usem tanto provedores de capacidade quanto tipos de inicialização. Um serviço pode ser atualizado para usar uma estratégia de provedor de capacidade em vez de um tipo de inicialização. Entretanto, você deve forçar uma nova implantação ao fazer isso.

Avisos de encerramento do Fargate Spot

Durante períodos de demanda extremamente alta, a capacidade do Fargate Spot pode estar indisponível. Isso pode atrasar as tarefas do Fargate Spot. Quando isso acontece, os serviços do Amazon ECS tentam iniciar tarefas novamente até que a capacidade necessária se torne disponível. O Fargate não substitui a capacidade spot pela capacidade sob demanda.

Quando as tarefas que usam a capacidade do Fargate Spot são interrompidas devido a uma interrupção do Spot, um aviso de dois minutos é enviado antes da interrupção de uma tarefa. O aviso é enviado como um evento de alteração de estado da tarefa para o Amazon EventBridge e como um sinal de SIGTERM para a tarefa em execução. Se você usar o Fargate Spot como parte de um serviço, então, nesse cenário, o programador de serviços receberá o sinal de interrupção e tentará iniciar tarefas adicionais no Fargate Spot se houver capacidade disponível. Um serviço com apenas uma tarefa será interrompido até que a capacidade esteja disponível. Para obter mais informações sobre um desligamento normal, consulte Desligamentos normais com o ECS.

Para garantir que seus contêineres saiam normalmente antes que a tarefa seja interrompida, é possível configurar o seguinte:

- Um valor de `stopTimeout` `120` segundos ou menos pode ser especificado na definição de contêiner que a tarefa está usando. O valor padrão de `stopTimeout` é 30 segundos. É possível especificar um valor mais longo de `stopTimeout` para ter mais tempo entre o momento em que o evento de alteração de estado da tarefa é recebido e o instante no qual o contêiner é interrompido forçosamente. Para obter mais informações, consulte Tempos limite de contêiner.

- Para executar ações de limpeza, o sinal de SIGTERM deve ser recebido de dentro do contêiner. A falha ao processar esse sinal fará com que a tarefa receba um sinal de SIGKILL após a configuração do `stopTimeout` e poderá causar perda ou corrupção de dados.

Veja a seguir o trecho de um evento de alteração de estado da tarefa. Esse trecho exibe o motivo e o código de uma interrupção do Fargate Spot.

```
{
  "version": "0",
  "id": "9bcdac79-b31f-4d3d-9410-fbd727c29fab",
  "detail-type": "ECS Task State Change",
  "source": "aws.ecs",
  "account": "111122223333",
  "resources": [
    "arn:aws:ecs:us-east-1:111122223333:task/b99d40b3-5176-4f71-9a52-9dbd6f1cebef"
  ],
  "detail": {
    "clusterArn": "arn:aws:ecs:us-east-1:111122223333:cluster/default",
    "createdAt": "2016-12-06T16:41:05.702Z",
    "desiredStatus": "STOPPED",
    "lastStatus": "RUNNING",
    "stoppedReason": "Your Spot Task was interrupted.",
    "stopCode": "SpotInterruption",
    "taskArn": "arn:aws:ecs:us-east-1:111122223333:task/
b99d40b3-5176-4f71-9a52-9dbd6fEXAMPLE",
    ...
  }
}
```

Veja a seguir um padrão de evento que é usado para criar uma regra do Eventbridge para eventos de alteração de estado da tarefa do Amazon ECS. Opcionalmente, você pode especificar um cluster no campo `detail`. Isso significa que você receberá eventos de alteração do estado da tarefa para esse cluster. Para obter mais informações sobre a criação de uma regra do EventBridge, consulte Conceitos básicos do Amazon EventBridge no Guia do usuário do Amazon EventBridge.

```
{
    "source": [
        "aws.ecs"
    ],
    "detail-type": [
        "ECS Task State Change"
```

```
    ],
    "detail": {
        "clusterArn": [
            "arn:aws:ecs:us-west-2:111122223333:cluster/default"
        ]
    }
}
```

Criação de um cluster do Amazon ECS para o tipo de inicialização do Fargate

Crie um cluster para definir a infraestrutura na qual suas tarefas e serviços são executados.

Antes de começar, é necessário concluir as etapas em Configuração para usar o Amazon ECS e atribuir a permissão adequada do IAM. Para obter mais informações, consulte the section called "Exemplos de clusters do Amazon ECS". O console do Amazon ECS cria os recursos necessários para um cluster do Amazon ECS ao criar uma pilha do AWS CloudFormation.

O console associa automaticamente os provedores de capacidade do Fargate e do Fargate Spot ao cluster.

É possível modificar as opções a seguir:

- Adicione um namespace ao cluster.

  Um namespace permite que os serviços que você cria no cluster possam se conectar a outros serviços no namespace sem configuração adicional. Para obter mais informações, consulte Interconexão de serviços do Amazon ECS.
- Ative o Container Insights ou o Container Insights com observabilidade aprimorada.

  O CloudWatch Container Insights coleta, agrega e resume métricas e logs das suas aplicações e microsserviços conteinerizados. O Container Insights também fornece informações de diagnóstico, como falhas de reinicialização de contêiner, que você usa para isolar problemas e resolvê-los rapidamente. Para obter mais informações, consulte the section called "Monitorar contêineres do Amazon ECS utilizando o Container Insights com observabilidade aprimorada".

  Em 2 de dezembro de 2024, a AWS lançou o Container Insights com observabilidade aprimorada para o Amazon ECS. Essa versão é compatível com a observabilidade aprimorada dos clusters do Amazon ECS usando os tipos de inicialização do Amazon EC2 e do Fargate. Depois de configurar o Container Insights com observabilidade melhorada no Amazon ECS, o Container Insights recolhe

automaticamente a telemetria detalhada da infraestrutura desde o nível do cluster até ao nível do contentor no seu ambiente e apresenta os dados em painéis que mostram uma variedade de métricas e dimensões. Em seguida, é possível utilizar esses painéis prontos para uso no console do Container Insights para compreender melhor a integridade e o desempenho do contêiner e atenuar problemas com mais rapidez, identificando anomalias.

Recomendamos que você use o Container Insights com observabilidade aprimorada em vez do Container Insights, pois ele fornece visibilidade detalhada no ambiente de contêiner, reduzindo o tempo médio de resolução.

- Adicione tags para ajudar a identificar seu cluster.
- Atribua uma chave do AWS KMS ao seu armazenamento gerenciado. Para obter informações sobre como criar uma chave, consulte Criar uma chave do KMS no Guia do usuário do AWS Key Management Service.
- Atribua uma chave do AWS KMS ao armazenamento efêmero do Fargate. Para obter informações sobre como criar uma chave, consulte Criar uma chave do KMS no Guia do usuário do AWS Key Management Service.

Procedimento

Para criar um novo cluster (console do Amazon ECS)

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, selecione a Região a ser usada.
3. No painel de navegação, escolha Clusters.
4. Na página Clusters, escolha Create Cluster (Criar cluster).
5. Em Configuração de cluster, configure o seguinte:
   - Em Nome do cluster, insira um nome exclusivo.

     O nome pode conter até 255 letras (minúsculas e maiúsculas), números e hifens.
   - (Opcional) Para que o namespace usado para o Service Connect seja diferente do nome do cluster, em Namespace, insira um nome exclusivo.
6. (Opcional) Utilize o Container Insights, expanda Monitoramento e escolha uma destas opções:
   - Para usar o Container Insights com observabilidade aprimorada (recomendado), escolha Container Insights com observabilidade aprimorada.

- Para usar o Container Insights, escolha Container Insights.

7. (Opcional), em Criptografia, você pode configurar o seguinte:

   - Criptografe os dados no armazenamento efêmero do Fargate. Em Criptografia, em Armazenamento efêmero do Fargate, insira o ARN da chave do AWS KMS que você deseja usar para criptografar os dados do armazenamento efêmero do Fargate.
   - Criptografe os dados no armazenamento gerenciado. Em Criptografia, em Armazenamento gerenciado, insira o ARN da chave do AWS KMS que você deseja usar para criptografar os dados do armazenamento gerenciado.

8. (Opcional) Para ajudar a identificar seu cluster, expanda Tags (Etiquetas) e configure suas etiquetas.

   [Adicionar uma tag] Selecione Add tag (Adicionar tag) e faça o seguinte:

   - Em Key (Chave), insira o nome da chave.
   - Em Value (Valor), insira o valor da chave.

   [Remover uma tag] Escolha Remover à direita da chave e do valor da tag.

9. Selecione Create (Criar).

Próximas etapas

Após criar o cluster, será possível criar definições de tarefas para suas aplicações e executá-las como tarefas autônomas ou como parte de um serviço. Para obter mais informações, consulte:

- Definições de tarefa do Amazon ECS
- Execução de uma aplicação como uma tarefa do Amazon ECS
- Criação de uma implantação de atualização contínua do Amazon ECS

Provedores de capacidade do Amazon ECS para o tipo de inicialização do EC2

Ao usar instâncias do Amazon EC2 para sua capacidade, você usa grupos do Auto Scaling para gerenciar as instâncias do Amazon EC2 registradas em seus clusters. O Auto Scaling ajuda a

garantir que você tenha o número adequado de instâncias do Amazon EC2 disponíveis para lidar com a carga da aplicação.

É possível usar o recurso de ajuste de escala gerenciado para que o Amazon ECS gerencie as ações reduzir a escala horizontalmente e aumentar a escala horizontalmente do grupo do Auto Scaling ou gerenciar as ações de ajuste de escala por si próprio. Para obter mais informações, consulte Gerenciamento automático da capacidade do Amazon ECS com ajuste de escala automático de cluster.

Recomendamos criar um novo grupo do Auto Scaling vazio. Se você usar um grupo do Auto Scaling existente, todas as instâncias do Amazon EC2 associadas ao grupo que já estavam em execução e registradas em um cluster do Amazon ECS antes de o grupo do Auto Scaling ser usado para criar um provedor de capacidade poderão não estar registradas corretamente no provedor de capacidade. Isso poderá causar problemas quando o provedor de capacidade for usado em uma estratégia de provedor de capacidade. Use `DescribeContainerInstances` para confirmar se uma instância de contêiner está associada a um provedor de capacidade ou não.

> **Note**
>
> Para criar um grupo do Auto Scaling vazio, defina a contagem desejada como zero. Depois da criação do provedor de capacidade e da sua associação a um cluster, é possível aumentar sua escala horizontalmente.
> Ao usar o console do Amazon ECS, o Amazon ECS cria um modelo de inicialização do Amazon EC2 e um grupo do Auto Scaling em seu nome como parte da pilha do AWS CloudFormation. Eles são prefixados com `EC2ContainerService-<ClusterName>`. É possível usar o grupo do Auto Scaling como um provedor de capacidade para aquele cluster.

Recomendamos usar a drenagem gerenciada de instâncias para permitir o encerramento tranquilo de instâncias do Amazon EC2 que não interrompe as workloads. Este recurso está ativado por padrão. Para obter mais informações, consulte Interrupção de workloads do Amazon ECS em execução em instâncias do EC2 com segurança.

Ao usar provedores de capacidade do grupo do Auto Scaling no console, considere o seguinte:

- Um grupo do Auto Scaling deve ter um `MaxSize` maior do que zero para o aumento de escala na horizontal.
- O grupo do Auto Scaling não pode ter configurações de ponderação da instância.

- Se o grupo do Auto Scaling não conseguir aumentar a escala horizontalmente para acomodar o número de tarefas executadas, as tarefas apresentarão falha na transição para além do estado `PROVISIONING`.
- Não modifique o recurso de política de escalabilidade associado aos grupos do Auto Scaling que são gerenciados por provedores de capacidade.
- Se a escalabilidade gerenciada estiver ativada ao ser criado um provedor de capacidade, a contagem desejada do grupo do Auto Scaling poderá ser definida como `0`. Quando a escalabilidade gerenciada está habilitada, o Amazon ECS gerencia as ações do grupo do Auto Scaling para reduzir e aumentar a escala horizontalmente.
- Você deve associar o provedor de capacidade a um cluster antes de associá-lo à estratégia do provedor de capacidade.
- É possível especificar, no máximo, 20 provedores de capacidade para uma estratégia do provedor de capacidade.
- Não é possível atualizar um serviço que usa um provedor de capacidade do grupo do Auto Scaling para usar um provedor de capacidade do Fargate. O inverso também é verdadeiro.
- Em uma estratégia de provedor de capacidade, se não houver um valor de `weight` especificado para um provedor de capacidade no console, será usado o valor padrão 1. Se a API ou a AWS CLI estiver sendo usada, será usado o valor padrão `0`.
- Quando vários fornecedores de capacidade são especificados dentro de uma estratégia de provedor de capacidade, pelo menos um dos provedores de capacidade deve ter um valor de peso superior a zero. Quaisquer provedores de capacidade com peso zero não serão usados na atribuição de tarefas. Se você especificar vários provedores de capacidade em uma estratégia em que todos tenham um peso zero, quaisquer ações `RunTask` ou `CreateService` que usarem a estratégia de provedor de capacidade apresentarão falha.
- Somente um provedor de capacidade em uma estratégia de provedor de capacidade pode ter um valor de base definido. Se nenhum valor de base for especificado, será usado o valor padrão de zero.
- Um cluster pode conter uma combinação de provedores de capacidade de grupo do Auto Scaling e provedores de capacidade do Fargate. Entretanto, uma estratégia de provedor de capacidade só pode conter provedores de capacidade do grupo do Auto Scaling ou do Fargate, mas não ambos.
- Um cluster pode conter uma combinação de serviços e tarefas autônomas que usem tanto provedores de capacidade quanto tipos de inicialização. Um serviço pode ser atualizado para usar uma estratégia de provedor de capacidade em vez de um tipo de inicialização. Entretanto, você deve forçar uma nova implantação ao fazer isso.

- O Amazon ECS oferece suporte a grupos de alta atividade do Amazon EC2 Auto Scaling. Um grupo de alta atividade é um grupo de instâncias do Amazon EC2 pré-inicializadas e prontas para serem colocadas em serviço. Sempre que a aplicação precisa aumentar a escala horizontalmente, o Amazon EC2 Auto Scaling usa as instâncias inicializadas previamente do grupo de aquecimento em vez de iniciar instâncias a frio. Isso permite que qualquer processo de inicialização final seja executado antes de a instância ser atribuída para um serviço. Para obter mais informações, consulte Configuração de instâncias inicializadas previamente para o grupo do Amazon ECS Auto Scaling.

Para obter mais informações sobre a criação de um modelo de execução do Amazon EC2 Auto Scaling, consulte Modelos de execução do Auto Scaling no Guia do usuário do Amazon EC2 Auto Scaling. Para obter mais informações sobre a criação de um grupo do Amazon EC2 Auto Scaling, consulte Grupos do Auto Scaling no Guia do usuário do Amazon EC2 Auto Scaling.

Considerações de segurança sobre a instância de contêiner do Amazon EC2 para o Amazon ECS

Você deve considerar uma única instância de contêiner e seu acesso dentro do seu modelo de ameaças. Por exemplo, uma única tarefa afetada pode aproveitar as permissões do IAM de uma tarefa não infectada na mesma instância.

Recomendamos que você use o indicado a seguir para ajudar a evitar isso:

- Não use privilégios de administrador ao executar suas tarefas.
- Atribua um perfil de tarefa com acesso de privilégio mínimo às suas tarefas.

  O agente de contêiner cria automaticamente um token com um ID de credencial exclusivo que é usado para acessar os recursos do Amazon ECS.
- Para evitar que contêineres executados por tarefas que usem o modo de rede `awsvpc` acessem as informações de credenciais fornecidas ao perfil de instância do Amazon EC2, e ainda permitindo que as permissões fornecidas pelo perfil da tarefa, defina a variável de configuração de agente `ECS_AWSVPC_BLOCK_IMDS` como verdadeira no arquivo de configuração do agente e reinicie o agente.
- Use o monitoramento de runtime do Amazon GuardDuty para detectar ameaças em clusters e contêineres no ambiente da AWS. O monitoramento de runtime usa um agente de segurança do GuardDuty que adiciona visibilidade de runtime às workloads individuais do Amazon ECS,

por exemplo, acesso a arquivos, execução de processos e conexões de rede. Para obter mais informações, consulte GuardDuty Runtime Monitoring no Guia do usuário do GuardDuty.

Criar um cluster do Amazon ECS para o tipo de inicialização do Amazon EC2

Crie um cluster para definir a infraestrutura na qual suas tarefas e serviços são executados.

Antes de começar, é necessário concluir as etapas em Configuração para usar o Amazon ECS e atribuir a permissão adequada do IAM. Para obter mais informações, consulte the section called "Exemplos de clusters do Amazon ECS". O console do Amazon ECS fornece uma maneira simples de criar os recursos necessários para um cluster do Amazon ECS criando uma pilha do AWS CloudFormation.

Para tornar o processo de criação do cluster o mais fácil possível, o console tem seleções padrão para muitas opções, que descrevemos abaixo. Também há painéis de ajuda disponíveis para a maioria das seções do console, que fornecem mais contexto.

É possível registrar instâncias do Amazon EC2 ao criar o cluster ou registrar instâncias adicionais no cluster após ele ter sido criado.

É possível modificar as opções padrão a seguir:

- Alterar as sub-redes em que suas instâncias são executadas
- Alterar os grupos de segurança usados para controlar o tráfego para as instâncias de contêiner
- Adicione um namespace ao cluster.

  Um namespace permite que os serviços que você cria no cluster possam se conectar a outros serviços no namespace sem configuração adicional. Para obter mais informações, consulte Interconexão de serviços do Amazon ECS.
- Ative o Container Insights ou o Container Insights com observabilidade aprimorada.

  O CloudWatch Container Insights coleta, agrega e resume métricas e logs das suas aplicações e microsserviços conteinerizados. O Container Insights também fornece informações de diagnóstico, como falhas de reinicialização de contêiner, que você usa para isolar problemas e resolvê-los rapidamente. Para obter mais informações, consulte the section called "Monitorar contêineres do Amazon ECS utilizando o Container Insights com observabilidade aprimorada".

Em 2 de dezembro de 2024, a AWS lançou o Container Insights com observabilidade aprimorada para o Amazon ECS. Essa versão é compatível com a observabilidade aprimorada dos clusters do Amazon ECS usando os tipos de inicialização do Amazon EC2 e do Fargate. Depois de configurar o Container Insights com observabilidade melhorada no Amazon ECS, o Container Insights recolhe automaticamente a telemetria detalhada da infraestrutura desde o nível do cluster até ao nível do contentor no seu ambiente e apresenta os dados em painéis que mostram uma variedade de métricas e dimensões. Em seguida, é possível utilizar esses painéis prontos para uso no console do Container Insights para compreender melhor a integridade e o desempenho do contêiner e atenuar problemas com mais rapidez, identificando anomalias.

Recomendamos que você use o Container Insights com observabilidade aprimorada em vez do Container Insights, pois ele fornece visibilidade detalhada no ambiente de contêiner, reduzindo o tempo médio de resolução.

- Atribua uma chave do AWS KMS ao seu armazenamento gerenciado. Para obter informações sobre como criar uma chave, consulte Criar uma chave do KMS no Guia do usuário do AWS Key Management Service.
- Atribua uma chave do AWS KMS ao armazenamento efêmero do Fargate. Para obter informações sobre como criar uma chave, consulte Criar uma chave do KMS no Guia do usuário do AWS Key Management Service.
- Adicione tags para ajudar a identificar seu cluster.

Opções do grupo do Auto Scaling

Ao usar instâncias do Amazon EC2, você deve especificar um grupo do Auto Scaling para gerenciar a infraestrutura em que suas tarefas e serviços são executados.

Quando você escolhe criar um novo grupo do Auto Scaling, ele é configurado automaticamente para o seguinte comportamento:

- O Amazon ECS gerencia as ações de redução e aumento de escala na horizontal do grupo do Auto Scaling.
- O Amazon ECS não impedirá que as instâncias do Amazon EC2 que contenham tarefas e estejam em um grupo do Auto Scaling sejam terminadas durante uma ação de redução de escala na horizontal. Para obter mais informações, consulte Proteção de instância no Guia do usuário do AWS Auto Scaling.

Você configura as seguintes propriedades do grupo do Auto Scaling que determinam o tipo e o número de instâncias a serem iniciadas para o grupo:

- As AMIs otimizadas para o Amazon ECS
- O tipo de instância.
- O par de chaves de SSH que prova sua identidade quando você se conecta à instância. Para obter informações sobre como criar chaves SSH, consulte Pares de chaves do Amazon EC2 e instância do Linux no Manual do usuário do Amazon EC2.
- O número mínimo de instâncias a iniciar para o grupo do Auto Scaling.
- O número máximo de instâncias iniciadas para o grupo do Auto Scaling.

  Para que o grupo sofra aumento de escala na horizontal, o máximo deve ser maior que 0.

O Amazon ECS cria um modelo de execução do Amazon EC2 Auto Scaling e um grupo do Auto Scaling em seu nome como parte da pilha do AWS CloudFormation. Os valores especificados para a AMI, os tipos de instância e o par de chaves de SSH fazem parte do modelo de inicialização. Os modelos possuem o prefixo `EC2ContainerService-<ClusterName>`, o que os torna fáceis de identificar. Os grupos do Auto Scaling são prefixados com `<ClusterName>-ECS-Infra-ECSAutoScalingGroup`.

As instâncias iniciadas para o grupo do Auto Scaling usam o modelo de inicialização.

Opções de rede

Por padrão, as instâncias são iniciadas nas sub-redes padrão da região. São usados os grupos de segurança, que controlam o tráfego para as instâncias de contêiner, atualmente associados às sub-redes. É possível alterar as sub-redes e os grupos de segurança das instâncias.

É possível escolher uma sub-rede existente. É possível usar um grupo de segurança existente ou criar um novo. Ao criar um novo grupo de segurança, você precisa especificar pelo menos uma regra de entrada.

As regras de entrada determinam qual tráfego pode alcançar suas instâncias de contêiner e incluem as propriedades a seguir:

- O protocolo a permitir
- O intervalo de portas a permitir
- O tráfego de entrada (origem)

Para permitir o tráfego de entrada de um endereço ou bloco CIDR específico, use Personalizada para Origem com o CIDR permitido.

Para permitir o tráfego de entrada de todos os destinos, use Qualquer lugar para Origem. Essa opção adiciona automaticamente o bloco CIDR IPv4 0.0.0.0/0 e o bloco CIDR IPv6 ::/0.

Para permitir o tráfego de entrada do seu computador local, use Grupo de origem para Origem. Isso adiciona automaticamente o endereço IP atual de seu computador local como a origem permitida.

Para criar um novo cluster (console do Amazon ECS)

Antes de começar, atribua a permissão apropriada do IAM. Para obter mais informações, consulte the section called “Exemplos de clusters do Amazon ECS”.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, selecione a Região a ser usada.
3. No painel de navegação, escolha Clusters.
4. Na página Clusters, escolha Create Cluster (Criar cluster).
5. Em Configuração de cluster, configure o seguinte:
   - Em Nome do cluster, insira um nome exclusivo.

     O nome pode conter até 255 letras (minúsculas e maiúsculas), números e hifens.
   - (Opcional) Para que o namespace usado para o Service Connect seja diferente do nome do cluster, em Namespace, insira um nome exclusivo.
6. Adicione instâncias do Amazon EC2 ao seu cluster, expanda Infraestrutura e selecione Instâncias do Amazon EC2.

   Em seguida, configure o grupo do Auto Scaling que atua como o provedor de capacidade:

   a. Para usar um grupo do Auto Scaling existente, em Auto Scaling group (ASG) (Grupo do Auto Scaling (ASG)), selecione o grupo.

   b. Para criar um grupo do Auto Scaling, a partir de Auto Scaling group (ASG) (Grupo do Auto Scaling (ASG)), selecione Create new group (Criar novo grupo) e, em seguida, forneça os seguintes detalhes sobre o grupo:

      - Em Modelo de provisionamento, escolha se deseja usar instâncias Sob demanda ou instâncias Spot.

- Caso opte por usar instâncias spot, em Estratégia de alocação, escolha quais grupos de capacidade spot (tipos de instância e zonas de disponibilidade) serão usados nas instâncias.

  Na maioria das workloads, você pode escolher Otimizada para capacidade de preço.

  Para obter mais informações, consulte Estratégias de alocação para Instâncias spot no Manual do usuário do Amazon EC2.
- Em Imagem de máquina das Amazon (AMI) da instância de contêiner, escolha a AMI otimizada para o Amazon ECS para as instâncias do grupo do Auto Scaling.
- Em EC2 instance type (Tipo de instância do EC2), escolha o tipo de instância para suas workloads.

  A escalabilidade gerenciada funciona melhor se o grupo do Auto Scaling usa os mesmos tipos de instância ou semelhantes.
- Em Perfil de instância do EC2, escolha um perfil de instância de contêiner existente ou crie um novo.

  Para obter mais informações, consulte Função do IAM de instância de contêiner do Amazon ECS.
- Em Capacity (Capacidade), insira o número mínimo e o número máximo de instâncias a serem iniciadas no grupo do Auto Scaling.
- Em SSH key pair (Par de chaves de SSH), escolha o par que prova sua identidade quando você se conecta à instância.
- Para permitir imagens e armazenamento maiores, em Tamanho do volume raiz do EBS, insira o valor em GiB.

7. (Opcional) Para alterar a VPC e as sub-redes, em Redes para instâncias do Amazon EC2, execute qualquer uma das operações a seguir:

   - Para remover uma sub-rede, em Subnets (Sub-redes), escolha X para cada sub-rede que você deseja remover.
   - Para mudar para uma VPC diferente da VPC padrão, em VPC, escolha uma VPC existente e, depois, em Sub-redes, selecione as sub-redes.
   - Escolha os grupos de segurança. Em Grupo de segurança, escolha uma das seguintes opções:

- Para usar um grupo de segurança existente, escolha Usar um grupo de segurança existente e, em seguida, selecione o grupo de segurança.
- Para criar um grupo de segurança, escolha Criar um novo grupo de segurança. Em seguida, escolha Adicionar regra para cada regra de entrada.

  Para obter informações sobre regras de entrada, consulte Opções de rede.

- Para atribuir automaticamente endereços IP públicos às suas instâncias de contêiner do Amazon EC2, em Atribuir automaticamente IP público, escolha uma das seguintes opções:
  - Usar configuração de sub-rede: atribui um endereço IP público às instâncias quando a sub-rede na qual as instâncias são executadas for uma sub-rede pública.
  - Ativar: atribui um endereço IP público às instâncias.

8. (Opcional) Utilize o Container Insights, expanda Monitoramento e escolha uma destas opções:

   - Para usar o Container Insights com observabilidade aprimorada (recomendado), escolha Container Insights com observabilidade aprimorada.
   - Para usar o Container Insights, escolha Container Insights.

9. (Opcional)

   Se você usa o monitoramento de runtime com a opção manual e deseja que esse cluster seja monitorado pelo GuardDuty, escolha Adicionar tag e faça o seguinte:

   - Em Chave, insira **`guardDutyRuntimeMonitoringManaged`**.
   - Em Valor, insira **`true`**.

10. (Opcional) Criptografe os dados no armazenamento gerenciado. Em Criptografia, em Armazenamento gerenciado, insira o ARN da chave do AWS KMS que você deseja usar para criptografar os dados do armazenamento gerenciado.
11. (Opcional) Para gerenciar as tags de cluster, expanda Tags e, em seguida, execute uma das seguintes operações:

    [Adicionar uma tag] Selecione Add tag (Adicionar tag) e faça o seguinte:

    - Em Key (Chave), insira o nome da chave.
    - Em Value (Valor), insira o valor da chave.

    [Remover uma tag] Escolha Remover à direita da chave e do valor da tag.

12. Selecione Create (Criar).

Próximas etapas

Após criar o cluster, será possível criar definições de tarefas para suas aplicações e executá-las como tarefas autônomas ou como parte de um serviço. Para obter mais informações, consulte:

- Definições de tarefa do Amazon ECS
- Execução de uma aplicação como uma tarefa do Amazon ECS
- Criação de uma implantação de atualização contínua do Amazon ECS

Gerenciamento automático da capacidade do Amazon ECS com ajuste de escala automático de cluster

O Amazon ECS pode gerenciar a escalabilidade de instâncias do Amazon EC2 registradas no seu cluster. Isso se chama ajuste de escala automático do cluster do Amazon ECS. Você ativa o ajuste de escala gerenciado ao criar o provedor de capacidade do grupo do Amazon ECS Auto Scaling. Em seguida, você define uma porcentagem para o destino (a `targetCapacity`) para a utilização da instância neste grupo do Auto Scaling. O Amazon ECS cria duas métricas personalizadas do CloudWatch e uma política de ajuste de escala de rastreamento de destino para o grupo do Auto Scaling. O Amazon ECS gerencia as ações reduzir a escala horizontalmente e aumentar a escala horizontalmente com base na utilização de recursos por parte das suas tarefas.

Para cada provedor de capacidade do grupo do Auto Scaling associado a um cluster, o Amazon ECS cria e gerencia os seguintes recursos:

- Um alarme do CloudWatch de baixo valor métrico
- Um alarme do CloudWatch de alto valor métrico
- Uma política de escalabilidade de rastreamento de destino.

> **Note**
>
> O Amazon ECS cria a política de escalabilidade de rastreamento de destino e a anexa ao grupo do Auto Scaling. Para atualizar a política de escalabilidade de rastreamento de destino, atualize as configurações de escalabilidade gerenciadas pelo provedor de capacidade em vez de atualizar a política de escalabilidade diretamente.

Quando a escalabilidade gerenciada é desativada ou o provedor de capacidade é desassociado de um cluster, o Amazon ECS remove as métricas do CloudWatch e os recursos da política de escalabilidade de rastreamento de destino.

O Amazon ECS usa as seguintes métricas para determinar quais ações realizar:

`CapacityProviderReservation`

A porcentagem de instâncias de contêiner em uso para um provedor de capacidade específico. O Amazon ECS gera essa métrica.

O Amazon ECS define o valor `CapacityProviderReservation` como um número entre 0 e 100. O Amazon ECS usa a fórmula a seguir para representar a proporção de quanta capacidade permanece no grupo do Auto Scaling. Em seguida, o Amazon ECS publica a métrica no CloudWatch. Para obter mais informações sobre como a métrica é calculada, consulte Deep Dive on Amazon ECS Cluster Auto Scaling.

```
CapacityProviderReservation = (number of instances needed) / (number of running
 instances) x 100
```

`DesiredCapacity`

A quantidade de capacidade para o grupo do Auto Scaling. Esta métrica não é publicada no CloudWatch.

O Amazon ECS publica a métrica `CapacityProviderReservation` no CloudWatch, no namespace `AWS/ECS/ManagedScaling`. A métrica `CapacityProviderReservation` faz com que ocorra uma das seguintes ações:

O valor **`CapacityProviderReservation`** é igual a **`targetCapacity`**

O grupo do Auto Scaling não precisa aumentar nem reduzir a escala horizontalmente. O percentual de utilização pretendido foi atingido.

O valor **`CapacityProviderReservation`** é maior que **`targetCapacity`**

Há mais tarefas usando um percentual maior da capacidade do que seu percentual de `targetCapacity`. O aumento do valor da métrica `CapacityProviderReservation` faz com que o alarme do CloudWatch associado se ative. Esse alarme atualiza o valor `DesiredCapacity` para o grupo do Auto Scaling. O grupo do Auto Scaling usa esse valor para iniciar instâncias do EC2 e registrá-las no cluster.

Quando a `targetCapacity` for o valor padrão de 100%, as novas tarefas ficam no estado `PENDING` durante o aumento de escala na horizontal, pois não há capacidade disponível nas instâncias para executar as tarefas. Depois que as novas instâncias se registrarem no ECS, essas tarefas serão iniciadas nas novas instâncias.

O valor **`CapacityProviderReservation`** é menor que **`targetCapacity`**

Há menos tarefas usando um percentual menor da capacidade do que seu percentual de `targetCapacity`, e há pelo menos uma instância que pode ser encerrada. O valor reduzido da métrica `CapacityProviderReservation` faz com que o alarme do CloudWatch associado se ative. Esse alarme atualiza o valor `DesiredCapacity` para o grupo do Auto Scaling. O grupo do Auto Scaling usa esse valor para terminar instâncias de contêiner do EC2 e, em seguida, cancelar seus registros no cluster.

O grupo do Auto Scaling segue as políticas de término de grupo para determinar quais instâncias ele encerrará primeiro durante os eventos de redução de escala na horizontal. Além disso, ele evita instâncias com a configuração de proteção contra redução de escala na horizontal de instâncias ativada. O ajuste de escala automático de clusters pode gerenciar quais instâncias têm a configuração de proteção contra redução de escala na horizontal de instância se você ativar a proteção contra encerramento gerenciada. Para obter mais informações sobre proteção contra encerramento gerenciada, consulte Controle das instâncias encerradas pelo Amazon ECS. Para obter mais informações sobre como os grupos do Auto Scaling encerram instâncias, consulte Controlar quais instâncias do Auto Scaling são encerradas durante uma redução de escala na horizontal no Guia do usuário do Amazon EC2 Auto Scaling.

Ao usar o ajuste de escala automático de clusters, leve em consideração o seguinte:

- Não altere nem gerencie a capacidade desejada para o grupo do Auto Scaling associado a um provedor de capacidade com quaisquer políticas de escalabilidade diferentes da que o Amazon ECS gerencia.
- Quando o Amazon ECS aumenta a escala horizontalmente a partir de 0 instâncias, ele inicia 2 instâncias automaticamente.
- O Amazon ECS usa o perfil do IAM vinculado ao serviço `AWSServiceRoleForECS` para as permissões das quais ele precisa para chamar o AWS Auto Scaling em seu nome. Para obter mais informações, consulte Uso de perfis vinculados ao serviço para o Amazon ECS.
- Ao usar provedores de capacidade com grupos do Auto Scaling, é necessária a permissão `autoscaling:CreateOrUpdateTags` para que o usuário, o grupo ou o perfil crie os provedores

de capacidade. Isso ocorre porque o Amazon ECS adiciona uma etiqueta ao grupo do Auto Scaling quando ele o associa ao provedor de capacidade.

> **Important**
>
> Certifique-se de usar ferramentas que não removam a tag `AmazonECSManaged` do grupo do Auto Scaling. Se essa tag for removida, o Amazon ECS não conseguirá gerenciar o ajuste de escala.

- O ajuste de escala automático de clusters não modifica a MinimumCapacity (Capacidade mínima) nem a MaximumCapacity (Capacidade máxima) do grupo. Para que o grupo aumente a escala horizontalmente, o valor de MaximumCapacity deve ser maior que zero.
- Quando o Auto Scaling (escalabilidade gerenciada) está ativado, um provedor de capacidade só pode ser conectado a um cluster ao mesmo tempo. Se o seu provedor de capacidade tiver a escalabilidade gerenciada desativada, será possível associá-lo a vários clusters.
- Quando a escalabilidade gerenciada está desativada, o provedor de capacidade não reduz nem aumenta a escala horizontalmente. É possível usar uma estratégia de provedor de capacidade para equilibrar as tarefas entre provedores de capacidade.
- A estratégia `binpack` é a mais eficiente em termos de capacidade.
- Quando a capacidade pretendida é menos de 100%, a estratégia de posicionamento precisa usar a estratégia `binpack` que deve ter uma ordem maior do que a estratégia `spread`. Isso evita que o provedor de capacidade aumente a escala horizontalmente até que cada tarefa tenha uma instância dedicada ou que o limite seja atingido.

Ativar o ajuste de escala automático de clusters

É possível ativar o ajuste de escala automático de clusters usando o console ou a AWS CLI.

Quando você cria um cluster para o tipo de execução do EC2 usando o console, o Amazon ECS cria um grupo do Auto Scaling em seu nome e define sua capacidade de destino. Para obter mais informações, consulte Criar um cluster do Amazon ECS para o tipo de inicialização do Amazon EC2.

Você também pode criar um grupo do Auto Scaling e depois atribuí-lo a um cluster. Para obter mais informações, consulte Atualização de um provedor de capacidade do Amazon ECS.

Ao usar a AWS CLI após criar o cluster

1. Antes de criar o provedor de capacidade, você precisa criar um grupo do Auto Scaling. Para obter mais informações, consulte Grupos do Auto Scaling no Guia do usuário do Amazon EC2 Auto Scaling.
2. Use `put-cluster-capacity-providers` para modificar o provedor de capacidade do cluster. Para obter mais informações, consulte Ativação do ajuste de escala automático de cluster do Amazon ECS.

Otimização do ajuste de escala automático de cluster do Amazon ECS

Os clientes que executam o Amazon ECS no Amazon EC2 podem aproveitar o ajuste de escala automático de cluster para gerenciar o ajuste de escala de grupos do Amazon EC2 Auto Scaling. Com o ajuste de escala automático de cluster, é possível configurar o Amazon ECS para escalar o grupo do Auto Scaling automaticamente e se concentrar apenas na execução das tarefas. O Amazon ECS garante que o grupo do Auto Scaling reduza ou aumente a escala horizontalmente, conforme necessário, sem a necessidade de intervenção adicional. Os provedores de capacidade do Amazon ECS são usados para gerenciar a infraestrutura do cluster ao garantir que haja instâncias de contêiner suficientes para atender às demandas da aplicação. Para saber como o ajuste de escala automático de cluster funciona internamente, consulte Deep Dive on Amazon ECS Cluster Auto Scaling.

O ajuste de escala automático de cluster depende de uma integração baseada no CloudWatch com o grupo do Auto Scaling para ajustar a capacidade do cluster. Portanto, ele tem uma latência inerente associada a

- Publicar as métricas do CloudWatch,
- O tempo necessário para a métrica `CapacityProviderReservation` violar os alarmes do CloudWatch (altos e baixos)
- O tempo gasto por uma instância do Amazon EC2 recém-iniciada para entrar em modo de operação estável. É possível executar as seguintes ações para tornar o ajuste de escala automático de cluster mais responsivo com a finalidade de obter implantações mais rápidas:

Tamanhos do ajuste de escala para as etapas do provedor de capacidade

Os provedores de capacidade do Amazon ECS aumentarão ou reduzirão as instâncias de contêiner para atender às demandas da aplicação. Por padrão, o número mínimo de instâncias que o Amazon ECS iniciará está definido como um. Isso poderá adicionar mais tempo às implantações, se forem necessárias diversas instâncias para a atribuição das tarefas pendentes. É possível aumentar o

`minimumScalingStepSize` por meio da API do Amazon ECS com a finalidade de aumentar o número mínimo de instâncias em que o Amazon ECS aumenta ou reduz a escala horizontalmente ao mesmo tempo. Um `maximumScalingStepSize` muito baixo pode limitar quantas instâncias de contêiner terão aumento ou redução da escala horizontalmente por vez, o que pode atrasar as implantações.

> **Note**
>
> No momento, essa configuração está disponível somente por meio das APIs `CreateCapacityProvider` ou `UpdateCapacityProvider`.

Período de aquecimento da instância

O período de aquecimento da instância corresponde ao período após o qual uma instância do Amazon EC2 iniciada recentemente pode contribuir para as métricas do CloudWatch relacionadas ao grupo do Auto Scaling. Depois que o período de aquecimento especificado expirar, a instância será contabilizada nas métricas agregadas do grupo do Auto Scaling, e o ajuste de escala automático de cluster continuará com sua próxima iteração de cálculos para estimar o número de instâncias necessárias.

O valor padrão para `instanceWarmupPeriod` é de 300 segundos, e você pode configurar para um valor mais baixo por meio das APIs `CreateCapacityProvider` ou `UpdateCapacityProvider` para obter um ajuste de escala mais responsivo. Recomendamos definir o valor como mais de 60 segundos para evitar provisionamento excessivo.

Capacidade não utilizada

Se o provedor de capacidade não tiver instâncias de contêiner disponíveis para a atribuição de tarefas, ele precisará aumentar (aumentar a escala horizontalmente) a capacidade do cluster ao iniciar instâncias do Amazon EC2 rapidamente e aguardar a inicialização antes de iniciar os contêineres nelas. Isso pode reduzir significativamente a taxa de inicialização de tarefas. Você tem duas opções disponíveis para lidar com isso.

Nesse caso, ter a capacidade não utilizada do Amazon EC2 já iniciada e pronta para executar tarefas aumentará a taxa efetiva de inicialização de tarefas. É possível usar a configuração `Target Capacity` para indicar que deseja manter a capacidade não utilizada em seus clusters. Por exemplo, ao definir a `Target Capacity` como 80%, você indica que o cluster precisa de 20% de capacidade não utilizada disponível em todos os momentos. Essa capacidade não utilizada

pode permitir que qualquer tarefa independente seja iniciada imediatamente, garantindo que a inicialização de tarefas não sofra o controle de utilização. A desvantagem desta abordagem é o potencial aumento dos custos de manutenção da capacidade não utilizada do cluster.

Uma abordagem alternativa que pode ser considerada é adicionar reserva de capacidade ao seu serviço e não ao provedor de capacidade. Isso significa que, em vez de reduzir a configuração `Target Capacity` para iniciar a capacidade não utilizada, é possível aumentar o número de réplicas em seu serviço ao modificar a métrica de ajuste de escala de rastreamento de destino ou os limites de ajuste de escala em etapas do ajuste de escala automático do serviço. Observe que essa abordagem será útil somente para workloads com picos, mas não terá efeito ao implantar novos serviços e escalar de zero a determinado número de tarefas pela primeira vez. Para obter mais informações sobre as políticas de ajuste de escala relacionadas, consulte Target Tracking Scaling Policies ou Step Scaling Policies no Guia do desenvolvedor do Amazon Elastic Container Service.

Comportamento de escalabilidade gerenciada do Amazon ECS

Quando você tem provedores de capacidade do grupo do Auto Scaling que usam ajuste de escala gerenciado, o Amazon ECS estima o número ideal de instâncias a serem adicionadas ao cluster e usa o valor para determinar quantas instâncias solicitar ou liberar.

Comportamento gerenciado de aumento

O Amazon ECS seleciona um provedor de capacidade para cada tarefa com base na estratégia do provedor de capacidade do serviço, da tarefa independente ou do padrão do cluster. O Amazon ECS segue o restante dessas etapas para um único provedor de capacidade.

Tarefas sem uma estratégia de provedor de capacidade são ignoradas pelos provedores de capacidade. Uma tarefa pendente que não tenha uma estratégia de provedor de capacidade não fará com que nenhum provedor de capacidade sofra aumento de escala na horizontal. Tarefas ou serviços não podem definir uma estratégia de provedor de capacidade se essa tarefa ou serviço definir um tipo de inicialização.

Veja a seguir a descrição do comportamento de aumento com mais detalhes.

- Agrupe todas as tarefas de provisionamento desse provedor de capacidade para que cada grupo tenha os mesmos exatos requisitos de recursos.
- Quando você usa vários tipos de instância em um grupo do Auto Scaling, os tipos de instância no grupo do Auto Scaling são ordenadas por seus parâmetros. Esses parâmetros incluem vCPU, memória, interfaces de rede elástica (ENIs), portas e GPUs. Os menores e maiores tipos de instância para cada parâmetro são selecionados. Para obter mais informações sobre como

escolher o tipo de instância, consulte Instâncias de contêiner do Amazon EC2 para o Amazon ECS.

> **Important**
>
> Se um grupo de tarefas tiver requisitos de recursos maiores do que o menor tipo de instância no grupo do Auto Scaling, esse grupo de tarefas não poderá ser executado com esse provedor de capacidade. O provedor de capacidade não escala o grupo do Auto Scaling. As tarefas permanecem no estado `PROVISIONING`.
> Para evitar que as tarefas permaneçam no estado `PROVISIONING`, recomendamos que você crie grupos do Auto Scaling e provedores de capacidade separados para diferentes requisitos mínimos de recursos. Ao executar tarefas ou criar serviços, adicione somente provedores de capacidade à estratégia do provedor de capacidade que possa executar a tarefa no menor tipo de instância no grupo do Auto Scaling. Para outros parâmetros, é possível usar restrições de posicionamento

- Para cada grupo de tarefas, o Amazon ECS calcula o número de instâncias necessárias para executar as tarefas não colocadas. Esse cálculo usa uma estratégia `binpack`. Essa estratégia leva em consideração os requisitos de vCPU, memória, interfaces de rede elásticas (ENI), portas e GPUs das tarefas. Ela também leva em consideração a disponibilidade de recursos das instâncias do Amazon EC2. Os valores para o maior tipo de instância são tratados como a contagem máxima de instâncias calculada. Os valores para o menor tipo de instância são usados como proteção. Se o menor tipo de instância não puder executar pelo menos uma instância da tarefa, o cálculo considerará a tarefa como não compatível. Como resultado, a tarefa será excluída do cálculo do aumento da escala horizontalmente. Quando todas as tarefas não são compatíveis com o menor tipo de instância, o ajuste de escala automático do cluster é interrompido e o valor `CapacityProviderReservation` permanece com o valor `targetCapacity`.
- O Amazon ECS publicará a métrica `CapacityProviderReservation` para o CloudWatch em relação ao `minimumScalingStepSize` se ocorrer qualquer um dos eventos a seguir.
  - A contagem máxima de instâncias calculadas é menor que o tamanho mínimo da etapa de ajuste de escala.
  - O valor mais baixo para `maximumScalingStepSize` ou para a contagem máxima de instâncias calculadas.
- Os alarmes do CloudWatch usam a métrica `CapacityProviderReservation` para provedores de capacidade. Quando a métrica `CapacityProviderReservation` é maior que o valor da `targetCapacity`, os alarmes também aumentam a `DesiredCapacity` do grupo do Auto

Scaling. O valor `targetCapacity` é uma configuração do provedor de capacidade que é enviada para o alarme do CloudWatch durante a fase de ativação da autoescalabilidade do cluster.

O padrão para `targetCapacity` é de 100%.

- O grupo do Auto Scaling inicia instâncias do EC2 adicionais. Para evitar o excesso de provisionamento, o Auto Scaling garante que a capacidade das instâncias do EC2 executadas recentemente esteja estabilizada antes de executar novas instâncias. O Auto Scaling verifica se todas as instâncias existentes passaram pelo `instanceWarmupPeriod` (agora, subtraindo o tempo de inicialização da instância). O aumento horizontal da escala é bloqueado para instâncias que estão dentro do `instanceWarmupPeriod`.

  O número padrão de segundos para o aquecimento de uma instância recém-ativada é 300.

Para obter mais informações, consulte Deep dive on Amazon ECS cluster auto scaling (Análise profunda da autoescalabilidade de clusters do Amazon ECS).

Considerações sobre aumento de escala na horizontal

Considere o seguinte para o processo de aumento de escala na horizontal:

- Embora existam várias restrições de posicionamento, recomendamos usar a restrição de posicionamento de tarefas `distinctInstance`. Isso impede que o processo de aumento de escala horizontalmente seja interrompido porque você está usando uma restrição de posicionamento que não é compatível com as instâncias amostradas.
- A escalabilidade gerenciada funciona melhor se o grupo do Auto Scaling usa os mesmos tipos de instância ou semelhantes.
- Quando um processo de aumento de escala na horizontal for necessário e não houver instâncias de contêiner em execução no momento, o Amazon ECS sempre aumentará a escala horizontalmente para duas instâncias no início e, em seguida, executará processos adicionais de redução ou aumento da escala na horizontal. Qualquer aumento de escala na horizontal adicional aguardará o período de aquecimento da instância. Para processos de redução de escala na horizontal, o Amazon ECS espera 15 minutos após um processo de aumento de escala na horizontal antes de iniciar os processos de redução de escala na horizontal a qualquer momento.
- A segunda etapa de aumento da escala na horizontal precisa aguardar até que o `instanceWarmupPeriod` expire, o que pode afetar o limite geral da escala. Caso precise reduzir esse tempo, certifique-se de que `instanceWarmupPeriod` seja grande o suficiente para que a

instância do EC2 seja executada e inicie o agente do Amazon ECS (o que impede o excesso de provisionamento).

- O ajuste de escala automático de clusters oferece suporte à configuração de execução, aos modelos de execução e a vários tipos de instâncias no grupo do Auto Scaling do provedor de capacidade. Também é possível usar a seleção de tipo de instância baseada em atributos sem vários tipos de instâncias.
- Ao usar um grupo do Auto Scaling com instâncias sob demanda e vários tipos de instância ou instâncias spot, coloque os tipos de instância maiores acima na lista de prioridades e não especifique um peso. Não há suporte para a especificação de um peso, no momento. Para obter mais informações, consulte Grupos do Auto Scaling com vários tipos de instância no Guia do usuário do AWS Auto Scaling.
- O Amazon ECS iniciará o `minimumScalingStepSize` se a contagem máxima de instâncias calculada for menor do que o tamanho mínimo da etapa de escalabilidade ou o que for menor entre `maximumScalingStepSize` e o valor máximo calculado da contagem de instâncias.
- Se um serviço ou `run-task` do Amazon ECS iniciar uma tarefa e as instâncias de contêiner do provedor de capacidade não tiverem recursos suficientes para iniciá-la, o Amazon ECS limitará o número de tarefas com esse status para cada cluster e impedirá que qualquer tarefa ultrapasse esse limite. Para obter mais informações, consulte Cotas de serviço.

Comportamento gerenciado de redução de escala na horizontal

O Amazon ECS monitora instâncias de contêiner para cada provedor de capacidade dentro de um cluster. Quando uma instância de contêiner não executa nenhuma tarefa, ela é considerada vazia e o Amazon ECS inicia o processo de redução de escala na horizontal.

Os alarmes de escalabilidade do CloudWatch exigem 15 pontos de dados (15 minutos) antes do início do processo de escalabilidade para o grupo do Auto Scaling. Depois que o processo de redução de escala na horizontal é iniciado, até que o Amazon ECS precise reduzir o número de instâncias de contêiner registradas, o grupo do Auto Scaling define o valor `DesireCapacity` como sendo maior que uma instância e menor que 50% a cada minuto.

Quando o Amazon ECS solicitar um aumento da escala horizontalmente (quando `CapacityProviderReservation` for maior que 100) enquanto um processo de redução da escala horizontalmente estiver em andamento, o processo de redução da escala horizontalmente será interrompido e começará do início, se necessário.

Veja a seguir a descrição do comportamento de redução de escala na horizontal com mais detalhes:

1. O Amazon ECS calcula o número de instâncias de contêiner vazias. Uma instância de contêiner é considerada vazia mesmo quando houver tarefas de daemon em execução.
2. O Amazon ECS define o valor `CapacityProviderReservation` como um número entre 0 e 100 que usa a fórmula a seguir para representar a proporção de quão grande o grupo do Auto Scaling precisa ser em relação ao quão grande ele realmente é, expressa como um percentual. Em seguida, o Amazon ECS publica a métrica no CloudWatch. Para obter mais informações sobre como a métrica é calculada, consulte Análise profunda do ajuste de escala automático de clusters do Amazon ECS.

   ```
   CapacityProviderReservation = (number of instances needed) / (number of running
    instances) x 100
   ```

3. A métrica `CapacityProviderReservation` gera um alarme do CloudWatch. Esse alarme atualiza o valor `DesiredCapacity` para o grupo do Auto Scaling. Em seguida, ocorre uma das seguintes ações:
   - Se você não usar o encerramento gerenciado do provedor de capacidade, o grupo do Auto Scaling selecionará instâncias do EC2 usando a política de encerramento do grupo do Auto Scaling e encerrará as instâncias até que o número de instâncias do EC2 atinja a `DesiredCapacity`. As instâncias de contêiner são canceladas do registro do cluster.
   - Se todas as instâncias de contêiner usarem proteção contra encerramento gerenciado, o Amazon ECS removerá a proteção de redução de escala na horizontal nas instâncias de contêiner que estiverem vazias. O grupo do Auto Scaling poderá, então, terminar as instâncias do EC2. As instâncias de contêiner são canceladas do registro do cluster.

Controle das instâncias encerradas pelo Amazon ECS

> **Important**
>
> Você deve ativar a proteção contra redução de escala na horizontal de instâncias do ajuste de escala automático no grupo do Auto Scaling para usar o recurso de proteção contra encerramento gerenciada do ajuste de escala automático de clusters.

A proteção gerenciada contra encerramento permite o ajuste de escala automático de cluster para controlar quais instâncias serão encerradas. Quando você usa a proteção gerenciada contra encerramento, o Amazon ECS encerra somente as instâncias do EC2 que não têm tarefas do Amazon ECS em execução. As tarefas executadas por um serviço que use a estratégia de

agendamento `DAEMON` serão ignoradas, e uma instância poderá ser encerrada pelo ajuste de escala automático de clusters mesmo quando a instância estiver executando essas tarefas. Isso ocorre porque todas as instâncias no cluster estão executando essas tarefas.

Primeiro, o Amazon ECS ativa a opção de proteção contra a redução de escala horizontalmente de instâncias para as instâncias do EC2 no grupo do Auto Scaling. Em seguida, o Amazon ECS coloca as tarefas nas instâncias. Quando todas as tarefas diferentes de daemon são interrompidas em uma instância, o Amazon ECS inicia o processo de redução da escala na horizontal e desativa a proteção contra redução da escala na horizontal para a instância do EC2. O grupo do Auto Scaling pode então terminar a instância.

A proteção contra redução de escala na horizontal de instâncias do ajuste de escala automático controla quais instâncias do EC2 podem ser encerradas pelo ajuste de escala automático. Instâncias com o recurso de redução da escala horizontalmente ativado não podem ser encerradas durante o processo de redução da escala horizontalmente. Para obter mais informações sobre a proteção contra redução de escala na horizontal de instâncias do Auto Scaling, consulte Uso de proteção contra redução de escala na horizontal de instâncias no Guia do usuário do Amazon EC2 Auto Scaling.

Você pode definir o percentual de `targetCapacity` para ter capacidade disponível. Isso ajuda a executar tarefas futuras mais rapidamente porque o grupo do Auto Scaling não precisa executar mais instâncias. O Amazon ECS usa o valor da capacidade de destino para gerenciar a métrica do CloudWatch criada pelo serviço. O Amazon ECS gerencia a métrica do CloudWatch. O grupo do Auto Scaling será tratado como um estado estável para que nenhuma ação de ajuste de escala seja necessária. Os valores podem ser de 0 a 100%. Por exemplo, para configurar o Amazon ECS para manter 10% de capacidade livre sobre a usada pelas tarefas do Amazon ECS, defina o valor de capacidade-alvo como 90%. Considere as informações a seguir quando definir o valor da `targetCapacity` em um provedor de capacidade.

- Um valor de `targetCapacity` inferior a 100% representa a quantidade de capacidade livre (instâncias do Amazon EC2) que precisam estar presentes no cluster. Capacidade livre significa que não há tarefas em execução.
- Restrições de posicionamento, como zonas de disponibilidade, sem `binpack` adicional, forçam o Amazon ECS a acabar executando uma tarefa por cada instância, o que pode não ser o comportamento desejado.

É necessário ativar a proteção contra redução de escala na horizontal de instâncias do ajuste de escala automático no grupo Auto Scaling para usar a proteção contra encerramento gerenciada.

Se você não ativar a proteção de redução de escala na horizontal, ativar a proteção contra encerramento gerenciada poderá levar a um comportamento indesejável. Por exemplo, é possível ter instâncias paralisadas em estado de drenagem. Para obter mais informações, consulte Uso de proteção contra redução de escala na horizontal da instância no Guia do usuário do Amazon EC2 Auto Scaling.

Ao usar a proteção contra encerramento com um provedor de capacidade, não execute nenhuma ação manual, como desvincular a instância, no grupo do Auto Scaling associado ao provedor de capacidade. Ações manuais podem interromper a operação de redução de escala na horizontal do provedor de capacidade. Se você desvincular uma instância do grupo do Auto Scaling, precisará também cancelar o registro da instância desvinculada no cluster do Amazon ECS.

Atualizar a proteção gerenciada contra encerramento para provedores de capacidade do Amazon ECS

Ao usar a configuração de proteção gerenciada contra encerramento, é necessário atualizar a configuração para provedores de capacidade existentes.

Console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, escolha o cluster.
3. Na página do cluster, escolha a guia Infraestrutura.
4. Escolha o provedor de capacidade.
5. Escolha Atualizar para modificar as configurações do provedor de capacidade.
6. Em Configurações do grupo do Auto Scaling, ative a opção Proteção gerenciada contra encerramento para habilitar ou desabilitar o recurso.
7. Selecione Atualizar.

AWS CLI

Você pode atualizar a configuração de proteção gerenciada contra encerramento um provedor de capacidade usando o comando `update-capacity-provider`:

Para habilitar a proteção gerenciada contra encerramento:

```
aws ecs update-capacity-provider \
```

```
  --capacity-provider CapacityProviderName \
  --auto-scaling-group-provider
"managedScaling={status=ENABLED,targetCapacity=70,minimumScalingStepSize=1,maximumScalingStepS
```

Para desabilitar a proteção gerenciada contra encerramento:

```
aws ecs update-capacity-provider \
  --capacity-provider CapacityProviderName \
  --auto-scaling-group-provider
"managedScaling={status=ENABLED,targetCapacity=70,minimumScalingStepSize=1,maximumScalingStepS
```

> **Note**
>
> Alguns minutos podem ser necessários para que as alterações entrem em vigor em todo o cluster. Quando a proteção gerenciada contra encerramento é habilitada, as instâncias que já estão executando tarefas serão protegidas contra eventos de redução da escala horizontal. Quando a proteção gerenciada contra encerramento é desabilitada, o sinalizador de proteção será removido das instâncias durante o próximo ciclo de gerenciamento do provedor de capacidade do ECS.

Console para a execução de tarefas

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, escolha o cluster.
3. Na página do cluster, escolha a guia Tarefas.
4. Escolha a tarefa.
5. Em Configuração, alterne Proteção gerenciada contra encerramento para habilitar ou desabilitar o recurso.
6. Escolha Configurar proteção de redução horizontal da escala de tarefas.

   A caixa de diálogo Configurar proteção de redução horizontal da escala de tarefas é exibida

   a. Em Proteção de redução horizontal da escala de tarefas, alterne para Habilitar.
   b. Em Expira em minutos, insira o número de minutos antes que a proteção de redução horizontal da escala de tarefas termine.
   c. Escolha Update (Atualizar)

Ativação do ajuste de escala automático de cluster do Amazon ECS

Você ativa o ajuste de escala automático de cluster, para que o Amazon ECS gerencie o ajuste de escala de instâncias do Amazon EC2 registradas no cluster.

Se você quiser usar o console para ativar o ajuste de escala automático do cluster, consulte Criação de um provedor de capacidade para o Amazon ECS.

Antes de você começar, crie um grupo do Auto Scaling e um provedor de capacidade. Para obter mais informações, consulte the section called “Provedores de capacidade para o tipo de inicialização do EC2”.

Para ativar o ajuste de escala automático de cluster, associe o provedor de capacidade ao cluster e, em seguida, ative o ajuste de escala automático de cluster.

1. Use o comando `put-cluster-capacity-providers` para associar um ou mais provedores de capacidade ao cluster.

   Para adicionar os provedores de capacidade do AWS Fargate, inclua os provedores de capacidade `FARGATE` e `FARGATE_SPOT` na solicitação. Para obter mais informações, consulte `put-cluster-capacity-providers` na Referência de comandos da AWS CLI.

   ```
   aws ecs put-cluster-capacity-providers \
     --cluster ClusterName \
     --capacity-providers CapacityProviderName FARGATE FARGATE_SPOT \
     --default-capacity-provider-strategy capacityProvider=CapacityProvider,weight=1
   ```

   Para adicionar um grupo do Auto Scaling para o tipo de execução do EC2, inclua o nome do grupo do Auto Scaling na solicitação. Para obter mais informações, consulte `put-cluster-capacity-providers` na Referência de comandos da AWS CLI.

   ```
   aws ecs put-cluster-capacity-providers \
     --cluster ClusterName \
     --capacity-providers CapacityProviderName \
     --default-capacity-provider-strategy capacityProvider=CapacityProvider,weight=1
   ```

2. Use o comando `describe-clusters` para verificar se a associação foi bem-sucedida. Para obter mais informações, consulte `describe-clusters` na Referência de comandos da AWS CLI.

```
aws ecs describe-clusters \
  --cluster ClusterName \
  --include ATTACHMENTS
```

3. Use o comando `update-capacity-provider` para ativar a autoescalabilidade gerenciada para o provedor de capacidade. Para obter mais informações, consulte `update-capacity-provider` na Referência de comandos da AWS CLI.

```
aws ecs update-capacity-provider \
  --capacity-providers CapacityProviderName \
  --auto-scaling-group-provider managedScaling=ENABLED
```

Desativação do ajuste de escala automático de cluster do Amazon ECS

Você desativa o ajuste de escala automático de cluster quando precisa de um controle mais granular das instâncias do EC2 registradas em seu cluster.

Para desativar o ajuste de escala automático de cluster para um cluster, é possível desassociar o provedor de capacidade com o ajuste de escala gerenciado ativado no cluster ou atualizar o provedor de capacidade para desativar o ajuste de escala gerenciado.

Desassociação do provedor de capacidade

Use as etapas a seguir para desassociar um provedor de capacidade de um cluster.

1. Use o comando `put-cluster-capacity-providers` para desassociar o provedor de capacidade do grupo do Auto Scaling do cluster. O cluster pode manter a associação com os provedores de capacidade do AWS Fargate. Para obter mais informações, consulte `put-cluster-capacity-providers` na Referência de comandos da AWS CLI.

```
aws ecs put-cluster-capacity-providers \
  --cluster ClusterName \
  --capacity-providers FARGATE FARGATE_SPOT \
  --default-capacity-provider-strategy '[]'
```

   Use o comando `put-cluster-capacity-providers` para desassociar o provedor de capacidade do grupo do Auto Scaling do cluster. Para obter mais informações, consulte `put-cluster-capacity-providers` na Referência de comandos da AWS CLI.

```
aws ecs put-cluster-capacity-providers \
  --cluster ClusterName \
  --capacity-providers [] \
  --default-capacity-provider-strategy '[]'
```

2. Use o comando `describe-clusters` para verificar se a desassociação foi bem-sucedida. Para obter mais informações, consulte `describe-clusters` na Referência de comandos da AWS CLI.

```
aws ecs describe-clusters \
  --cluster ClusterName \
  --include ATTACHMENTS
```

Desativar a escalabilidade gerenciada do provedor de capacidade

Use as etapas a seguir para desativar a escalabilidade gerenciada do provedor de capacidade.

- Use o comando `update-capacity-provider` para desativar a autoescalabilidade gerenciada para o provedor de capacidade. Para obter mais informações, consulte `update-capacity-provider` na Referência de comandos da AWS CLI.

```
aws ecs update-capacity-provider \
  --capacity-providers CapacityProviderName \
  --auto-scaling-group-provider managedScaling=DISABLED
```

Criação de um provedor de capacidade para o Amazon ECS

Depois que a criação do cluster for concluída, será possível criar um novo provedor de capacidade (grupo do Auto Scaling) para o tipo de inicialização do EC2. Os provedores de capacidade ajudam a gerenciar e escalar a infraestrutura para suas aplicações.

Antes de criar o provedor de capacidade, você precisa criar um grupo do Auto Scaling. Para obter mais informações, consulte Grupos do Auto Scaling no Guia do usuário do Amazon EC2 Auto Scaling.

Para criar um provedor de capacidade para o cluster (console do Amazon ECS)

1. Abra o console em https://console.aws.amazon.com/ecs/v2.

2. No painel de navegação, escolha Clusters.
3. Na página Clusters, escolha o cluster.
4. Na página Cluster: ***name*** (Cluster: nome), escolha Infrastructure (Infraestrutura) e, em seguida, escolha Create (Criar).
5. Na página Create capacity providers (Criar provedores de capacidade), configure as opções a seguir.

    a. Em Basic details (Detalhes básicos), em Capacity provider name (Nome do provedor de capacidade), insira um nome exclusivo de provedor de capacidade.

    b. Em Auto Scaling group (grupo do Auto Scaling), em Use an existing Auto Scaling group (Usar um grupo existente do Auto Scaling), escolha o grupo do Auto Scaling.

    c. (Opcional) Para configurar uma política de escalabilidade, em Scaling policies (Políticas de escalabilidade), configure as opções a seguir.

    - Para que o Amazon ECS gerencie as ações de redução e aumento da escala horizontalmente, selecione Turn on managed scaling (Ativar escalabilidade gerenciada).
    - Para evitar que uma instância do EC2 com tarefas do Amazon ECS em execução seja encerrada, selecione Turn on scaling protection (Ativar proteção de escalabilidade).
    - Em Set target capacity (Definir capacidade de destino), insira o valor de destino para a métrica do CloudWatch usada na política de escalabilidade de rastreamento de destino gerenciada pelo Amazon ECS.

6. Escolha Criar.

Atualização de um provedor de capacidade do Amazon ECS

Quando você usa um grupo do Auto Scaling como provedor de capacidade, é possível modificar a política de escalabilidade do grupo.

Para atualizar um provedor de capacidade do cluster (console do Amazon ECS)

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Clusters.
3. Na página Clusters, escolha o cluster.
4. Na página Cluster: ***name*** (Cluster: nome), escolha Infrastructure (Infraestrutura) e, em seguida, escolha Update (Atualizar).

5. Na página Create capacity providers (Criar provedores de capacidade), configure as opções a seguir.

   - Em Grupo do Auto Scaling, em Políticas de escalabilidade, configure as opções a seguir.

     - Para que o Amazon ECS gerencie as ações de redução e aumento da escala horizontalmente, selecione Turn on managed scaling (Ativar escalabilidade gerenciada).
     - Para evitar que instâncias do EC2 com tarefas do Amazon ECS em execução sejam encerradas, selecione Ativar proteção de escalabilidade.
     - Em Set target capacity (Definir capacidade de destino), insira o valor de destino para a métrica do CloudWatch usada na política de escalabilidade de rastreamento de destino gerenciada pelo Amazon ECS.

6. Selecione Atualizar.

Exclusão de um provedor de capacidade do Amazon ECS

Se você tiver terminado de usar um provedor de capacidade de grupo do Auto Scaling, poderá excluí-lo. Depois que o grupo for excluído, o provedor de capacidade do grupo do Auto Scaling fará a transição para o estado `INACTIVE`. Os clusters com um status `INACTIVE` podem permanecer detectáveis em sua conta durante algum tempo. No entanto, esse comportamento está sujeito a alterações no futuro, então você não deve confiar na persistência de provedores de capacidade `INACTIVE`. Antes que o provedor de capacidade de grupo do Auto Scaling seja excluído, o provedor de capacidade deverá ser removido da estratégia de provedor de capacidade de todos os serviços. É possível usar a API `UpdateService` ou o fluxo de trabalho do serviço de atualização no console do Amazon ECS para remover um provedor de capacidade da estratégia do provedor de capacidade de um serviço. Use a opção Forçar uma nova implantação para garantir que todas as tarefas que usam a capacidade de instância do Amazon EC2 fornecida pelo provedor de capacidade sejam transferidas para usar a capacidade dos provedores de capacidade restantes.

Para excluir um provedor de capacidade para o cluster (console do Amazon ECS)

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Clusters.
3. Na página Clusters, escolha o cluster.
4. Na página Cluster: ***name*** (Cluster: nome), escolha Infrastructure (Infraestrutura), o grupo do Auto Scaling e, em seguida, escolha Delete (Excluir).

5. Na caixa de confirmação, insira delete ***nome do grupo do Auto Scaling***.
6. Escolha Excluir.

Interrupção de workloads do Amazon ECS em execução em instâncias do EC2 com segurança

A drenagem gerenciada de instância facilita a interrupção normal de instâncias do Amazon EC2. Isso permite que as workloads sejam interrompidas com segurança e reprogramadas para instâncias sem interrupção. A manutenção e as atualizações da infraestrutura são realizadas sem se preocupar com a interrupção das workloads. Ao usar a drenagem gerenciada de instâncias, você simplifica os fluxos de trabalho de gerenciamento da infraestrutura que exigem a substituição das instâncias do Amazon EC2 e, ao mesmo tempo, garante a resiliência e a disponibilidade das aplicações.

A drenagem gerenciada de instâncias do Amazon ECS funciona com substituições de instâncias do grupo do Auto Scaling. Com base na atualização da instância e na vida útil máxima dela, os clientes podem garantir a conformidade com os requisitos mais recentes de sistema operacional e segurança da capacidade.

A drenagem gerenciada de instâncias pode ser usada somente com provedores de capacidade do Amazon ECS. É possível ativar a drenagem gerenciada de instâncias ao criar ou ao atualizar provedores de capacidade de grupo do Auto Scaling usando o console do Amazon ECS, a AWS CLI ou o SDK.

Os eventos a seguir são cobertos pela drenagem gerenciada de instâncias do Amazon ECS.

- Atualização de instâncias do grupo do Auto Scaling: use a atualização de instâncias para realizar a substituição contínua das instâncias do Amazon EC2 no grupo do Auto Scaling em vez de fazer isso manualmente em lotes. Isso é útil quando você precisa substituir um grande número de instâncias. Uma atualização de instâncias é iniciada por meio do console do Amazon EC2 ou da API `StartInstanceRefresh`. Certifique-se de selecionar `Replace` para ativar a proteção na redução da escala ao chamar `StartInstanceRefresh` se estiver usando a proteção de encerramento gerenciada.
- Vida útil máxima da instância: você pode definir uma vida útil máxima quando se trata de substituir instâncias do grupo do Auto Scaling. Isso é útil para programar instâncias de substituição com base em políticas internas de segurança ou conformidade.
- Redução horizontal de escala no grupo do Auto Scaling: com base em políticas de ajuste de escala e ações de ajuste de escala programadas, o grupo do Auto Scaling oferece suporte ao

ajuste de escala automático de instâncias. Ao usar um grupo do Auto Scaling como provedor de capacidade do Amazon ECS, é possível reduzir a escala horizontalmente para instâncias de grupos do Auto Scaling quando nenhuma tarefa estiver em execução nessas instâncias.

- Verificações de integridade no grupo do Auto Scaling: o grupo do Auto Scaling oferece suporte a várias verificações de integridade para gerenciar o encerramento de instâncias não íntegras.
- Atualizações da pilha do AWS CloudFormation: é possível adicionar um atributo `UpdatePolicy` à pilha do AWS CloudFormation para executar atualizações cumulativas quando o grupo sofre alterações.
- Rebalanceamento de capacidade do spot: o grupo do Auto Scaling tenta substituir, de forma proativa, as instâncias spot que apresentam maior risco de interrupção com base no aviso de rebalanceamento de capacidade do Amazon EC2. O grupo do Auto Scaling encerra a instância antiga quando a substituição é executada e está íntegra. A drenagem gerenciada de instâncias do Amazon ECS drena a instância spot da mesma forma que drena uma instância não spot.
- Interrupção de spot: as instâncias spot são encerradas com um aviso prévio de dois minutos. A drenagem de instâncias gerenciada pelo Amazon ECS coloca a instância em estado de drenagem em resposta.

Ganchos do ciclo de vida do Amazon EC2 Auto Scaling com drenagem gerenciada de instâncias

Os ganchos do ciclo de vida do grupo do Auto Scaling permitem ao cliente criar soluções que são acionadas por determinados eventos no ciclo de vida da instância e executar uma ação personalizada quando esse evento ocorre. Um grupo do Auto Scaling permite até 50 hooks. Diversos ganchos de encerramento podem existir e serem executados em paralelo, e o grupo do Auto Scaling aguardará a conclusão de todos os hooks antes de encerrar uma instância.

Além do encerramento de hook gerenciado pelo Amazon ECS, você pode configurar seus próprios hooks de encerramento do ciclo de vida. Os hooks do ciclo de vida têm uma `default action`, e recomendamos definir `continue` como o padrão para garantir que outros hooks, como o hook gerenciado pelo Amazon ECS, não sejam afetados por erros de hooks personalizados.

Se você já configurou um hook do ciclo de vida de encerramento para um grupo do Auto Scaling e também habilitou a drenagem gerenciada de instâncias do Amazon ECS, ambos os hooks do ciclo de vida serão executados. No entanto, não há garantia para os horários relativos. Os ganchos do ciclo de vida têm uma configuração `default action` para especificar a ação a ser executada quando o tempo limite expirar. Em caso de falhas, recomendamos usar `continue` como resultado padrão no hook personalizado. Isso garante que outros hooks, principalmente os que são gerenciados pelo Amazon ECS, não sejam afetados por erros presentes no hook de ciclo de

vida personalizado. O resultado alternativo de `abandon` faz com que todos os outros hooks sejam ignorados e deve ser evitado. Para obter mais informações sobre os ganchos do ciclo de vida para o grupo do Auto Scaling, consulte Amazon EC2 Auto Scaling lifecycle hooks no Guia do usuário do Amazon EC2 Auto Scaling.

Tarefas e drenagem gerenciada de instâncias

A drenagem gerenciada de instâncias do Amazon ECS usa o recurso de drenagem existente encontrado nas instâncias de contêineres. O recurso de drenagem de instância de contêiner realiza a substituição e interrupção das tarefas de réplica que pertencem a um serviço do Amazon ECS. Uma tarefa independente, como uma invocada por `RunTask`, que está no estado `PENDING` ou `RUNNING`, permanecerá inalterada. Você terá que esperar que as tarefas sejam concluídas ou interrompê-las de forma manual. A instância de contêiner permanece no estado `DRAINING` até que todas as tarefas sejam interrompidas ou até que 48 horas tenham se passado. As tarefas do daemon são as últimas a serem interrompidas depois que todas as tarefas de réplica são interrompidas.

Drenagem gerenciada de instâncias e proteção gerenciada de encerramento

A drenagem gerenciada de instâncias funciona mesmo se o encerramento gerenciado estiver desabilitado. Para obter informações sobre a proteção contra o encerramento gerenciado, consulte Controle das instâncias encerradas pelo Amazon ECS.

A tabela a seguir resume o comportamento de diferentes combinações de encerramento e drenagem gerenciados.

Encerramento gerenciado	Drenagem gerenciada	Outcome
Habilitada	Habilitado	O Amazon ECS protege as instâncias do Amazon EC2 que estão executando tarefas de serem encerradas por

Encerramento gerenciado	Drenagem gerenciada	Outcome
		eventos de redução horizontal de escala. Todas as instâncias que estão sendo encerradas, como aquelas que não têm proteção de encerramento definida, receberam interrupção spot ou são forçadas pela atualização da instância, são drenadas normalmente.

Encerramento gerenciado	Drenagem gerenciada	Outcome
Desabilitado	Habilitada	O Amazon ECS não oferece proteção para as instâncias do Amazon EC2 que executam tarefas contra a redução da escala horizontalmente. No entanto, todas as instâncias que estão sendo encerradas são drenadas normalmente.

Encerramento gerenciado	Drenagem gerenciada	Outcome
Habilitada	Desabilitado	O Amazon ECS protege as instâncias do Amazon EC2 que estão executando tarefas de serem encerradas por eventos de redução horizontal de escala. No entanto, as instâncias ainda podem ser encerradas por interrupção spot ou atualização forçada da instância, ou se não estiverem executando nenhuma tarefa. O Amazon

Encerramento gerenciado	Drenagem gerenciada	Outcome
		ECS não realiza uma drenagem normal dessas instâncias e executa tarefas de serviço de substituição depois que elas são interrompidas.

Encerramento gerenciado	Drenagem gerenciada	Outcome
Desabilitado	Desabilitado	As instâncias do Amazon EC2 podem reduzir a escala horizontalmente ou ser encerradas a qualquer momento, mesmo que estejam executando tarefas do Amazon ECS. O Amazon ECS executará tarefas de serviço de substituição depois que elas forem encerradas.

Drenagem gerenciada de instâncias e drenagem de instância spot

Com a drenagem de instância spot, é possível definir uma variável de ambiente `ECS_ENABLE_SPOT_INSTANCE_DRAINING` no agente do Amazon ECS que habilita que o Amazon ECS atribua uma instância no status de drenagem em resposta à interrupção de spot de dois minutos. A drenagem gerenciada de instâncias do Amazon ECS facilita o desligamento normal de instâncias do Amazon EC2 que estão sendo encerradas por vários motivos, não apenas por interrupções spot. Por exemplo, é possível usar o rebalanceamento de capacidade do Amazon EC2 Auto Scaling para substituir, de forma proativa, a instância spot com risco elevado de interrupção, e a drenagem da instância gerenciada executa o encerramento tranquilo da instância spot que está sendo substituída. Ao usar a drenagem gerenciada de instâncias, não é necessário habilitar a drenagem de instâncias spot separadamente. Portanto, usar `ECS_ENABLE_SPOT_INSTANCE_DRAINING` nos dados do usuário do grupo do Auto Scaling é redundante. Para obter mais informações sobre a drenagem de instância spot, consulte Instâncias spot.

Como funciona a drenagem gerenciada de instâncias com o EventBridge

Os eventos de drenagem gerenciada de instâncias do Amazon ECS são publicados no Amazon EventBridge, e o Amazon ECS cria uma regra gerenciada do EventBridge no barramento padrão da sua conta para oferecer suporte à drenagem gerenciada de instâncias. Você pode filtrar esses eventos para outros serviços da AWS, como o Lambda, o Amazon SNS e o Amazon SQS para monitorar e solucionar problemas.

- O Amazon EC2 Auto Scaling envia um evento para o EventBridge quando um hook de ciclo de vida é invocado.
- Avisos de interrupção spot são publicados no EventBridge.
- O Amazon ECS gera mensagens de erro que você pode recuperar por meio do console e das APIs do Amazon ECS.
- O EventBridge tem mecanismos de repetição incorporados como mitigações para falhas temporárias.

Configuração de provedores de capacidade do Amazon ECS para o desligamento de instâncias com segurança

É possível ativar a drenagem gerenciada de instâncias ao criar ou ao atualizar provedores de capacidade do grupo do Auto Scaling usando o console do Amazon ECS e a AWS CLI.

> **Note**
>
> Por padrão, a drenagem gerenciada de instâncias é ativada quando você cria um provedor de capacidade.

Veja a seguir exemplos de uso da AWS CLI para criar um provedor de capacidade com a drenagem gerenciada de instâncias ativada e habilitar a drenagem gerenciada de instâncias para o provedor de capacidade existente de um cluster.

Criação de um provedor de capacidade com a drenagem gerenciada de instâncias habilitada

Para criar um provedor de capacidade com a drenagem gerenciada de instâncias habilitada, use o comando `create-capacity-provider`. Defina o parâmetro `managedDraining` como `ENABLED`.

```
aws ecs create-capacity-provider \
--name capacity-provider \
--auto-scaling-group-provider '{
  "autoScalingGroupArn": "asg-arn",
  "managedScaling": {
    "status": "ENABLED",
    "targetCapacity": 100,
    "minimumScalingStepSize": 1,
    "maximumScalingStepSize": 1
  },
  "managedDraining": "ENABLED",
  "managedTerminationProtection": "ENABLED",
}'
```

Resposta:

```
{
    "capacityProvider": {
        "capacityProviderArn": "capacity-provider-arn",
        "name": "capacity-provider",
        "status": "ACTIVE",
        "autoScalingGroupProvider": {
            "autoScalingGroupArn": "asg-arn",
            "managedScaling": {
                "status": "ENABLED",
                "targetCapacity": 100,
                "minimumScalingStepSize": 1,
```

```
                "maximumScalingStepSize": 1
            },
            "managedTerminationProtection": "ENABLED"
            "managedDraining": "ENABLED"
        }
    }
}
```

Como habilitar a drenagem gerenciada de instâncias para o provedor de capacidade existente de um cluster

Habilitar a drenagem gerenciada de instâncias para o provedor de capacidade existente de um cluster usa o comando `update-capacity-provider`. Você nota que `managedDraining` atualmente diz `DISABLED` e `updateStatus` diz `UPDATE_IN_PROGRESS`.

```
aws ecs update-capacity-provider \
--name cp-draining \
--auto-scaling-group-provider '{
  "managedDraining": "ENABLED"
}
```

Resposta:

```
{
    "capacityProvider": {
        "capacityProviderArn": "cp-draining-arn",
        "name": "cp-draining",
        "status": "ACTIVE",
        "autoScalingGroupProvider": {
            "autoScalingGroupArn": "asg-draining-arn",
            "managedScaling": {
                "status": "ENABLED",
                "targetCapacity": 100,
                "minimumScalingStepSize": 1,
                "maximumScalingStepSize": 1,
                "instanceWarmupPeriod": 300
            },
            "managedTerminationProtection": "DISABLED",
            "managedDraining": "DISABLED" // before update
        },
        "updateStatus": "UPDATE_IN_PROGRESS", // in progress and need describe again to
 find out the result
```

```
        "tags": [
        ]
    }
}
```

Use o comando `describe-clusters` e inclua `ATTACHMENTS`. O `status` do anexo de drenagem gerenciada de instâncias é `PRECREATED`, e o `attachmentsStatus` geral é `UPDATING`.

```
aws ecs describe-clusters --clusters cluster-name --include ATTACHMENTS
```

Resposta:

```
{
    "clusters": [
        {
            ...

            "capacityProviders": [
                "cp-draining"
            ],
            "defaultCapacityProviderStrategy": [],
            "attachments": [
                # new precreated managed draining attachment
                {
                    "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
                    "type": "managed_draining",
                    "status": "PRECREATED",
                    "details": [
                        {
                            "name": "capacityProviderName",
                            "value": "cp-draining"
                        },
                        {
                            "name": "autoScalingLifecycleHookName",
                            "value": "ecs-managed-draining-termination-hook"
                        }
                    ]
                },

                ...

            ],
```

```
            "attachmentsStatus": "UPDATING"
        }
    ],
    "failures": []
}
```

Quando a atualização estiver concluída, use `describe-capacity-providers` e você verá que `managedDraining` agora é `ENABLED`.

```
aws ecs describe-capacity-providers --capacity-providers cp-draining
```

Resposta:

```
{
    "capacityProviders": [
        {
            "capacityProviderArn": "cp-draining-arn",
            "name": "cp-draining",
            "status": "ACTIVE",
            "autoScalingGroupProvider": {
                "autoScalingGroupArn": "asg-draning-arn",
                "managedScaling": {
                    "status": "ENABLED",
                    "targetCapacity": 100,
                    "minimumScalingStepSize": 1,
                    "maximumScalingStepSize": 1,
                    "instanceWarmupPeriod": 300
                },
                "managedTerminationProtection": "DISABLED",
                "managedDraining": "ENABLED" // successfully update
            },
            "updateStatus": "UPDATE_COMPLETE",
            "tags": []
        }
    ]
}
```

Solução de problemas de esgotamento de instância gerenciada do Amazon ECS

Talvez seja necessário solucionar problemas relacionados com a drenagem gerenciada de instâncias. A seguir, apresentamos um exemplo de problema que você pode encontrar ao utilizá-la e sua resolução.

As instâncias não são encerradas após o tempo de vida máximo da instância ser excedido ao usar o ajuste de escala automático.

Se as instâncias não forem encerradas mesmo depois de atingir e exceder o tempo de vida máximo da instância ao usar um grupo do Auto Scaling, pode ser porque elas estão protegidas contra a redução da escala horizontalmente. É possível desativar o encerramento gerenciado e permitir a drenagem gerenciada para lidar com a reciclagem de instâncias.

Criar recursos para ajuste de escala automático de cluster do Amazon ECS usando o AWS Management Console

Saiba como criar os recursos para o ajuste de escala automático de cluster usando o AWS Management Console. Quando os recursos requerem um nome, usamos o prefixo `ConsoleTutorial` para garantir que todos tenham nomes exclusivos e facilitar sua localização.

Tópicos

- Pré-requisitos
- Etapa 1: criar um cluster do Amazon ECS
- Etapa 2: registrar uma definição de tarefa
- Etapa 3: executar uma tarefa
- Etapa 4: verificar
- Etapa 5: limpar

Pré-requisitos

Este tutorial pressupõe que os seguintes pré-requisitos foram concluídos:

- As etapas em Configuração para usar o Amazon ECS foram concluídas.
- O usuário do IAM tem as permissões necessárias especificadas no exemplo de política do IAM de AmazonECS_FullAccess.
- A função do IAM de instância de contêiner do Amazon ECS é criada. Para obter mais informações, consulte Função do IAM de instância de contêiner do Amazon ECS.
- A função do IAM vinculada ao serviço do Amazon ECS é criada. Para obter mais informações, consulte Uso de perfis vinculados ao serviço para o Amazon ECS.

- A função do IAM vinculada ao serviço do Auto Scaling é criada. Para obter mais informações, consulte Funções vinculadas ao serviço para o Amazon EC2 Auto Scaling no Guia do usuário do Amazon EC2 Auto Scaling.
- Você tem uma VPC e um grupo de segurança criados para uso. Para obter mais informações, consulte the section called “Criar uma nuvem privada virtual”.

Etapa 1: criar um cluster do Amazon ECS

Use as etapas a seguir para criar um cluster do Amazon ECS.

O Amazon ECS cria um modelo de execução do Amazon EC2 Auto Scaling e um grupo do Auto Scaling em seu nome como parte da pilha do AWS CloudFormation.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, selecione Clusters e, em seguida, Criar cluster.
3. Em Configuração do cluster, em Nome do cluster, insira `ConsoleTutorial-cluster`.
4. Em Infraestrutura, desmarque AWS Fargate (sem servidor) e selecione Instâncias do Amazon EC2. Em seguida, configure o grupo do Auto Scaling que atua como o provedor de capacidade.
    - Em Grupo do Auto Scaling (ASG). Selecione Criar novo ASG, e forneça os detalhes a seguir sobre o grupo:
        - Em Sistema/arquitetura operacional, escolha Amazon Linux 2.
        - Em Tipo de instância do EC2, escolha t3.nano.
        - Em Capacity (Capacidade), insira o número mínimo e o número máximo de instâncias a serem iniciadas no grupo do Auto Scaling.
5. (Opcional) Para gerenciar as tags de cluster, expanda Tags e, em seguida, execute uma das seguintes operações:

    [Adicionar uma tag] Selecione Add tag (Adicionar tag) e faça o seguinte:

    - Em Key (Chave), insira o nome da chave.
    - Em Value (Valor), insira o valor da chave.

    [Remover uma tag] Escolha Remover à direita da chave e do valor da tag.
6. Escolha Criar.

Etapa 2: registrar uma definição de tarefa

Para executar uma tarefa no seu cluster, você deve registrar uma definição de tarefa. As definições de tarefa são listas de contêineres agrupados. O exemplo a seguir é uma definição de tarefa simples que usa uma imagem `amazonlinux` do Docker Hub e simplesmente hiberna por 360 segundos. Para obter mais informações sobre os parâmetros de definição de tarefa disponíveis, consulte Definições de tarefa do Amazon ECS.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Task definitions (Definições de tarefa).
3. Escolha Create new task definition (Criar nova definição de tarefa), Create new task definition with JSON (Criar nova definição de tarefa com JSON).
4. Na caixa Editor JSON, cole o conteúdo a seguir.

```
{
    "family": "ConsoleTutorial-taskdef",
    "containerDefinitions": [
        {
            "name": "sleep",
            "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
            "memory": 20,
            "essential": true,
            "command": [
                "sh",
                "-c",
                "sleep infinity"
            ]
        }
    ],
    "requiresCompatibilities": [
        "EC2"
    ]
}
```

5. Escolha Criar.

Etapa 3: executar uma tarefa

Depois de registrar uma definição de tarefa para sua conta, você pode executar uma tarefa no cluster. Para este tutorial, execute cinco instâncias da definição de tarefa `ConsoleTutorial-taskdef` em seu cluster do `ConsoleTutorial-cluster`.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, escolha ConsoleTutorial-cluster.
3. Em Tarefas, escolha Executar nova tarefa.
4. Na seção Ambiente, em Opções de computação, escolha Estratégia do provedor de capacidade.
5. Em Configuração de implantação, para Tipo de aplicação, escolha Tarefa.
6. Escolha ConsoleTutorial-taskdef na lista suspensa Família.
7. Em Tarefas desejadas, insira 5.
8. Escolha Criar.

Etapa 4: verificar

Neste ponto do tutorial, você deve ter um cluster com cinco tarefas em execução e um grupo do Auto Scaling com um provedor de capacidade. O provedor de capacidade tem a escalabilidade gerenciada do Amazon ECS habilitada.

Podemos verificar se tudo está funcionando corretamente visualizando as métricas do CloudWatch, as configurações do grupo do Auto Scaling e, finalmente, a contagem de tarefas de cluster do Amazon ECS.

Para visualizar as métricas do CloudWatch do cluster

1. Abra o console do CloudWatch, em https://console.aws.amazon.com/cloudwatch/.
2. Na barra de navegação na parte superior da tela, selecione uma Região .
3. No painel de navegação, em Métricas, escolha Todas as métricas.
4. Na página Todas as métricas, na guia Procurar, escolha `AWS/ECS/ManagedScaling`.
5. Escolha CapacityProviderName, ClusterName.
6. Marque a caixa de seleção que corresponde ao `ConsoleTutorial-cluster` ClusterName.
7. Na guia Métricas em gráfico, altere Período para 30 segundos e Estatística para Máximo.

O valor exibido no gráfico mostra o valor da capacidade do destino para o provedor de capacidade. Deve começar em `100`, que era o percentual de capacidade alvo que definimos. Você deve vê-lo aumentar até `200`, o que acionará um alarme para a política de escalabilidade de rastreamento de destino. O alarme irá acionar o grupo do Auto Scaling para expansão.

Use as etapas a seguir para visualizar os detalhes do grupo do Auto Scaling e confirmar se a ação de aumento ocorreu.

Para verificar se o grupo do Auto Scaling foi aumentado

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. Na barra de navegação na parte superior da tela, selecione uma Região .
3. No painel de navegação, em Auto Scaling, escolha Auto Scaling Groups (Grupos de Auto Scaling).
4. Escolha o grupo do Auto Scaling `ConsoleTutorial-cluster` criado neste tutorial. Veja o valor em Capacidade desejada e veja as instâncias na guia Gerenciamento de instâncias para confirmar se o seu grupo sofreu aumento de escala na horizontal para duas instâncias.

Use as etapas a seguir para visualizar o cluster do Amazon ECS e confirmar se as instâncias do Amazon EC2 foram registradas no cluster e suas tarefas, transferidas para um status de `RUNNING`.

Para verificar as instâncias no grupo do Auto Scaling

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Clusters.
3. Na página Clusters, escolha o cluster `ConsoleTutorial-cluster`.
4. Na guia Tarefas, confirme que você vê cinco tarefas no status `RUNNING`.

Etapa 5: limpar

Ao concluir este tutorial, limpe os recursos associados a ele para evitar cobranças por recursos que você não está usando. A exclusão de provedores de capacidade e de definições de tarefa não é compatível, mas não há custo associado a esses recursos.

Para limpar os recursos do tutorial

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Clusters.
3. Na página Clusters, escolha ConsoleTutorial-cluster.
4. Na página ConsoleTutorial-cluster, escolha a guia Tarefas e, em seguida, escolha Interromper, Interromper tudo.
5. No painel de navegação, escolha Clusters.
6. Na página Clusters, escolha ConsoleTutorial-cluster.
7. Na parte superior direita da página, escolha Excluir cluster.
8. Na caixa de confirmação, insira excluir ConsoleTutorial-cluster e escolha Excluir.
9. Exclua os grupos do Auto Scaling usando as etapas a seguir.
    a. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
    b. Na barra de navegação na parte superior da tela, selecione uma Região .
    c. No painel de navegação, em Auto Scaling, escolha Auto Scaling Groups (Grupos de Auto Scaling).
    d. Selecione o grupo do Auto Scaling `ConsoleTutorial-cluster` e escolha Ações.
    e. No menu Actions (Ações), escolha Delete (Excluir). Insira excluir na caixa de confirmação e, em seguida, escolha Excluir.

Instâncias de contêiner do Amazon EC2 para o Amazon ECS

Uma instância de contêiner do Amazon ECS corresponde a uma instância do Amazon EC2 que executa o agente de contêiner do Amazon ECS e está registrada em um cluster. Quando você executa tarefas com o Amazon ECS usando o tipo de inicialização do EC2 ou um provedor de capacidade do grupo do Auto Scaling, as tarefas são colocadas nas instâncias de contêiner ativas. Você é responsável pelo gerenciamento e manutenção da instância de contêiner.

Embora você possa criar sua própria AMI de instância do Amazon EC2 que atenda às especificações básicas necessárias para executar workloads em contêineres no Amazon ECS, as AMIs otimizadas para Amazon ECS são pré-configuradas e testadas no Amazon ECS por engenheiros da AWS. É a maneira mais simples de você começar e fazer com que seus contêineres sejam executados na AWS rapidamente.

Ao criar um cluster usando o console, o Amazon ECS cria um modelo de inicialização para as instâncias com a AMI mais recente associada ao sistema operacional selecionado.

Quando você usa o AWS CloudFormation para criar um cluster, o parâmetro SSM faz parte do modelo de inicialização do Amazon EC2 para instâncias do grupo do Auto Scaling. Você pode configurar o modelo para usar um parâmetro dinâmico do Systems Manager para determinar qual AMI otimizada do Amazon ECS deve ser implantada. Esse parâmetro garante que toda vez que você implantar a pilha, ele verificará se há uma atualização disponível que precise ser aplicada às instâncias do EC2. Para obter um exemplo de como usar o parâmetro do Systems Manager, consulte Criar um cluster Amazon ECS com a AMI do Amazon Linux 2023 otimizada para o Amazon ECS no Guia do usuário do AWS CloudFormation.

- Recuperação de metadados da AMI do Linux otimizada para o Amazon ECS
- Recuperação dos metadados da AMI do Bottlerocket otimizada para o Amazon ECS
- Recuperação de metadados da AMI do Windows otimizada para o Amazon ECS

É possível escolher entre os tipos de instância que são compatíveis com sua aplicação. Com instâncias maiores, é possível iniciar mais tarefas ao mesmo tempo. Com instâncias menores, é possível aumentar a escala horizontalmente de maneira mais otimizada para economizar custos. Não é preciso escolher um único tipo de instância do Amazon EC2 que se adapte a todas as aplicações em seu cluster. Em vez disso, você pode criar diversos grupos do Auto Scaling de forma que cada grupo tenha um tipo de instância diferente. Em seguida, é possível criar um provedor de capacidade do Amazon EC2 para cada um desses grupos.

Use as seguintes diretrizes para determinar os tipos de família de instâncias e o tipo de instância a ser usado:

- Elimine os tipos ou as famílias de instâncias que não atendem aos requisitos específicos da aplicação. Por exemplo, se a sua aplicação exigir uma GPU, será possível excluir qualquer tipo de instância que não tenha uma GPU.
- Considere os requisitos, incluindo o throughput e o armazenamento da rede.
- Considere a CPU e a memória. Como regra geral, a CPU e a memória devem ser grandes o suficiente para armazenar pelo menos uma réplica da tarefa que você deseja executar.

Instâncias spot

A capacidade spot pode proporcionar economias de custo significativas em relação às instâncias sob demanda. A capacidade spot é o excesso de capacidade cujo preço é significativamente menor do que a capacidade sob demanda ou a capacidade reservada. A capacidade spot é adequada para workloads de processamento em lote e machine learning, além de ambientes de desenvolvimento e preparação. Em geral, é adequada para qualquer workload que tolere tempo de inatividade temporário.

Entenda as consequências a seguir, pois a capacidade spot pode não estar disponível o tempo todo.

- Durante períodos de demanda extremamente alta, a capacidade spot pode estar indisponível. Isso pode atrasar a execução de instâncias spot do Amazon EC2. Nesses casos, os serviços do Amazon ECS tentam executar tarefas novamente, e os grupos do Amazon EC2 Auto Scaling também tentam iniciar instâncias novamente, até que a capacidade necessária esteja disponível. O Amazon EC2 não substitui a capacidade spot por capacidade sob demanda.
- Quando a demanda geral por capacidade aumenta, as instâncias spot e as tarefas podem ser encerradas com apenas um aviso de dois minutos. Depois que o aviso for enviado, as tarefas devem iniciar um desligamento ordenado, se necessário, antes que a instância seja totalmente encerrada. Isso ajuda a minimizar a possibilidade de erros. Para obter mais informações sobre um desligamento normal, consulte Desligamentos normais com o ECS.

Para ajudar a minimizar a escassez de capacidade Spot, considere as recomendações a seguir:

- Use várias regiões e zonas de disponibilidade: a capacidade spot varia de acordo com a região e a zona de disponibilidade. É possível melhorar a disponibilidade de spot executando suas workloads em várias regiões e zonas de disponibilidade. Se possível, especifique sub-redes em todas as zonas de disponibilidade nas regiões em que você executar suas tarefas e instâncias.
- Use vários tipos de instâncias do Amazon EC2: quando você usa políticas de instâncias mistas com o Amazon EC2 Auto Scaling, vários tipos de instâncias são iniciadas em seu grupo do Auto Scaling. Isso garante que uma solicitação por capacidade spot possa ser atendida quando necessário. Para maximizar a confiabilidade e minimizar a complexidade, use tipos de instâncias com aproximadamente a mesma quantidade de CPU e memória em sua Política de instâncias mistas. Essas instâncias podem ser de uma geração diferente ou de variantes do mesmo tipo de instância base. Observe que elas podem vir com recursos adicionais que você talvez não precise. Um exemplo dessa lista poderia incluir m4.large, m5.large, m5a.large, m5d.large, m5n.large,

m5dn.large e m5ad.large. Para obter mais informações, consulte Grupos de Auto Scaling com vários tipos de instância e opções de compra no Manual do usuário do Amazon EC2 Auto Scaling.

- Use a estratégia de alocação spot otimizada para capacidade: com o Amazon EC2 Spot, é possível escolher entre as estratégias de alocação otimizadas para capacidade e custo. Se você escolher a estratégia de capacidade otimizada ao iniciar uma nova instância, o Amazon EC2 Spot selecionará o tipo de instância com maior disponibilidade na zona de disponibilidade selecionada. Isso ajudará a reduzir a possibilidade de a instância ser encerrada logo após sua inicialização.

Para obter informações sobre como configurar avisos de encerramento de spot em suas instâncias de contêiner, consulte:

- Configuração de instâncias de contêiner do Linux no Amazon ECS para receber avisos de instância spot
- Configuração de instâncias de contêiner do Windows no Amazon ECS para receber avisos de instância spot

AMIs do Linux otimizadas para o Amazon ECS

> **Important**
>
> A AMI do Amazon Linux 2 otimizada para Amazon ECS chega ao fim de sua vida útil em 30 de junho de 2026, espelhando a mesma data de EOL do sistema operacional upstream Amazon Linux 2 (para obter mais informações, consulte as perguntas frequentes sobre o Amazon Linux 2). Incentivamos os clientes a atualizar suas aplicações para usar o Amazon Linux 2023, que inclui suporte de longo prazo até 2028. Para obter informações sobre a migração do Amazon Linux 2 para o Amazon Linux 2023, consulte Migrating from the Amazon Linux 2 Amazon ECS-optimized AMI to the Amazon Linux 2023 Amazon ECS-optimized AMI.

O Amazon ECS fornece as AMIs otimizadas para Amazon ECS que são pré-configuradas com os requisitos e as recomendações para executar as workloads de contêiner. Recomendamos que você use a AMI do Amazon Linux 2023 otimizada para Amazon ECS para suas instâncias do Amazon EC2. A execução das suas instâncias de contêiner na AMI mais recente otimizada para Amazon ECS garante que você receba a versão atual das atualizações de segurança e do agente

de contêiner. Para obter mais informações sobre como iniciar uma instância, consulte Iniciar uma instância de contêiner do Linux do Amazon ECS.

Ao criar um cluster usando o console, o Amazon ECS cria um modelo de inicialização para as instâncias com a AMI mais recente associada ao sistema operacional selecionado.

Quando você usa o AWS CloudFormation para criar um cluster, o parâmetro SSM faz parte do modelo de inicialização do Amazon EC2 para instâncias do grupo do Auto Scaling. Você pode configurar o modelo para usar um parâmetro dinâmico do Systems Manager para determinar qual AMI otimizada do Amazon ECS deve ser implantada. Esse parâmetro garante que toda vez que você implantar a pilha, ele verificará se há uma atualização disponível que precise ser aplicada às instâncias do EC2. Para obter um exemplo de como usar o parâmetro do Systems Manager, consulte Criar um cluster Amazon ECS com a AMI do Amazon Linux 2023 otimizada para o Amazon ECS no Guia do usuário do AWS CloudFormation.

Se você precisar personalizar a AMI otimizada para o Amazon ECS, consulte Amazon ECS Optimized AMI Build Recipes no GitHub.

As seguintes variantes da AMI otimizada para o Amazon ECS estão disponíveis para instâncias do Amazon EC2 com o sistema operacional do Amazon Linux 2023.

Sistema operacional	AMI	Descrição	Configuração de armazenamento
Amazon Linux 2023	AMI do Amazon Linux 2023 otimizada para Amazon ECS	O Amazon Linux 2023 é a próxima geração do Amazon Linux da AWS. Para a maioria dos casos, é recomendado para iniciar suas instâncias do Amazon EC2 para workloads do Amazon ECS. Para obter mais informações, consulte O que é o Amazon Linux 2023 no Guia do Usuário	Por padrão, a AMI do Amazon Linux 2023 otimizada para Amazon ECS é fornecida com um único volume raiz de 30 GiB. É possível modificar o tamanho de 30 GiB do volume raiz no momento da inicialização para aumentar o armazenamento disponível em sua

Sistema operacional	AMI	Descrição	Configuração de armazenamento
		do Amazon Linux 2023.	instância de contêiner. Esse armazenamento é usado para o sistema operacional e imagens e metadados de Docker. O sistema de arquivos padrão para a AMI do Amazon Linux 2023 otimizada para Amazon ECS é `xfs` e o Docker usa o driver de armazenamento `overlay2`. Para obter mais informações, consulte Usar o driver de armazenamento OverlayFS na documentação do Docker.

Sistema operacional	AMI	Descrição	Configuração de armazenamento
Amazon Linux 2023 (arm64)	AMI do Amazon Linux 2023 (arm64) otimizada para Amazon ECS	Com base no Amazon Linux 2023, recomenda-se usar esta AMI ao inicializar suas instâncias do Amazon EC2 para processadores AWS Graviton/Graviton 2/Graviton 3/Graviton 4 baseados em Arm, para suas workloads do Amazon ECS. Para obter mais informações, consulte Especificações para instâncias de uso geral do Amazon EC2 no Guia de tipos de instâncias do Amazon EC2.	Por padrão, a AMI do Amazon Linux 2023 otimizada para Amazon ECS é fornecida com um único volume raiz de 30 GiB. É possível modificar o tamanho de 30 GiB do volume raiz no momento da inicialização para aumentar o armazenamento disponível em sua instância de contêiner. Esse armazenamento é usado para o sistema operacional e imagens e metadados de Docker. O sistema de arquivos padrão para a AMI do Amazon Linux 2023 otimizada para Amazon ECS é `xfs` e o Docker usa o driver de armazenamento `overlay2`. Para obter mais informações, consulte Usar o driver de armazenamento OverlayFS na

Sistema operacional	AMI	Descrição	Configuração de armazenamento
			documentação do Docker.

Sistema operacional	AMI	Descrição	Configuração de armazenamento
Amazon Linux 2023 (Neuron)	AMI do Amazon Linux 2023 otimizada para Amazon ECS	Baseada no Amazon Linux 2023, esta AMI destina-se a instâncias Inf1, Trn1 ou Inf2 do Amazon EC2. Ela vem pré-configurada com drivers do AWS Inferentia e do AWS Trainium, e o runtime do AWS Neuron para Docker, o que facilita a execução de workloads de inferência de machine learning no Amazon ECS. Para obter mais informações, consulte Definições de tarefa do Amazon ECS para workloads de machine learning do AWS Neuron. A AMI do Amazon Linux 2023 (Neuron) otimizada para o Amazon ECS não é fornecida com a AWS CLI instalada previamente.	Por padrão, a AMI do Amazon Linux 2023 otimizada para Amazon ECS é fornecida com um único volume raiz de 30 GiB. É possível modificar o tamanho de 30 GiB do volume raiz no momento da inicialização para aumentar o armazenamento disponível em sua instância de contêiner. Esse armazenamento é usado para o sistema operacional e imagens e metadados de Docker. O sistema de arquivos padrão para a AMI do Amazon Linux 2023 otimizada para Amazon ECS é `xfs` e o Docker usa o driver de armazenamento `overlay2`. Para obter mais informações, consulte Usar o driver de armazenamento OverlayFS na

Sistema operacional	AMI	Descrição	Configuração de armazenamento
			documentação do Docker.

Sistema operacional	AMI	Descrição	Configuração de armazenamento
GPU Amazon Linux 2023	AMI de GPU Amazon Linux 2023 otimizada para Amazon ECS	Com base no Amazon Linux 2023, recomenda-se usar esta AMI na inicialização de instâncias baseadas em GPU do Amazon EC2 para suas workloads do Amazon ECS. Ela vem pré-configurada com drivers de kernel NVIDIA e um runtime da GPU do Docker, que facilitam a execução de workloads que aproveitam as GPUs no Amazon ECS. Para obter mais informações, consulte Definições de tarefa do Amazon ECS para workloads de GPU.	Por padrão, a AMI do Amazon Linux 2023 otimizada para Amazon ECS é fornecida com um único volume raiz de 30 GiB. É possível modificar o tamanho de 30 GiB do volume raiz no momento da inicialização para aumentar o armazenamento disponível em sua instância de contêiner. Esse armazenamento é usado para o sistema operacional e imagens e metadados de Docker. O sistema de arquivos padrão para a AMI do Amazon Linux 2023 otimizada para Amazon ECS é `xfs` e o Docker usa o driver de armazenamento `overlay2`. Para obter mais informações, consulte Usar o driver de armazenamento OverlayFS na

Sistema operacional	AMI	Descrição	Configuração de armazenamento
			documentação do Docker.

As seguintes variantes da AMI otimizada para o Amazon ECS estão disponíveis para instâncias do Amazon EC2 com o sistema operacional do Amazon Linux 2.

Sistema operacional	AMI	Descrição	Configuração de armazenamento
Amazon Linux 2	AMI do Amazon Linux 2 kernel 5.10 otimizada para Amazon ECS	Com base no Amazon Linux 2, esta AMI é para ser usada ao inicializar suas instâncias do Amazon EC2 e quando você quiser usar o kernel do Linux 5.10 em vez do kernel 4.14 para suas workloads do Amazon ECS. A AMI do Amazon Linux 2 kernel 5.10 otimizada para Amazon ECS não vem com a AWS CLI pré-instalada.	Por padrão, as AMIs otimizadas para Amazon ECS baseadas no Amazon Linux 2 (AMI do Amazon Linux 2 otimizado para Amazon ECS, AMI do Amazon Linux 2 (arm64) otimizada para Amazon ECS e AMI otimizada para GPU do Amazon ECS) são enviadas com um único volume raiz de 30 GiB. É possível modificar o tamanho de 30 GiB do volume raiz no momento da inicialização para aumentar o armazenamento disponível em sua instância de contêiner

Sistema operacional	AMI	Descrição	Configuração de armazenamento
			. Esse armazenamento é usado para o sistema operacional e imagens e metadados de Docker. O sistema de arquivos padrão para a AMI do Amazon Linux 2 otimizada para Amazon ECS é `xfs` e o Docker usa o driver de armazenamento `overlay2`. Para obter mais informações, consulte Usar o driver de armazenamento OverlayFS na documentação do Docker.

Sistema operacional	AMI	Descrição	Configuração de armazenamento
Amazon Linux 2	AMI do Amazon Linux 2 otimizada para Amazon ECS	Isso é para suas workloads do Amazon ECS. A AMI do Amazon Linux 2 otimizada para Amazon ECS não vem com a AWS CLI pré-instalada.	Por padrão, as AMIs otimizadas para Amazon ECS baseadas no Amazon Linux 2 (AMI do Amazon Linux 2 otimizado para Amazon ECS, AMI do Amazon Linux 2 (arm64) otimizada para Amazon ECS e AMI otimizada para GPU do Amazon ECS) são enviadas com um único volume raiz de 30 GiB. É possível modificar o tamanho de 30 GiB do volume raiz no momento da inicialização para aumentar o armazenamento disponível em sua instância de contêiner. Esse armazenamento é usado para o sistema operacional e imagens e metadados de Docker. O sistema de arquivos padrão para a AMI do Amazon Linux 2 otimizada para

Sistema operacional	AMI	Descrição	Configuração de armazenamento
			Amazon ECS é `xfs` e o Docker usa o driver de armazenamento `overlay2`. Para obter mais informações, consulte Usar o driver de armazenamento OverlayFS na documentação do Docker.

Sistema operacional	AMI	Descrição	Configuração de armazenamento
Amazon Linux 2 (arm64)	AMI do Amazon Linux 2 kernel 5.10 (arm64) otimizada para Amazon ECS	Com base no Amazon Linux 2, esta AMI é para suas instâncias do Amazon EC2 para processadores AWS Graviton/Graviton 2/ Graviton 3/Graviton 4 baseados em Arm, e quando você quiser usar o kernel 5.10 em vez do kernel 4.14 do Linux para suas workloads do Amazon ECS. Para obter mais informações, consulte Especificações para instâncias de uso geral do Amazon EC2 no Guia de tipos de instâncias do Amazon EC2. A AMI do Amazon Linux 2 (arm64) otimizada para Amazon ECS não vem com a AWS CLI pré-instalada.	Por padrão, as AMIs otimizadas para Amazon ECS baseadas no Amazon Linux 2 (AMI do Amazon Linux 2 otimizado para Amazon ECS, AMI do Amazon Linux 2 (arm64) otimizada para Amazon ECS e AMI otimizada para GPU do Amazon ECS) são enviadas com um único volume raiz de 30 GiB. É possível modificar o tamanho de 30 GiB do volume raiz no momento da inicialização para aumentar o armazenamento disponível em sua instância de contêiner. Esse armazenamento é usado para o sistema operacional e imagens e metadados de Docker. O sistema de arquivos padrão para a AMI do Amazon Linux 2 otimizada para

Sistema operacional	AMI	Descrição	Configuração de armazenamento
			Amazon ECS é `xfs` e o Docker usa o driver de armazenamento `overlay2`. Para obter mais informações, consulte Usar o driver de armazenamento OverlayFS na documentação do Docker.

Sistema operacional	AMI	Descrição	Configuração de armazenamento
Amazon Linux 2 (arm64)	AMI do Amazon Linux 2 (arm64) otimizada para Amazon ECS	Com base no Amazon Linux 2, essa AMI deve ser usada ao inicializar suas instâncias do Amazon EC2 para processadores AWS Graviton/Graviton 2/Graviton 3/Graviton 4 baseados em Arm, para suas workloads do Amazon ECS. A AMI do Amazon Linux 2 (arm64) otimizada para Amazon ECS não vem com a AWS CLI pré-instalada.	Por padrão, as AMIs otimizadas para Amazon ECS baseadas no Amazon Linux 2 (AMI do Amazon Linux 2 otimizado para Amazon ECS, AMI do Amazon Linux 2 (arm64) otimizada para Amazon ECS e AMI otimizada para GPU do Amazon ECS) são enviadas com um único volume raiz de 30 GiB. É possível modificar o tamanho de 30 GiB do volume raiz no momento da inicialização para aumentar o armazenamento disponível em sua instância de contêiner. Esse armazenamento é usado para o sistema operacional e imagens e metadados de Docker. O sistema de arquivos padrão para a AMI do Amazon Linux 2 otimizada para

Sistema operacional	AMI	Descrição	Configuração de armazenamento
			Amazon ECS é `xfs` e o Docker usa o driver de armazenamento `overlay2`. Para obter mais informações, consulte Usar o driver de armazenamento OverlayFS na documentação do Docker.

Sistema operacional	AMI	Descrição	Configuração de armazenamento
Amazon Linux 2 (GPU)	AMI do kernel 5.10 otimizada para GPU do Amazon ECS	Com base no Amazon Linux 2, recomenda-se usar essa AMI na execução de instâncias baseadas em GPU do Amazon EC2 com o kernel 5.10 do Linux nas workloads do Amazon ECS. Ela vem pré-configurada com drivers de kernel NVIDIA e um runtime da GPU do Docker, que facilitam a execução de workloads que aproveitam as GPUs no Amazon ECS. Para obter mais informações, consulte Definições de tarefa do Amazon ECS para workloads de GPU.	Por padrão, as AMIs otimizadas para Amazon ECS baseadas no Amazon Linux 2 (AMI do Amazon Linux 2 otimizado para Amazon ECS, AMI do Amazon Linux 2 (arm64) otimizada para Amazon ECS e AMI otimizada para GPU do Amazon ECS) são enviadas com um único volume raiz de 30 GiB. É possível modificar o tamanho de 30 GiB do volume raiz no momento da inicialização para aumentar o armazenamento disponível em sua instância de contêiner. Esse armazenamento é usado para o sistema operacional e imagens e metadados de Docker. O sistema de arquivos padrão para a AMI do Amazon Linux 2 otimizada para

Sistema operacional	AMI	Descrição	Configuração de armazenamento
			Amazon ECS é `xfs` e o Docker usa o driver de armazenamento `overlay2`. Para obter mais informações, consulte Usar o driver de armazenamento OverlayFS na documentação do Docker.

Sistema operacional	AMI	Descrição	Configuração de armazenamento
Amazon Linux 2 (GPU)	AMI otimizada para GPU do Amazon ECS	Com base no Amazon Linux 2, recomenda-se usar essa AMI na execução de instâncias baseadas em GPU do Amazon EC2 com o kernel 4.14 do Linux nas workloads do Amazon ECS. Ela vem pré-configurada com drivers de kernel NVIDIA e um runtime da GPU do Docker, que facilitam a execução de workloads que aproveitam as GPUs no Amazon ECS. Para obter mais informações, consulte Definições de tarefa do Amazon ECS para workloads de GPU.	Por padrão, as AMIs otimizadas para Amazon ECS baseadas no Amazon Linux 2 (AMI do Amazon Linux 2 otimizado para Amazon ECS, AMI do Amazon Linux 2 (arm64) otimizada para Amazon ECS e AMI otimizada para GPU do Amazon ECS) são enviadas com um único volume raiz de 30 GiB. É possível modificar o tamanho de 30 GiB do volume raiz no momento da inicialização para aumentar o armazenamento disponível em sua instância de contêiner. Esse armazenamento é usado para o sistema operacional e imagens e metadados de Docker. O sistema de arquivos padrão para a AMI do Amazon Linux 2 otimizada para

Sistema operacional	AMI	Descrição	Configuração de armazenamento
			Amazon ECS é `xfs` e o Docker usa o driver de armazenamento `overlay2`. Para obter mais informações, consulte Usar o driver de armazenamento OverlayFS na documentação do Docker.

Sistema operacional	AMI	Descrição	Configuração de armazenamento
Amazon Linux 2 (Neuron)	AMI do kernel 5.10 do Amazon Linux 2 (Neuron) otimizada para o Amazon ECS	Com base no Amazon Linux 2, essa AMI é para instâncias Inf1, Trn1 ou Inf2 do Amazon EC2. Ela vem pré-configurada com drivers do AWS Inferentia com kernel do Linux 5.10 e AWS Trainium e o runtime do AWS Neuron para Docker, que facilita a execução de workloads de inferência de machine learning no Amazon ECS. Para obter mais informações, consulte Definições de tarefa do Amazon ECS para workloads de machine learning do AWS Neuron. A AMI do Amazon Linux 2 (Neuron) otimizada para Amazon ECS não é fornecida com a AWS CLI pré-instalada.	Por padrão, as AMIs otimizadas para Amazon ECS baseadas no Amazon Linux 2 (AMI do Amazon Linux 2 otimizado para Amazon ECS, AMI do Amazon Linux 2 (arm64) otimizada para Amazon ECS e AMI otimizada para GPU do Amazon ECS) são enviadas com um único volume raiz de 30 GiB. É possível modificar o tamanho de 30 GiB do volume raiz no momento da inicialização para aumentar o armazenamento disponível em sua instância de contêiner. Esse armazenamento é usado para o sistema operacional e imagens e metadados de Docker. O sistema de arquivos padrão para a AMI do Amazon Linux 2 otimizada para

Sistema operacional	AMI	Descrição	Configuração de armazenamento
			Amazon ECS é `xfs` e o Docker usa o driver de armazenamento `overlay2`. Para obter mais informações, consulte Usar o driver de armazenamento OverlayFS na documentação do Docker.

Sistema operacional	AMI	Descrição	Configuração de armazenamento
Amazon Linux 2 (Neuron)	AMI do Amazon Linux 2 (Neuron) otimizada para Amazon ECS	Com base no Amazon Linux 2, essa AMI é para instâncias Inf1, Trn1 ou Inf2 do Amazon EC2. Ela vem pré-configurada com drivers do AWS Inferentia e do AWS Trainium, e o runtime do AWS Neuron para Docker, o que facilita a execução de workloads de inferência de machine learning no Amazon ECS. Para obter mais informações, consulte Definições de tarefa do Amazon ECS para workloads de machine learning do AWS Neuron. A AMI do Amazon Linux 2 (Neuron) otimizada para Amazon ECS não é fornecida com a AWS CLI pré-instalada.	Por padrão, as AMIs otimizadas para Amazon ECS baseadas no Amazon Linux 2 (AMI do Amazon Linux 2 otimizado para Amazon ECS, AMI do Amazon Linux 2 (arm64) otimizada para Amazon ECS e AMI otimizada para GPU do Amazon ECS) são enviadas com um único volume raiz de 30 GiB. É possível modificar o tamanho de 30 GiB do volume raiz no momento da inicialização para aumentar o armazenamento disponível em sua instância de contêiner. Esse armazenamento é usado para o sistema operacional e imagens e metadados de Docker. O sistema de arquivos padrão para a AMI do Amazon Linux 2 otimizada para

Sistema operacional	AMI	Descrição	Configuração de armazenamento
			Amazon ECS é `xfs` e o Docker usa o driver de armazenamento `overlay2`. Para obter mais informações, consulte Usar o driver de armazenamento OverlayFS na documentação do Docker.

O Amazon ECS fornece um log de alterações para a variante do Linux da AMI otimizada para Amazon ECS no GitHub. Para obter mais informações, consulte Log de alterações.

As variantes do Linux da AMI otimizada para Amazon ECS usam a AMI do Amazon Linux 2 ou a AMI do Amazon Linux 2023 como base. O nome da AMI para cada variante pode ser recuperado consultando a API do Systems Manager Parameter Store. Para obter mais informações, consulte Recuperação de metadados da AMI do Linux otimizada para o Amazon ECS. As notas de versão do Amazon Linux 2 AMI também estão disponíveis. Para obter mais informações, consulte Notas de lançamento do Amazon Linux 2. As notas de versão do Amazon Linux 2023 também estão disponíveis. Para obter mais informações, consulte Notas de lançamento do Amazon Linux 2023.

As páginas a seguir fornecem informações adicionais sobre as alterações:

- Notas de lançamento da AMI de origem no GitHub
- Notas de lançamento do Docker Engine na documentação do Docker
- Documentação do driver NVIDIA na documentação da NVIDIA
- Amazon ECS agent changelog no GitHub

  O código-fonte da aplicação `ecs-init`, os scripts e a configuração para empacotar o agente agora fazem parte do repositório do agente. Para versões mais antigas de `ecs-init` e pacotes, consulte Log de alterações do Amazon ecs-init no GitHub

Aplicação de atualizações de segurança à AMI otimizada para o Amazon ECS

As AMIs otimizadas para o Amazon ECS baseadas no Amazon Linux contêm uma versão personalizada do cloud-init. O Cloud-init é um pacote usado para inicializar imagens do Linux em um ambiente de computação em nuvem e realizar ações desejadas ao executar uma instância. Por padrão, todas as AMIs otimizadas para o Amazon ECS baseadas no Amazon Linux lançadas antes de 12 de junho de 2024 têm todas as atualizações de segurança "críticas" e "importantes" aplicadas quando a instância é inicializada.

A partir da versão de 12 de junho, nos lançamentos de 2024 das AMIs otimizadas para o Amazon ECS baseadas no Amazon Linux 2, o comportamento padrão não inclui mais a atualização de pacotes na inicialização. Em vez disso, recomendamos atualizar para uma nova AMI otimizada para o Amazon ECS à medida que as versões forem disponibilizadas. As AMIs otimizadas para o Amazon ECS são lançadas quando há atualizações de segurança disponíveis ou alterações de base da AMI. Isso garante que você receba as versões mais recentes do pacote e as atualizações de segurança, e que as versões do pacote sejam imutáveis por meio de execuções de instâncias. Para obter mais informações sobre como recuperar a AMI otimizada para o Amazon ECS mais recente, consulte Recuperação de metadados da AMI do Linux otimizada para o Amazon ECS.

Recomendamos automatizar seu ambiente para atualizar para novas AMIs à medida que forem disponibilizadas. Para obter informações sobre as opções disponíveis, consulte Amazon ECS enables easier EC2 capacity management, with managed instance draining.

Para continuar aplicando manualmente as atualizações de segurança "críticas" e "importantes" em uma versão da AMI, você pode executar o comando a seguir na sua instância do Amazon EC2.

```
yum update --security
```

Se quiser habilitar novamente as atualizações de segurança na execução, você pode adicionar a linha a seguir à seção `#cloud-config` de dados do usuário do cloud-init ao executar a instância do Amazon EC2. Para obter mais informações, consulte Using cloud-init on Amazon Linux 2 no Guia do usuário do Amazon Linux.

```
#cloud-config
repo_upgrade: security
```

Recuperação de metadados da AMI do Linux otimizada para o Amazon ECS

Você pode recuperar programaticamente os metadados da AMI otimizada para o Amazon ECS. Os metadados incluem o nome da AMI, a versão do agente de contêiner do Amazon ECS e a versão de runtime do Amazon ECS, que inclui a versão do Docker.

Ao criar um cluster usando o console, o Amazon ECS cria um modelo de inicialização para as instâncias com a AMI mais recente associada ao sistema operacional selecionado.

Quando você usa o AWS CloudFormation para criar um cluster, o parâmetro SSM faz parte do modelo de inicialização do Amazon EC2 para instâncias do grupo do Auto Scaling. Você pode configurar o modelo para usar um parâmetro dinâmico do Systems Manager para determinar qual AMI otimizada do Amazon ECS deve ser implantada. Esse parâmetro garante que toda vez que você implantar a pilha, ele verificará se há uma atualização disponível que precise ser aplicada às instâncias do EC2. Para obter um exemplo de como usar o parâmetro do Systems Manager, consulte Criar um cluster Amazon ECS com a AMI do Amazon Linux 2023 otimizada para o Amazon ECS no Guia do usuário do AWS CloudFormation.

O ID da AMI, o nome da imagem, o sistema operacional, a versão do agente de contêiner, o nome da imagem da fonte e a versão do runtime para cada variante das AMIs otimizadas para Amazon ECS podem ser recuperados de maneira programática, consultando a API da Systems Manager Parameter Store. Para obter mais informações sobre a API do Systems Manager Parameter Store, consulte GetParameters e GetParametersByPath.

> **Note**
>
> O usuário administrador precisa ter as seguintes permissões do IAM para recuperar os metadados da AMI otimizada para Amazon ECS. Essas permissões foram adicionadas à política `AmazonECS_FullAccess` do IAM.
>
> - ssm:GetParameters
> - ssm:GetParameter
> - ssm:GetParametersByPath

Formato de parâmetro do Systems Manager Parameter Store

Veja a seguir o formato do nome do parâmetro para cada variante da AMI otimizada para Amazon ECS.

AMIs do Linux otimizadas para Amazon ECS

- Metadados da AMI do Amazon Linux 2023:

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/<version>
```

- Metadados da AMI do Amazon Linux 2023 (arm64):

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/arm64/<version>
```

- Metadados da AMI do Amazon Linux 2023 (Neuron):

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/neuron/<version>
```

- Metadados da AMI do Amazon Linux 2023 (GPU):

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/gpu/<version>
```

  Metadados da AMI do Amazon Linux 2:

```
/aws/service/ecs/optimized-ami/amazon-linux-2/<version>
```

- Metadados da AMI do Amazon Linux 2 kernel 5.10:

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/<version>
```

- Metadados da AMI do Amazon Linux 2 (arm64):

```
/aws/service/ecs/optimized-ami/amazon-linux-2/arm64/<version>
```

- Metadados da AMI do Amazon Linux 2 kernel 5.10 (arm64):

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/arm64/<version>
```

- Metadados da AMI do kernel 5.10 otimizada para GPU do Amazon ECS:

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/gpu/<version>
```

- Metadados da AMI do Amazon Linux 2 (GPU):

```
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/<version>
```

- Metadados de AMI do kernel 5.10 do Amazon Linux 2 (Neuron) otimizada para o Amazon ECS:

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/inf/<version>
```

- Metadados da AMI do Amazon Linux 2 (Neuron):

```
/aws/service/ecs/optimized-ami/amazon-linux-2/inf/<version>
```

O formato de nome de parâmetro a seguir recupera o ID de imagem da versão recomendada mais recente da AMI do Amazon Linux 2 otimizada para Amazon ECS usando o subparâmetro `image_id`.

```
/aws/service/ecs/optimized-ami/amazon-linux-2/recommended/image_id
```

O seguinte formato de nome de parâmetro recupera os metadados de uma versão específica da AMI otimizada para Amazon ECS especificando o nome da AMI.

- Metadados da AMI do Amazon Linux 2 otimizada para Amazon ECS:

```
/aws/service/ecs/optimized-ami/amazon-linux-2/amzn2-ami-ecs-hvm-2.0.20181112-x86_64-ebs
```

> **Note**
>
> Todas as versões da AMI do Amazon Linux 2 otimizada para Amazon ECS estão disponíveis para recuperação. Somente versões `amzn-ami-2017.09.l-amazon-ecs-optimized` e posteriores da AMI otimizada para Amazon ECS (Linux) podem ser recuperadas.

Exemplos

Os exemplos a seguir mostram maneiras como você pode recuperar os metadados de cada variante da AMI otimizada para Amazon ECS.

Recuperar os metadados da AMI otimizada para Amazon ECS recomendada mais recente

É possível recuperar a AMI otimizada para Amazon ECS recomendada mais recente por meio da AWS CLI, usando os comandos da AWS CLI a seguir.

AMIs do Linux otimizadas para Amazon ECS

- Para as AMIs do Amazon Linux 2023 otimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
recommended --region us-east-1
```

- Para as AMIs do Amazon Linux 2023 (arm64) otimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
arm64/recommended --region us-east-1
```

- Para as AMIs do Amazon Linux 2023 (Neuron) otimizadas para o Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
neuron/recommended --region us-east-1
```

- Para as AMIs de GPU Amazon Linux 2023 otimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/gpu/
recommended --region us-east-1
```

- Para as AMIs do Amazon Linux 2 kernel 5.10 otimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/recommended --region us-east-1
```

- Para as AMIs do Amazon Linux 2 otimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
recommended --region us-east-1
```

- Para as AMIs do Amazon Linux 2 kernel 5.10 (arm64) otimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/arm64/recommended --region us-east-1
```

- Para as AMIs do Amazon Linux 2 (arm64) otimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/arm64/
recommended --region us-east-1
```

- Para as AMIs do kernel 5.10 otimizadas para GPU do Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/gpu/recommended --region us-east-1
```

- Para as AMIs otimizadas para GPU do Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended --region us-east-1
```

- Para as AMIs do kernel 5.10 do Amazon Linux 2 (Neuron) otimizadas para o Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/inf/recommended --region us-east-1
```

- Para as AMIs do Amazon Linux 2 (Neuron) otimizadas para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/inf/
recommended --region us-east-1
```

Recuperar o ID de imagem da AMI do Amazon Linux 2023 otimizada para Amazon ECS mais recente e recomendada

É possível recuperar o ID de imagem da AMI do Amazon Linux 2023 otimizada para Amazon ECS mais recente e recomendada usando o subparâmetro `image_id`.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-
linux-2023/recommended/image_id --region us-east-1
```

Para recuperar o valor `image_id` somente, você pode consultar o valor de parâmetro específico; por exemplo:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
recommended/image_id --region us-east-1 --query "Parameters[0].Value"
```

Recuperar os metadados de uma versão específica da AMI do Amazon Linux 2 otimizada para Amazon ECS

Recupere os metadados de uma versão específica da AMI do Amazon Linux otimizada para Amazon ECS por meio da AWS CLI, usando o seguinte comando da AWS CLI. Substitua o nome da AMI pelo nome da AMI do Amazon Linux otimizada para Amazon ECS a ser recuperado.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/amzn2-ami-ecs-hvm-2.0.20200928-x86_64-ebs --region us-east-1
```

Recuperação de metadados da AMI do kernel 5.10 do Amazon Linux 2 otimizada para o Amazon ECS usando a API GetParametersByPath do Systems Manager

Recupere os metadados da AMI do Amazon Linux 2 otimizada para Amazon ECS com a API GetParametersByPath do Systems Manager, usando a AWS CLI com o seguinte comando.

```
aws ssm get-parameters-by-path --path /aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/ --region us-east-1
```

Recuperar o ID de imagem da AMI do Amazon Linux 2 com kernel 5.10 otimizada para o Amazon ECS mais recente e recomendada

É possível recuperar o ID de imagem do ID de AMI do kernel 5.10 do Amazon Linux 2 otimizado para o Amazon ECS recomendado mais recente usando o subparâmetro `image_id`.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/recommended/image_id --region us-east-1
```

Para recuperar o valor `image_id` somente, você pode consultar o valor de parâmetro específico; por exemplo:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/recommended/image_id --region us-east-1 --query "Parameters[0].Value"
```

Usar a mais recente e recomendada AMI otimizada para Amazon ECS em um modelo do AWS CloudFormation

É possível referenciar a mais recente e recomendada AMI otimizada para Amazon ECS em um modelo do AWS CloudFormation fazendo referência ao nome do armazenamento de parâmetros do Systems Manager.

Exemplo do Linux

```
Parameters:kernel-5.10
  LatestECSOptimizedAMI:
    Description: AMI ID
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/recommended/
image_id
```

Migrar de um Amazon Linux 2 para uma AMI do Amazon Linux 2023 otimizada para Amazon ECS

Após o Amazon Linux, o Amazon ECS encerrará o suporte padrão para AMIs do Amazon Linux 2 otimizadas para Amazon ECS a partir de 30 de junho de 2026. Após essa data, a versão do agente do Amazon ECS será fixada e as novas AMIs do Amazon Linux 2 otimizadas para Amazon ECS só serão publicadas quando a AMI de origem do Amazon Linux 2 for atualizada. O fim da vida útil (EOL) ocorrerá em 30 de junho de 2026, após o qual nenhuma outra AMI do Amazon Linux 2 otimizada para Amazon ECS será publicada, mesmo que a AMI de origem seja atualizada.

O Amazon Linux 2023 fornece uma abordagem segura por padrão com políticas de segurança pré-configuradas, SELinux no modo permissivo, modo somente IMDSv2 habilitado por padrão, tempos de inicialização otimizados e gerenciamento aprimorado de pacotes para maior segurança e performance.

Há um alto grau de compatibilidade entre as AMIs do Amazon Linux 2 e do Amazon Linux 2023 otimizadas para Amazon ECS, e a maioria dos clientes verá mudanças mínimas, ou nenhuma, em suas workloads entre os dois sistemas operacionais.

Para obter mais informações, consulte Comparing Amazon Linux 2 and Amazon Linux 2023, no Guia do usuário do Amazon Linux 2023, e as Perguntas frequentes sobre o Amazon Linux 2023.

Considerações sobre compatibilidade

Gerenciamento de pacotes e atualizações do sistema operacional

Diferentemente das versões anteriores do Amazon Linux, as AMIs do Amazon Linux 2023 otimizadas para o Amazon ECS estão bloqueadas em uma versão específica do repositório do Amazon Linux. Isso evita que os usuários atualizem inadvertidamente pacotes que possam trazer alterações indesejadas ou significativas. Para obter mais informações, consulte Managing repositories and OS updates in Amazon Linux 2023 no Guia do usuário do Amazon Linux 2023.

versão de kernel do Linux

As AMIs do Amazon Linux 2 são baseadas nos kernels Linux 4.14 e 5.10, enquanto o Amazon Linux 2023 usa os kernels Linux 6.1 e 6.12. Para obter mais informações, consulte Comparing Amazon Linux 2 and Amazon Linux 2023 kernels no Guia do usuário do Amazon Linux 2023.

Alterações na disponibilidade de pacotes

Confira abaixo as alterações importantes de pacotes no Amazon Linux 2023:

- Alguns pacotes binários de fontes no Amazon Linux 2 não estão mais disponíveis no Amazon Linux 2023. Para obter mais informações, consulte Packages removed from Amazon Linux 2023 nas Notas da versão do Amazon Linux 2023.
- Mudanças na forma como o Amazon Linux é compatível com diferentes versões de pacotes. O sistema `amazon-linux-extras` usado no Amazon Linux 2 não existe no Amazon Linux 2023. Todos os pacotes estão simplesmente disponíveis no repositório “principal”.
- Os pacotes extras para Enterprise Linux (EPEL) não são compatíveis no Amazon Linux 2023. Para obter mais informações, consulte EPEL compatibility in Amazon Linux 2023 no Guia do usuário do Amazon Linux 2023.
- As aplicações de 32 bits não são compatíveis no Amazon Linux 2023. Para obter mais informações, consulte Deprecated features from Amazon Linux 2 no Guia do usuário do Amazon Linux 2023.

Alterações nos grupos de controle (cgroups)

Um grupo de controle (cgroup) é um recurso do kernel Linux para organizar hierarquicamente os processos e distribuir os recursos do sistema entre eles. Os grupos de controle são usados extensivamente para implementar um tempo de execução de contêiner e por `systemd`.

O agente do Amazon ECS, o Docker e o containerd são compatíveis tanto com o cgroupv1 quanto com o cgroupv2. O agente do Amazon ECS e o runtime do contêiner gerenciam os cgroups para você, portanto, os clientes do Amazon ECS não precisam fazer nenhuma alteração nessa atualização de cgroup subjacente.

Para obter mais detalhes sobre o cgroupv2, consulte Control groups v2 in Amazon Linux 2023 no Guia do usuário do Amazon Linux 2023.

Alterações no serviço de metadados de instância (IMDS)

O Amazon Linux 2023 requer o serviço de metadados de instância versão 2 (IMDSv2) por padrão. O IMDSv2 tem diversos benefícios que ajudam a aprimorar a postura de segurança. Ele usa um método de autenticação orientado por sessão que requer a criação de um token secreto em uma solicitação HTTP PUT simples para iniciar a sessão. O token de uma sessão pode ter validade de um segundo a seis horas.

Para obter mais informações sobre como fazer a transição do IMDSv1 para o IMDSv2, consulte Transição para usar o Serviço de metadados da instância versão 2 no Guia do usuário do Amazon EC2.

Se desejar usar o IMDSv1, você ainda poderá fazê-lo ao substituir manualmente as configurações usando as propriedades de inicialização da opção de metadados da instância.

Alterações na troca de memória

A troca de memória por contêiner não é compatível com o Amazon Linux 2023 e os cgroups v2. Para obter mais informações, consulte Gerenciar o espaço de memória de troca de contêiner no Amazon ECS.

Alterações na validação do FIPS

O Amazon Linux 2 é certificado sob o FIPS 140-2, e o Amazon Linux 2023 é certificado sob o FIPS 140-3.

Para habilitar o modo FIPS no Amazon Linux 2023, instale os pacotes necessários na sua instância do Amazon EC2 e siga as etapas de configuração usando as instruções em Enable FIPS Mode on Amazon Linux 2023 no Guia do usuário do Amazon Linux 2023.

Compatibilidade acelerada de instâncias

As AMIs do Amazon Linux 2023 otimizadas para Amazon ECS são compatíveis com os tipos de instâncias aceleradas por GPU e Neuron. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.

Criar AMIs personalizadas

Embora recomendemos migrar para AMIs otimizadas para Amazon ECS oficialmente compatíveis e publicadas para o Amazon Linux 2023, você pode continuar a criar AMIs personalizadas do Amazon

Linux 2 otimizadas para Amazon ECS usando os scripts de criação de código aberto usados para criar as variantes do Linux da AMI otimizada para Amazon ECS. Para obter mais informações, consulte Script de compilação da AMI do Linux otimizada para o Amazon ECS.

Estratégias de migração

Recomendamos criar e implementar um plano de migração que inclua testes completos de aplicações. As seções a seguir descrevem diferentes estratégias de migração com base em como você gerencia sua infraestrutura do Amazon ECS.

Migração com provedores de capacidade do Amazon ECS

1. Crie um novo provedor de capacidade com um novo modelo de inicialização. Isso deve fazer referência a um grupo do Auto Scaling com um modelo de inicialização semelhante ao seu existente, mas em vez da AMI do Amazon Linux 2 otimizada para Amazon ECS, ele deve especificar uma das variantes do Amazon Linux 2023. Adicione esse novo provedor de capacidade ao seu cluster existente do Amazon ECS.
2. Atualize a estratégia de provedor de capacidade padrão do seu cluster para incluir tanto o provedor de capacidade do Amazon Linux 2 existente quanto o novo provedor de capacidade do Amazon Linux 2023. Comece com um peso maior no provedor do Amazon Linux 2 e um peso menor no provedor do Amazon Linux 2023 (por exemplo, Amazon Linux 2: peso 80, Amazon Linux 2023: peso 20). Isso faz com que o Amazon ECS comece a provisionar instâncias do Amazon Linux 2023 à medida que novas tarefas são agendadas. Verifique se as instâncias estão registradas corretamente e se as tarefas podem ser executadas com êxito nas novas instâncias.
3. Ajuste gradualmente os pesos do provedor de capacidade na estratégia padrão do seu cluster, aumentando o peso do provedor do Amazon Linux 2023 e diminuindo o peso do provedor do Amazon Linux 2 ao longo do tempo (por exemplo, 60/40, depois 40/60 e depois 20/80). Você também pode atualizar as estratégias individuais do provedor de capacidade de serviço para priorizar as instâncias do Amazon Linux 2023. Monitore o posicionamento das tarefas para garantir que elas sejam executadas com êxito nas instâncias do Amazon Linux 2023.
4. Opcionalmente, drene as instâncias de contêineres do Amazon Linux 2 para acelerar a migração de tarefas. Se você tiver capacidade suficiente de substituição do Amazon Linux 2023, poderá drenar manualmente suas instâncias de contêineres do Amazon Linux 2 por meio do console do Amazon ECS ou da AWS CLI para acelerar a transição de suas tarefas do Amazon Linux 2 para o Amazon Linux 2023. Depois que a migração for concluída, remova o provedor de capacidade do Amazon Linux 2 do seu cluster e exclua o grupo do Auto Scaling associado.

Migração com um grupo do Amazon EC2 Auto Scaling

1. Crie um novo grupo do Amazon EC2 Auto Scaling com um novo modelo de inicialização. Isso deve ser semelhante ao seu modelo de inicialização existente, mas em vez da AMI do Amazon Linux 2 otimizada para Amazon ECS, ele deve especificar uma das variantes do Amazon Linux 2023. Esse novo grupo do Auto Scaling pode inicializar instâncias em seu cluster existente.
2. Aumente a escala verticalmente do grupo do Auto Scaling para que você comece a ter instâncias do Amazon Linux 2023 registradas em seu cluster. Verifique se as instâncias estão registradas corretamente e se as tarefas podem ser executadas com êxito nas novas instâncias.
3. Depois de verificar que suas tarefas funcionam no Amazon Linux 2023, aumente a escala verticalmente do grupo do Auto Scaling do Amazon Linux 2023 e, ao mesmo tempo, reduza a escala vertical e gradualmente do grupo do Auto Scaling do Amazon Linux 2, até substituir completamente todas as instâncias do Amazon Linux 2.
4. Se você tiver capacidade suficiente de substituição do Amazon Linux 2023, poderá querer drenar explicitamente suas instâncias de contêineres para acelerar a transição de suas tarefas do Amazon Linux 2 para o Amazon Linux 2023. Para obter mais informações, consulte Drenagem de instâncias de contêiner do Amazon ECS.

Migração com instâncias gerenciadas manualmente

1. Inicializar manualmente (ou ajustar scripts que inicializam) novas instâncias do Amazon EC2 usando a AMI do Amazon Linux 2023 otimizada para Amazon ECS em vez do Amazon Linux 2. Certifique-se de que essas instâncias usem os mesmos grupos de segurança, sub-redes, perfis do IAM e configuração de cluster que suas instâncias existentes do Amazon Linux 2. As instâncias devem se registrar automaticamente em seu cluster existente do Amazon ECS após a inicialização.
2. Verifique se as novas instâncias do Amazon Linux 2023 estão sendo registradas com êxito no seu cluster do Amazon ECS e se estão em um estado `ACTIVE`. Teste se as tarefas podem ser agendadas e executadas adequadamente nessas novas instâncias, aguardando o posicionamento natural das tarefas ou interrompendo e iniciando manualmente algumas tarefas para acionar o reagendamento.
3. Substitua gradualmente suas instâncias do Amazon Linux 2 inicializando instâncias adicionais do Amazon Linux 2023, conforme necessário, e, em seguida, drenando e encerrando manualmente as instâncias do Amazon Linux 2, uma a uma. Você pode drenar instâncias por meio do console do Amazon ECS definindo o status `DRAINING` da instância, o que deixará

de colocar novas tarefas nela e permitirá que as tarefas existentes sejam concluídas ou reagendadas em outro lugar.

Script de compilação da AMI do Linux otimizada para o Amazon ECS

O Amazon ECS abriu o código dos scripts de compilação usados para criar as variantes do Linux da AMI otimizada para o Amazon ECS. Esses scripts de compilação agora estão disponíveis no GitHub. Para obter mais informações, consulte amazon-ecs-ami no GitHub.

Se você precisar personalizar a AMI otimizada para o Amazon ECS, consulte Amazon ECS Optimized AMI Build Recipies no GitHub.

O repositório de scripts de compilação inclui um modelo do HashiCorp packer e scripts de compilação para gerar cada uma das variantes de Linux da AMI otimizada para o Amazon ECS. Esses scripts são a fonte da verdade para compilações das AMIs otimizadas para o Amazon ECS. Por isso, você pode acompanhar o repositório GitHub para monitorar alterações em nossas AMIs. Por exemplo, talvez você queira que sua AMI use a mesma versão do Docker que a equipe do Amazon ECS usa para a AMI oficial.

Para obter mais informações, consulte o repositório de AMIs do Amazon ECS em aws/amazon-ecs-ami no GitHub

Para compilar uma AMI do Linux otimizada para o Amazon ECS

1. Clone o repositório `aws/amazon-ecs-ami` do GitHub.

   ```
   git clone https://github.com/aws/amazon-ecs-ami.git
   ```

2. Adicione uma variável de ambiente para a região da AWS a ser usada ao criar a AMI. Substitua o valor `us-west-2` com a região a ser usada.

   ```
   export REGION=us-west-2
   ```

3. Um makefile é fornecido para compilar a AMI. A partir do diretório raiz do repositório clonado, use um dos seguintes comandos, correspondente à variante Linux da AMI otimizada para Amazon ECS que você deseja compilar.

   - AMI do Amazon Linux 2 otimizada para Amazon ECS

     ```
     make al2
     ```

- AMI do Amazon Linux 2 (arm64) otimizada para Amazon ECS

```
make al2arm
```

- AMI otimizada para GPU do Amazon ECS

```
make al2gpu
```

- AMI do Amazon Linux 2 (Neuron) otimizada para Amazon ECS

```
make al2inf
```

- AMI do Amazon Linux 2023 otimizada para Amazon ECS

```
make al2023
```

- AMI do Amazon Linux 2023 (arm64) otimizada para Amazon ECS

```
make al2023arm
```

- AMI do Amazon Linux 2023 (Neuron) otimizada para Amazon ECS

```
make al2023neu
```

AMIs Bottlerocket otimizadas para Amazon ECS

O Bottlerocket é um sistema operacional de código aberto baseado no Linux que foi criado pela AWS especificamente para executar contêineres em máquinas virtuais ou hosts bare metal. A AMI Bottlerocket otimizada para Amazon ECS é segura e contém apenas o número mínimo de pacotes necessários para executar contêineres. Isso melhora o uso de recursos, reduz a superfície de ataque de segurança e ajuda a reduzir a sobrecarga de gerenciamento. A AMI Bottlerocket também é integrada ao Amazon ECS para ajudar a reduzir a sobrecarga operacional envolvida na atualização de instâncias de contêineres em um cluster.

O Bottlerocket difere do Amazon Linux das maneiras a seguir:

- O Bottlerocket não inclui um gerenciador de pacotes e seu software só pode ser executado como contêineres. As atualizações do Bottlerocket são aplicadas e podem ser revertidas em uma única etapa, o que reduz a probabilidade de erros de atualização.

- O principal mecanismo para gerenciar hosts do Bottlerocket é com um programador de contêineres. Ao contrário do Amazon Linux, o login em instâncias individuais do Bottlerocket deve ser uma operação pouco frequente apenas para fins avançados de depuração e solução de problemas.

Para obter mais informações sobre o Bottlerocket, consulte a documentação e os comunicados no GitHub.

Existem variantes da AMI do Bottlerocket otimizada para o Amazon ECS para o kernel 6.1 e o kernel 5.10.

As seguintes variantes usam o kernel 6.1:

- `aws-ecs-2`
- `aws-ecs-2-nvidia`

As seguintes variantes usam o kernel 5.10:

- `aws-ecs-1`
- `aws-ecs-1-nvidia`

  Para obter mais informações sobre a variante `aws-ecs-1-nvidia`, consulte Anúncio do suporte à GPU NVIDIA para Bottlerocket no Amazon ECS.

Considerações

Considere o seguinte ao usar uma AMI Bottlerocket com o Amazon ECS.

- Bottlerocket oferece suporte a instâncias do Amazon EC2 com processadores `x86_64` e `arm64`. O AMI Bottlerocket não é recomendado para uso com instâncias do Amazon EC2 com um chip Inferentia.
- As imagens do Bottlerocket não incluem um servidor ou shell SSH. No entanto, é possível usar ferramentas de gerenciamento fora de banda para se obter acesso de administrador ao SSH e realizar a inicialização. Para obter mais informações, consulte essas seções no bottlerocket README.md no GitHub:
  - Exploration (Exploração)
  - Admin container (Contêiner Admin)

- Por padrão, o Bottlerocket tem um contêiner de controle que está habilitado. Esse contêiner executa o agente do AWS Systems Manager que você pode usar para executar comandos ou iniciar sessões de shell em instâncias Bottlerocket do Amazon EC2. Para obter mais informações, consulte Configurar gerenciador de sessões no Guia do usuário do AWS Systems Manager.

- O Bottlerocket é otimizado para workloads de contêiner e tem foco na segurança. O Bottlerocket não inclui um gerenciador de pacotes e é imutável. Para obter informações sobre as orientações e recursos de segurança, consulte Recursos de segurança e Orientação de segurança no GitHub.
- Há suporte para o modo de rede `awsvpc` para a versão `1.1.0` ou posterior da AMI do Bottlerocket.
- Há suporte para o App Mesh em uma definição de tarefa para versão `1.15.0` do Bottlerocket AMI ou posterior.
- O parâmetro de definição de tarefas `initProcessEnabled` é compatível com a versão `1.19.0` ou posterior da AMI do Bottlerocket.
- As AMIs do Bottlerocket também não oferecem suporte com os serviços e recursos a seguir:
  - ECS Anywhere
  - Service Connect
  - Amazon EFS em modo criptografado
  - Amazon EFS em modo de rede `awsvpc`
  - Os volumes do Amazon EBS não podem ser montados
  - Acelerador de inferência elástica

Recuperação dos metadados da AMI do Bottlerocket otimizada para o Amazon ECS

É possível recuperar o ID da imagem de máquina da Amazon (AMI) para AMIs otimizadas para o Amazon ECS consultando a API Parameter Store do AWS Systems Manager. Ao usar esse parâmetro, não será necessário pesquisar manualmente IDs de AMIs otimizadas para o Amazon ECS. Para obter mais informações sobre a API Systems Manager Parameter Store, consulte GetParameter. O usuário que você usou deve ter a permissão `ssm:GetParameter` do IAM para recuperar os metadados da AMI otimizada para o Amazon ECS.

Variante **`aws-ecs-2`** da AMI do Bottlerocket

É possível recuperar a última variante estável da AMI do Bottlerocket `aws-ecs-2` por Região da AWS e arquitetura com a AWS CLI ou o AWS Management Console.

- AWS CLI: é possível recuperar o ID de imagem da AMI do Bottlerocket otimizada para o Amazon ECS recomendada mais recentemente com o comando AWS CLI a seguir, usando o subparâmetro `image_id`. Substitua a *region* pelo código da região para o qual você deseja o ID da AMI. Para obter informações sobre as Regiões da AWS com suporte, consulte Encontrar uma AMI no GitHub. Para recuperar uma versão diferente da mais recente, substitua `latest` pelo número de versão.
  - Para a arquitetura de 64 bits (`x86_64`):

    ```
    aws ssm get-parameter --region us-east-2 --name "/aws/service/bottlerocket/aws-ecs-2/x86_64/latest/image_id" --query Parameter.Value --output text
    ```

  - Para a arquitetura Arm de 64 bits (`arm64`):

    ```
    aws ssm get-parameter --region us-east-2 --name "/aws/service/bottlerocket/aws-ecs-2/arm64/latest/image_id" --query Parameter.Value --output text
    ```

- AWS Management Console – É possível consultar o ID da AMI otimizada para o Amazon ECS recomendada, usando um URL no AWS Management Console. O URL abre o console do Amazon EC2 Systems Manager com o valor do ID para o parâmetro. No URL a seguir, substitua a *region* pelo código da região para o qual você deseja o ID da AMI. Para obter informações sobre as Regiões da AWS com suporte, consulte Encontrar uma AMI no GitHub.
  - Para a arquitetura de 64 bits (`x86_64`):

    ```
    https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/aws-ecs-2/x86_64/latest/image_id/description?region=region#
    ```

  - Para a arquitetura Arm de 64 bits (`arm64`):

    ```
    https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/aws-ecs-2/arm64/latest/image_id/description?region=region#
    ```

Variante **`aws-ecs-2-nvidia`** da AMI do Bottlerocket

É possível recuperar a última variante estável da AMI do Bottlerocket `aws-ecs-2-nvdia` por região e arquitetura com a AWS CLI ou o AWS Management Console.

- AWS CLI: é possível recuperar o ID de imagem da AMI do Bottlerocket otimizada para o Amazon ECS recomendada mais recentemente com o comando AWS CLI a seguir, usando o subparâmetro `image_id`. Substitua a *region* pelo código da região para o qual você deseja o ID da AMI. Para

obter informações sobre as Regiões da AWS com suporte, consulte Encontrar uma AMI no GitHub. Para recuperar uma versão diferente da mais recente, substitua `latest` pelo número de versão.

- Para a arquitetura de 64 bits (`x86_64`):

```
aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-ecs-2-nvidia/x86_64/latest/image_id" --query Parameter.Value --output text
```

- Para a arquitetura Arm de 64 bits (`arm64`):

```
aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-ecs-2-nvidia/arm64/latest/image_id" --query Parameter.Value --output text
```

- AWS Management Console: É possível consultar o ID da AMI otimizada para o Amazon ECS recomendada, usando um URL no AWS Management Console. O URL abre o console do Amazon EC2 Systems Manager com o valor do ID para o parâmetro. No URL a seguir, substitua a *region* pelo código da região para o qual você deseja o ID da AMI. Para obter informações sobre as Regiões da AWS com suporte, consulte Encontrar uma AMI no GitHub.

  - Para a arquitetura de 64 bits (`x86_64`):

```
https://regionconsole.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/aws-ecs-2-nvidia/x86_64/latest/image_id/description?region=region#
```

  - Para a arquitetura Arm de 64 bits (`arm64`):

```
https://regionconsole.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/aws-ecs-2-nvidia/arm64/latest/image_id/description?region=region#
```

Variante **`aws-ecs-1`** da AMI do Bottlerocket

É possível recuperar a última variante estável da AMI do Bottlerocket `aws-ecs-1` por Região da AWS e arquitetura com a AWS CLI ou o AWS Management Console.

- AWS CLI: é possível recuperar o ID de imagem da AMI do Bottlerocket otimizada para o Amazon ECS recomendada mais recentemente com o comando AWS CLI a seguir, usando o subparâmetro `image_id`. Substitua a *region* pelo código da região para o qual você deseja o ID da AMI. Para obter informações sobre as Regiões da AWS com suporte, consulte Encontrar uma AMI no GitHub. Para recuperar uma versão diferente da mais recente, substitua `latest` pelo número de versão.

  - Para a arquitetura de 64 bits (`x86_64`):

```
aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
ecs-1/x86_64/latest/image_id" --query Parameter.Value --output text
```

- Para a arquitetura Arm de 64 bits (`arm64`):

```
aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
ecs-1/arm64/latest/image_id" --query Parameter.Value --output text
```

- AWS Management Console – É possível consultar o ID da AMI otimizada para o Amazon ECS recomendada, usando um URL no AWS Management Console. O URL abre o console do Amazon EC2 Systems Manager com o valor do ID para o parâmetro. No URL a seguir, substitua a *`region`* pelo código da região para o qual você deseja o ID da AMI. Para obter informações sobre as Regiões da AWS com suporte, consulte Encontrar uma AMI no GitHub.
  - Para a arquitetura de 64 bits (`x86_64`):

```
https://region.console.aws.amazon.com/systems-manager/parameters/aws/service/
bottlerocket/aws-ecs-1/x86_64/latest/image_id/description
```

  - Para a arquitetura Arm de 64 bits (`arm64`):

```
https://region.console.aws.amazon.com/systems-manager/parameters/aws/service/
bottlerocket/aws-ecs-1/arm64/latest/image_id/description
```

Variante **`aws-ecs-1-nvidia`** da AMI do Bottlerocket

É possível recuperar a última variante estável da AMI do Bottlerocket `aws-ecs-1-nvdia` por região e arquitetura com a AWS CLI ou o AWS Management Console.

- AWS CLI: é possível recuperar o ID de imagem da AMI do Bottlerocket otimizada para o Amazon ECS recomendada mais recentemente com o comando AWS CLI a seguir, usando o subparâmetro `image_id`. Substitua a *`region`* pelo código da região para o qual você deseja o ID da AMI. Para obter informações sobre as Regiões da AWS com suporte, consulte Encontrar uma AMI no GitHub. Para recuperar uma versão diferente da mais recente, substitua `latest` pelo número de versão.
  - Para a arquitetura de 64 bits (`x86_64`):

```
aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
ecs-1-nvidia/x86_64/latest/image_id" --query Parameter.Value --output text
```

- Para a arquitetura Arm de 64 bits (`arm64`):

```
aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-ecs-1-nvidia/arm64/latest/image_id" --query Parameter.Value --output text
```

- AWS Management Console: É possível consultar o ID da AMI otimizada para o Amazon ECS recomendada, usando um URL no AWS Management Console. O URL abre o console do Amazon EC2 Systems Manager com o valor do ID para o parâmetro. No URL a seguir, substitua a *region* pelo código da região para o qual você deseja o ID da AMI. Para obter informações sobre as Regiões da AWS com suporte, consulte Encontrar uma AMI no GitHub.
  - Para a arquitetura de 64 bits (`x86_64`):

```
https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/aws-ecs-1-nvidia/x86_64/latest/image_id/description?region=region#
```

  - Para a arquitetura Arm de 64 bits (`arm64`):

```
https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/aws-ecs-1-nvidia/arm64/latest/image_id/description?region=region#
```

Próximas etapas

Para obter um tutorial detalhado sobre como começar a usar o sistema operacional Bottlerocket no Amazon ECS, consulte Using a Bottlerocket AMI with Amazon ECS no GitHub e Getting started with Bottlerocket and Amazon ECS no site do blog da AWS.

Para obter mais informações sobre como executar uma instância do Bottlerocket, consulte Iniciar uma instância do Bottlerocket para o Amazon ECS

Iniciar uma instância do Bottlerocket para o Amazon ECS

Você pode executar uma instância do Bottlerocket para executar as workloads de contêiner.

É possível usar a AWS CLI para executar a instância do Bottlerocket.

1. Crie um arquivo chamado `userdata.toml`. Esse arquivo será usado para dados do usuário da instância. Substitua *cluster-name* pelo nome do seu cluster.

```
[settings.ecs]
```

```
cluster = "cluster-name"
```

2. Use um dos comandos incluídos em the section called “Recuperação dos metadados da AMI do Bottlerocket otimizada para o Amazon ECS” para obter o ID da AMI do Bottlerocket. Você usará isso na etapa a seguir.
3. Execute o comando a seguir para iniciar a instância do Bottlerocket. Lembre-se de substituir os parâmetros a seguir:
   - Substitua *sub-rede* pelo ID da sub-rede pública ou privada na qual sua instância será iniciada.
   - Substitua *bottlerocket_ami* pelo ID da AMI da etapa anterior.
   - Substitua *t3.large* pelo tipo de instância que você deseja usar.
   - Substitua *região* pelo código da região.

```
aws ec2 run-instances --key-name ecs-bottlerocket-example \
   --subnet-id subnet \
   --image-id bottlerocket_ami \
   --instance-type t3.large \
   --region region \
   --tag-specifications
 'ResourceType=instance,Tags=[{Key=bottlerocket,Value=example}]' \
   --user-data file://userdata.toml \
   --iam-instance-profile Name=ecsInstanceRole
```

4. Execute o comando a seguir para verificar se a instância de contêiner está registrada no cluster. Ao executar esse comando, lembre-se de substituir os parâmetros a seguir:
   - Substitua *cluster* pelo nome do seu cluster.
   - Substitua *região* pelo código da sua região.

```
aws ecs list-container-instances --cluster cluster-name --region region
```

Para obter uma demonstração detalhada dos conceitos básicos do sistema operacional Bottlerocket no Amazon ECS, consulte Uso de uma AMI do Bottlerocket com o Amazon ECS, no GitHub, e Conceitos básicos do Bottlerocket e Amazon ECS no site do blog da AWS.

Gerenciamento de instâncias de contêiner do Linux no Amazon ECS

Ao usar as instâncias do EC2 para workloads do Amazon ECS, você é responsável pela manutenção das instâncias.

Procedimentos de gerenciamento

- Iniciar uma instância de contêiner do Linux do Amazon ECS
- Inicialização de instâncias de contêiner do Linux no Amazon ECS para transmitir dados
- Configuração de instâncias de contêiner do Linux no Amazon ECS para receber avisos de instância spot
- Execução de um script ao iniciar uma instância de contêiner do Linux no Amazon ECS
- Aumento das interfaces de rede de instâncias de contêiner do Linux no Amazon ECS
- Reserva de memória da instância de contêiner do Linux no Amazon ECS
- Gerenciamento de instâncias de contêiner do Amazon ECS remotamente usando o AWS Systems Manager
- Uso de um proxy HTTP para instâncias de contêiner do Linux no Amazon ECS
- Configuração de instâncias inicializadas previamente para o grupo do Amazon ECS Auto Scaling
- Atualizar o agente de contêiner do Amazon ECS

Cada agente de contêiner do Amazon ECS oferece suporte a um conjunto diferente de recursos e fornece correções de erros de versões anteriores. Quando possível, sempre recomendamos usar a versão mais recente do agente de contêiner do Amazon ECS. Para atualizar o agente de contêiner para a versão mais recente, consulte Atualizar o agente de contêiner do Amazon ECS.

Para ver quais recursos e aprimoramentos estão incluídos em cada versão de agente, consulte https://github.com/aws/amazon-ecs-agent/releases.

> **Important**
>
> A versão mínima do Docker para métricas confiáveis é a versão Docker `v20.10.13` e posteriores, que está incluída na AMI otimizada para o Amazon ECS `20220607` e posteriores.
> As versões `1.20.0` e posteriores do agente do Amazon ECS descontinuaram o suporte para as versões do Docker anteriores à `18.01.0`.

Iniciar uma instância de contêiner do Linux do Amazon ECS

É possível criar instâncias de contêiner do Amazon ECS usando o console do Amazon EC2.

É possível iniciar uma instância usando vários métodos, incluindo o console do Amazon EC2, a AWS CLI e o SDK. Para obter informações sobre outros métodos de inicialização de uma instância, consulte Iniciar uma instância no Manual do usuário do Amazon EC2.

Para obter mais informações sobre o assistente de inicialização, consulte Iniciar uma instância usando o novo assistente de inicialização de instância no Manual do usuário do Amazon EC2.

Antes de começar, conclua as etapas em Configuração para usar o Amazon ECS.

É possível usar o assistente do Amazon EC2 para iniciar uma instância. O assistente de lançamento de instâncias especifica todos os parâmetros de início necessários para iniciar uma instância.

Parâmetros para configuração de instâncias

- Procedimento
- Nome e tags
- Imagens de aplicações e sistemas operacionais (imagem de máquina da Amazon)
- Tipo de instância
- Par de chaves (login)
- Configurações de rede
- Configurar armazenamento
- Detalhes avançados

Procedimento

Antes de começar, conclua as etapas em Configuração para usar o Amazon ECS.

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. Na barra de navegação na parte superior da tela, a região atual AWS é exibida [por exemplo, Leste dos EUA (Ohio)]. Selecione uma região na qual a instância será iniciada.
3. No painel do console do Amazon EC2, selecione Launch instance (Executar instância).

Nome e tags

O nome da instância é uma tag em que a chave é Name (Nome) e o valor é o nome que você especificar. É possível aplicar tags na instância, sos volumes e nos elementos gráficos elásticos. Para instâncias spot, é possível marcar apenas a solicitação de instância spot.

A especificação de um nome de instância e de tags adicionais é opcional.

- Em Name (Nome), insira um nome descritivo para a instância. Se você não especificar um nome, a instância poderá ser identificada por seu ID, que é gerado automaticamente quando você inicia a instância.
- Para adicionar mais tags, selecione Add additional tag (Adicionar outra tag). Escolha Add tag (Adicionar tag), insira uma chave e um valor, e selecione o tipo de recurso a aplicar a tag. Escolha Add tag (Adicionar tag) para cada tag adicional a acrescentar.

Imagens de aplicações e sistemas operacionais (imagem de máquina da Amazon)

Uma imagem de máquina da Amazon (AMI) contém as informações necessárias para criar uma instância. Por exemplo, uma AMI pode conter o software necessário para atuar como servidor Web, por exemplo, Apache e seu site.

Use a barra Search (Pesquisa) para encontrar uma AMI otimizada para o Amazon ECS publicada pela AWS.

1. Insira um dos seguintes termos na barra Search (Pesquisa).
   - **`ami-ecs`**
   - O Value (Valor) de uma AMI otimizada para o Amazon ECS.

     Para obter as mais recentes AMIs otimizadas para o Amazon ECS e seus valores, consulte AMI do Linux otimizada para o Amazon ECS.
2. Pressione Enter.
3. Na página Escolher uma imagem de máquina da Amazon AMI), selecione a guia AMIs do AWS Marketplace.
4. No painel esquerdo Refine results (Refinar os resultados), selecione Amazon Web Services como o Publisher (Editor).
5. Escolha Select (Selecionar) na linha da AMI que você deseja usar.

Como alternativa, escolha Cancel (Cancelar) (no canto superior direito) para retornar ao assistente de inicialização de instância sem escolher uma AMI. Uma AMI padrão será selecionada. Confirme se a AMI atende aos requisitos descritos nas AMIs do Linux otimizadas para Amazon ECS.

Tipo de instância

O tipo de instância define a configuração do hardware e o tamanho da instância. Os tipos de instâncias maiores têm mais CPU e memória. Para obter mais informações, consulte Tipos de instância.

- Em Instance type (Tipo de instância), selecione o tipo de instância da instância.

  O tipo de instância que você selecionar determinará os recursos disponíveis para execução de suas tarefas.

Par de chaves (login)

Em Key pair name (Nome do par de chaves), escolha um par de chaves existente ou escolha Create new key pair (Criar um novo par de chaves) para criar um novo.

> **Important**
>
> Se você escolher a opção Proceed without key pair (Not recommended) (Prosseguir sem par de chaves, não recomendado), não conseguirá se conectar à instância a menos que escolha uma AMI configurada para permitir que os usuários façam login de outro modo.

Configurações de rede

Defina as configurações de rede, conforme necessário.

- Networking platform (Plataforma de rede): escolha Virtual Private Cloud (VPC) (Nuvem privada virtual - VPC) e especifique a sub-rede na seção Network interfaces (Interfaces de rede).
- VPC: selecione uma VPC existente na qual criar o grupo de segurança.
- Subnet (Sub-rede): é possível executar uma instância em uma sub-rede associada a uma zona de disponibilidade, a uma zona local, a uma zona Wavelength ou a um Outpost.

Para iniciar a instância em uma zona de disponibilidade, selecione a sub-rede na qual a instância será iniciada. Para criar uma nova sub-rede, escolha Create new subnet (Criar nova sub-rede) para acessar o console da Amazon VPC. Quando tiver concluído, retorne ao assistente de inicialização da instância e escolha o ícone Refresh (Atualizar) para carregar sua sub-rede na lista.

Para iniciar a instância em uma zona local, selecione uma sub-rede que você criou na zona local.

Para iniciar uma instância em um Outpost, selecione uma sub-rede em uma VPC associada ao Outpost.

- Auto-assign Public IP (Atribuir IP público automaticamente): se sua instância tiver de ser acessada pela Internet pública, verifique se o campo Auto-assign Public IP (Atribuir IP público automaticamente) está definido como Enable (Habilitar). Em caso negativo, defina esse campo como Disable (Desabilitar).

> **Note**
>
> Instâncias de contêiner precisam de acesso para se comunicar com o endpoint do serviço do Amazon ECS. Isso pode ser feito por meio de uma interface do endpoint da VPC ou por meio dos recursos de computação das instâncias de contêiner que tenham endereços IP públicos.
> Para saber mais sobre endpoints da VPC de interface, consulte Endpoints da VPC de interface do Amazon ECS (AWS PrivateLink).
> Se você não tiver um endpoint da VPC de interface configurado e suas instâncias de contêiner não tiverem endereços IP públicos, elas deverão usar a conversão de endereço de rede (NAT) para fornecer esse acesso. Para obter mais informações, consulte Gateways NAT no Guia do usuário da Amazon VPC e Uso de um proxy HTTP para instâncias de contêiner do Linux no Amazon ECS neste guia.

- Firewall (security groups) (Firewall, grupos de segurança): use um grupo de segurança para definir regras de firewall da sua instância de contêiner. Essas regras especificam qual tráfego de rede de entrada será fornecido para sua instância de contêiner. Todo o tráfego é ignorado.
  - Para selecionar um grupo de segurança existente, escolha Select an existing security group (Selecionar um grupo de segurança existente) e escolha o grupo de segurança criado em Configuração para usar o Amazon ECS.

Configurar armazenamento

A AMI que você selecionou inclui um ou mais volumes de armazenamento, incluindo o volume raiz. É possível especificar volumes adicionais a serem anexados à instância.

É possível usar a exibição Simple (Simples).

- Storage type (Tipo de armazenamento): configure o armazenamento para a sua instância de contêiner.

  Se você estiver usando a AMI do Amazon Linux 2 otimizada para Amazon ECS, sua instância terá um único volume de 30 GiB configurado, que é compartilhado entre o sistema operacional e o Docker.

  Se você estiver usando a AMI otimizada para Amazon ECS, a instância terá dois volumes configurados. O volume Raiz é para uso do sistema operacional e o segundo volume do Amazon EBS (anexado a `/dev/xvdcz`) é para uso do Docker.

  Você também pode aumentar ou diminuir os tamanhos de volumes para a sua instância para atender às necessidades dos seus aplicativos.

Detalhes avançados

Em Advanced details (Detalhes avançados), expanda a seção para visualizar os campos e especifique quaisquer parâmetros adicionais para a instância.

- Purchasing option (Opção de compra): escolha Request Spot instances (Solicitar instâncias Spot) para solicitar instâncias Spot. Também é necessário definir os outros campos relacionados a instâncias spot. Para obter mais informações, consulte Solicitações de instâncias spot.

  > **Note**
  >
  > Se você estiver usando instâncias spot e vir uma mensagem `Not available`, pode ser necessário escolher um tipo de instância diferente.

  .

- IAM instance profile (Perfil de instância do IAM): selecione a função do IAM de instância de contêiner. Isso geralmente é chamado de `ecsInstanceRole`.

> **Important**
>
> Se você não iniciar a instância de contêiner com as permissões apropriadas do IAM, o agente do Amazon ECS não poderá se conectar ao cluster. Para obter mais informações, consulte Função do IAM de instância de contêiner do Amazon ECS.

- Dados do usuário: configure a instância de contêiner do Amazon ECS com os dados do usuário, como as variáveis de ambiente de agente de Configuração do agente de contêiner do Amazon ECS. Os scripts de dados do usuário do Amazon EC2 são executados apenas uma vez, quando a instância é iniciada pela primeira vez. Veja a seguir exemplos comuns de quais dados do usuário são usados.
  - Por padrão, sua instância de contêiner é executada em seu cluster padrão. Para executar em um cluster que não seja padrão, escolha a lista Detalhes avançados. Em seguida, cole o seguinte script no campo Dados do usuário substituindo *your_cluster_name* pelo nome do seu cluster.

```
#!/bin/bash
echo ECS_CLUSTER=your_cluster_name >> /etc/ecs/ecs.config
```

  - Se você tiver um arquivo `ecs.config` no Amazon S3 e tiver habilitado o acesso somente leitura do Amazon S3 à função de instância de contêiner, escolha a lista Advanced Details (Detalhes avançados). Em seguida, cole o seguinte script no campo User data (Dados do usuário) substituindo *your_bucket_name* pelo nome do seu bucket para instalar a AWS CLI e gravar seu arquivo de configuração no momento da inicialização.

> **Note**
>
> Para obter mais informações sobre essa configuração, consulte Armazenamento da configuração da instância de contêiner do Amazon ECS no Amazon S3.

```
#!/bin/bash
yum install -y aws-cli
aws s3 cp s3://your_bucket_name/ecs.config /etc/ecs/ecs.config
```

- Especifique tags para sua instância de contêiner usando o parâmetro de configuração `ECS_CONTAINER_INSTANCE_TAGS`. Isso cria etiquetas associadas somente ao Amazon ECS, elas não podem ser listadas usando a API do Amazon EC2.

  > **Important**
  >
  > Se você iniciar as instâncias de contêiner usando um grupo do Amazon EC2 Auto Scaling, deverá usar o parâmetro de configuração do agente ECS_CONTAINER_INSTANCE_TAGS para adicionar etiquetas. Isso é decorrente da maneira como as etiquetas são adicionadas às instâncias do Amazon EC2 que são iniciadas por meio de grupos do Auto Scaling.

  ```
  #!/bin/bash
  cat <<'EOF' >> /etc/ecs/ecs.config
  ECS_CLUSTER=your_cluster_name
  ECS_CONTAINER_INSTANCE_TAGS={"tag_key": "tag_value"}
  EOF
  ```

- Especifique etiquetas para a instância de contêiner e use o parâmetro de configuração `ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM` para propagá-las do Amazon EC2 para o Amazon ECS

  Veja a seguir um exemplo de um script de dados do usuário que poderia propagar as tags associadas a uma instância de contêiner, bem como registrar a instância do contêiner com um cluster denominado `your_cluster_name`.

  ```
  #!/bin/bash
  cat <<'EOF' >> /etc/ecs/ecs.config
  ECS_CLUSTER=your_cluster_name
  ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM=ec2_instance
  EOF
  ```

Para obter mais informações, consulte Inicialização de instâncias de contêiner do Linux no Amazon ECS para transmitir dados.

Inicialização de instâncias de contêiner do Linux no Amazon ECS para transmitir dados

Ao iniciar uma instância do Amazon EC2, é possível transmitir dados do usuário para a instância do EC2. Os dados podem ser usados para realizar tarefas de configuração automatizadas em comum e até mesmo executar scripts na inicialização da instância. Para o Amazon ECS, os casos de uso mais comuns para dados de usuário devem transmitir informações de configuração para o daemon do Docker e o agente de contêiner do Amazon ECS.

É possível transmitir vários tipos de dados de usuário para o Amazon EC2, inclusive boothooks de nuvem, scripts de shell e diretivas `cloud-init`. Para obter mais informações sobre esses e outros tipos de formato, consulte a documentação Cloud-Init.

Para transmitir os dados do usuário ao usar o assistente de inicialização do Amazon EC2, consulte Iniciar uma instância de contêiner do Linux do Amazon ECS.

É possível configurar a instância de contêiner para transmitir dados na configuração do agente de contêiner ou na configuração do daemon do Docker.

Agente do contêiner do Amazon ECS

As variantes do Linux da AMI otimizada para Amazon ECS procuram os dados da configuração do agente no arquivo `/etc/ecs/ecs.config` quando o agente de contêiner é iniciado. É possível especificar esses dados de configuração na inicialização com os dados de usuário do Amazon EC2. Para obter mais informações sobre as variáveis de configuração do agente de contêiner do Amazon ECS disponíveis, consulte Configuração do agente de contêiner do Amazon ECS.

Para definir apenas uma única variável de configuração do agente, como o nome do cluster, use echo a fim de copiar a variável para o arquivo de configuração:

```
#!/bin/bash
echo "ECS_CLUSTER=MyCluster" >> /etc/ecs/ecs.config
```

Caso você tenha várias variáveis a serem gravadas em `/etc/ecs/ecs.config`, use o formato `heredoc` a seguir. Esse formato grava tudo entre as linhas que começam com cat e `EOF` no arquivo de configuração.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_ENGINE_AUTH_TYPE=docker
```

```
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"username":"my_name","password":"my_password","email":"email@example.com"}}
ECS_LOGLEVEL=debug
ECS_WARM_POOLS_CHECK=true
EOF
```

Para definir atributos de instância personalizados, defina a variável de ambiente `ECS_INSTANCE_ATTRIBUTES`.

```
#!/bin/bash
cat <<'EOF' >> ecs.config
ECS_INSTANCE_ATTRIBUTES={"envtype":"prod"}
EOF
```

Daemon do Docker

É possível especificar informações de configuração do daemon do Docker com os dados de usuário do Amazon EC2. Para mais informações sobre as opções de configuração, consulte a documentação de daemon do Docker.

> **Note**
>
> A AWS não oferece suporte a configurações personalizadas do Docker, pois ocasionalmente elas podem entrar em conflito com futuras alterações ou recursos do Amazon ECS sem aviso prévio.

No exemplo abaixo, as opções personalizadas são adicionadas ao arquivo de configuração do daemon do Docker, `/etc/docker/daemon.json`, que é então especificado nos dados de usuário quando a instância é iniciada.

```
#!/bin/bash
cat <<EOF >/etc/docker/daemon.json
{"debug": true}
EOF
systemctl restart docker --no-block
```

No exemplo abaixo, as opções personalizadas são adicionadas ao arquivo de configuração do daemon do Docker, `/etc/docker/daemon.json`, que é então especificado nos dados de usuário

quando a instância é iniciada. Este exemplo mostra como desabilitar o docker-proxy no arquivo de configuração do daemon do Docker.

```
#!/bin/bash
cat <<EOF >/etc/docker/daemon.json
{"userland-proxy": false}
EOF
systemctl restart docker --no-block
```

Configuração de instâncias de contêiner do Linux no Amazon ECS para receber avisos de instância spot

O Amazon EC2 encerra, interrompe ou coloca a instância spot em hibernação quando o preço spot excede o preço máximo da solicitação ou a capacidade não está mais disponível. O Amazon EC2 fornece um aviso de interrupção de dois minutos de instância spot para ações de terminar e interromper. Ele não fornece o aviso de dois minutos para a ação de hibernação. Se a drenagem da instância spot do Amazon ECS estiver ativada na instância, o Amazon ECS receberá o aviso de interrupção da instância spot e atribuirá a instância ao status `DRAINING`.

> **Important**
>
> O Amazon ECS não recebe um aviso do Amazon EC2 quando as instâncias são removidas pelo rebalanceamento de capacidade do Auto Scaling. Para obter mais informações, consulte Rebalanceamento de capacidade do Amazon EC2 Auto Scaling.

Quando uma instância de contêiner é definida como `DRAINING`, o Amazon ECS impede que novas tarefas sejam programadas para posicionamento na instância de contêiner. As tarefas de serviço nas instâncias de contêiner de drenagem que estão com o status de `PENDING` são interrompidas imediatamente. Se houver instâncias de contêiner no cluster disponíveis, as tarefas de serviço de substituição serão iniciadas nelas.

Por padrão, a drenagem de instância spot está desativada.

É possível ativar a drenagem de instância spot ao iniciar uma instância. Adicione o script apresentado a seguir ao campo Dados do usuário. Substitua *`MyCluster`* pelo nome do cluster no qual deseja registrar a instância de contêiner.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
```

```
ECS_CLUSTER=MyCluster
ECS_ENABLE_SPOT_INSTANCE_DRAINING=true
EOF
```

Para obter mais informações, consulte Iniciar uma instância de contêiner do Linux do Amazon ECS.

Para ativar a drenagem de instâncias spot para uma instância de contêiner existente

1. Conecte-se à instância spot pelo SSH.
2. Edite o arquivo `/etc/ecs/ecs.config` e adicione o seguinte:

   ```
   ECS_ENABLE_SPOT_INSTANCE_DRAINING=true
   ```

3. Reinicie o serviço `ecs`.

   - Para a AMI do Amazon Linux 2 otimizada para o Amazon ECS:

     ```
     sudo systemctl restart ecs
     ```

4. (Opcional) É possível verificar se o agente está em execução e consultar algumas informações sobre sua nova instância de contêiner consultando a operação da API de introspecção do agente. Para obter mais informações, consulte the section called “Introspecção de contêiner”.

   ```
   curl http://localhost:51678/v1/metadata
   ```

Execução de um script ao iniciar uma instância de contêiner do Linux no Amazon ECS

Talvez seja necessário executar um contêiner específico em cada instância de contêiner para lidar com operações ou questões de segurança, como monitoramento, segurança, métricas, descoberta de serviços ou registros em log.

Para fazer isso, você pode configurar suas instâncias de contêiner de forma que elas chamem o comando docker run com esse script de dados de usuário na execução ou em algum sistema init como Upstart ou systemd. Quando esse método funciona, existem algumas desvantagens, pois o Amazon ECS não tem qualquer conhecimento de contêiner e não pode monitorar a CPU, a memória, as portas ou quaisquer outros recursos utilizados. Para garantir que o Amazon ECS possa assumir adequadamente todos os recursos de tarefas, crie uma definição de tarefa para que o contêiner seja executado nas instâncias de contêiner. Em seguida, use o Amazon ECS para trazer a tarefa até o momento da inicialização com dados de usuário do Amazon EC2.

No procedimento a seguir, o script de dados de usuário do Amazon EC2 usa a API de introspecção do Amazon ECS para identificar a instância de contêiner. Em seguida, ele usa a AWS CLI e o comando start-task para executar uma tarefa específica nele durante o startup.

Para iniciar uma tarefa no momento da execução da instância de contêiner

1. Modifique a função do IAM `ecsInstanceRole` para adicionar permissões para a operação de API `StartTask`. Para obter mais informações, consulte Atualizar permissões para um perfil no Guia do usuário do AWS Identity and Access Management.
2. Inicie uma ou mais instâncias de contêiner usando a AMI do Amazon Linux 2 otimizada para o Amazon ECS. Inicie novas instâncias de contêiner e use o script de exemplo apresentado a seguir nos dados do usuário do EC2. Substitua *`your_cluster_name`* pelo cluster no qual a instância de contêiner será registrada e *`my_task_def`* pela definição de tarefa a ser executada na instância na inicialização.

   Para obter mais informações, consulte Iniciar uma instância de contêiner do Linux do Amazon ECS.

   > **Note**
   >
   > O conteúdo MIME de várias partes abaixo usa um script de shell para definir valores de configuração e instalar os pacotes. Ele também usa um trabalho systemd para iniciar a tarefa depois que o serviço ecs estiver em execução e a API de introspecção ficar disponível.

```
Content-Type: multipart/mixed; boundary="==BOUNDARY=="
MIME-Version: 1.0

--==BOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
# Specify the cluster that the container instance should register into
cluster=your_cluster_name

# Write the cluster configuration variable to the ecs.config file
# (add any other configuration variables here also)
echo ECS_CLUSTER=$cluster >> /etc/ecs/ecs.config
```

```
START_TASK_SCRIPT_FILE="/etc/ecs/ecs-start-task.sh"
cat <<- 'EOF' > ${START_TASK_SCRIPT_FILE}
 exec 2>>/var/log/ecs/ecs-start-task.log
 set -x

 # Install prerequisite tools
 yum install -y jq aws-cli

 # Wait for the ECS service to be responsive
 until curl -s http://localhost:51678/v1/metadata
 do
  sleep 1
 done

 # Grab the container instance ARN and AWS Region from instance metadata
 instance_arn=$(curl -s http://localhost:51678/v1/metadata | jq -r '.
 | .ContainerInstanceArn' | awk -F/ '{print $NF}' )
 cluster=$(curl -s http://localhost:51678/v1/metadata | jq -r '. | .Cluster' | awk
 -F/ '{print $NF}' )
 region=$(curl -s http://localhost:51678/v1/metadata | jq -r '.
 | .ContainerInstanceArn' | awk -F: '{print $4}')

 # Specify the task definition to run at launch
 task_definition=my_task_def

 # Run the AWS CLI start-task command to start your task on this container instance
 aws ecs start-task --cluster $cluster --task-definition $task_definition --
container-instances $instance_arn --started-by $instance_arn --region $region
EOF

# Write systemd unit file
UNIT="ecs-start-task.service"
cat <<- EOF > /etc/systemd/system/${UNIT}
      [Unit]
      Description=ECS Start Task
      Requires=ecs.service
      After=ecs.service

      [Service]
      Restart=on-failure
      RestartSec=30
      ExecStart=/usr/bin/bash ${START_TASK_SCRIPT_FILE}
```

```
      [Install]
      WantedBy=default.target
EOF

# Enable our ecs.service dependent service with `--no-block` to prevent systemd
 deadlock
# See https://github.com/aws/amazon-ecs-agent/issues/1707
systemctl enable --now --no-block "${UNIT}"
--==BOUNDARY==--
```

3. Verifique se as suas instâncias de contêiner são executadas no cluster correto e se as suas tarefas foram iniciadas.

   a. Abra o console em https://console.aws.amazon.com/ecs/v2.

   b. Na barra de navegação, selecione a região em que seu cluster está localizado.

   c. No painel de navegação, escolha Clusters e selecione o cluster que hospeda as suas instâncias de contêiner.

   d. Na página Cluster, escolha Tarefas e, em seguida, escolha suas tarefas.

      Cada instância de contêiner que você executou deve ter sua tarefa em execução nela.

      Se você não vir suas tarefas, faça login em suas instâncias de contêiner com SSH e verifique o arquivo `/var/log/ecs/ecs-start-task.log` para obter informações de depuração.

Aumento das interfaces de rede de instâncias de contêiner do Linux no Amazon ECS

> **Note**
>
> Esse recurso não está disponível no Fargate.

Cada tarefa que usa o modo de rede `awsvpc` recebe sua própria interface de rede elástica (ENI), que está conectada à instância de contêiner que a hospeda. Existe um limite padrão para o número de interfaces de rede que podem ser anexadas a uma instância do Amazon EC2, e a interface de rede primária conta como uma delas. Por exemplo, por padrão, uma instância `c5.large` pode ter até três ENIs associadas a ela. A interface de rede principal para a instância conta como uma, por isso você pode associar mais duas ENIs à instância. Como cada tarefa que usa o modo de rede

`awsvpc` exige uma ENI, em geral, você pode executar somente duas dessas tarefas nesse tipo de instância.

O Amazon ECS oferece suporte à execução de instâncias de contêiner com maior densidade de ENI usando tipos de instâncias do Amazon EC2 compatíveis. Quando você usa esses tipos de instância e ativa a configuração da conta `awsvpcTrunking`, ENIs adicionais ficam disponíveis em instâncias de contêiner iniciadas recentemente. Essa configuração permite que você coloque mais tarefas em cada instância de contêiner. Para usar o console para ativar o recurso, consulte Modificar configurações de conta do Amazon ECS. Para usar a AWS CLI para ativar o recurso, consulte Gerenciar configurações de conta do Amazon ECS usando o AWS CLI.

Por exemplo, uma instância `c5.large` com `awsvpcTrunking` tem um limite de ENI aumentado para doze. A instância de contêiner terá a interface de rede prmária e o Amazon ECS cria e anexa uma interface de rede "tronco" à instância de contêiner. Portanto, essa configuração permite que você inicie 10 tarefas na instância de contêiner, em vez das duas tarefas atuais.

A interface de rede tronco é totalmente gerenciada pelo Amazon ECS e é excluída quando você encerra ou cancela o registro da instância de contêiner do cluster. Para obter mais informações, consulte Opções de redes de tarefas do Amazon ECS para o tipo de inicialização do EC2.

Considerações

Considere as informações apresentadas a seguir ao usar o recurso de entroncamento de ENI.

- Somente variantes do Linux da AMI otimizada para o Amazon ECS ou outras variantes do Amazon Linux com a versão `1.28.1` ou posterior do agente de contêiner e a versão `1.28.1-2` ou posterior do pacote ecs-init oferecem suporte a limites maiores de ENI. Se você usar a variante do Linux mais recente da AMI otimizada para Amazon ECS, esses requisitos serão atendidos. Os contêineres do Windows não são suportados no momento.
- Somente novas instâncias do Amazon EC2 executadas depois de habilitar `awsvpcTrunking` recebem o aumento de limites de ENI e a interface de rede de truncamento. Instâncias executadas anteriormente não recebem esses recursos, independentemente das ações executadas.
- As instâncias do Amazon EC2 devem ter solicitações de DNS de IPv4 baseadas em recursos desativadas. Para desabilitar essa opção, desmarque a opção Habilitar pedidos de DNS baseados em solicitações IPv4 (registo A) ao criar uma nova instância no console do Amazon EC2. Para desabilitar essa opção usando o AWS CLI, use o comando a seguir.

```
aws ec2 modify-private-dns-name-options --instance-id i-xxxxxxx --no-enable-resource-name-dns-a-record --no-dry-run
```

- As instâncias do Amazon EC2 em sub-redes compartilhadas não são compatíveis. Elas falharão ao registrar em um cluster se forem usadas.
- Suas tarefas devem usar o modo de rede `awsvpc` e o tipo de execução do EC2. As tarefas que usam o tipo de execução do Fargate sempre recebem uma ENI dedicada, independentemente de quantas são executadas. Por isso, esse recurso não é necessário.
- Suas tarefas devem ser inicializadas na mesma Amazon VPC que a instância de contêiner. Suas tarefas falharão ao iniciar com um erro de atributo se não estiverem na mesma VPC.
- Ao executar uma nova instância de contêiner, a instância muda para o status `REGISTERING` enquanto a interface de rede elástica tronco é provisionada para a instância. Se ocorrer uma falha no registro, a instância mudará para o status `REGISTRATION_FAILED`. É possível solucionar problemas de um registro com falha descrevendo a instância de contêiner para visualizar o campo `statusReason`, que descreve o motivo da falha. Em seguida, o registro da instância de contêiner pode ser cancelado manualmente ou encerrado. Depois que o registro da instância de contêiner for cancelado ou encerrado com êxito, o Amazon ECS exclui a ENI de truncamento.

  > **Note**
  >
  > O Amazon ECS emite eventos de alteração de estado de instância de contêiner que você pode monitorar para instâncias que fazem a transição para um estado `REGISTRATION_FAILED`. Para obter mais informações, consulte Eventos de alteração no estado da instância de contêiner do Amazon ECS.

- Assim que a instância de contêiner é encerrada, ela muda para o status `DEREGISTERING` enquanto a interface de rede elástica tronco é desprovisionada. Depois, a instância muda para o status `INACTIVE`.
- Se uma instância de contêiner em uma sub-rede pública com o aumento dos limites de ENI for encerrada e depois reiniciada, a instância perderá o endereço IP público e o agente de contêiner perderá a conexão.
- Quando você habilita `awsvpcTrunking`, as instâncias de contêiner recebem uma ENI adicional que usa o grupo de segurança padrão da VPC e é gerenciada pelo Amazon ECS.

  Uma VPC padrão vem com uma sub-rede pública em cada zona de disponibilidade, um gateway da Internet e configurações para habilitar a resolução de DNS. A sub-rede é pública, porque a tabela de rotas principal envia o tráfego da sub-rede destinado à internet para o gateway da internet. É possível transformar uma sub-rede padrão em uma sub-rede privada removendo a rota do destino 0.0.0.0/0 para o gateway da internet. Porém, se você fizer isso, nenhuma instância de

container sendo executada nessa sub-rede poderá acessar a internet. Você pode adicionar ou excluir regras de grupo de segurança para controlar tráfego de entrada e saída das sub-redes. Para obter mais informações, consulte Regras de grupo de segurança no Manual do usuário da Amazon Virtual Private Cloud.

Pré-requisitos

Antes de executar uma instância de contêiner com limites de ENI maiores, os pré-requisitos a seguir devem ser atendidos.

- A função vinculada ao serviço para Amazon ECS deve ser criada. A função vinculada ao serviço do Amazon ECS fornece ao Amazon ECS as permissões para fazer chamadas a outros serviços da AWS em seu nome. Essa função será automaticamente criada quando você criar um cluster ou se criar ou atualizar um serviço no AWS Management Console. Para obter mais informações, consulte Uso de perfis vinculados ao serviço para o Amazon ECS. Também é possível criar a função vinculada ao serviço com o comando da AWS CLI a seguir.

```
aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
```

- Sua conta ou perfil do IAM da instância de contêiner deve habilitar a configuração de conta `awsvpcTrunking`. Recomendamos que você crie dois perfis de instância de contêiner (`ecsInstanceRole`). Em seguida, você pode habilitar a configuração da conta `awsvpcTrunking` para um perfil e usá-lo em tarefas que exigem truncamento de ENI. Para obter informações sobre o perfil da instância de contêiner, consulte Função do IAM de instância de contêiner do Amazon ECS.

Assim que os pré-requisitos forem atendidos, é possível executar uma nova instância de contêiner usando um dos tipos de instância do Amazon EC2 compatíveis, e a instância terá os limites de ENI maiores. Para obter uma lista dos tipos de instâncias compatíveis, consulte Instâncias com suporte para o aumento de interfaces de rede de contêineres do Amazon ECS. A instância de contêiner deve ter a versão `1.28.1` ou posterior do agente de contêiner e a versão `1.28.1-2` ou posterior do pacote ecs-init. Se você usar a variante do Linux mais recente da AMI otimizada para Amazon ECS, esses requisitos serão atendidos. Para obter mais informações, consulte Iniciar uma instância de contêiner do Linux do Amazon ECS.

> **Important**
>
> As instâncias do Amazon EC2 devem ter solicitações de DNS de IPv4 baseadas em recursos desativadas. Para desabilitar essa opção, certifique-se de que a opção Enable resource-based IPV4 (A record) DNS requests (Habilitar solicitações de DNS de IPV4 (registro A) baseadas em recursos) esteja desmarcada ao criar uma nova instância usando o console do Amazon EC2. Para desabilitar essa opção usando o AWS CLI, use o comando a seguir.
>
> ```
> aws ec2 modify-private-dns-name-options --instance-id i-xxxxxxx --no-enable-resource-name-dns-a-record --no-dry-run
> ```

Para visualizar as instâncias de contêiner com limites de ENI maiores com a AWS CLI

Cada instância de contêiner tem uma interface de rede padrão, conhecida como interface de rede tronco. Use o comando a seguir para listar as instâncias de contêiner com limites maiores de ENI consultando o atributo `ecs.awsvpc-trunk-id`, que indica a existência de uma interface de rede de truncamento.

- list-attributes (AWS CLI)

```
aws ecs list-attributes \
      --target-type container-instance \
      --attribute-name ecs.awsvpc-trunk-id \
      --cluster cluster_name \
      --region us-east-1
```

- Get-ECSAttributeList (AWS Tools for Windows PowerShell)

```
Get-ECSAttributeList -TargetType container-instance -AttributeName ecs.awsvpc-trunk-id -Region us-east-1
```

Instâncias com suporte para o aumento de interfaces de rede de contêineres do Amazon ECS

A seguir, são mostrados os tipos de instância do Amazon EC2 compatíveis e o número de tarefas que usam o modo de rede `awsvpc` que podem ser iniciadas em cada tipo de instância antes e após a habilitação da configuração da conta `awsvpcTrunking`.

> **Important**
>
> Embora outros tipos de instância sejam compatíveis na mesma família de instâncias, os tipos de instância `a1.metal`, `c5.metal`, `c5a.8xlarge`, `c5ad.8xlarge`, `c5d.metal`, `m5.metal`, `p3dn.24xlarge`, `r5.metal`, `r5.8xlarge` e `r5d.metal` não são compatíveis. As famílias de instâncias `c5n`, `d3`, `d3en`, `g3`, `g3s`, `g4dn`, `i3`, `i3en`, `inf1`, `m5dn`, `m5n`, `m5zn`, `mac1`, `r5b`, `r5n`, `r5dn`, `u-12tb1`, `u-6tb1`, `u-9tb1` e `z1d` não são compatíveis.

Tópicos

- Uso geral
- Otimizadas para computação
- Otimizado para memória
- Otimizada para armazenamento
- Computação acelerada
- Computação de alta performance

Uso geral

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
a1.medium	1	10
a1.large	2	10
a1.xlarge	3	20
a1.2xlarge	3	40
a1.4xlarge	7	60
m5.large	2	10
m5.xlarge	3	20
m5.2xlarge	3	40

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
m5.4xlarge	7	60
m5.8xlarge	7	60
m5.12xlarge	7	60
m5.16xlarge	14	120
m5.24xlarge	14	120
m5a.large	2	10
m5a.xlarge	3	20
m5a.2xlarge	3	40
m5a.4xlarge	7	60
m5a.8xlarge	7	60
m5a.12xlarge	7	60
m5a.16xlarge	14	120
m5a.24xlarge	14	120
m5ad.large	2	10
m5ad.xlarge	3	20
m5ad.2xlarge	3	40
m5ad.4xlarge	7	60
m5ad.8xlarge	7	60
m5ad.12xlarge	7	60
m5ad.16xlarge	14	120

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
m5ad.24xlarge	14	120
m5d.large	2	10
m5d.xlarge	3	20
m5d.2xlarge	3	40
m5d.4xlarge	7	60
m5d.8xlarge	7	60
m5d.12xlarge	7	60
m5d.16xlarge	14	120
m5d.24xlarge	14	120
m5d.metal	14	120
m6a.large	2	10
m6a.xlarge	3	20
m6a.2xlarge	3	40
m6a.4xlarge	7	60
m6a.8xlarge	7	90
m6a.12xlarge	7	120
m6a.16xlarge	14	120
m6a.24xlarge	14	120
m6a.32xlarge	14	120
m6a.48xlarge	14	120

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
m6a.metal	14	120
m6g.medium	1	4
m6g.large	2	10
m6g.xlarge	3	20
m6g.2xlarge	3	40
m6g.4xlarge	7	60
m6g.8xlarge	7	60
m6g.12xlarge	7	60
m6g.16xlarge	14	120
m6g.metal	14	120
m6gd.medium	1	4
m6gd.large	2	10
m6gd.xlarge	3	20
m6gd.2xlarge	3	40
m6gd.4xlarge	7	60
m6gd.8xlarge	7	60
m6gd.12xlarge	7	60
m6gd.16xlarge	14	120
m6gd.metal	14	120
m6i.large	2	10

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
m6i.xlarge	3	20
m6i.2xlarge	3	40
m6i.4xlarge	7	60
m6i.8xlarge	7	90
m6i.12xlarge	7	120
m6i.16xlarge	14	120
m6i.24xlarge	14	120
m6i.32xlarge	14	120
m6i.metal	14	120
m6id.large	2	10
m6id.xlarge	3	20
m6id.2xlarge	3	40
m6id.4xlarge	7	60
m6id.8xlarge	7	90
m6id.12xlarge	7	120
m6id.16xlarge	14	120
m6id.24xlarge	14	120
m6id.32xlarge	14	120
m6id.metal	14	120
m6idn.large	2	10

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
m6idn.xlarge	3	20
m6idn.2xlarge	3	40
m6idn.4xlarge	7	60
m6idn.8xlarge	7	90
m6idn.12xlarge	7	120
m6idn.16xlarge	14	120
m6idn.24xlarge	14	120
m6idn.32xlarge	15	120
m6idn.metal	15	120
m6in.large	2	10
m6in.xlarge	3	20
m6in.2xlarge	3	40
m6in.4xlarge	7	60
m6in.8xlarge	7	90
m6in.12xlarge	7	120
m6in.16xlarge	14	120
m6in.24xlarge	14	120
m6in.32xlarge	15	120
m6in.metal	15	120
m7a.medium	1	4

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
m7a.large	2	10
m7a.xlarge	3	20
m7a.2xlarge	3	40
m7a.4xlarge	7	60
m7a.8xlarge	7	90
m7a.12xlarge	7	120
m7a.16xlarge	14	120
m7a.24xlarge	14	120
m7a.32xlarge	14	120
m7a.48xlarge	14	120
m7a.metal-48xl	14	120
m7g.medium	1	4
m7g.large	2	10
m7g.xlarge	3	20
m7g.2xlarge	3	40
m7g.4xlarge	7	60
m7g.8xlarge	7	60
m7g.12xlarge	7	60
m7g.16xlarge	14	120
m7g.metal	14	120

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
m7gd.medium	1	4
m7gd.large	2	10
m7gd.xlarge	3	20
m7gd.2xlarge	3	40
m7gd.4xlarge	7	60
m7gd.8xlarge	7	60
m7gd.12xlarge	7	60
m7gd.16xlarge	14	120
m7gd.metal	14	120
m7i.large	2	10
m7i.xlarge	3	20
m7i.2xlarge	3	40
m7i.4xlarge	7	60
m7i.8xlarge	7	90
m7i.12xlarge	7	120
m7i.16xlarge	14	120
m7i.24xlarge	14	120
m7i.48xlarge	14	120
m7i.metal-24xl	14	120
m7i.metal-48xl	14	120

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
m7i-flex.large	2	4
m7i-flex.xlarge	3	10
m7i-flex.2xlarge	3	20
m7i-flex.4xlarge	7	40
m7i-flex.8xlarge	7	60
m7i-flex.12xlarge	7	120
m7i-flex.16xlarge	14	120
m8g.medium	1	4
m8g.large	2	10
m8g.xlarge	3	20
m8g.2xlarge	3	40
m8g.4xlarge	7	60
m8g.8xlarge	7	60
m8g.12xlarge	7	60
m8g.16xlarge	14	120
m8g.24xlarge	14	120
m8g.48xlarge	14	120
m8g.metal-24xl	14	120
m8g.metal-48xl	14	120
m8gd.medium	1	4

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
m8gd.large	2	10
m8gd.xlarge	3	20
m8gd.2xlarge	3	40
m8gd.4xlarge	7	60
m8gd.8xlarge	7	60
m8gd.12xlarge	7	60
m8gd.16xlarge	14	120
m8gd.24xlarge	14	120
m8gd.48xlarge	14	120
m8gd.metal-24xl	14	120
m8gd.metal-48xl	14	120
mac2.metal	7	12
mac2-m1ultra.metal	7	12
mac2-m2.metal	7	12
mac2-m2pro.metal	7	12

Otimizadas para computação

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
c5.large	2	10

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
c5.xlarge	3	20
c5.2xlarge	3	40
c5.4xlarge	7	60
c5.9xlarge	7	60
c5.12xlarge	7	60
c5.18xlarge	14	120
c5.24xlarge	14	120
c5a.large	2	10
c5a.xlarge	3	20
c5a.2xlarge	3	40
c5a.4xlarge	7	60
c5a.12xlarge	7	60
c5a.16xlarge	14	120
c5a.24xlarge	14	120
c5ad.large	2	10
c5ad.xlarge	3	20
c5ad.2xlarge	3	40
c5ad.4xlarge	7	60
c5ad.12xlarge	7	60
c5ad.16xlarge	14	120

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
c5ad.24xlarge	14	120
c5d.large	2	10
c5d.xlarge	3	20
c5d.2xlarge	3	40
c5d.4xlarge	7	60
c5d.9xlarge	7	60
c5d.12xlarge	7	60
c5d.18xlarge	14	120
c5d.24xlarge	14	120
c6a.large	2	10
c6a.xlarge	3	20
c6a.2xlarge	3	40
c6a.4xlarge	7	60
c6a.8xlarge	7	90
c6a.12xlarge	7	120
c6a.16xlarge	14	120
c6a.24xlarge	14	120
c6a.32xlarge	14	120
c6a.48xlarge	14	120
c6a.metal	14	120

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
c6g.medium	1	4
c6g.large	2	10
c6g.xlarge	3	20
c6g.2xlarge	3	40
c6g.4xlarge	7	60
c6g.8xlarge	7	60
c6g.12xlarge	7	60
c6g.16xlarge	14	120
c6g.metal	14	120
c6gd.medium	1	4
c6gd.large	2	10
c6gd.xlarge	3	20
c6gd.2xlarge	3	40
c6gd.4xlarge	7	60
c6gd.8xlarge	7	60
c6gd.12xlarge	7	60
c6gd.16xlarge	14	120
c6gd.metal	14	120
c6gn.medium	1	4
c6gn.large	2	10

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
c6gn.xlarge	3	20
c6gn.2xlarge	3	40
c6gn.4xlarge	7	60
c6gn.8xlarge	7	60
c6gn.12xlarge	7	60
c6gn.16xlarge	14	120
c6i.large	2	10
c6i.xlarge	3	20
c6i.2xlarge	3	40
c6i.4xlarge	7	60
c6i.8xlarge	7	90
c6i.12xlarge	7	120
c6i.16xlarge	14	120
c6i.24xlarge	14	120
c6i.32xlarge	14	120
c6i.metal	14	120
c6id.large	2	10
c6id.xlarge	3	20
c6id.2xlarge	3	40
c6id.4xlarge	7	60

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
c6id.8xlarge	7	90
c6id.12xlarge	7	120
c6id.16xlarge	14	120
c6id.24xlarge	14	120
c6id.32xlarge	14	120
c6id.metal	14	120
c6in.large	2	10
c6in.xlarge	3	20
c6in.2xlarge	3	40
c6in.4xlarge	7	60
c6in.8xlarge	7	90
c6in.12xlarge	7	120
c6in.16xlarge	14	120
c6in.24xlarge	14	120
c6in.32xlarge	15	120
c6in.metal	15	120
c7a.medium	1	4
c7a.large	2	10
c7a.xlarge	3	20
c7a.2xlarge	3	40

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
c7a.4xlarge	7	60
c7a.8xlarge	7	90
c7a.12xlarge	7	120
c7a.16xlarge	14	120
c7a.24xlarge	14	120
c7a.32xlarge	14	120
c7a.48xlarge	14	120
c7a.metal-48xl	14	120
c7g.medium	1	4
c7g.large	2	10
c7g.xlarge	3	20
c7g.2xlarge	3	40
c7g.4xlarge	7	60
c7g.8xlarge	7	60
c7g.12xlarge	7	60
c7g.16xlarge	14	120
c7g.metal	14	120
c7gd.medium	1	4
c7gd.large	2	10
c7gd.xlarge	3	20

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
c7gd.2xlarge	3	40
c7gd.4xlarge	7	60
c7gd.8xlarge	7	60
c7gd.12xlarge	7	60
c7gd.16xlarge	14	120
c7gd.metal	14	120
c7gn.medium	1	4
c7gn.large	2	10
c7gn.xlarge	3	20
c7gn.2xlarge	3	40
c7gn.4xlarge	7	60
c7gn.8xlarge	7	60
c7gn.12xlarge	7	60
c7gn.16xlarge	14	120
c7gn.metal	14	120
c7i.large	2	10
c7i.xlarge	3	20
c7i.2xlarge	3	40
c7i.4xlarge	7	60
c7i.8xlarge	7	90

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
c7i.12xlarge	7	120
c7i.16xlarge	14	120
c7i.24xlarge	14	120
c7i.48xlarge	14	120
c7i.metal-24xl	14	120
c7i.metal-48xl	14	120
c7i-flex.large	2	4
c7i-flex.xlarge	3	10
c7i-flex.2xlarge	3	20
c7i-flex.4xlarge	7	40
c7i-flex.8xlarge	7	60
c7i-flex.12xlarge	7	120
c7i-flex.16xlarge	14	120
c8g.medium	1	4
c8g.large	2	10
c8g.xlarge	3	20
c8g.2xlarge	3	40
c8g.4xlarge	7	60
c8g.8xlarge	7	60
c8g.12xlarge	7	60

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
c8g.16xlarge	14	120
c8g.24xlarge	14	120
c8g.48xlarge	14	120
c8g.metal-24xl	14	120
c8g.metal-48xl	14	120
c8gd.medium	1	4
c8gd.large	2	10
c8gd.xlarge	3	20
c8gd.2xlarge	3	40
c8gd.4xlarge	7	60
c8gd.8xlarge	7	60
c8gd.12xlarge	7	60
c8gd.16xlarge	14	120
c8gd.24xlarge	14	120
c8gd.48xlarge	14	120
c8gd.metal-24xl	14	120
c8gd.metal-48xl	14	120
c8gn.medium	1	4
c8gn.large	2	10
c8gn.xlarge	3	20

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
c8gn.2xlarge	3	40
c8gn.4xlarge	7	60
c8gn.8xlarge	9	60
c8gn.12xlarge	11	60
c8gn.16xlarge	15	120
c8gn.24xlarge	23	120
c8gn.48xlarge	23	120
c8gn.metal-24xl	23	120
c8gn.metal-48xl	23	120

Otimizado para memória

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
r5.large	2	10
r5.xlarge	3	20
r5.2xlarge	3	40
r5.4xlarge	7	60
r5.12xlarge	7	60
r5.16xlarge	14	120
r5.24xlarge	14	120

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
r5a.large	2	10
r5a.xlarge	3	20
r5a.2xlarge	3	40
r5a.4xlarge	7	60
r5a.8xlarge	7	60
r5a.12xlarge	7	60
r5a.16xlarge	14	120
r5a.24xlarge	14	120
r5ad.large	2	10
r5ad.xlarge	3	20
r5ad.2xlarge	3	40
r5ad.4xlarge	7	60
r5ad.8xlarge	7	60
r5ad.12xlarge	7	60
r5ad.16xlarge	14	120
r5ad.24xlarge	14	120
r5b.16xlarge	14	120
r5d.large	2	10
r5d.xlarge	3	20
r5d.2xlarge	3	40

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
r5d.4xlarge	7	60
r5d.8xlarge	7	60
r5d.12xlarge	7	60
r5d.16xlarge	14	120
r5d.24xlarge	14	120
r5dn.16xlarge	14	120
r6a.large	2	10
r6a.xlarge	3	20
r6a.2xlarge	3	40
r6a.4xlarge	7	60
r6a.8xlarge	7	90
r6a.12xlarge	7	120
r6a.16xlarge	14	120
r6a.24xlarge	14	120
r6a.32xlarge	14	120
r6a.48xlarge	14	120
r6a.metal	14	120
r6g.medium	1	4
r6g.large	2	10
r6g.xlarge	3	20

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
r6g.2xlarge	3	40
r6g.4xlarge	7	60
r6g.8xlarge	7	60
r6g.12xlarge	7	60
r6g.16xlarge	14	120
r6g.metal	14	120
r6gd.medium	1	4
r6gd.large	2	10
r6gd.xlarge	3	20
r6gd.2xlarge	3	40
r6gd.4xlarge	7	60
r6gd.8xlarge	7	60
r6gd.12xlarge	7	60
r6gd.16xlarge	14	120
r6gd.metal	14	120
r6i.large	2	10
r6i.xlarge	3	20
r6i.2xlarge	3	40
r6i.4xlarge	7	60
r6i.8xlarge	7	90

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
r6i.12xlarge	7	120
r6i.16xlarge	14	120
r6i.24xlarge	14	120
r6i.32xlarge	14	120
r6i.metal	14	120
r6idn.large	2	10
r6idn.xlarge	3	20
r6idn.2xlarge	3	40
r6idn.4xlarge	7	60
r6idn.8xlarge	7	90
r6idn.12xlarge	7	120
r6idn.16xlarge	14	120
r6idn.24xlarge	14	120
r6idn.32xlarge	15	120
r6idn.metal	15	120
r6in.large	2	10
r6in.xlarge	3	20
r6in.2xlarge	3	40
r6in.4xlarge	7	60
r6in.8xlarge	7	90

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
r6in.12xlarge	7	120
r6in.16xlarge	14	120
r6in.24xlarge	14	120
r6in.32xlarge	15	120
r6in.metal	15	120
r6id.large	2	10
r6id.xlarge	3	20
r6id.2xlarge	3	40
r6id.4xlarge	7	60
r6id.8xlarge	7	90
r6id.12xlarge	7	120
r6id.16xlarge	14	120
r6id.24xlarge	14	120
r6id.32xlarge	14	120
r6id.metal	14	120
r7a.medium	1	4
r7a.large	2	10
r7a.xlarge	3	20
r7a.2xlarge	3	40
r7a.4xlarge	7	60

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
r7a.8xlarge	7	90
r7a.12xlarge	7	120
r7a.16xlarge	14	120
r7a.24xlarge	14	120
r7a.32xlarge	14	120
r7a.48xlarge	14	120
r7a.metal-48xl	14	120
r7g.medium	1	4
r7g.large	2	10
r7g.xlarge	3	20
r7g.2xlarge	3	40
r7g.4xlarge	7	60
r7g.8xlarge	7	60
r7g.12xlarge	7	60
r7g.16xlarge	14	120
r7g.metal	14	120
r7gd.medium	1	4
r7gd.large	2	10
r7gd.xlarge	3	20
r7gd.2xlarge	3	40

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
r7gd.4xlarge	7	60
r7gd.8xlarge	7	60
r7gd.12xlarge	7	60
r7gd.16xlarge	14	120
r7gd.metal	14	120
r7i.large	2	10
r7i.xlarge	3	20
r7i.2xlarge	3	40
r7i.4xlarge	7	60
r7i.8xlarge	7	90
r7i.12xlarge	7	120
r7i.16xlarge	14	120
r7i.24xlarge	14	120
r7i.48xlarge	14	120
r7i.metal-24xl	14	120
r7i.metal-48xl	14	120
r7iz.large	2	10
r7iz.xlarge	3	20
r7iz.2xlarge	3	40
r7iz.4xlarge	7	60

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
r7iz.8xlarge	7	90
r7iz.12xlarge	7	120
r7iz.16xlarge	14	120
r7iz.32xlarge	14	120
r7iz.metal-16xl	14	120
r7iz.metal-32xl	14	120
r8g.medium	1	4
r8g.large	2	10
r8g.xlarge	3	20
r8g.2xlarge	3	40
r8g.4xlarge	7	60
r8g.8xlarge	7	60
r8g.12xlarge	7	60
r8g.16xlarge	14	120
r8g.24xlarge	14	120
r8g.48xlarge	14	120
r8g.metal-24xl	14	120
r8g.metal-48xl	14	120
r8gd.medium	1	4
r8gd.large	2	10

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
r8gd.xlarge	3	20
r8gd.2xlarge	3	40
r8gd.4xlarge	7	60
r8gd.8xlarge	7	60
r8gd.12xlarge	7	60
r8gd.16xlarge	14	120
r8gd.24xlarge	14	120
r8gd.48xlarge	14	120
r8gd.metal-24xl	14	120
r8gd.metal-48xl	14	120
u-3tb1.56xlarge	7	12
u-6tb1.56xlarge	14	12
u-18tb1.112xlarge	14	12
u-18tb1.metal	14	12
u-24tb1.112xlarge	14	12
u-24tb1.metal	14	12
u7i-6tb.112xlarge	14	120
u7i-8tb.112xlarge	14	120
u7i-12tb.224xlarge	14	120
u7in-16tb.224xlarge	15	120

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
u7in-24tb.224xlarge	15	120
u7in-32tb.224xlarge	15	120
u7inh-32tb.480xlarge	15	120
x2gd.medium	1	10
x2gd.large	2	10
x2gd.xlarge	3	20
x2gd.2xlarge	3	40
x2gd.4xlarge	7	60
x2gd.8xlarge	7	60
x2gd.12xlarge	7	60
x2gd.16xlarge	14	120
x2gd.metal	14	120
x2idn.16xlarge	14	120
x2idn.24xlarge	14	120
x2idn.32xlarge	14	120
x2idn.metal	14	120
x2iedn.xlarge	3	13
x2iedn.2xlarge	3	29
x2iedn.4xlarge	7	60
x2iedn.8xlarge	7	120

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
x2iedn.16xlarge	14	120
x2iedn.24xlarge	14	120
x2iedn.32xlarge	14	120
x2iedn.metal	14	120
x2iezn.2xlarge	3	64
x2iezn.4xlarge	7	120
x2iezn.6xlarge	7	120
x2iezn.8xlarge	7	120
x2iezn.12xlarge	14	120
x2iezn.metal	14	120
x8g.medium	1	4
x8g.large	2	10
x8g.xlarge	3	20
x8g.2xlarge	3	40
x8g.4xlarge	7	60
x8g.8xlarge	7	60
x8g.12xlarge	7	60
x8g.16xlarge	14	120
x8g.24xlarge	14	120
x8g.48xlarge	14	120

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
x8g.metal-24xl	14	120
x8g.metal-48xl	14	120

Otimizada para armazenamento

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
i4g.large	2	10
i4g.xlarge	3	20
i4g.2xlarge	3	40
i4g.4xlarge	7	60
i4g.8xlarge	7	60
i4g.16xlarge	14	120
i4i.xlarge	3	8
i4i.2xlarge	3	28
i4i.4xlarge	7	58
i4i.8xlarge	7	118
i4i.12xlarge	7	118
i4i.16xlarge	14	248
i4i.24xlarge	14	118
i4i.32xlarge	14	498

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
i4i.metal	14	498
i7i.large	2	10
i7i.xlarge	3	20
i7i.2xlarge	3	40
i7i.4xlarge	7	60
i7i.8xlarge	7	90
i7i.12xlarge	7	90
i7i.16xlarge	14	120
i7i.24xlarge	14	120
i7i.48xlarge	14	120
i7i.metal-24xl	14	120
i7i.metal-48xl	14	120
i7ie.large	2	20
i7ie.xlarge	3	29
i7ie.2xlarge	3	29
i7ie.3xlarge	3	29
i7ie.6xlarge	7	60
i7ie.12xlarge	7	60
i7ie.18xlarge	14	120
i7ie.24xlarge	14	120

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
i7ie.48xlarge	14	120
i7ie.metal-24xl	14	120
i7ie.metal-48xl	14	120
i8g.large	2	10
i8g.xlarge	3	20
i8g.2xlarge	3	40
i8g.4xlarge	7	60
i8g.8xlarge	7	60
i8g.12xlarge	7	60
i8g.16xlarge	14	120
i8g.24xlarge	14	120
i8g.48xlarge	14	120
i8g.metal-24xl	14	120
im4gn.large	2	10
im4gn.xlarge	3	20
im4gn.2xlarge	3	40
im4gn.4xlarge	7	60
im4gn.8xlarge	7	60
im4gn.16xlarge	14	120
is4gen.medium	1	4

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
is4gen.large	2	10
is4gen.xlarge	3	20
is4gen.2xlarge	3	40
is4gen.4xlarge	7	60
is4gen.8xlarge	7	60

Computação acelerada

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
dl1.24xlarge	59	120
dl2q.24xlarge	14	120
f2.6xlarge	7	90
f2.12xlarge	7	120
f2.48xlarge	14	120
g4ad.xlarge	1	12
g4ad.2xlarge	1	12
g4ad.4xlarge	2	12
g4ad.8xlarge	3	12
g4ad.16xlarge	7	12
g5.xlarge	3	6

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
g5.2xlarge	3	19
g5.4xlarge	7	40
g5.8xlarge	7	90
g5.12xlarge	14	120
g5.16xlarge	7	120
g5.24xlarge	14	120
g5.48xlarge	6	120
g5g.xlarge	3	20
g5g.2xlarge	3	40
g5g.4xlarge	7	60
g5g.8xlarge	7	60
g5g.16xlarge	14	120
g5g.metal	14	120
g6.xlarge	3	20
g6.2xlarge	3	40
g6.4xlarge	7	60
g6.8xlarge	7	90
g6.12xlarge	7	120
g6.16xlarge	14	120
g6.24xlarge	14	120

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
g6.48xlarge	14	120
g6e.xlarge	3	20
g6e.2xlarge	3	40
g6e.4xlarge	7	60
g6e.8xlarge	7	90
g6e.12xlarge	9	120
g6e.16xlarge	14	120
g6e.24xlarge	19	120
g6e.48xlarge	39	120
g6f.large	1	10
g6f.xlarge	3	20
g6f.2xlarge	3	40
g6f.4xlarge	7	60
gr6.4xlarge	7	60
gr6.8xlarge	7	90
gr6f.4xlarge	7	60
inf2.xlarge	3	20
inf2.8xlarge	7	90
inf2.24xlarge	14	120
inf2.48xlarge	14	120

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
p4d.24xlarge	59	120
p4de.24xlarge	59	120
p5.48xlarge	63	242
p5e.48xlarge	63	242
p5en.48xlarge	63	242
p6-b200.48xlarge	31	242
p6e-gb200.36xlarge	38	120
trn1.2xlarge	3	19
trn1.32xlarge	39	120
trn1n.32xlarge	79	242
trn2.48xlarge	31	242
trn2u.48xlarge	31	242
vt1.3xlarge	3	40
vt1.6xlarge	7	60
vt1.24xlarge	14	120

Computação de alta performance

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
hpc6a.48xlarge	1	120

Tipo de instância	Limite de tarefas sem truncamento da ENI	Limite de tarefas com truncamento da ENI
hpc6id.32xlarge	1	120
hpc7g.4xlarge	3	120
hpc7g.8xlarge	3	120
hpc7g.16xlarge	3	120

Reserva de memória da instância de contêiner do Linux no Amazon ECS

Quando o agente de contêiner do Amazon ECS registra uma instância de contêiner em um cluster, o agente deve determinar a quantidade de memória que a instância de contêiner tem disponível com a finalidade de reservar para as tarefas. Devido à sobrecarga de memória da plataforma e à memória ocupada pelo kernel do sistema, esse número é diferente da memória instalada anunciada para instâncias do Amazon EC2. Por exemplo, uma instância `m4.large` tem 8 GiB de memória instalada. No entanto, isso nem sempre significa que exatos 8.192 MiB de memória estão disponíveis para as tarefas quando a instância de contêiner é registrada.

O agente de contêiner do Amazon ECS fornece uma variável de configuração denominada `ECS_RESERVED_MEMORY`, que pode ser usada para remover um número específico de MiB de memória do grupo alocado para as tarefas. Isso reserva de forma efetiva a memória para processos críticos do sistema.

Se você ocupar toda a memória em uma instância de contêiner com as tarefas, é possível que elas disputem a memória com processos essenciais do sistema e iniciem uma falha no sistema.

Por exemplo, se você especificar `ECS_RESERVED_MEMORY=256` no arquivo de configuração do agente de contêiner, o agente registrará a memória total menos 256 MiB para essa instância, e 256 MiB de memória não poderão ser atribuídos para tarefas do ECS. Para obter mais informações sobre as variáveis de configuração do agente e como configurá-las, consulte Configuração do agente de contêiner do Amazon ECS e Inicialização de instâncias de contêiner do Linux no Amazon ECS para transmitir dados.

Se você especificar 8.192 MiB para a tarefa e nenhuma de suas instâncias de contêiner tiver 8.192 MiB ou mais de memória disponível para atender a esse requisito, a tarefa não poderá ser

posicionada no cluster. Se estiver usando um ambiente de computação gerenciado, o AWS Batch deverá executar um tipo de instância maior para acomodar a solicitação.

Você também deve reservar uma quantidade específica de memória para o agente de contêiner do Amazon ECS e outros processos do sistema essenciais nas instâncias de contêiner, para que os contêineres da tarefa não disputem a mesma memória e não iniciem uma possível falha do sistema.

O agente de contêiner do Amazon ECS usa a função `ReadMemInfo()` do Docker para consultar a memória disponível total para o sistema operacional. Tanto o Linux quanto o Windows oferecem utilitários de linha de comando para determinar a memória total.

Example - Determinar a memória total do Linux

O comando free retorna a memória total reconhecida pelo sistema operacional.

```
$ free -b
```

Exemplo de saída para uma instância `m4.large` executando a AMI do Amazon Linux otimizada para Amazon ECS.

```
             total       used       free     shared    buffers     cached
Mem:    8373026816  348180480 8024846336      90112   25534464  205418496
-/+ buffers/cache:  117227520 8255799296
```

Essa instância tem 8.373.026.816 bytes de memória total, ou seja, 7.985 MiB estão disponíveis para tarefas.

Example - Determinar a memória total do Windows

O comando wmic retorna a memória total reconhecida pelo sistema operacional.

```
C:\> wmic ComputerSystem get TotalPhysicalMemory
```

Exemplo de saída para uma instância `m4.large` executando a AMI do Windows otimizada para o Amazon ECS.

```
TotalPhysicalMemory
8589524992
```

Essa instância tem 8.589.524.992 bytes de memória total, ou seja, 8.191 MiB estão disponíveis para tarefas.

Visualização da memória da instância de contêiner

É possível visualizar a quantidade de memória com a qual uma instância de contêiner é registrada no console do Amazon ECS (ou com a operação da API DescribeContainerInstances). Se estiver tentando maximizar a utilização de recursos fornecendo às suas tarefas o máximo de memória possível para um tipo de instância específico, você pode observar a memória disponível para essa instância de contêiner e atribuir essa quantidade de memória às tarefas.

Para visualizar a memória da instância de contêiner

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Clusters e selecione o cluster que hospeda a instância de contêiner.
3. Escolha Infraestrutura e, em Instâncias de contêiner, selecione uma instância de contêiner.
4. A seção Recursos mostra a memória registrada e disponível para a instância de contêiner.

   O valor da memória Registrada é o que a instância de contêiner registrou no Amazon ECS quando foi executada pela primeira vez, e o valor da memória Disponível é o que ainda não foi alocado para tarefas.

Gerenciamento de instâncias de contêiner do Amazon ECS remotamente usando o AWS Systems Manager

É possível usar o recurso Run Command do AWS Systems Manager (Systems Manager) para gerenciar de forma remota e segura a configuração das instâncias de contêiner do Amazon ECS. O Run Command oferece uma maneira simples de executar tarefas administrativas comuns sem fazer login localmente na instância. É possível gerenciar as alterações de configuração entre clusters executando comandos simultaneamente nas várias instâncias de contêiner. O Run Command reporta o status e os resultados de cada comando.

Veja alguns exemplos dos tipos de tarefas que você pode executar com o Run Command:

- Instale ou desinstale pacotes.
- Execute atualizações de segurança.
- Limpe imagens de docker.

- Interrompa ou inicie serviços.
- Visualize os recursos do sistema.
- Visualize os arquivos de log.
- Execute operações de arquivos.

Para obter mais informações sobre o Run Command, consulte Run Command do AWS Systems Manager no Guia do usuário do AWS Systems Manager.

A seguir, apresentamos os pré-requisitos para usar o Systems Manager com o Amazon ECS.

1. Você deve conceder permissões para acessar as APIs do Systems Manager ao perfil de instância de contêiner (ecsInstanceRole). É possível fazer isso ao atribuir AmazonSSMManagedInstanceCore ao perfil `ecsInstanceRole`. Para obter informações sobre como anexar uma política a um perfil, consulte Atualizar permissões para um perfil no Guia do usuário do AWS Identity and Access Management.
2. Verifique se o SSM Agent está instalado nas instâncias de contêiner. Para obter mais informações, consulte Instalar e desinstalar manualmente o SSM Agent em instâncias do EC2 para Linux.

Depois de anexar políticas gerenciadas do Systems Manager a `ecsInstanceRole` e verificar se o Agente (SSM Agent) do AWS Systems Manager está instalado nas instâncias de contêiner, será possível começar a usar o Run Command para enviar comandos às instâncias de contêiner. Para obter informações sobre a execução de comandos e scripts de shell nas instâncias e visualizar a saída resultante, consulte Executar comandos usando o Run Command do Systems Manager e Demonstrações do Run Command no Guia do usuário do AWS Systems Manager.

Um caso de uso comum é atualizar o software da instância de contêiner com Executar comando. Você pode seguir os procedimentos no Guia do usuário do AWS Systems Manager com os parâmetros a seguir.

Parameter	Valor
Documento de comando	`AWS-RunShellScript`
Comando	`$ yum update -y`

Parameter	Valor	
Instâncias de destino	Suas instâncias de contêiner	

Uso de um proxy HTTP para instâncias de contêiner do Linux no Amazon ECS

É possível configurar as instâncias de contêiner do Amazon ECS para usar um proxy HTTP para o agente de contêiner do Amazon ECS e o daemon do Docker. Isso é útil se suas as instâncias de contêiner não têm acesso à rede externa através de um gateway da Internet da Amazon VPC, um gateway NAT ou uma instância.

Para configurar sua instância de contêiner do Linux do Amazon ECS para usar um proxy HTTP, defina as variáveis a seguir nos arquivos relevantes no momento da inicialização (com dados de usuário do Amazon EC2). Além disso, é possível editar manualmente o arquivo de configuração e, em seguida, reiniciar o agente.

`/etc/ecs/ecs.config` (Amazon Linux 2 e AmazonLinux AMI)

`HTTP_PROXY=10.0.0.131:3128`

Defina esse valor para o nome do host (ou endereço IP) e o número de porta de um proxy HTTP a serem usados pelo agente do Amazon ECS para se conectar à Internet. Por exemplo, as instâncias de contêiner podem não ter acesso à rede externa através de um gateway da Internet da Amazon VPC, um gateway NAT ou uma instância.

`NO_PROXY=169.254.169.254,169.254.170.2,/var/run/docker.sock`

Defina esse valor como `169.254.169.254,169.254.170.2,/var/run/docker.sock` para filtrar metadados da instância do EC2, funções do IAM para tarefas e o tráfego do daemon do Docker proveniente do proxy.

`/etc/systemd/system/ecs.service.d/http-proxy.conf` (somente Amazon Linux 2)

`Environment="HTTP_PROXY=10.0.0.131:3128/"`

Defina esse valor como o nome do host (ou endereço IP) e o número de porta de um proxy HTTP a serem usados pelo `ecs-init` para se conectar à Internet. Por exemplo, as instâncias de contêiner podem não ter acesso à rede externa através de um gateway da Internet da Amazon VPC, um gateway NAT ou uma instância.

```
Environment="NO_PROXY=169.254.169.254,169.254.170.2,/var/run/
docker.sock"
```

Defina esse valor como `169.254.169.254,169.254.170.2,/var/run/docker.sock` para filtrar metadados da instância do EC2, funções do IAM para tarefas e o tráfego do daemon do Docker proveniente do proxy.

`/etc/init/ecs.override` (Somente Amazon Linux AMI)

```
env HTTP_PROXY=10.0.0.131:3128
```

Defina esse valor como o nome do host (ou endereço IP) e o número de porta de um proxy HTTP a serem usados pelo `ecs-init` para se conectar à Internet. Por exemplo, as instâncias de contêiner podem não ter acesso à rede externa através de um gateway da Internet da Amazon VPC, um gateway NAT ou uma instância.

```
env NO_PROXY=169.254.169.254,169.254.170.2,/var/run/docker.sock
```

Defina esse valor como `169.254.169.254,169.254.170.2,/var/run/docker.sock` para filtrar metadados da instância do EC2, funções do IAM para tarefas e o tráfego do daemon do Docker proveniente do proxy.

`/etc/systemd/system/docker.service.d/http-proxy.conf` (somente Amazon Linux 2)

```
Environment="HTTP_PROXY=http://10.0.0.131:3128"
```

Defina este valor para o nome do host (ou endereço IP) e o número de porta de um proxy HTTP a serem usados pelo daemon do Docker para se conectar à Internet. Por exemplo, as instâncias de contêiner podem não ter acesso à rede externa através de um gateway da Internet da Amazon VPC, um gateway NAT ou uma instância.

```
Environment="NO_PROXY=169.254.169.254,169.254.170.2"
```

Defina este valor como `169.254.169.254,169.254.170.2` para filtrar os metadados da instância EC2 no proxy.

`/etc/sysconfig/docker` (somente AMI do Amazon Linux AMI e Amazon Linux 2)

```
export HTTP_PROXY=http://10.0.0.131:3128
```

Defina este valor para o nome do host (ou endereço IP) e o número de porta de um proxy HTTP a serem usados pelo daemon do Docker para se conectar à Internet. Por exemplo, as instâncias de contêiner podem não ter acesso à rede externa através de um gateway da Internet da Amazon VPC, um gateway NAT ou uma instância.

`export NO_PROXY=169.254.169.254,169.254.170.2`

Defina este valor como `169.254.169.254,169.254.170.2` para filtrar os metadados da instância EC2 no proxy.

Definir essas variáveis de ambiente nos arquivos acima só afeta o agente de contêiner do Amazon ECS, `ecs-init`, e o daemon do Docker. Elas não configuram nenhum outro serviço (como yum) para usar o proxy.

Para obter informações sobre como configurar o proxy, consulte How do I set up an HTTP proxy for Docker and the Amazon ECS container agent in Amazon Linux 2 or AL2023.

Configuração de instâncias inicializadas previamente para o grupo do Amazon ECS Auto Scaling

O Amazon ECS oferece suporte a grupos de alta atividade do Amazon EC2 Auto Scaling. Um grupo de alta atividade é um grupo de instâncias do Amazon EC2 pré-inicializadas e prontas para serem colocadas em serviço. Sempre que sua aplicação precisar sofrer aumento de escala na horizontal, o Amazon EC2 Auto Scaling usa as instâncias pré-inicializadas do grupo de alta atividade em vez de iniciar instâncias frias, permite que qualquer processo de inicialização final seja executado e, em seguida, coloca a instância em serviço.

Para saber mais sobre grupos de alta atividade e como adicionar um grupo de alta atividade ao seu grupo do Auto Scaling, consulte Grupos de alta atividade para o Amazon EC2 Auto Scaling no Guia do usuário do Amazon EC2 Auto Scaling.

Ao criar ou atualizar um grupo de alta atividade para um grupo do Auto Scaling para o Amazon ECS, não é possível definir a opção que retorna instâncias para o grupo de alta atividade ao reduzir a escala horizontalmente (`ReuseOnScaleIn`). Para obter mais informações, consulte put-warm-pool na Referência da AWS Command Line Interface.

Para usar grupos de alta atividade com seu cluster do Amazon ECS, defina a variável de configuração do agente `ECS_WARM_POOLS_CHECK` como `true` no campo User data (Dados do usuário) do seu modelo de inicialização do grupo do Amazon EC2 Auto Scaling.

A seguir há um exemplo de como a variável de configuração do agente pode ser especificada no campo User data (Dados do usuário) de um modelo de inicialização do Amazon EC2. Substitua *`MyCluster`* pelo nome do seu cluster.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
```

```
ECS_CLUSTER=MyCluster
ECS_WARM_POOLS_CHECK=true
EOF
```

Só há suporte para a variável `ECS_WARM_POOLS_CHECK` nas versões `1.59.0` e posteriores do agente. Para obter mais informações sobre as variáveis, consulte Configuração do agente de contêiner do Amazon ECS.

Atualizar o agente de contêiner do Amazon ECS

Ocasionalmente, pode ser necessário atualizar o agente de contêiner do Amazon ECS para obter correções de erros e novos recursos. A atualização do agente de contêiner do Amazon ECS não interrompe a execução das tarefas ou dos serviços na instância de contêiner. O processo de atualização do agente varia, dependendo do local em que a instância de contêiner foi iniciada, se na AMI otimizada para Amazon ECS ou em outro sistema operacional.

> **Note**
>
> As atualizações de agente não se aplicam a instâncias de contêiner do Windows. É recomendável executar novas instâncias de contêiner para atualizar a versão do agente nos clusters do Windows.

Verificar a versão do agente de contêiner do Amazon ECS

É possível verificar a versão do agente de contêiner que está sendo executada nas instâncias de contêiner para saber se é preciso atualizá-la. A exibição da instância de contêiner no console do Amazon ECS fornece a versão do agente. Use o procedimento a seguir para verificar a versão do agente.

Amazon ECS console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, escolha a região em que sua instância externa está registrada.
3. No painel de navegação, escolha Clusters e selecione o cluster que hospeda a instância externa.
4. Na página Cluster : ***name***, escolha a guia Infrastructure (Infraestrutura).
5. Em Container instances (Instâncias de contêiner), observe a coluna Agent version (Versão do agente) para suas instâncias de contêiner. Se a instância de contêiner não contiver a

versão mais recente do agente de contêiner, o console o alertará com uma mensagem e sinalizará a versão desatualizada do agente.

Se a sua versão do agente estiver desatualizada, será possível atualizar seu agente de contêiner com os procedimentos a seguir:

- Se sua instância de contêiner estiver executando a AMI otimizada para o Amazon ECS, consulte Atualizar o agente de contêiner do Amazon ECS em uma AMI otimizada para Amazon ECS.
- Se sua instância de contêiner não estiver executando a AMI otimizada para o Amazon ECS, consulte Atualizar manualmente o agente de contêiner do Amazon ECS (para AMIs não otimizadas para Amazon ECS).

> **Important**
>
> Para atualizar versões do agente do Amazon ECS anteriores à v1.0.0 na AMI otimizada para Amazon ECS, recomendamos que você encerre a instância de contêiner atual e execute uma nova instância com a versão mais recente da AMI. Todas as instâncias de contêiner que usam uma versão de pré-visualização devem ser retiradas e substituídas pela AMI mais recente. Para obter mais informações, consulte Iniciar uma instância de contêiner do Linux do Amazon ECS.

Amazon ECS container agent introspection API

Também é possível usar a API de introspecção do agente de contêiner do Amazon ECS para verificar a versão do agente na própria instância de contêiner. Para obter mais informações, consulte Introspecção de contêiner do Amazon ECS.

Para verificar se o agente de contêiner do Amazon ECS está executando a versão mais recente com a API de introspecção

1. Faça login em sua instância de contêiner via SSH.
2. Consulte a API de introspecção.

```
[ec2-user ~]$ curl -s 127.0.0.1:51678/v1/metadata | python3 -mjson.tool
```

> **Note**
>
> A API de introspecção incluiu informações de `Version` na versão v1.0.0 do agente de contêiner do Amazon ECS. Se `Version` não estiver presente ao consultar a API de introspecção ou ela não estiver presente no seu agente, a versão que você está executando é v0.0.3 ou anterior. Você deve atualizar a versão.

Atualizar o agente de contêiner do Amazon ECS em uma AMI otimizada para Amazon ECS

Se você estiver usando a AMI otimizada para Amazon ECS, terá várias opções para obter a versão mais recente do agente de contêiner do Amazon ECS (mostrado por ordem de recomendação):

- Encerre as instâncias de contêiner e inicie a versão mais recente da AMI do Amazon Linux 2 otimizada para Amazon ECS (manualmente ou atualizando a configuração de execução do Auto Scaling com a AMI mais recente). O resultado será uma instância de contêiner atualizada com as versões testadas e validadas mais recentes do Amazon Linux, do Docker, do `ecs-init` e do agente de contêiner do Amazon ECS. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.
- Conecte-se à instância com o SSH e atualize o pacote do `ecs-init` (e suas dependências) para a versão mais recente. Essa operação fornecerá as versões testadas e validadas mais recentes do Docker e do `ecs-init` que estão disponíveis nos repositórios do Amazon Linux e na versão mais recente do agente de contêiner do Amazon ECS. Para obter mais informações, consulte Para atualizar o pacote `ecs-init` em uma AMI otimizada para o Amazon ECS.
- Atualize o agente de contêiner com a operação de API do `UpdateContainerAgent`, através do console ou com a AWS CLI ou os SDKs da AWS. Para obter mais informações, consulte Atualizar o agente de contêiner do Amazon ECS com a operação de API do `UpdateContainerAgent`.

> **Note**
>
> As atualizações de agente não se aplicam a instâncias de contêiner do Windows. É recomendável executar novas instâncias de contêiner para atualizar a versão do agente nos clusters do Windows.

Para atualizar o pacote **`ecs-init`** em uma AMI otimizada para o Amazon ECS

1. Faça login em sua instância de contêiner via SSH.
2. Atualize o pacote `ecs-init` com o comando a seguir.

```
sudo yum update -y ecs-init
```

> **Note**
>
> O pacote `ecs-init` e o agente de contêiner do Amazon ECS são atualizados imediatamente. No entanto, as versões mais recentes do Docker não serão carregadas até que o daemon do Docker seja reiniciado. Inicie novamente reinicializando a instância ou executando os seguintes comandos em sua instância:
>
> - AMI do Amazon Linux 2 otimizada para Amazon ECS:
>
>   ```
>   sudo systemctl restart docker
>   ```
>
> - AMI do Amazon Linux otimizada para Amazon ECS:
>
>   ```
>   sudo service docker restart && sudo start ecs
>   ```

Atualizar o agente de contêiner do Amazon ECS com a operação de API do **`UpdateContainerAgent`**

> **Important**
>
> A API do `UpdateContainerAgent` só é compatível com variantes Linux da AMI otimizada para o Amazon ECS, com exceção da AMI do Amazon Linux 2 (arm64) otimizada para Amazon ECS. Para instâncias de contêiner que usam a AMI do Amazon Linux 2 (arm64) otimizada para Amazon ECS, atualize o pacote `ecs-init` para atualizar o agente. Para instâncias de contêiner que estão executando outros sistemas operacionais, consulte Atualizar manualmente o agente de contêiner do Amazon ECS (para AMIs não otimizadas para Amazon ECS). Se você está usando instâncias de container do Windows. recomendamos que você inicie novas instâncias de contêiner para atualizar a versão do agente nos clusters do Windows.

O processo da API do `UpdateContainerAgent` começa quando você solicita uma atualização do agente, por meio do console ou com a AWS CLI ou AWS SDKs. O Amazon ECS compara a versão atual do agente com a versão mais recente disponível e verifica se é possível realizar uma atualização. Se uma atualização não estiver disponível, por exemplo, se o agente já estiver executando a versão mais recente, um `NoUpdateAvailableException` será retornado.

Os estágios do processo de atualização mostrados acima são os seguintes:

`PENDING`

Uma atualização de agente está disponível, e o processo de atualização iniciou.

`STAGING`

O agente começou a baixar a atualização do agente. Se o agente não conseguir baixar a atualização ou se o conteúdo da atualização estiver incorreto ou corrompido, o agente enviará uma notificação da falha, e a atualização passará para o estado `FAILED`.

`STAGED`

O download do agente foi concluído e o conteúdo do agente foi verificado.

`UPDATING`

O serviço `ecs-init` é reiniciado e obtém a nova versão do agente. Se, por algum motivo, não for possível reiniciar o agente, a atualização passará para o estado `FAILED`; do contrário, o agente informará ao Amazon ECS que a atualização foi concluída.

> **Note**
>
> As atualizações de agente não se aplicam a instâncias de contêiner do Windows. É recomendável executar novas instâncias de contêiner para atualizar a versão do agente nos clusters do Windows.

Para atualizar o agente de contêiner do Amazon ECS em uma AMI otimizada para Amazon ECS no console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, escolha a região em que sua instância externa está registrada.
3. No painel de navegação, escolha Clusters e selecione o cluster.

4. Na página Cluster : ***name***, escolha a guia Infrastructure (Infraestrutura).
5. Em Instâncias de contêiner, selecione as instâncias a serem atualizadas e escolha Ações, Agente de atualização.

Atualizar manualmente o agente de contêiner do Amazon ECS (para AMIs não otimizadas para Amazon ECS)

Ocasionalmente, pode ser necessário atualizar o agente de contêiner do Amazon ECS para obter correções de erros e novos recursos. A atualização do agente de contêiner do Amazon ECS não interrompe a execução das tarefas ou dos serviços na instância de contêiner.

> **Note**
>
> As atualizações de agente não se aplicam a instâncias de contêiner do Windows. É recomendável executar novas instâncias de contêiner para atualizar a versão do agente nos clusters do Windows.

1. Faça login em sua instância de contêiner via SSH.
2. Verifique se o seu agente usa a variável de ambiente `ECS_DATADIR` para salvar seu estado.

   ```
   ubuntu:~$ docker inspect ecs-agent | grep ECS_DATADIR
   ```

   Saída:

   ```
   "ECS_DATADIR=/data",
   ```

   > **Important**
   >
   > Se o comando anterior não retornar a variável de ambiente `ECS_DATADIR`, você deverá interromper todas as tarefas em execução nessa instância de contêiner antes de atualizar seu agente. Agentes mais novos com a variável de ambiente `ECS_DATADIR` salvam seu estado, e você pode atualizá-los enquanto as tarefas são executadas sem problemas.

3. Interrompa o agente de contêiner do Amazon ECS.

```
ubuntu:~$ docker stop ecs-agent
```

4. Exclua o contêiner de agente.

```
ubuntu:~$ docker rm ecs-agent
```

5. Verifique se o diretório `/etc/ecs` e o arquivo de configuração do agente de contêiner do Amazon ECS existem em `/etc/ecs/ecs.config`.

```
ubuntu:~$ sudo mkdir -p /etc/ecs && sudo touch /etc/ecs/ecs.config
```

6. Edite o arquivo `/etc/ecs/ecs.config` e verifique se ele contém pelo menos as seguintes instruções de variável. Se você não quiser que sua instância de contêiner seja registrada no cluster padrão, especifique seu nome de cluster como o valor para `ECS_CLUSTER`.

```
ECS_DATADIR=/data
ECS_ENABLE_TASK_IAM_ROLE=true
ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST=true
ECS_LOGFILE=/log/ecs-agent.log
ECS_AVAILABLE_LOGGING_DRIVERS=["json-file","awslogs"]
ECS_LOGLEVEL=info
ECS_CLUSTER=default
```

Para obter mais informações sobre essas e outras opções de runtime de agente, consulte Configuração do agente de contêiner do Amazon ECS.

> **Note**
>
> É possível, opcionalmente, armazenar suas variáveis de ambiente do agente no Amazon S3 (que podem ser baixadas para as instâncias de contêiner no momento da inicialização usando os dados de usuário do Amazon EC2). Isso é recomendado para informações confidenciais, como credenciais de autenticação para repositórios privados. Para obter mais informações, consulte Armazenamento da configuração da instância de contêiner do Amazon ECS no Amazon S3 e Uso de imagens de contêiner que não são da AWS no Amazon ECS.

7. Extraia a imagem de agente de contêiner do Amazon ECS mais recente do Amazon Elastic Container Registry Public.

```
ubuntu:~$ docker pull public.ecr.aws/ecs/amazon-ecs-agent:latest
```

Saída:

```
Pulling repository amazon/amazon-ecs-agent
a5a56a5e13dc: Download complete
511136ea3c5a: Download complete
9950b5d678a1: Download complete
c48ddcf21b63: Download complete
Status: Image is up to date for amazon/amazon-ecs-agent:latest
```

8. Execute o agente de contêiner do Amazon ECS mais recente na instância de contêiner.

> **Note**
>
> Use políticas de reinicialização do Docker ou um gerenciador de processos (como upstart ou systemd) para tratar o agente de contêiner como um serviço ou um daemon e garantir que ele seja reiniciado após sair. A AMI otimizada para Amazon ECS usa o RPM `ecs-init` para essa finalidade, e você pode visualizar o código-fonte para esse RPM no GitHub.

O comando de execução de agente do exemplo a seguir é dividido em linhas separadas para mostrar cada opção. Para obter mais informações sobre essas e outras opções de runtime de agente, consulte Configuração do agente de contêiner do Amazon ECS.

> **Important**
>
> Os sistemas operacionais com o SELinux habilitado requerem a opção `--privileged` em seu comando docker run. Além disso, para instâncias de contêiner habilitadas para SELinux, recomendamos que você adicione a opção `:Z` às montagens de volume `/log` e `/data`. No entanto, as montagens de host para esses volumes devem existir para que você execute o comando; caso contrário, você receberá um erro `no such file or directory`. Execute a seguinte ação se você tiver dificuldades para executar o agente do Amazon ECS em uma instância de contêiner habilitada para o SELinux:
>
> - Crie os pontos de montagem de volume de host na instância de contêiner.

```
ubuntu:~$ sudo mkdir -p /var/log/ecs /var/lib/ecs/data
```

- Adicione a opção `--privileged` ao comando docker run abaixo.
- Anexe a opção `:Z` às montagens de volume de contêiner `/log` e `/data` (por exemplo, `--volume=/var/log/ecs/:/log:Z`) ao comando docker run abaixo.

```
ubuntu:~$ sudo docker run --name ecs-agent \
--detach=true \
--restart=on-failure:10 \
--volume=/var/run:/var/run \
--volume=/var/log/ecs/:/log \
--volume=/var/lib/ecs/data:/data \
--volume=/etc/ecs:/etc/ecs \
--volume=/etc/ecs:/etc/ecs/pki \
--net=host \
--env-file=/etc/ecs/ecs.config \
amazon/amazon-ecs-agent:latest
```

> **Note**
>
> Se você receber uma mensagem `Error response from daemon: Cannot start container`, poderá excluir o contêiner com falha com o comando sudo docker rm ecs-agent e tentar executar o comando novamente.

AMIs do Windows otimizadas para o Amazon ECS

As AMIs otimizadas para Amazon ECS são pré-configuradas com os componentes necessários para a execução de workloads do ECS. Embora você possa criar sua própria AMI de instância de contêiner que atenda às especificações básicas necessárias para executar workloads em contêineres no Amazon ECS, as AMIs otimizadas para Amazon ECS são pré-configuradas e testadas no Amazon ECS por engenheiros da AWS. É a maneira mais simples de você começar e fazer com que seus contêineres sejam executados na AWS rapidamente.

Para cada variante, os metadados da AMI otimizada para o Amazon ECS, incluindo o nome da AMI, a versão do agente do contêiner do Amazon ECS e a versão do runtime do Amazon ECS, que inclui a versão do Docker, podem ser recuperados de forma programática. Para obter mais informações,

consulte the section called “Recuperação de metadados da AMI do Windows otimizada para o Amazon ECS”.

> **Important**
>
> Todas as variantes da AMI otimizadas para o ECS produzidas depois de agosto de 2022 serão migradas do Docker EE (Mirantis) para o Docker CE (projeto Moby).
> Para garantir que os clientes tenham as atualizações de segurança mais recentes por padrão, o Amazon ECS mantém pelo menos as três últimas AMIs otimizadas para Amazon ECS do Windows. Depois de lançar novas AMIs otimizadas para Amazon ECS do Windows, o Amazon ECS torna privadas as AMIs otimizadas para Amazon ECS do Windows mais antigas. Se você precisar ter acesso a uma AMI privada, avise-nos abrindo um tíquete no suporte à nuvem.

Variantes da AMI otimizada para Amazon ECS

As seguintes variantes do Windows Server da AMI otimizada para Amazon ECS estão disponíveis para instâncias do Amazon EC2.

> **Important**
>
> Todas as variantes de AMI otimizadas para o ECS produzidas após agosto serão migradas do Docker EE (Mirantis) para o Docker CE (projeto Moby).

- AMI do Windows Server 2025 Full otimizada para Amazon ECS
- AMI do Windows Server 2025 Core otimizada para Amazon ECS
- AMI do Windows Server 2022 Full otimizada para Amazon ECS
- AMI do Windows Server 2022 Core otimizada para Amazon ECS
- AMI do Windows Server 2019 Full otimizada para Amazon ECS
- AMI do Windows Server 2019 Core otimizada para Amazon ECS
- AMI do Windows Server 2016 Full otimizada para Amazon ECS

> **Important**
>
> O Windows Server 2016 não oferece suporte à versão mais recente do Docker, por exemplo, 25.x.x. Portanto, as AMIs do Windows Server 2016 Full não receberão patches de segurança ou bugs no runtime do Docker. Recomendamos que você mude para uma das seguintes plataformas do Windows:
>
> - Windows Server 2022 Full
> - Windows Server 2022 Core
> - Windows Server 2019 Full
> - Windows Server 2019 Core

Em 9 de agosto de 2022, a AMI do Windows Server 20H2 Core otimizada para Amazon ECS atingiu sua data de término do suporte. Nenhuma nova versão dessa AMI será lançada. Para obter mais informações, consulte Windows Server release information.

O Windows Server 2025, Windows Server 2022, Windows Server 2019 e o Windows Server 2016 são versões de canal de manutenção em longo prazo (LTSC). O Windows Server 20H2 é uma versão do canal semestral (SAC). Para obter mais informações, consulte Windows Server release information.

Considerações

Veja algumas coisas que você deve saber sobre contêineres do Windows do Amazon EC2 e o Amazon ECS.

- Os contêineres do Windows não podem ser executados em instâncias de contêiner do Linux, e o oposto também ocorre. Para uma melhor colocação de tarefas para Windows e Linux, mantenha as instâncias de contêiner do Windows e do Linux em clusters separados e só coloque tarefas do Windows em clusters do Windows. É possível garantir que as definições de tarefa do Windows só sejam colocadas em instâncias do Windows, para tal definindo a restrição de posicionamento a seguir: `memberOf(ecs.os-type=='windows')`.
- Há suporte dos contêineres do Windows para tarefas que usem o tipo de inicialização do EC2 e do Fargate.
- Os contêineres do Windows e as instâncias de contêiner não podem oferecer suporte a todos os parâmetros de definição de tarefa disponíveis para contêineres e instâncias de contêiner do Linux.

Para alguns parâmetros, eles sequer são compatíveis e outros se comportam de maneira diferente no Windows em comparação com o Linux. Para obter mais informações, consulte Diferenças de definição de tarefa do Amazon ECS para instâncias do EC2 executando o Windows.

- Para o recurso de funções do IAM para tarefas, é necessário configurar as instâncias de contêiner do Windows para permitir o recurso na inicialização. Seus contêineres devem executar algum código do PowerShell fornecido quando usam o recurso. Para obter mais informações, consulte Configuração adicional de instância do Windows no Amazon EC2.
- O recurso de funções do IAM para tarefas usa um proxy de credencial para fornecer credenciais aos contêineres. Como esse proxy de credencial ocupa a porta 80 na instância de contêiner, caso você use funções do IAM para as tarefas, a porta 80 não permanecerá disponível para tarefas. Para contêineres de serviço da Web, você pode usar o Application Load Balancer e o mapeamento de porta dinâmico para fornecer conexões de porta 80 HTTP padrão aos contêineres. Para obter mais informações, consulte Uso do balanceamento de carga para distribuir o tráfego de serviço do Amazon ECS.
- As imagens do Docker do servidor do Windows são grandes (9 GiB). Dessa forma, as instâncias de contêiner do Windows exigem mais espaço de armazenamento do que as instâncias de contêiner do Linux.
- Para executar um contêiner do Windows em um Windows Server, a versão do sistema operacional da imagem base do contêiner deve corresponder à do host. Para obter mais informações, consulte Compatibilidade de versão do contêiner do Windows no site de documentação da Microsoft. Caso seu cluster execute várias versões do Windows, é possível garantir que uma tarefa seja colocada em uma instância do EC2 em execução na mesma versão usando a restrição de posicionamento: `memberOf(attribute:ecs.os-family == WINDOWS_SERVER_<OS_Release>_<FULL or CORE>)`. Para obter mais informações, consulte the section called “Recuperação de metadados da AMI do Windows otimizada para o Amazon ECS”.

Recuperação de metadados da AMI do Windows otimizada para o Amazon ECS

O ID da AMI, o nome da imagem, o sistema operacional, a versão do agente de contêiner e a versão do runtime para cada variante das AMIs otimizadas para Amazon ECS podem ser recuperados de maneira programática, consultando a API do Systems Manager Parameter Store. Para obter mais informações sobre a API do Systems Manager Parameter Store, consulte GetParameters e GetParametersByPath.

> **Note**
>
> O usuário administrador precisa ter as seguintes permissões do IAM para recuperar os metadados da AMI otimizada para Amazon ECS. Essas permissões foram adicionadas à política `AmazonECS_FullAccess` do IAM.
>
> - ssm:GetParameters
> - ssm:GetParameter
> - ssm:GetParametersByPath

Formato de parâmetro do Systems Manager Parameter Store

> **Note**
>
> Os parâmetros a seguir da API do Systems Manager Parameter Store estão obsoletos e não devem ser usados para recuperar as AMIs mais recentes do Windows:
>
> - `/aws/service/ecs/optimized-ami/windows_server/2016/english/full/recommended/image_id`
> - `/aws/service/ecs/optimized-ami/windows_server/2019/english/full/recommended/image_id`

Veja a seguir o formato do nome do parâmetro para cada variante da AMI otimizada para Amazon ECS.

- Metadados da AMI do Windows Server 2025 Full:

```
/aws/service/ami-windows-latest/Windows_Server-2025-English-Full-ECS_Optimized
```

- Metadados da AMI do Windows Server 2025 Core:

```
/aws/service/ami-windows-latest/Windows_Server-2025-English-Core-ECS_Optimized
```

- Metadados da AMI do Windows Server 2022 Full:

```
/aws/service/ami-windows-latest/Windows_Server-2022-English-Full-ECS_Optimized
```

- Metadados da AMI do Windows Server 2022 Core:

```
/aws/service/ami-windows-latest/Windows_Server-2022-English-Core-ECS_Optimized
```

- Metadados da AMI do Windows Server 2019 Full:

```
/aws/service/ami-windows-latest/Windows_Server-2019-English-Full-ECS_Optimized
```

- Metadados da AMI do Windows Server 2019 Core:

```
/aws/service/ami-windows-latest/Windows_Server-2019-English-Core-ECS_Optimized
```

- Metadados da AMI do Windows Server 2016 Full:

```
/aws/service/ami-windows-latest/Windows_Server-2016-English-Full-ECS_Optimized
```

O formato de nome de parâmetro a seguir recupera os metadados da última versão estável da AMI do Windows Server 2019 Full.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-
English-Full-ECS_Optimized
```

Veja a seguir um exemplo do objeto JSON retornado para o valor do parâmetro.

```
{
    "Parameters": [
        {
            "Name": "/aws/service/ami-windows-latest/Windows_Server-2019-English-Full-
ECS_Optimized",
            "Type": "String",
            "Value": "{\"image_name\":\"Windows_Server-2019-English-Full-
ECS_Optimized-2023.06.13\",\"image_id\":\"ami-0debc1fb48e4aee16\",\"ecs_runtime_version
\":\"Docker (CE) version 20.10.21\",\"ecs_agent_version\":\"1.72.0\"}",
            "Version": 58,
            "LastModifiedDate": "2023-06-22T19:37:37.841000-04:00",
            "ARN": "arn:aws:ssm:us-east-1::parameter/aws/service/ami-windows-latest/
Windows_Server-2019-English-Full-ECS_Optimized",
            "DataType": "text"
        }
    ],
    "InvalidParameters": []
```

```
}
```

Cada um dos campos na saída acima está disponível para ser consultado como subparâmetros. Crie o caminho do parâmetro para um subparâmetro anexando o nome do subparâmetro ao caminho para a AMI selecionada. Os seguintes subparâmetros estão disponíveis:

- `schema_version`
- `image_id`
- `image_name`
- `os`
- `ecs_agent_version`
- `ecs_runtime_version`

Exemplos

Os exemplos a seguir mostram maneiras como você pode recuperar os metadados de cada variante da AMI otimizada para Amazon ECS.

Recuperar os metadados da AMI otimizada para Amazon ECS estável mais recente

É possível recuperar a AMI otimizada para Amazon ECS estável mais recente por meio da AWS CLI, usando os comandos da AWS CLI a seguir.

- Para a AMI do Windows Server 2025 Full otimizada para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2025-English-Full-ECS_Optimized --region us-east-1
```

- Para a AMI do Windows Server 2025 Core otimizada para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2025-English-Core-ECS_Optimized --region us-east-1
```

- Para a AMI do Windows Server 2022 Full otimizada para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-English-Full-ECS_Optimized --region us-east-1
```

- Para a AMI do Windows Server 2022 Core otimizada para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-English-Core-ECS_Optimized --region us-east-1
```

- Para a AMI do Windows Server 2019 Full otimizada para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-English-Full-ECS_Optimized --region us-east-1
```

- Para a AMI do Windows Server 2019 Core otimizada para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-English-Core-ECS_Optimized --region us-east-1
```

- Para a AMI do Windows Server 2016 Full otimizada para Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2016-English-Full-ECS_Optimized --region us-east-1
```

Usar a mais recente e recomendada AMI otimizada para Amazon ECS em um modelo do AWS CloudFormation

É possível referenciar a mais recente e recomendada AMI otimizada para Amazon ECS em um modelo do AWS CloudFormation fazendo referência ao nome do armazenamento de parâmetros do Systems Manager.

```
Parameters:
  LatestECSOptimizedAMI:
    Description: AMI ID
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ami-windows-latest/Windows_Server-2019-English-Full-ECS_Optimized/image_id
```

Versões da AMI do Windows otimizada para o Amazon ECS

Visualize as versões atuais e anteriores das AMIs otimizadas para o Amazon ECS e suas versões correspondentes do agente de contêiner do Amazon ECS, do Docker e do pacote `ecs-init`.

Os metadados da AMI otimizada para Amazon ECS, incluindo o ID da AMI, podem ser recuperados de maneira programática em cada variante. Para obter mais informações, consulte the section called “Recuperação de metadados da AMI do Windows otimizada para o Amazon ECS”.

As guias a seguir exibem uma lista de versões de AMIs do Windows otimizadas para Amazon ECS. Para obter detalhes sobre como referenciar o parâmetro do Systems Manager Parameter Store em um modelo do AWS CloudFormation, consulte Usar a mais recente e recomendada AMI otimizada para Amazon ECS em um modelo do AWS CloudFormation.

Important

Para garantir que os clientes tenham as atualizações de segurança mais recentes por padrão, o Amazon ECS mantém pelo menos as três últimas AMIs otimizadas para Amazon ECS do Windows. Depois de lançar novas AMIs otimizadas para Amazon ECS do Windows, o Amazon ECS torna privadas as AMIs otimizadas para Amazon ECS do Windows mais antigas. Se você precisar ter acesso a uma AMI privada, avise-nos abrindo um tíquete no suporte à nuvem.

O Windows Server 2016 não oferece suporte à versão mais recente do Docker, por exemplo, 25.x.x. Portanto, as AMIs do Windows Server 2016 Full não receberão patches de segurança ou bugs no runtime do Docker. Recomendamos que você mude para uma das seguintes plataformas do Windows:

- Windows Server 2022 Full
- Windows Server 2022 Core
- Windows Server 2019 Full
- Windows Server 2019 Core

Windows Server 2025 Full AMI versions

A tabela a seguir lista as versões atuais e anteriores da AMI do Windows Server 2025 Full otimizada para Amazon ECS e suas versões correspondentes do agente de contêiner do Amazon ECS e do Docker.

AMI do Windows Server 2025 Full otimizada para Amazon ECS	Versão do agente de contêiner do Amazon ECS	Versão do Docker	Visibilidade
Windows_Server-2025-English-Full-ECS_Optimized-2025.06.13	1.94.0	25.0.6 (Docker CE)	Public

Use o seguinte comando da AWS CLI para recuperar a AMI do Windows Server 2025 Full otimizada para Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2025-English-Full-ECS_Optimized
```

Windows Server 2025 Core AMI versions

A tabela a seguir lista as versões atuais e anteriores da AMI do Windows Server 2025 Core otimizada para Amazon ECS e suas versões correspondentes do agente de contêiner do Amazon ECS e do Docker.

AMI do Windows Server 2025 Core otimizada para Amazon ECS	Versão do agente de contêiner do Amazon ECS	Versão do Docker	Visibilidade
Windows_Server-2025-English-Core-ECS_Optimized-2025.06.13	1.94.0	25.0.6 (Docker CE)	Public

Use o seguinte comando da AWS CLI para recuperar a AMI do Windows Server 2025 Core otimizada para Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2025-English-Core-ECS_Optimized
```

Windows Server 2022 Full AMI versions

A tabela a seguir lista as versões atuais e anteriores da AMI do Windows Server 2022 Full otimizada para Amazon ECS e suas versões correspondentes do agente de contêiner do Amazon ECS e do Docker.

AMI do Windows Server 2022 Full otimizada para Amazon ECS	Versão do agente de contêiner do Amazon ECS	Versão do Docker	Visibilidade
Windows_Server-2022-English-Full-ECS_Optimized-2025.06.13	1.94.0	25.0.6 (Docker CE)	Public
Windows_Server-2022-English-Full-ECS_Optimized-2025.05.17	1.93.1	25.0.6 (Docker CE)	Public
Windows_Server-2022-English-Full-ECS_Optimized-2025.4.14	1.91.2	25.0.6 (Docker CE)	Public
Windows_Server-2022-English-Full-ECS_Optimized-2025.3.17	1.91.0	25.0.6 (Docker CE)	Public
Windows_Server-2022-English-Full-ECS_Optimized-2025.2.14	1.90.0	25.0.6 (Docker CE)	Public

Use o seguinte comando da AWS CLI para recuperar a AMI do Windows Server 2022 Full otimizada para Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-English-Full-ECS_Optimized
```

Windows Server 2022 Core AMI versions

A tabela a seguir lista as versões atuais e anteriores da AMI do Windows Server 2022 Core otimizada para Amazon ECS e suas versões correspondentes do agente de contêiner do Amazon ECS e do Docker.

AMI do Windows Server 2022 Core otimizada para Amazon ECS	Versão do agente de contêiner do Amazon ECS	Versão do Docker	Visibilidade
Windows_Server-2022-English-Core-ECS_Optimized-2025.06.13	1.94.0	25.0.6 (Docker CE)	Public
Windows_Server-2022-English-Core-ECS_Optimized-2025.05.17	1.93.1	25.0.6 (Docker CE)	Public
Windows_Server-2022-English-Core-ECS_Optimized-2025.4.14	1.91.2	25.0.6 (Docker CE)	Public
Windows_Server-2022-English-Core-ECS_Optimized-2025.3.17	1.91.0	25.0.6 (Docker CE)	Public
Windows_Server-2022-English-Core-ECS	1.90.0	25.0.6 (Docker CE)	Public

AMI do Windows Server 2022 Core otimizada para Amazon ECS	Versão do agente de contêiner do Amazon ECS	Versão do Docker	Visibilidade
_Optimized-2025.2.14			

Use o seguinte comando da AWS CLI para recuperar a AMI do Windows Server 2022 Full otimizada para Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-English-Core-ECS_Optimized
```

Windows Server 2019 Full AMI versions

A tabela a seguir lista as versões atuais e anteriores da AMI do Windows Server 2019 Full otimizada para Amazon ECS e suas versões correspondentes do agente de contêiner do Amazon ECS e do Docker.

AMI do Windows Server 2019 Full otimizada para Amazon ECS	Versão do agente de contêiner do Amazon ECS	Versão do Docker	Visibilidade
Windows_Server-2019-English-Full-ECS_Optimized-2025.06.13	1.94.0	25.0.6 (Docker CE)	Public
Windows_Server-2019-English-Full-ECS_Optimized-2025.05.17	1.93.1	25.0.6 (Docker CE)	Public
Windows_Server-2019-English-Full-ECS	1.91.2	25.0.6 (Docker CE)	Public

AMI do Windows Server 2019 Full otimizada para Amazon ECS	Versão do agente de contêiner do Amazon ECS	Versão do Docker	Visibilidade
_Optimized-2025.4.14			
Windows_Server-2019-English-Full-ECS_Optimized-2025.3.17	1.91.0	25.0.6 (Docker CE)	Public
Windows_Server-2019-English-Full-ECS_Optimized-2025.2.14	1.90.0	25.0.6 (Docker CE)	Public

Use o seguinte comando da AWS CLI para recuperar a AMI do Windows Server 2019 Full otimizada para Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-English-Full-ECS_Optimized
```

Windows Server 2019 Core AMI versions

A tabela a seguir lista as versões atuais e anteriores da AMI do Windows Server 2019 Core otimizada para Amazon ECS e suas versões correspondentes do agente de contêiner do Amazon ECS e do Docker.

AMI do Windows Server 2019 Core otimizada para Amazon ECS	Versão do agente de contêiner do Amazon ECS	Versão do Docker	Visibilidade
Windows_Server-2019-English-Core-ECS	1.94.0	25.0.6 (Docker CE)	Public

AMI do Windows Server 2019 Core otimizada para Amazon ECS	Versão do agente de contêiner do Amazon ECS	Versão do Docker	Visibilidade
_Optimized-2025.06.13			
Windows_Server-2019-English-Core-ECS_Optimized-2025.05.17	1.93.1	25.0.6 (Docker CE)	Public
Windows_Server-2019-English-Core-ECS_Optimized-2025.4.14	1.91.2	25.0.6 (Docker CE)	Public
Windows_Server-2019-English-Core-ECS_Optimized-2025.3.17	1.91.0	25.0.6 (Docker CE)	Public
Windows_Server-2019-English-Core-ECS_Optimized-2025.2.14	1.90.0	25.0.6 (Docker CE)	Public

Use o seguinte comando da AWS CLI para recuperar a AMI do Windows Server 2019 Full otimizada para Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-English-Core-ECS_Optimized
```

Windows Server 2016 Full AMI versions

> **Important**
>
> O Windows Server 2016 não oferece suporte à versão mais recente do Docker, por exemplo, 25.x.x. Portanto, as AMIs do Windows Server 2016 Full não receberão patches de segurança ou bugs no runtime do Docker. Recomendamos que você mude para uma das seguintes plataformas do Windows:
>
> - Windows Server 2022 Full
> - Windows Server 2022 Core
> - Windows Server 2019 Full
> - Windows Server 2019 Core

A tabela a seguir lista as versões atuais e anteriores da AMI do Windows Server 2016 Full otimizada para Amazon ECS e suas versões correspondentes do agente de contêiner do Amazon ECS e do Docker.

AMI do Windows Server 2016 Full otimizada para Amazon ECS	Versão do agente de contêiner do Amazon ECS	Versão do Docker	Visibilidade
Windows_Server-2016-English-Full-ECS_Optimized-2025.06.13	`1.94.0`	`20.10.23 (Docker CE)`	Public
Windows_Server-2016-English-Full-ECS_Optimized-2025.05.17	`1.93.1`	`20.10.23 (Docker CE)`	Public
Windows_Server-2016-English-Full-ECS	`1.91.2`	`20.10.23 (Docker CE)`	Public

AMI do Windows Server 2016 Full otimizada para Amazon ECS	Versão do agente de contêiner do Amazon ECS	Versão do Docker	Visibilidade
_Optimized-2025.4.14			
Windows_Server-2016-English-Full-ECS_Optimized-2025.3.17	1.91.0	20.10.23 (Docker CE)	Public
Windows_Server-2016-English-Full-ECS_Optimized-2025.2.14	1.90.0	20.10.23 (Docker CE)	Public

Use a seguinte AWS CLI AMI do Windows Server 2016 Full otimizada para Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2016-English-Full-ECS_Optimized
```

Criar sua própria AMI do Windows otimizada para Amazon ECS

Use o EC2 Image Builder para criar sua própria AMI personalizada do Windows otimizada para o Amazon ECS. Isso facilita o uso de uma AMI do Windows com sua própria licença no Amazon ECS. O Amazon ECS fornece um componente gerenciado do Image Builder que fornece a configuração do sistema necessária para executar instâncias do Windows para hospedar os contêineres. Cada componente gerenciado do Amazon ECS inclui um agente de contêiner específico e uma versão do Docker. É possível personalizar a imagem para usar o componente gerenciado do Amazon ECS mais recente ou, se for necessário um agente de contêiner ou uma versão do Docker mais antiga, poderá especificar um componente diferente.

Para obter uma demonstração completa do uso do EC2 Image Builder, consulte Conceitos básicos do EC2 Image Builder no Guia do usuário do EC2 Image Builder.

Ao criar sua própria AMI do Windows otimizada para Amazon ECS usando o EC2 Image Builder, você cria uma receita de imagem. Sua receita de imagem deve atender aos seguintes requisitos:

- A imagem de origem deve ser baseada no Windows Server 2019 Core, no Windows Server 2019 Full, no Windows Server 2022 Core ou no Windows Server 2022 Full. Nenhum outro sistema operacional Windows tem suporte. E, nesse caso, pode haver incompatibilidade com o componente.
- Ao especificar os Componentes de criação, o componente `ecs-optimized-ami-windows` é obrigatório. O componente `update-windows` é recomendado, o que garante que a imagem contenha as atualizações de segurança mais recentes.

  Para especificar uma versão de componente diferente, expanda o menu Versioning options (Opções de versionamento) e especifique a versão do componente que quer usar. Para obter mais informações, consulte Listar as versões do componente `ecs-optimized-ami-windows`.

Listar as versões do componente **`ecs-optimized-ami-windows`**

Ao criar uma receita do EC2 Image Builder e especificar o componente `ecs-optimized-ami-windows`, você pode usar a opção padrão ou especificar uma versão do componente específica. Para determinar quais versões do componente estão disponíveis, juntamente com o agente de contêiner do Amazon ECS e as versões do Docker contidas no componente, use o AWS Management Console.

Para listar as versões do componente **`ecs-optimized-ami-windows`** disponíveis

1. Abra o console do EC2 Image Builder em https://console.aws.amazon.com/imagebuilder/.
2. Na barra de navegação, selecione a Região em que a imagem está sendo criada.
3. No painel de navegação, no menu Saved configurations (Configurações salvas), escolha Components (Componentes).
4. Na página Components (Componentes), na barra de pesquisa digite `ecs-optimized-ami-windows`, abra o menu de qualificação e selecione Quick start (Amazon-managed) (Início rápido [gerenciado pela Amazon]) .
5. Use a coluna Description (Descrição) para determinar a versão do componente com o agente de contêiner do Amazon ECS e a versão do Docker de que sua imagem precisa.

Gerenciamento de instâncias de contêiner do Windows do Amazon ECS

Ao usar as instâncias do EC2 para workloads do Amazon ECS, você é responsável pela manutenção das instâncias.

As atualizações de agente não se aplicam a instâncias de contêiner do Windows. É recomendável executar novas instâncias de contêiner para atualizar a versão do agente nos clusters do Windows.

Procedimentos de gerenciamento

- Iniciar uma instância de contêiner do Windows do Amazon ECS
- Inicialização de instâncias de contêiner do Windows no Amazon ECS para transmitir dados
- Uso de um proxy HTTP para instâncias de contêiner do Windows no Amazon ECS
- Configuração de instâncias de contêiner do Windows no Amazon ECS para receber avisos de instância spot

Iniciar uma instância de contêiner do Windows do Amazon ECS

As instâncias de contêiner do Amazon ECS são criadas por meio do console do Amazon EC2. Antes de começar, é necessário concluir as etapas em Configuração para usar o Amazon ECS.

Para obter mais informações sobre o assistente de inicialização, consulte Iniciar uma instância usando o novo assistente de inicialização de instância no Manual do usuário do Amazon EC2.

É possível usar o assistente do Amazon EC2 para iniciar uma instância. É possível usar a seguinte lista para os parâmetros e deixar os parâmetros não listados como padrão. As instruções a seguir orientam você por cada grupo de parâmetros.

Procedimento

Antes de começar, conclua as etapas em Configuração para usar o Amazon ECS.

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. Na barra de navegação na parte superior da tela, a região atual AWS é exibida [por exemplo, Leste dos EUA (Ohio)]. Selecione uma região na qual a instância será iniciada. Essa escolha é importante, pois alguns recursos do Amazon EC2 podem ser compartilhados entre regiões, enquanto outros não podem.
3. No painel do console do Amazon EC2, selecione Launch instance (Executar instância).

Nome e tags

O nome da instância é uma tag em que a chave é Name (Nome) e o valor é o nome que você especificar. É possível aplicar tags na instância, sos volumes e nos elementos gráficos elásticos. Para instâncias spot, é possível marcar apenas a solicitação de instância spot.

A especificação de um nome de instância e de tags adicionais é opcional.

- Em Name (Nome), insira um nome descritivo para a instância. Se você não especificar um nome, a instância poderá ser identificada por seu ID, que é gerado automaticamente quando você inicia a instância.
- Para adicionar mais tags, selecione Add additional tag (Adicionar outra tag). Escolha Add tag (Adicionar tag), insira uma chave e um valor, e selecione o tipo de recurso a aplicar a tag. Escolha Add tag (Adicionar tag) para cada tag adicional a acrescentar.

Imagens de aplicações e sistemas operacionais (imagem de máquina da Amazon)

Uma imagem de máquina da Amazon (AMI) contém as informações necessárias para criar uma instância. Por exemplo, uma AMI pode conter o software necessário para atuar como servidor Web, por exemplo, Apache e seu site.

Para obter as mais recentes AMIs otimizadas para o Amazon ECS e seus valores, consulte AMI do Windows otimizada para o Amazon ECS.

Use a barra Search (Pesquisa) para encontrar uma AMI otimizada para o Amazon ECS publicada pela AWS.

1. Com base em seus requisitos, insira uma das AMIs a seguir na barra Search (Pesquisa) e pressione Enter.
   - Windows_Server-2022-English-Full-ECS_Optimized
   - Windows_Server-2022-English-Core-ECS_Optimized
   - Windows_Server-2019-English-Full-ECS_Optimized
   - Windows_Server-2019-English-Core-ECS_Optimized
   - Windows_Server-2016-English-Full-ECS_Optimized
2. Na página Choose an Amazon Machine Image (AMI) (Escolher uma imagem de máquina da Amazon [AMI]), selecione a guia Community AMIs (AMIs da comunidade).
3. Na lista exibida, escolha uma AMI verificada pela Microsoft com a data de publicação mais recente e clique em Select (Selecionar).

Tipo de instância

O tipo de instância define a configuração do hardware e o tamanho da instância. Os tipos de instâncias maiores têm mais CPU e memória. Para obter mais informações, consulte Tipos de instância.

- Em Instance type (Tipo de instância), selecione o tipo de instância da instância.

  O tipo de instância que você selecionar determinará os recursos disponíveis para execução de suas tarefas.

Par de chaves (login)

Em Key pair name (Nome do par de chaves), escolha um par de chaves existente ou escolha Create new key pair (Criar um novo par de chaves) para criar um novo.

> **Important**
>
> Se você escolher a opção Proceed without key pair (Not recommended) (Prosseguir sem par de chaves, não recomendado), não conseguirá se conectar à instância a menos que escolha uma AMI configurada para permitir que os usuários façam login de outro modo.

Configurações de rede

Defina as configurações de rede, conforme necessário.

- Networking platform (Plataforma de rede): escolha Virtual Private Cloud (VPC) (Nuvem privada virtual - VPC) e especifique a sub-rede na seção Network interfaces (Interfaces de rede).
- VPC: selecione uma VPC existente na qual criar o grupo de segurança.
- Subnet (Sub-rede): é possível executar uma instância em uma sub-rede associada a uma zona de disponibilidade, a uma zona local, a uma zona Wavelength ou a um Outpost.

  Para iniciar a instância em uma zona de disponibilidade, selecione a sub-rede na qual a instância será iniciada. Para criar uma nova sub-rede, escolha Create new subnet (Criar nova sub-rede) para acessar o console da Amazon VPC. Quando tiver concluído, retorne ao assistente de inicialização da instância e escolha o ícone Refresh (Atualizar) para carregar sua sub-rede na lista.

  Para iniciar a instância em uma zona local, selecione uma sub-rede que você criou na zona local.

Para iniciar uma instância em um Outpost, selecione uma sub-rede em uma VPC associada ao Outpost.

- Auto-assign Public IP (Atribuir IP público automaticamente): se sua instância tiver de ser acessada pela Internet pública, verifique se o campo Auto-assign Public IP (Atribuir IP público automaticamente) está definido como Enable (Habilitar). Em caso negativo, defina esse campo como Disable (Desabilitar).

> **Note**
>
> Instâncias de contêiner precisam de acesso para se comunicar com o endpoint do serviço do Amazon ECS. Isso pode ser feito por meio de uma interface do endpoint da VPC ou por meio dos recursos de computação das instâncias de contêiner que tenham endereços IP públicos.
> Para saber mais sobre endpoints da VPC de interface, consulte Endpoints da VPC de interface do Amazon ECS (AWS PrivateLink).
> Se você não tiver um endpoint da VPC de interface configurado e suas instâncias de contêiner não tiverem endereços IP públicos, elas deverão usar a conversão de endereço de rede (NAT) para fornecer esse acesso. Para obter mais informações, consulte Gateways NAT no Guia do usuário da Amazon VPC e Uso de um proxy HTTP para instâncias de contêiner do Linux no Amazon ECS neste guia.

- Firewall (security groups) (Firewall, grupos de segurança): use um grupo de segurança para definir regras de firewall da sua instância de contêiner. Essas regras especificam qual tráfego de rede de entrada será fornecido para sua instância de contêiner. Todo o tráfego é ignorado.
  - Para selecionar um grupo de segurança existente, escolha Select an existing security group (Selecionar um grupo de segurança existente) e escolha o grupo de segurança criado em Configuração para usar o Amazon ECS

Configurar armazenamento

A AMI que você selecionou inclui um ou mais volumes de armazenamento, incluindo o volume raiz. É possível especificar volumes adicionais a serem anexados à instância.

É possível usar a exibição Simple (Simples).

- Storage type (Tipo de armazenamento): configure o armazenamento para a sua instância de contêiner.

Se você estiver usando o Amazon Linux AMI otimizado para Amazon ECS, a instância terá dois volumes configurados. O volume Raiz é para uso do sistema operacional e o segundo volume do Amazon EBS (anexado a `/dev/xvdcz`) é para uso do Docker.

Você também pode aumentar ou diminuir os tamanhos de volumes para a sua instância para atender às necessidades dos seus aplicativos.

Detalhes avançados

Em Advanced details (Detalhes avançados), expanda a seção para visualizar os campos e especifique quaisquer parâmetros adicionais para a instância.

- Purchasing option (Opção de compra): escolha Request Spot instances (Solicitar instâncias Spot) para solicitar instâncias Spot. Também é necessário definir os outros campos relacionados a instâncias spot. Para obter mais informações, consulte Solicitações de instâncias spot.

  > **Note**
  >
  > Se você estiver usando instâncias spot e vir uma mensagem `Not available`, pode ser necessário escolher um tipo de instância diferente.

  .

- IAM instance profile (Perfil de instância do IAM): selecione a função do IAM de instância de contêiner. Isso geralmente é chamado de `ecsInstanceRole`.

  > **Important**
  >
  > Se você não iniciar a instância de contêiner com as permissões apropriadas do IAM, o agente do Amazon ECS não poderá se conectar ao cluster. Para obter mais informações, consulte Função do IAM de instância de contêiner do Amazon ECS.

- (Opcional) User data (Dados do usuário): configure a instância de contêiner do Amazon ECS com os dados do usuário, como as variáveis de ambiente de agente de Configuração do agente de contêiner do Amazon ECS. Os scripts de dados do usuário do Amazon EC2 são executados apenas uma vez, quando a instância é iniciada pela primeira vez. Veja a seguir exemplos comuns de quais dados do usuário são usados.

- Por padrão, sua instância de contêiner é executada em seu cluster padrão. Para executar em um cluster que não seja padrão, escolha a lista Detalhes avançados. Em seguida, cole o seguinte script no campo Dados do usuário substituindo *your_cluster_name* pelo nome do seu cluster.

  O `EnableTaskIAMRole` ativa o recurso de funções de tarefas do IAM para as tarefas.

  Além disso, as seguintes opções estão disponíveis quando for usado o modo de rede `awsvpc`.

  - `EnableTaskENI`:este sinalizador ativa a rede de tarefas e é necessário quando é usado o modo de rede `awsvpc`.
  - `AwsvpcBlockIMDS`: este sinalizador opcional bloqueia o acesso IMDS para os contentores de tarefas em execução no modo de rede `awsvpc`.
  - `AwsvpcAdditionalLocalRoutes`: este sinalizador opcional permite que você tenha rotas adicionais no namespace da tarefa.

    Substitua `ip-address` pelo o endereço IP das rotas adicionais, por exemplo, 172.31.42.23/32.

```
<powershell>
Import-Module ECSTools
Initialize-ECSAgent -Cluster your_cluster_name -EnableTaskIAMRole -EnableTaskENI -
AwsvpcBlockIMDS -AwsvpcAdditionalLocalRoutes
'["ip-address"]'
</powershell>
```

Inicialização de instâncias de contêiner do Windows no Amazon ECS para transmitir dados

Ao iniciar uma instância do Amazon EC2, é possível transmitir dados do usuário para a instância do EC2. Os dados podem ser usados para realizar tarefas de configuração automatizadas em comum e até mesmo executar scripts na inicialização da instância. Para o Amazon ECS, os casos de uso mais comuns para dados de usuário devem transmitir informações de configuração para o daemon do Docker e o agente de contêiner do Amazon ECS.

É possível transmitir vários tipos de dados de usuário para o Amazon EC2, inclusive boothooks de nuvem, scripts de shell e diretivas `cloud-init`. Para obter mais informações sobre esses e outros tipos de formato, consulte a documentação Cloud-Init.

É possível transmitir esses dados do usuário ao usar o assistente de inicialização do Amazon EC2. Para obter mais informações, consulte Iniciar uma instância de contêiner do Linux do Amazon ECS.

Dados dos usuário do Windows padrão

Este exemplo de script de dados do usuário mostra os dados do usuário padrão que as instâncias de contêiner do Windows receberão se você usar o console. O script abaixo faz o seguinte:

- Define o nome do cluster para o nome que você inseriu.
- Define as funções do IAM para tarefas.
- Define `json-file` e `awslogs` como os drivers de registro em log disponíveis.

Além disso, as seguintes opções estão disponíveis quando for usado o modo de rede `awsvpc`.

- `EnableTaskENI`:este sinalizador ativa a rede de tarefas e é necessário quando é usado o modo de rede `awsvpc`.
- `AwsvpcBlockIMDS`: este sinalizador opcional bloqueia o acesso IMDS para os contêineres de tarefas em execução no modo de rede `awsvpc`.
- `AwsvpcAdditionalLocalRoutes`: este sinalizador opcional permite que você tenha rotas adicionais.

  Substitua `ip-address` pelo o endereço IP das rotas adicionais, por exemplo, 172.31.42.23/32.

É possível usar esse script nas próprias instâncias de contêiner (desde que elas sejam iniciadas em AMIs Windows Server otimizadas para o Amazon ECS).

Substitua a linha `-Cluster` *`cluster-name`* para especificar o nome do seu próprio cluster.

```
<powershell>
Initialize-ECSAgent -Cluster cluster-name -EnableTaskIAMRole -LoggingDrivers '["json-
file","awslogs"]' -EnableTaskENI -AwsvpcBlockIMDS -AwsvpcAdditionalLocalRoutes
'["ip-address"]'
</powershell>
```

Para tarefas do Windows configuradas para usar o driver de log `awslogs`, também é necessário definir a variável de ambiente `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE` na instância de contêiner. Use a sintaxe a seguir.

Substitua a linha `-Cluster` *`cluster-name`* para especificar o nome do seu próprio cluster.

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE",
 $TRUE, "Machine")
Initialize-ECSAgent -Cluster cluster-name -EnableTaskIAMRole -LoggingDrivers '["json-
file","awslogs"]'
</powershell>
```

Dados do usuário da instalação do agente do Windows

Este exemplo de script de dados do usuário instala o agente de contêiner do Amazon ECS em uma instância iniciada com uma AMI Windows_Server-2016-English-Full-Containers. Ele foi adaptado com base nas instruções de instalação do agente na página README Repositório do GitHub do Amazon ECS Container Agent.

> **Note**
>
> Esse script é compartilhado com finalidades de exemplo. É muito mais fácil começar a usar os contêineres do Windows usando a AMI do Windows Server otimizada para Amazon ECS. Para obter mais informações, consulte Criação de um cluster do Amazon ECS para o tipo de inicialização do Fargate.

Para obter informações sobre como instalar o agente do Amazon ECS no Windows Server 2022 Full, consulte o problema 3753 no GitHub.

É possível usar esse script para suas próprias instâncias de contêiner (desde que elas sejam executadas com uma versão da AMI Windows_Server-2016-English-Full-Contêineres). Não se esqueça de substituir a linha *windows* para especificar o nome de seu próprio cluster (caso não esteja usando um cluster chamado `windows`).

```
<powershell>
# Set up directories the agent uses
New-Item -Type directory -Path ${env:ProgramFiles}\Amazon\ECS -Force
New-Item -Type directory -Path ${env:ProgramData}\Amazon\ECS -Force
New-Item -Type directory -Path ${env:ProgramData}\Amazon\ECS\data -Force
# Set up configuration
$ecsExeDir = "${env:ProgramFiles}\Amazon\ECS"
[Environment]::SetEnvironmentVariable("ECS_CLUSTER", "windows", "Machine")
[Environment]::SetEnvironmentVariable("ECS_LOGFILE", "${env:ProgramData}\Amazon\ECS\log
\ecs-agent.log", "Machine")
```

```
[Environment]::SetEnvironmentVariable("ECS_DATADIR", "${env:ProgramData}\Amazon\ECS
\data", "Machine")
# Download the agent
$agentVersion = "latest"
$agentZipUri = "https://s3.amazonaws.com/amazon-ecs-agent/ecs-agent-windows-
$agentVersion.zip"
$zipFile = "${env:TEMP}\ecs-agent.zip"
Invoke-RestMethod -OutFile $zipFile -Uri $agentZipUri
# Put the executables in the executable directory.
Expand-Archive -Path $zipFile -DestinationPath $ecsExeDir -Force
Set-Location ${ecsExeDir}
# Set $EnableTaskIAMRoles to $true to enable task IAM roles
# Note that enabling IAM roles will make port 80 unavailable for tasks.
[bool]$EnableTaskIAMRoles = $false
if (${EnableTaskIAMRoles}) {
  $HostSetupScript = Invoke-WebRequest https://raw.githubusercontent.com/aws/amazon-
ecs-agent/master/misc/windows-deploy/hostsetup.ps1
  Invoke-Expression $($HostSetupScript.Content)
}
# Install the agent service
New-Service -Name "AmazonECS" `
        -BinaryPathName "$ecsExeDir\amazon-ecs-agent.exe -windows-service" `
        -DisplayName "Amazon ECS" `
        -Description "Amazon ECS service runs the Amazon ECS agent" `
        -DependsOn Docker `
        -StartupType Manual
sc.exe failure AmazonECS reset=300 actions=restart/5000/restart/30000/restart/60000
sc.exe failureflag AmazonECS 1
Start-Service AmazonECS
</powershell>
```

Uso de um proxy HTTP para instâncias de contêiner do Windows no Amazon ECS

É possível configurar as instâncias de contêiner do Amazon ECS para usar um proxy HTTP para o agente de contêiner do Amazon ECS e o daemon do Docker. Isso é útil se suas as instâncias de contêiner não têm acesso à rede externa através de um gateway da Internet da Amazon VPC, um gateway NAT ou uma instância.

Para configurar sua instância de contêiner do Windows do Amazon ECS para usar um proxy HTTP, defina as variáveis a seguir no momento da inicialização (com dados de usuário do Amazon EC2).

```
[Environment]::SetEnvironmentVariable("HTTP_PROXY",
"http://proxy.mydomain:port", "Machine")
```

Defina HTTP_PROXY como o nome do host (ou endereço IP) e o número de porta de um proxy HTTP a serem usados pelo agente do Amazon ECS para se conectar à Internet. Por exemplo, as instâncias de contêiner podem não ter acesso à rede externa através de um gateway da Internet da Amazon VPC, um gateway NAT ou uma instância.

```
[Environment]::SetEnvironmentVariable("NO_PROXY",
"169.254.169.254,169.254.170.2,\\.\pipe\docker_engine", "Machine")
```

Defina NO_PROXY como `169.254.169.254,169.254.170.2,\\.\pipe\docker_engine` para filtrar os metadados da instância do EC2, as funções do IAM para tarefas e o tráfego do daemon do Docker proveniente do proxy.

Example Script de dados do usuário do proxy HTTP do Windows

O exemplo de script PowerShell de dados de usuário abaixo configura o agente de contêiner do Amazon ECS e o daemon do Docker para usar um proxy HTTP especificado por você. Você também pode especificar um cluster no qual a instância de contêiner será registrada.

Para usar esse script ao executar uma instância de contêiner, siga as etapas em the section called "Iniciar uma instância de contêiner". Basta copiar e colar o script do PowerShell abaixo no campo Dados do usuário (substitua os valores de exemplo vermelhos pelas suas informações de proxy e cluster).

> **Note**
>
> A opção `-EnableTaskIAMRole` é necessária para habilitar funções do IAM para tarefas. Para obter mais informações, consulte Configuração adicional de instância do Windows no Amazon EC2.

```
<powershell>
Import-Module ECSTools

$proxy = "http://proxy.mydomain:port"
[Environment]::SetEnvironmentVariable("HTTP_PROXY", $proxy, "Machine")
[Environment]::SetEnvironmentVariable("NO_PROXY", "169.254.169.254,169.254.170.2,\\.
\pipe\docker_engine", "Machine")
```

```
Restart-Service Docker
Initialize-ECSAgent -Cluster MyCluster -EnableTaskIAMRole
</powershell>
```

Configuração de instâncias de contêiner do Windows no Amazon ECS para receber avisos de instância spot

O Amazon EC2 encerra, interrompe ou coloca a instância spot em hibernação quando o preço spot excede o preço máximo da solicitação ou a capacidade não está mais disponível. O Amazon EC2 fornece um aviso de interrupção da instância spot, enviando à instância um aviso de dois minutos antes que ela seja interrompida. Se a drenagem da instância spot do Amazon ECS estiver habilitada na instância, o ECS receberá o aviso de interrupção da instância spot e colocará a instância no status `DRAINING`.

> **Important**
>
> O Amazon ECS monitora os avisos de interrupção da instância spot que têm as ações de instância `terminate` e `stop`. Se você especificou o comportamento de interrupção `hibernate` da instância ao solicitar as instâncias spot ou a frota spot, a drenagem de instâncias spot do Amazon ECS não é compatível com essas instâncias.

Quando uma instância de contêiner é definida como `DRAINING`, o Amazon ECS impede que novas tarefas sejam programadas para posicionamento na instância de contêiner. As tarefas de serviço nas instâncias de contêiner de drenagem que estão com o status de `PENDING` são interrompidas imediatamente. Se houver instâncias de contêiner no cluster disponíveis, as tarefas de serviço de substituição serão iniciadas nelas.

É possível ativar a drenagem de instância spot ao iniciar uma instância. Você deve definir o parâmetro `ECS_ENABLE_SPOT_INSTANCE_DRAINING` antes de iniciar o agente de contêiner. Substitua *`my-cluster`* pelo nome do cluster.

```
[Environment]::SetEnvironmentVariable("ECS_ENABLE_SPOT_INSTANCE_DRAINING", "true",
 "Machine")

# Initialize the agent
Initialize-ECSAgent -Cluster my-cluster
```

Para obter mais informações, consulte the section called “Iniciar uma instância de contêiner”.

Clusters do Amazon ECS para o tipo de inicialização externa

O Amazon ECS Anywhere fornece suporte para registrar uma Instância externa, como um servidor on-premises ou uma máquina virtual (VM), no cluster do Amazon ECS. As instâncias externas são otimizadas para executar aplicações que geram tráfego de saída ou dados de processo. Se sua aplicação exigir tráfego de entrada, a falta de suporte do Elastic Load Balancing torna a execução dessas workloads menos eficiente. O Amazon ECS adicionou um novo tipo de inicialização `EXTERNAL` que você pode usar para criar serviços ou executar tarefas nas instâncias externas.

Sistemas operacionais e arquiteturas de sistema compatíveis

Veja a seguir a lista de sistemas operacionais e arquiteturas de sistemas compatíveis.

- Amazon Linux 2
- Amazon Linux 2023
- CentOS Stream 9
- RHEL 7, RHEL 8, RHEL 9: os repositórios de pacotes abertos do Docker e do RHEL não são compatíveis com a instalação nativa do Docker no RHEL. É preciso garantir que o Docker esteja instalado antes da execução do script de instalação descrito neste documento.
- Fedora 32, Fedora 33, Fedora 40
- openSUSE Tumbleweed
- Ubuntu 18, Ubuntu 20, Ubuntu 22, Ubuntu 24
- Debian 10

> **Important**
>
> O suporte a longo prazo do Debian 9 (suporte a LTS) terminou em 30 de junho de 2022 e não há mais suporte pelo Amazon ECS Anywhere.

- Debian 11
- Debian 12
- SUSE Enterprise Server 15
- As arquiteturas de CPU `x86_64` e `ARM64` são compatíveis.
- Há suporte para as seguintes versões do sistema operacional Windows:
  - Windows Server 2022

- Windows Server 2019
- Windows Server 2016
- Windows Server 20H2

Considerações

Antes de começar a usar instâncias externas, esteja ciente das seguintes considerações.

- É possível registrar uma instância externa em um cluster de cada vez. Para obter instruções sobre como registrar uma instância externa em um cluster diferente, consulte Cancelamento do registro de uma instância externa do Amazon ECS.
- As instâncias externas exigem uma função do IAM que permita que elas se comuniquem com APIs da AWS. Para obter mais informações, consulte Perfil do IAM para o Amazon ECS Anywhere.
- As instâncias externas não devem ter uma cadeia de credenciais de instânciaa pré-configurada definida localmente, pois isso interferirá com o script de registro.
- Para enviar logs de contêiner para o CloudWatch Logs, crie e especifique uma função do IAM de execução de tarefa na definição de tarefa.
- Quando uma instância externa é registrada em um cluster, o atributo `ecs.capability.external` é associado à instância. Esse atributo identifica a instância como uma instância externa. É possível adicionar atributos personalizados às instâncias externas para usar como uma restrição de posicionamento de tarefas. Para obter mais informações, consulte Atributos personalizados.
- É possível adicionar etiquetas de recurso à instância externa. Para obter mais informações, consulte Instâncias de contêiner externas.
- Há suporte para o ECS Exec em instâncias externas. Para obter mais informações, consulte Monitoramento de contêineres do Amazon ECS com o ECS Exec.
- Veja a seguir considerações adicionais que são específicas para as redes com instâncias externas. Para obter mais informações, consulte Redes .
  - O balanceamento de carga do serviço não é compatível.
  - A descoberta de serviço não é compatível.
  - Tarefas executadas em instâncias externas devem usar os modos de rede `bridge`, `host` ou `none`. O modo de rede `awsvpc` não é compatível.
  - Há domínios de serviço do Amazon ECS em cada região da AWS. Esses domínios de serviço devem ter permissão para enviar tráfego para as instâncias externas.

  - O SSM Agent instalado na instância externa mantém as credenciais do IAM que são alternadas a cada 30 minutos com o uso de uma impressão digital de hardware. Se a instância externa perder conexão com a AWS, o SSM Agent atualiza automaticamente as credenciais após a conexão ser restabelecida. Para obter mais informações, consulte Validar servidores on-premises e máquinas virtuais usando uma impressão digital de hardware no Guia do usuário do AWS Systems Manager.
- A API `UpdateContainerAgent` não é compatível. Para obter instruções sobre como atualizar o SSM Agent ou o agente do Amazon ECS nas instâncias externas, consulte Atualização do agente do AWS Systems Manager e o agente de contêiner do Amazon ECS em uma instância externa.
- Provedores de capacidade do Amazon ECS não são compatíveis. Para criar um serviço ou executar uma tarefa autônoma nas instâncias externas, use o tipo de inicialização `EXTERNAL`.
- SELinux não é compatível.
- Usar volumes do Amazon EFS ou especificar um `EFSVolumeConfiguration` não é compatível.
- A integração com o App Mesh não é compatível.
- Se você usar o console para criar uma definição de tarefa de instância externa, deverá criar a definição da tarefa com o editor JSON do console.
- Quando você executa o ECS Anywhere no Windows, você deve usar sua própria licença do Windows na infraestrutura on-premises.
- Ao usar uma AMI não otimizada para o Amazon ECS, execute os comandos a seguir na instância de contêiner externa para configurar regras para usar perfis do IAM em tarefas. Para obter mais informações, consulte Configuração adicional para instância externa.

```
$ sysctl -w net.ipv4.conf.all.route_localnet=1
$ iptables -t nat -A PREROUTING -p tcp -d 169.254.170.2 --dport 80 -j DNAT --to-
destination 127.0.0.1:51679
$ iptables -t nat -A OUTPUT -d 169.254.170.2 -p tcp -m tcp --dport 80 -j REDIRECT --
to-ports 51679
```

Redes

As instâncias externas do Amazon ECS são otimizadas para executar aplicações que geram tráfego de saída ou dados de processo. Se sua aplicação exigir tráfego de entrada, como um serviço da Web, a falta de suporte do Elastic Load Balancing torna a execução dessas workloads menos eficiente porque não há suporte para colocar essas workloads atrás de um balanceador de carga.

Veja a seguir considerações adicionais que são específicas para as redes com instâncias externas.

- O balanceamento de carga do serviço não é compatível.
- A descoberta de serviço não é compatível.
- Tarefas do Linux executadas em instâncias externas devem usar os modos de rede `bridge`, `host` ou `none`. O modo de rede `awsvpc` não é compatível.

  Para obter mais informações sobre cada modo de rede, consulte Opções de rede de tarefas do Amazon ECS para o tipo de lançamento do EC2.
- Tarefas do Windows executadas em instâncias externas devem usar o modo de rede `default`.
- Há domínios de serviço do Amazon ECS em cada região e você deve ter permissão para enviar tráfego para as instâncias externas.
- O SSM Agent instalado na instância externa mantém as credenciais do IAM que são alternadas a cada 30 minutos com o uso de uma impressão digital de hardware. Se a instância externa perder conexão com a AWS, o SSM Agent atualiza automaticamente as credenciais após a conexão ser restabelecida. Para obter mais informações, consulte Validar servidores on-premises e máquinas virtuais usando uma impressão digital de hardware no Guia do usuário do AWS Systems Manager.

Os domínios a seguir são usados para comunicação entre o serviço do Amazon ECS e o agente do Amazon ECS instalado na instância externa. Certifique-se de que o tráfego seja permitido e que a resolução DNS funcione. Para cada endpoint, *`região`* representa o identificador da região da AWS com suporte do Amazon ECS, como `us-east-2`, para a região Leste dos EUA (Ohio). Os endpoints de todas as regiões que você usa devem ser permitidos. Para os endpoints `ecs-a` e `ecs-t`, você deve incluir um asterisco (por exemplo, `ecs-a-*`).

- `ecs-a-*.`*`region`*`.amazonaws.com`: este endpoint é usado quando são gerenciadas tarefas.
- `ecs-t-*.`*`region`*`.amazonaws.com`: este endpoint é usado para gerenciar métricas de tarefas e contêineres.
- `ecs.`*`region`*`.amazonaws.com`: este é um endpoint de serviço do Amazon ECS.
- `ssm.`*`region`*`.amazonaws.com` : este é um endpoint de serviço para o AWS Systems Manager.
- `ec2messages.`*`region`*`.amazonaws.com`: este é o endpoint de serviço que o AWS Systems Manager usa para se comunicar entre o agente Systems Manager e o serviço Systems Manager na nuvem.
- `ssmmessages.`*`region`*`.amazonaws.com`: este é o endpoint de serviço necessário para criar e excluir canais de sessão com o serviço Session Manager na nuvem.

- Se as tarefas requerem comunicação com quaisquer outros serviços da AWS, certifique-se de que esses endpoints sejam permitidos. Exemplos da aplicações incluem o uso do Amazon ECR para extrair imagens de contêiner ou o uso do CloudWatch para CloudWatch Logs. Para obter mais informações, consulte Endpoints de serviço na Referência geral da AWS.

Amazon FSx for Windows File Server com ECS Anywhere

Para usar o Amazon FSx for Windows File Server com instâncias externas do Amazon ECS, você deve estabelecer uma conexão do entre seu data center on-premises e o Nuvem AWS. Para obter mais informações sobre as opções para conexão de sua rede a sua VPC, consulte Opções de conectividade do Amazon Virtual Private Cloud.

gMSA com ECS Anywhere

Há suporte para os casos de uso a seguir no ECS Anywhere.

- O Active Directory está na Nuvem AWS: para essa configuração, você cria uma conexão da entre sua rede on-premises e a Nuvem AWS usando uma conexão AWS Direct Connect. Para obter informações sobre como criar a conexão, consulte Opções de conectividade da Amazon Virtual Private Cloud. Você cria um Active Directory na Nuvem AWS. Para obter informações sobre como começar a usar o AWS Directory Service, consulte Configuração do AWS Directory Service no Guia de administração do AWS Directory Service. Em seguida, você pode associar suas instâncias externas ao domínio usando a conexão AWS Direct Connect. Para obter informações sobre como trabalhar com o gMSA com o Amazon ECS, consulte the section called “Saiba como usar gMSAs para contêineres do Windows do EC2”.
- O Active Directory está no data center on-premises. - Para essa configuração, você une suas instâncias externas ao Active Directory on-premises. Em seguida, você usa as credenciais disponíveis localmente ao executar as tarefas do Amazon ECS.

Criação de um cluster do Amazon ECS para o tipo de inicialização externa

Crie um cluster para definir a infraestrutura na qual suas tarefas e serviços são executados.

Antes de começar, é necessário concluir as etapas em Configuração para usar o Amazon ECS e atribuir a permissão adequada do IAM. Para obter mais informações, consulte the section called “Exemplos de clusters do Amazon ECS”. O console do Amazon ECS fornece uma maneira simples de criar os recursos necessários para um cluster do Amazon ECS criando uma pilha do AWS CloudFormation.

Para tornar o processo de criação do cluster o mais fácil possível, o console tem seleções padrão para muitas opções, que descrevemos abaixo. Também há painéis de ajuda disponíveis para a maioria das seções do console, que fornecem mais contexto.

É possível modificar as opções a seguir:

- Adicione um namespace ao cluster.

  Um namespace permite que os serviços que você cria no cluster possam se conectar a outros serviços no namespace sem configuração adicional. Para obter mais informações, consulte Interconexão de serviços do Amazon ECS.
- Configurar o cluster para instâncias externas
- Ative o Container Insights ou o Container Insights com observabilidade aprimorada.

  O CloudWatch Container Insights coleta, agrega e resume métricas e logs das suas aplicações e microsserviços conteinerizados. O Container Insights também fornece informações de diagnóstico, como falhas de reinicialização de contêiner, que você usa para isolar problemas e resolvê-los rapidamente. Para obter mais informações, consulte the section called “Monitorar contêineres do Amazon ECS utilizando o Container Insights com observabilidade aprimorada”.

  Em 2 de dezembro de 2024, a AWS lançou o Container Insights com observabilidade aprimorada para o Amazon ECS. Essa versão é compatível com a observabilidade aprimorada dos clusters do Amazon ECS usando os tipos de inicialização do Amazon EC2 e do Fargate. Depois de configurar o Container Insights com observabilidade melhorada no Amazon ECS, o Container Insights recolhe automaticamente a telemetria detalhada da infraestrutura desde o nível do cluster até ao nível do contentor no seu ambiente e apresenta os dados em painéis que mostram uma variedade de métricas e dimensões. Em seguida, é possível utilizar esses painéis prontos para uso no console do Container Insights para compreender melhor a integridade e o desempenho do contêiner e atenuar problemas com mais rapidez, identificando anomalias.

  Recomendamos que você use o Container Insights com observabilidade aprimorada em vez do Container Insights, pois ele fornece visibilidade detalhada no ambiente de contêiner, reduzindo o tempo médio de resolução.
- Atribua uma chave do AWS KMS ao seu armazenamento gerenciado. Para obter informações sobre como criar uma chave, consulte Criar uma chave do KMS no Guia do usuário do AWS Key Management Service.
- Adicione tags para ajudar a identificar seu cluster.

Para criar um novo cluster (console do Amazon ECS)

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, selecione a Região a ser usada.
3. No painel de navegação, escolha Clusters.
4. Na página Clusters, escolha Create Cluster (Criar cluster).
5. Em Configuração de cluster, configure o seguinte:

   - Em Nome do cluster, insira um nome exclusivo.

     O nome pode conter até 255 letras (minúsculas e maiúsculas), números e hifens.
   - (Opcional) Para que o namespace usado para o Service Connect seja diferente do nome do cluster, em Namespace, insira um nome exclusivo.
6. (Opcional) Utilize o Container Insights, expanda Monitoramento e escolha uma destas opções:

   - Para usar o Container Insights com observabilidade aprimorada (recomendado), escolha Container Insights com observabilidade aprimorada.
   - Para usar o Container Insights, escolha Container Insights.
7. (Opcional) Criptografe os dados no armazenamento gerenciado. Em Criptografia, em Armazenamento gerenciado, insira o ARN da chave do AWS KMS que você deseja usar para criptografar os dados do armazenamento gerenciado.
8. (Opcional) Para ajudar a identificar seu cluster, expanda Tags (Etiquetas) e configure suas etiquetas.

   [Adicionar uma tag] Selecione Add tag (Adicionar tag) e faça o seguinte:

   - Em Key (Chave), insira o nome da chave.
   - Em Valor insira o valor da chave.
9. Selecione Create (Criar).

Próximas etapas

Você deve registrar as instâncias no cluster. Para obter mais informações, consulte Registro de uma instância externa para um cluster do Amazon ECS.

Crie uma definição de tarefa para o tipo de execução externa. Para obter mais informações, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

Execute suas aplicações como tarefas autônomas ou como parte de um serviço. Para obter mais informações, consulte:

- Execução de uma aplicação como uma tarefa do Amazon ECS
- Criação de uma implantação de atualização contínua do Amazon ECS

Registro de uma instância externa para um cluster do Amazon ECS

Cada instância externa registrada em um cluster do Amazon ECS deve ter o SSM Agent, o agente de contêiner do Amazon ECS e o Docker instalados. Para registrar a instância externa em um cluster do Amazon ECS, ela deve primeiro ser registrada como uma instância gerenciada do AWS Systems Manager. É possível criar o script de instalação com alguns cliques no console do Amazon ECS. O script de instalação inclui uma chave de ativação do Systems Manager e comandos para instalar cada um dos agentes necessários e o Docker. O script de instalação deve ser executado no servidor on-premises ou na VM para concluir as etapas de instalação e registro.

> **Note**
>
> Antes de registrar sua instância externa do Linux com o cluster, crie o arquivo `/etc/ecs/ecs.config` na instância externa e adicione todos os parâmetros de configuração do agente de contêiner desejados. Isso não pode ser feito depois do registro da instância externa em um cluster. Para obter mais informações, consulte Configuração do agente de contêiner do Amazon ECS.

AWS Management Console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, selecione a Região a ser usada.
3. No painel de navegação, escolha Clusters.
4. Na página Clusters, escolha o cluster no qual a instância externa será registrada.
5. Na página Cluster : ***name***, escolha a guia Infrastructure (Infraestrutura).
6. Na página Register external instances (Registrar instâncias externas), execute as etapas a seguir.

    a. Em Activation key duration (in days) (Duração da chave de ativação (em dias)), insira o número de dias durante os quais a chave de ativação permanece ativa. Depois da

decorrência número de dias que você inseriu, a chave não funcionará mais ao ser registrada uma instância externa.

b. Em Number of instances (Número de instâncias), insira o número de instâncias externas que você quer registrar no cluster com a chave de ativação.

c. Em Instance role (Função da instância), escolha a função do IAM a ser associada às instâncias externas. Se uma função ainda não tiver sido criada, escolha Create new role (Criar nova função) para que o Amazon ECS crie uma função em seu nome. Para obter mais informações sobre quais permissões do IAM são necessárias para instâncias externas, consulte Perfil do IAM para o Amazon ECS Anywhere.

d. Copie o comando de registro. Esse comando deve ser executado em cada instância externa que você quer registrar no cluster.

> **Important**
>
> A parte bash do script deve ser executada como raiz. Se o comando não for executado como root, será retornado um erro.

e. Escolha Fechar.

AWS CLI for Linux operating systems

1. Crie um par de ativação do Systems Manager. Isso é usado para a ativação de instâncias gerenciadas do Systems Manager. A saída inclui um `ActivationId` e um `ActivationCode`. Eles serão usados em uma etapa posterior. Especifique a função do IAM do ECS Anywhere que você criou. Para obter mais informações, consulte Perfil do IAM para o Amazon ECS Anywhere.

```
aws ssm create-activation --iam-role ecsAnywhereRole | tee ssm-activation.json
```

2. No servidor on-premises ou na máquina virtual (VM), baixe o script de instalação.

```
curl --proto "https" -o "/tmp/ecs-anywhere-install.sh" "https://amazon-ecs-agent.s3.amazonaws.com/ecs-anywhere-install-latest.sh"
```

3. (Opcional) No servidor on-premises ou na máquina virtual (VM), use as etapas a seguir para verificar o script de instalação usando o arquivo de assinatura de script.

a. Baixe e instale GnuPG. Para obter mais informações sobre a GNUpg, consulte o site da GnuPG. Para sistemas Linux, instale gpg usando o gerenciador de pacotes no seu tipo de Linux.

b. Recuperar a chave pública PGP do Amazon ECS.

```
gpg --keyserver hkp://keys.gnupg.net:80 --recv BCE9D9A42D51784F
```

c. Baixe a assinatura do script de instalação. Ela consiste em uma assinatura PGP desvinculada ascii armazenada em um arquivo com a extensão `.asc`.

```
curl --proto "https" -o "/tmp/ecs-anywhere-install.sh.asc" "https://amazon-ecs-agent.s3.amazonaws.com/ecs-anywhere-install-latest.sh.asc"
```

d. Verifique o arquivo de script de instalação usando a chave.

```
gpg --verify /tmp/ecs-anywhere-install.sh.asc /tmp/ecs-anywhere-install.sh
```

A saída esperada é mostrada a seguir.

```
gpg: Signature made Tue 25 May 2021 07:16:29 PM UTC
gpg:                using RSA key 50DECCC4710E61AF
gpg: Good signature from "Amazon ECS <ecs-security@amazon.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the
 owner.
Primary key fingerprint: F34C 3DDA E729 26B0 79BE  AEC6 BCE9 D9A4 2D51 784F
     Subkey fingerprint: D64B B6F9 0CF3 77E9 B5FB  346F 50DE CCC4 710E 61AF
```

4. No servidor on-premises ou na máquina virtual (VM), execute o script de instalação. Especifique o nome do cluster, a região e o ID de ativação do Systems Manager e o código de ativação a partir da primeira etapa.

```
sudo bash /tmp/ecs-anywhere-install.sh \
    --region $REGION \
    --cluster $CLUSTER_NAME \
    --activation-id $ACTIVATION_ID \
    --activation-code $ACTIVATION_CODE
```

Para um servidor on-premises ou máquina virtual (VM) que tenha o driver NVIDIA instalado para workloads de GPU, você deve adicionar o sinalizador `--enable-gpu` ao script de instalação. Quando esse sinalizador é especificado, o script de instalação verifica se o driver NVIDIA está sendo executado e, em seguida, adiciona as variáveis de configuração necessárias para executar suas tarefas do Amazon ECS. Para obter mais informações sobre como executar workloads de GPU e especificar requisitos de GPU em uma definição de tarefa, consulte Especificar GPUs em uma definição de tarefa do Amazon ECS.

```
sudo bash /tmp/ecs-anywhere-install.sh \
    --region $REGION \
    --cluster $CLUSTER_NAME \
    --activation-id $ACTIVATION_ID \
    --activation-code $ACTIVATION_CODE \
    --enable-gpu
```

Use as etapas a seguir para registrar uma instância externa existente em um cluster diferente.

Para registrar uma instância externa existente em um cluster diferente

1. Interrompa o agente de contêiner do Amazon ECS.

```
sudo systemctl stop ecs.service
```

2. Edite o arquivo `/etc/ecs/ecs.config` e, na linha `ECS_CLUSTER`, verifique se o nome do cluster corresponde ao nome do cluster no qual a instância externa será registrada.
3. Remova os dados do agente do Amazon ECS existente.

```
sudo rm /var/lib/ecs/data/agent.db
```

4. Inicie o agente de contêiner do Amazon ECS.

```
sudo systemctl start ecs.service
```

AWS CLI for Windows operating systems

1. Crie um par de ativação do Systems Manager. Isso é usado para a ativação de instâncias gerenciadas do Systems Manager. A saída inclui um `ActivationId` e um

`ActivationCode`. Eles serão usados em uma etapa posterior. Especifique a função do IAM do ECS Anywhere que você criou. Para obter mais informações, consulte Perfil do IAM para o Amazon ECS Anywhere.

```
aws ssm create-activation --iam-role ecsAnywhereRole | tee ssm-activation.json
```

2. No servidor on-premises ou na máquina virtual (VM), baixe o script de instalação.

```
Invoke-RestMethod -URI "https://amazon-ecs-agent.s3.amazonaws.com/ecs-anywhere-install.ps1" -OutFile “ecs-anywhere-install.ps1”
```

3. (Opcional) O script do PowerShell é assinado pela Amazon e, portanto, o Windows executa automaticamente a validação do certificado no mesmo. Não há necessidade de realizar nenhuma validação manual.

   Para verificar manualmente o certificado, clique com o botão direito do mouse no arquivo, navegue até propriedades e use a guia Digital Signatures (Assinaturas digitais) para obter mais detalhes.

   Essa opção só está disponível quando o host tiver o certificado no armazenamento de certificados.

   Essa verificação deve retornar informações semelhante às seguintes:

```
# Verification (PowerShell)
Get-AuthenticodeSignature -FilePath .\ecs-anywhere-install.ps1

SignerCertificate                         Status      Path
-----------------                         ------      ----
EXAMPLECERTIFICATE                        Valid       ecs-anywhere-install.ps1

...

Subject              : CN="Amazon Web Services, Inc.",...

----
```

4. No servidor on-premises ou na máquina virtual (VM), execute o script de instalação. Especifique o nome do cluster, a região e o ID de ativação do Systems Manager e o código de ativação a partir da primeira etapa.

```
.\ecs-anywhere-install.ps1 -Region $Region -Cluster $Cluster -
ActivationID $ActivationID -ActivationCode $ActivationCode
```

5. Verifique se o agente de contêiner do Amazon ECS está em execução.

```
Get-Service AmazonECS

Status   Name               DisplayName
------   ----               -----------
Running  AmazonECS          Amazon ECS
```

Use as etapas a seguir para registrar uma instância externa existente em um cluster diferente.

Para registrar uma instância externa existente em um cluster diferente

1. Interrompa o agente de contêiner do Amazon ECS.

```
Stop-Service AmazonECS
```

2. Modifique o parâmetro `ECS_CLUSTER` para que o nome do cluster corresponda ao nome do cluster no qual a instância externa será registrada.

```
[Environment]::SetEnvironmentVariable("ECS_CLUSTER", $ECSCluster,
 [System.EnvironmentVariableTarget]::Machine)
```

3. Remova os dados do agente do Amazon ECS existente.

```
Remove-Item -Recurse -Force $env:ProgramData\Amazon\ECS\data\*
```

4. Inicie o agente de contêiner do Amazon ECS.

```
Start-Service AmazonECS
```

A AWS CLI pode ser usada para criar uma ativação do Systems Manager antes da execução do script de instalação para a conclusão do processo de registro da instância externa.

Cancelamento do registro de uma instância externa do Amazon ECS

Recomendamos que você cancele o registro da instância no Amazon ECS e no AWS Systems Manager quando concluir o uso de uma instância externa. Depois do cancelamento do registro, a instância externa não poderá mais aceitar novas tarefas.

Se você tiver tarefas em execução na instância de contêiner quando cancelar o registro, estas tarefas permanecerão em execução até serem interrompidas por outros meios. Contudo, essas tarefas não serão mais monitoradas ou gerenciadas pelo Amazon ECS. Se essas tarefas da instância externa fizerem parte de um serviço do Amazon ECS, o programador de serviços iniciará uma nova cópia das tarefas em uma instância diferente, se possível.

Após cancelar o registro da instância, limpe os recursos da AWS restantes na instância. Depois, você pode registrá-la em um novo cluster.

Procedimento

AWS Management Console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, escolha a região em que sua instância externa está registrada.
3. No painel de navegação, escolha Clusters e selecione o cluster que hospeda a instância externa.
4. Na página Cluster : ***name***, escolha a guia Infrastructure (Infraestrutura).
5. Em Container instances (Instâncias de contêiner), selecione o ID da instância externa para cancelar o registro. Você será redirecionado para a página de detalhes da instância do contêiner.
6. Na página Container Instance : ***id***, escolha Deregister.
7. Revise a mensagem de cancelamento do registro. Selecione Cancelar o registro no AWS Systems Manager para também cancelar o registro da instância externa como uma instância gerenciada no Systems Manager. Escolha Cancelar registro.

> **Note**
>
> É possível cancelar o registro da instância externa como uma instância gerenciada do Systems Manager no console do Systems Manager. Para obter instruções,

consulte Cancelar o registro de nós gerenciados em um ambiente híbrido e multinuvem no Guia do usuário do AWS Systems Manager.

8. Após cancelar o registro da instância, limpe os recursos da AWS no servidor on-premises ou na VM.

Sistema operacional	Etapas

Linux	a. Interrompa o agente de contêiner do Amazon ECS e os serviços do SSM Agent na instância.
	`sudo systemctl stop ecs amazon-ssm-agent`
	b. Remova os pacotes do Amazon ECS e do Systems Manager.
	Para CentOS 7, CentOS 8 e RHEL 7
	`sudo yum remove -y amazon-ecs-init amazon-ssm-agent`
	Para SUSE Enterprise Server 15
	`sudo zypper remove -y amazon-ecs-init amazon-ssm-agent`
	Para Debian e Ubuntu

Sistema operacional	Etapas

	`sudo apt remove -y amazon-ecs-init amazon-ssm-agent`
	c. Remova os diretórios restantes.
	`sudo rm -rf /var/lib/ecs /etc/ecs /var/lib/amazon/ssm /var/log/ecs /var/log/amazon/ssm`

Windows	a. Interrompa o agente de contêiner do Amazon ECS e os serviços do SSM Agent na instância.
	`Stop-Service AmazonECS`
	`Stop-Service AmazonSSMAgent`
	b. Remova o pacote do Amazon ECS.
	`.\ecs-anywhere-install.ps1 -Uninstall`

AWS CLI

1. Você precisa do ID da instância e do ARN da instância de contêiner para cancelar o registro da instância de contêiner. Se você não tiver esses valores, execute os comandos a seguir

Execute o comando a seguir para obter o ID da instância.

Você usa o ID da instância (`instanceID`) para obter o ARN (`containerInstanceARN`) da instância de contêiner.

```
instanceId=$(aws ssm describe-instance-information --region "{{ region }}" |
 jq ".InstanceInformationList[] |select(.IPAddress==\"{{ IPv4 Address }}\")
 | .InstanceId" | tr -d'"'
```

Execute os seguintes comandos.

Você usa o `containerInstanceArn` como parâmetro no comando para cancelar o registro da instância (`deregister-container-instance`).

```
instances=$(aws ecs list-container-instances --cluster "{{ cluster }}" --region
 "{{ region }}" | jq -c '.containerInstanceArns')
containerInstanceArn=$(aws ecs describe-container-instances --cluster
 "{{ cluster }}" --region "{{ region }}" --container-instances $instances
 | jq ".containerInstances[] | select(.ec2InstanceId==\"{{ instanceId }}\")
 | .containerInstanceArn" | tr -d '"')
```

2. Execute o comando a seguir para drenar a instância.

```
aws ecs update-container-instances-state --cluster "{{ cluster }}" --region
 "{{ region }}" --container-instances "{{ containerInstanceArn }}" --status
 DRAINING
```

3. Depois que a instância de contêiner terminar de ser drenada, execute o comando a seguir para cancelar o registro da instância.

```
aws ecs deregister-container-instance --cluster "{{ cluster }}" --region
 "{{ region }}" --container-instance "{{ containerInstanceArn }}"
```

4. Execute o comando a seguir para remover instâncias de contêiner do SSM.

```
aws ssm deregister-managed-instance --region "{{ region }}" --instance-id
 "{{ instanceId }}"
```

5. Após cancelar o registro da instância, limpe os recursos da AWS no servidor on-premises ou na VM.

Sistema operacional	Etapas

Linux	a. Interrompa o agente de contêiner do Amazon ECS e os serviços do SSM Agent na instância.
	`sudo systemctl stop ecs amazon-ssm-agent`
	b. Remova os pacotes do Amazon ECS e do Systems Manager.
	`sudo (yum/apt/zypper) remove amazon-ecs-init amazon-ssm-agent`
	c. Remova os diretórios restantes.
	`sudo rm -rf /var/lib/ecs /etc/ecs /var/lib/amazon/ssm /var/log/ecs /var/log/amazon/ssm`

Sistema operacional	Etapas

Windows	a. Interrompa o agente de contêiner do Amazon ECS e os serviços do SSM Agent na instância.
	`Stop-Service AmazonECS`
	`Stop-Service AmazonSSMAgent`
	b. Remova o pacote do Amazon ECS.
	`.\ecs-anywhere-install.ps1 -Uninstall`

Atualização do agente do AWS Systems Manager e o agente de contêiner do Amazon ECS em uma instância externa

O servidor on-premises ou a VM deve executar o agente do AWS Systems Manager (SSM Agent) e o agente de contêiner do Amazon ECS ao executar workloads do Amazon ECS. A AWS lança novas versões desses agentes quando qualquer recurso é adicionado ou atualizado. Se as instâncias externas estiverem usando uma versão anterior de qualquer agente, será possível atualizá-las usando os procedimentos a seguir.

Atualizar o SSM Agent em uma instância externa

O AWS Systems Manager recomenda que você automatize o processo de atualização do SSM Agent nas instâncias. São fornecidos vários métodos para automatizar as atualizações. Para obter mais informações, consulte Automatizar atualizações no SSM Agent no Guia do usuário do AWS Systems Manager.

Atualizar o agente do Amazon ECS em uma instância externa

Nas instâncias externas, o agente de contêiner do Amazon ECS é atualizado por meio do upgrade do pacote `ecs-init`. A atualização do agente de contêiner do Amazon ECS não interrompe a execução de tarefas ou serviços. O Amazon ECS fornece o pacote `ecs-init` e o arquivo de assinatura em um bucket do Amazon S3 em cada região. Começando com a versão `1.52.1-1` do código `ecs-init`, o Amazon ECS fornece pacotes `ecs-init` separados que dependem do sistema operacional e da arquitetura de sistema que a instância externa usa.

Use a tabela a seguir para determinar o pacote `ecs-init` que você deve baixar com base no sistema operacional e na arquitetura de sistema que a instância externa usa.

> **Note**
>
> É possível determinar o sistema operacional e a arquitetura de sistema que a instância externa usa por meio dos comandos a seguir.
>
> ```
> cat /etc/os-release
> uname -m
> ```

Sistemas operacionais (arquitetura)	Pacote ecs-init
CentOS 7 (x86_64) CentOS 8 (x86_64) CentOS Stream 9 (x86_64) SUSE Enterprise Server 15 (x86_64) RHEL 7 (x86_64) RHEL 8 (x86_64)	`amazon-ecs-init-latest.x86_64.rpm`
CentOS 7 (aarch64) CentOS 8 (aarch64) CentOS Stream 9 (aarch64)	`amazon-ecs-init-latest.aarch64.rpm`

Sistemas operacionais (arquitetura)	Pacote ecs-init
RHEL 7 (aarch64)	
Debian 9 (x86_64) Debian 10 (x86_64) Debian 11 (x86_64) Debian 12 (x86_64) Ubuntu 18 (x86_64) Ubuntu 20 (x86_64) Ubuntu 22 (x86_64) Ubuntu 24 (x86_64)	`amazon-ecs-init-latest.amd64.deb`
Debian 9 (aarch64) Debian 10 (aarch64) Debian 11 (aarch64) Debian 12 (aarch64) Ubuntu 18 (aarch64) Ubuntu 20 (aarch64) Ubuntu 22 (aarch64) Ubuntu 24 (aarch64)	`amazon-ecs-init-latest.arm64.deb`

Siga estas etapas para atualizar o agente do Amazon ECS.

Para atualizar o agente do Amazon ECS

1. Confirme a versão do agente do Amazon ECS que você está executando.

```
curl -s 127.0.0.1:51678/v1/metadata | python3 -mjson.tool
```

2. Baixe o pacote `ecs-init` para seu sistema operacional e arquitetura de sistema. O Amazon ECS fornece o arquivo de pacote `ecs-init` em um bucket do Amazon S3 em cada região. Substitua o identificador *<region>* no comando pelo nome da região (por exemplo, `us-west-2`) mais próxima de você geograficamente.

   amazon-ecs-init-latest.x86_64.rpm

   ```
   curl -o amazon-ecs-init.rpm https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.x86_64.rpm
   ```

   amazon-ecs-init-latest.aarch64.rpm

   ```
   curl -o amazon-ecs-init.rpm https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.aarch64.rpm
   ```

   amazon-ecs-init-latest.amd64.deb

   ```
   curl -o amazon-ecs-init.deb https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.amd64.deb
   ```

   amazon-ecs-init-latest.arm64.deb

   ```
   curl -o amazon-ecs-init.deb https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.arm64.deb
   ```

3. (Opcional) Verifique a validade do arquivo de pacote `ecs-init` usando a assinatura PGP.

   a. Baixe e instale GnuPG. Para obter mais informações sobre a GNUpg, consulte o site da GnuPG. Para sistemas Linux, instale `gpg` usando o gerenciador de pacotes no seu tipo de Linux.

   b. Recuperar a chave pública PGP do Amazon ECS.

      ```
      gpg --keyserver hkp://keys.gnupg.net:80 --recv BCE9D9A42D51784F
      ```

   c. Baixe a assinatura do pacote `ecs-init`. Ela consiste em uma assinatura PGP desvinculada ASCII armazenada em um arquivo com a extensão `.asc`. O Amazon ECS

fornece o arquivo de assinatura em um bucket do Amazon S3 em cada região. Substitua o identificador *<region>* no comando pelo nome da região (por exemplo, `us-west-2`) mais próxima de você geograficamente.

amazon-ecs-init-latest.x86_64.rpm

```
curl -o amazon-ecs-init.rpm.asc https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.x86_64.rpm.asc
```

amazon-ecs-init-latest.aarch64.rpm

```
curl -o amazon-ecs-init.rpm.asc https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.aarch64.rpm.asc
```

amazon-ecs-init-latest.amd64.deb

```
curl -o amazon-ecs-init.deb.asc https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.amd64.deb.asc
```

amazon-ecs-init-latest.arm64.deb

```
curl -o amazon-ecs-init.deb.asc https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.arm64.deb.asc
```

d. Verifique o arquivo de pacote `ecs-init` usando a chave.

Para os pacotes **`rpm`**

```
gpg --verify amazon-ecs-init.rpm.asc ./amazon-ecs-init.rpm
```

Para os pacotes **`deb`**

```
gpg --verify amazon-ecs-init.deb.asc ./amazon-ecs-init.deb
```

A saída esperada é mostrada a seguir.

```
gpg: Signature made Fri 14 May 2021 09:31:36 PM UTC
gpg:                using RSA key 50DECCC4710E61AF
gpg: Good signature from "Amazon ECS <ecs-security@amazon.com>" [unknown]
```

```
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: F34C 3DDA E729 26B0 79BE  AEC6 BCE9 D9A4 2D51 784F
     Subkey fingerprint: D64B B6F9 0CF3 77E9 B5FB  346F 50DE CCC4 710E 61AF
```

4. Instale o pacote `ecs-init`.

   Para o pacote **`rpm`** no CentOS 7, CentOS 8 e RHEL 7

   ```
   sudo yum install -y ./amazon-ecs-init.rpm
   ```

   Para o pacote **`rpm`** no SUSE Enterprise Server 15

   ```
   sudo zypper install -y --allow-unsigned-rpm ./amazon-ecs-init.rpm
   ```

   Para o pacotes **`deb`**

   ```
   sudo dpkg -i ./amazon-ecs-init.deb
   ```

5. Reinicie o serviço `ecs`.

   ```
   sudo systemctl restart ecs
   ```

6. Verifique se a versão do agente do Amazon ECS foi atualizada.

   ```
   curl -s 127.0.0.1:51678/v1/metadata | python3 -mjson.tool
   ```

Atualização de um cluster do Amazon ECS

É possível modificar o as propriedades do cluster a seguir:

- Definir um provedor de capacidade padrão

  Cada cluster pode ter um ou mais provedores de capacidade e uma estratégia opcional de provedor de capacidade. A estratégia do provedor de capacidade determina como as tarefas são distribuídas entre os provedores de capacidade de clusters. Ao executar uma tarefa autônoma ou criar um serviço, você pode usar a estratégia padrão de provedor de capacidade do cluster ou uma estratégia de provedor de capacidade que substitua a estratégia padrão.
- Ative o Container Insights.

O CloudWatch Container Insights coleta, agrega e resume métricas e logs das suas aplicações e microsserviços conteinerizados. O Container Insights também fornece informações de diagnóstico, como falhas de reinicialização de contêiner, que você usa para isolar problemas e resolvê-los rapidamente. Para obter mais informações, consulte the section called “Monitorar contêineres do Amazon ECS utilizando o Container Insights com observabilidade aprimorada”.

- Adicione tags para ajudar a identificar seus clusters.

Procedimento

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Clusters.
3. Na página Clusters, escolha o cluster.
4. Na página Cluster : ***nome***, escolha Atualizar cluster.
5. Para definir o provedor de capacidade padrão, em Estratégia do provedor de capacidade padrão, escolha Adicionar mais.
   a. Em Provedor de capacidade, escolha o provedor de capacidade.
   b. (Opcional) Em Base, insira o número mínimo de tarefas que serão executadas no provedor de capacidade.

      Você só pode definir um valor Base para um provedor de capacidade.
   c. (Opcional) Em Peso, insira o percentual relativo do número total de tarefas executadas que usem o provedor de capacidade especificado.
   d. (Opcional) Repita as etapas para qualquer provedor de capacidade adicional.
6. Para ativar ou desativar o Container Insights, expanda Monitoramento e, em seguida, ative Usar o Conteiner Insights.
7. Para ajudar a identificar seu cluster, expanda Tags e configure suas tags.

   [Adicionar uma tag] Selecione Add tag (Adicionar tag) e faça o seguinte:

   - Em Key (Chave), insira o nome da chave.
   - Em Value (Valor), insira o valor da chave.

   [Remover uma tag] Escolha Remover à direita da chave e do valor da tag.

8. Selecione Atualizar.

Exclusão de um cluster do Amazon ECS

Se você terminou de usar um cluster, poderá excluí-lo. Depois de excluir o cluster, ele faz a transição para o estado `INACTIVE`. Os clusters com um status `INACTIVE` podem permanecer detectáveis em sua conta por um período. No entanto, esse comportamento está sujeito a alterações no futuro, então você não deve confiar na persistência de clusters `INACTIVE`.

Antes de excluir um cluster, você deve executar as seguintes operações:

- Exclua todos os serviços no cluster. Para obter mais informações, consulte the section called “Excluir um serviço”.
- Interrompa todas as tarefas em execução no momento. Para obter mais informações, consulte the section called “Interrupção de uma tarefa”.
- Cancele o registro de todas as instâncias de contêiner registradas no cluster. Para obter mais informações, consulte the section called “Cancelamento do registro de uma instância de contêiner”.
- Excluir o namespace de . Para obter mais informações, consulte Deleting namespaces no Guia do desenvolvedor do AWS Cloud Map.

Procedimento

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, selecione a Região a ser usada.
3. No painel de navegação, escolha Clusters.
4. Na página Clusters, selecione o cluster a ser excluído.
5. Na parte superior direita da página, escolha Delete Cluster (Excluir cluster).

   Uma mensagem é exibida quando você não excluiu todos os recursos associados ao cluster.
6. Na caixa de confirmação, insira delete ***`nome do cluster`***.

Cancelamento do registro de uma instância de contêiner do Amazon ECS

> **Important**
>
> Este tópico abrange apenas instâncias de contêiner criadas no Amazon EC2. Para obter mais informações sobre como cancelar o registro de instâncias externas, consulte Cancelamento do registro de uma instância externa do Amazon ECS.

Quando você concluir uma instância de contêiner baseada no Amazon EC2, deverá cancelar o registro dela no cluster. Depois do cancelamento do registro, a instância de contêiner poderá mais aceitar novas tarefas.

Se você tiver tarefas em execução na instância de contêiner quando cancelar o registro, essas tarefas permanecerão em execução até que você conclua a instância ou até que as tarefas sejam interrompidas por outros meios. Contudo, essas tarefas são órfãs, o que significa que elas não são mais monitoradas ou gerenciadas pelo Amazon ECS. Se uma tarefa órfã na instância de contêiner fizer parte de um serviço do Amazon ECS, o programador de serviços iniciará uma nova cópia desta tarefa em uma instância de contêiner diferente, se possível. Todos os contêineres em tarefas de serviço órfãs registradas em um grupo de destino do Application Load Balancer terão seu registro cancelado. Eles começarão a drenagem da conexão de acordo com as definições do load balancer ou do grupo de destino. Se uma tarefa órfã estiver usando o modo de rede `awsvpc`, as interfaces de rede elásticas serão excluídas.

Se você pretende usar a instância de contêiner para qualquer outro propósito depois do cancelamento de registro, você deve interromper todas as tarefas em execução na instância de contêiner antes do cancelamento. Isso impede que as tarefas órfãs consumam recursos.

Ao cancelar o registro de uma instância de contêiner do, esteja ciente das seguintes considerações.

- Como cada instância de contêiner tem informações de estado exclusivas, o registro delas não deve ser cancelado de um cluster e realizado novamente em outro. Para relocar recursos de instância de contêiner, recomendamos que você encerre as instâncias de contêiner de um cluster e inicie novas instâncias de contêiner no novo cluster. Para obter mais informações, consulte Encerrar a instância no Manual do usuário do Amazon EC2 e em Iniciar uma instância de contêiner do Linux do Amazon ECS.

- Se a instância de contêiner for gerenciada por um grupo do Auto Scaling ou uma pilha do AWS CloudFormation, encerre a instância atualizando o grupo do Auto Scaling ou a pilha do AWS CloudFormation. Caso contrário, o grupo do Auto Scaling ou o AWS CloudFormation criará uma nova instância depois que você a encerrar.
- Se você encerra uma instância de contêiner em execução com um agente de contêiner do Amazon ECS conectado, o agente automaticamente cancela o registro da instância no cluster. O registro de instâncias de contêiner interrompidas ou instâncias com agentes desconectados não é automaticamente cancelado quando concluído.
- O cancelamento de registro de uma instância de contêiner remove a instância de um cluster, mas não encerra a instância do Amazon EC2. Se você tiver terminado de usar a instância, certifique-se de encerrá-la também para interromper o faturamento. Para obter mais informações, consulte Terminar sua instância no Guia do usuário do Amazon EC2.

Procedimento

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, escolha a região em que sua instância externa está registrada.
3. No painel de navegação, escolha Clusters e selecione o cluster que hospeda a instância.
4. Na página Cluster : ***name***, escolha a guia Infrastructure (Infraestrutura).
5. Em Container instances (Instâncias de contêiner), selecione o ID da instância para cancelar o registro. Você será redirecionado para a página de detalhes da instância do contêiner.
6. Na página Container Instance : ***id***, escolha Deregister.
7. Na tela de confirmação, escolha Cancelar registro.
8. Se você tiver terminado de trabalhar com a instância de contêiner, encerre a instância subjacente do Amazon EC2. Para obter mais informações, consulte Encerrar a instâncias no Manual do usuário do Amazon EC2.

Drenagem de instâncias de contêiner do Amazon ECS

Poderá haver momentos em que você precisará remover uma instância de contêiner do cluster, por exemplo, para executar atualizações do sistema ou para reduzir a escala verticalmente da capacidade do cluster. O Amazon ECS fornece a capacidade de fazer a transição de uma instância de contêiner para um status `DRAINING`. Esse recurso é denominado drenagem de instância de

contêiner. Quando uma instância de contêiner é definida como `DRAINING`, o Amazon ECS impede que novas tarefas sejam programadas para posicionamento na instância de contêiner.

Comportamento de drenagem para serviços

Todas as tarefas que fazem parte de um serviço e que estão no estado `PENDING` são interrompidas de imediato. Se houver capacidade de instância de contêiner disponível no cluster, o programador de serviços iniciará as tarefas de substituição. Se não houver capacidade suficiente de instância de contêiner, será enviada uma mensagem de evento de serviço indicando o problema.

Tarefas que fazem parte de um serviço na instância de contêiner e que estão no estado `RUNNING` são transferidas para o estado `STOPPED`. O programador de serviços tenta substituir as tarefas de acordo com os parâmetros de tipo de implantação e de configuração de implantação do serviço, `minimumHealthyPercent` e `maximumPercent`. Para obter mais informações, consulte Serviços do Amazon ECS e Parâmetros de definição de serviço do Amazon ECS.

- Se `minimumHealthyPercent` estiver abaixo de 100%, o programador pode ignorar `desiredCount` temporariamente durante a substituição de tarefas. Por exemplo, se `desiredCount` for quatro tarefas, um mínimo de 50% permitem que o programador interrompa duas tarefas existentes antes de iniciar duas novas tarefas. Se o mínimo é 100%, a programador de serviço não pode remover tarefas existentes até que as tarefas de substituição sejam consideradas saudáveis. Se as tarefas para serviços que não usam um load balancer estiverem com o status de `RUNNING`, elas serão consideradas saudáveis. As tarefas para serviços que usam um load balancer são consideradas saudáveis se estiverem com o status de `RUNNING` e se a instância de contêiner que as hospedam são relatadas como saudáveis pelo load balancer.

  > **Important**
  >
  > Se você usar Instâncias Spot e a `minimumHealthyPercent` for maior ou igual a 100%, o serviço não terá tempo suficiente para substituir a tarefa antes que a instância spot seja encerrada.

- O parâmetro `maximumPercent` representa um limite maior no número de tarefas em execução durante a substituição de tarefas, o que permite que você defina o tamanho do lote de substituição. Por exemplo, se `desiredCount` for quatro tarefas, um máximo de 200% inicia quatro novas tarefas antes que elas sejam drenadas (considerando que os recursos de cluster necessários fazer isso estejam disponíveis). Se o máximo for 100%, as tarefas de substituição não podem começar até que tarefas de drenagem sejam interrompidas.

> **Important**
>
> Se tanto `minimumHealthyPercent` quanto `maximumPercent` estiverem em 100%, então o serviço não pode remover tarefas existentes e também não pode iniciar tarefas de substituição. Isso evita a drenagem com êxito da instância de contêiner e evita a realização de novas implantações.

Comportamento de drenagem para tarefas autônomas

Quaisquer tarefas autônomas no estado `PENDING` ou `RUNNING` não são afetadas. Você deve esperar até que eles sejam interrompidas por conta própria ou interrompa-as manualmente. A instância de contêiner permanecerá no status `DRAINING`.

Uma instância de contêiner terá concluído a drenagem quando todas as tarefas em execução na instância tiverem feito a transição para o estado `STOPPED`. A instância de contêiner permanece no estado `DRAINING` até que seja reativada ou excluída. É possível verificar o estado das tarefas na instância de contêiner usando a operação ListTasks com o parâmetro `containerInstance`para obter uma lista de tarefas na instância, seguida por uma operação DescribeTasks com o nome do recurso da Amazon (ARN) ou o ID de cada tarefa para verificar o estado da tarefa.

Quando você estiver pronto para que a instância de contêiner comece a hospedar tarefas novamente, altere o estado da instância de contêiner de `DRAINING` para `ACTIVE`. O programador do serviço do Amazon ECS considerará a instância de contêiner novamente para a colocação de tarefas.

Procedimento

As etapas a seguir podem ser usadas para definir uma instância de contêiner a ser drenada usando o novo AWS Management Console.

Você também pode usar a ação de API UpdateContainerInstancesState ou o comando update-container-instances-state a fim de alterar o status de uma instância de contêiner para `DRAINING`.

AWS Management Console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Clusters.

3. Na página Clusters, escolha um cluster que hospede suas instâncias.
4. Na página Cluster : ***name***, escolha a guia Infrastructure (Infraestrutura). Em seguida, em Container instances (Instâncias de contêiner), marque a caixa de seleção de cada instância de contêiner que você deseja drenar.
5. Escolha Ações, Drenar.

Instâncias de contêineres do EC2 para Amazon ECS

O agente do Amazon ECS é um processo executado em cada instância de contêiner registrada em seu cluster. Ele facilita a comunicação entre as instâncias de contêiner e o Amazon ECS.

> **Note**
>
> Em instâncias de contêiner Linux, o contêiner do agente monta diretórios de nível superior, como `/lib`, `/lib64` e `/proc`. Isso é necessário para recursos e funcionalidades do ECS, como volumes do Amazon EBS, modo de rede `awsvpc`, Amazon ECS Service Connect e FireLens para Amazon ECS.

Cada agente de contêiner do Amazon ECS oferece suporte a um conjunto diferente de recursos e fornece correções de erros de versões anteriores. Quando possível, sempre recomendamos usar a versão mais recente do agente de contêiner do Amazon ECS. Para atualizar o agente de contêiner para a versão mais recente, consulte Atualizar o agente de contêiner do Amazon ECS.

O agente de contêiner do Amazon ECS contém a imagem `amazon-ecs-pause`. O Amazon ECS usa essa imagem em tarefas que usam o modo de rede `awsvpc`.

Para ver quais recursos e aprimoramentos estão incluídos em cada versão de agente, consulte https://github.com/aws/amazon-ecs-agent/releases.

> **Important**
>
> A versão mínima do Docker para métricas confiáveis é a versão Docker `v20.10.13` e posteriores, que está incluída na AMI otimizada para o Amazon ECS `20220607` e posteriores.
> As versões `1.20.0` e posteriores do agente do Amazon ECS descontinuaram o suporte para as versões do Docker anteriores à `18.01.0`.

Ciclo de vida

Quando o agente de contêiner do Amazon ECS registra uma instância do Amazon EC2 no cluster, a instância do Amazon EC2 relata seu status como `ACTIVE` e o status de conexão do agente como `TRUE`. Essa instância de contêiner pode aceitar solicitações de tarefas de processamento.

Se você interrompe (sem concluir) uma instância de contêiner, o status permanece como `ACTIVE`, mas o status de conexão do agente muda para `FALSE` em instantes. As tarefas que estavam sendo executadas na instância de contêiner são interrompidas. Se você reiniciar a instância de contêiner, o agente de contêiner se reconectará com o serviço do Amazon ECS e será possível, novamente, executar tarefas na instância.

> **Important**
>
> Se você interromper e iniciar uma instância de contêiner ou se reinicializar esta instância, algumas versões mais antigas do agente de contêiner do Amazon ECS registrarão a instância novamente, sem cancelar o registro do ID da instância de contêiner original. Nesse caso, o Amazon ECS listará mais instâncias de contêiner no cluster do que você realmente possui. (Se você tiver IDs de instância de contêiner duplicados para o mesmo ID de instância do Amazon EC2, poderá, com segurança, cancelar o registro das duplicatas listadas como `ACTIVE` com status da conexão do agente de `FALSE`.) Esse problema foi corrigido na versão atual do agente de contêiner do Amazon ECS. Para obter mais informações sobre como atualizar a versão atual, consulte Atualizar o agente de contêiner do Amazon ECS.

Se você alterar o status de uma instância de contêiner para `DRAINING`, as novas tarefas não serão posicionadas na instância de contêiner. Todas as tarefas de serviço em execução na instância de contêiner são removidas, se possível, de modo que você possa realizar atualizações de sistema. Para obter mais informações, consulte Drenagem de instâncias de contêiner do Amazon ECS.

Se você cancela o registro ou encerra uma instância de contêiner, seu status muda para `INACTIVE` imediatamente, e ela não é mais referida não quando você lista suas instâncias de contêiner. No entanto, você ainda pode descrever a instância de contêiner por uma hora depois do encerramento. Depois desse período, a descrição de instância não estará mais disponível.

> **Important**
>
> É possível drenar as instâncias manualmente ou criar um hook do ciclo de vida do grupo do Auto Scaling para definir o status da instância como `DRAINING`. Para obter mais informações

sobre hooks do ciclo de vida do Auto Scaling, consulte Hooks do ciclo de vida do Amazon EC2 Auto Scaling.

Compatibilidade com o Docker

O Amazon ECS é compatível com as duas últimas versões principais do Docker publicadas no Amazon Linux. Atualmente, isso inclui o Docker 20.10.x e o Docker 25.x.

A versão mínima exigida do Docker para o Amazon ECS pode ser encontrada no arquivo de especificação do agente do Amazon ECS no GitHub.

Ao usar a AMI otimizada para Amazon ECS, o Docker é pré-instalado e configurado para funcionar com o agente de contêiner do Amazon ECS. A AMI inclui uma versão do Docker que é testada e compatível com o Amazon ECS.

> **Note**
>
> Embora o Amazon ECS seja compatível com várias versões do Docker, recomendamos usar a versão do Docker que vem com a AMI otimizada para Amazon ECS para obter a melhor compatibilidade e suporte.

AMIs otimizadas para Amazon ECS

As variantes do Linux da AMI otimizada para Amazon ECS usam a AMI do Amazon Linux 2 como base. O nome da AMI de origem do Amazon Linux 2 para cada variante pode ser recuperado consultando a API da Systems Manager Parameter Store. Para obter mais informações, consulte Recuperação de metadados da AMI do Linux otimizada para o Amazon ECS. Quando você inicia nossas instâncias de contêiner a partir da AMI do Amazon Linux 2 mais recente otimizada para o Amazon ECS você recebe a versão atual do agente de contêiner. Para iniciar uma instância de contêiner com a AMI do Amazon Linux 2 mais recente otimizada pelo Amazon ECS, consulte Iniciar uma instância de contêiner do Linux do Amazon ECS.

Mais informações

As páginas a seguir fornecem informações adicionais sobre as alterações:

- Log de alterações do agente do Amazon ECS no GitHub

- O código-fonte da aplicação `ecs-init`, os scripts e a configuração para empacotar o agente agora fazem parte do repositório do agente. Para versões mais antigas de `ecs-init` e pacotes, consulte Log de alterações do Amazon ecs-init no GitHub
- Notas de release do Amazon Linux 2.
- Notas de lançamento do Docker Engine na documentação do Docker
- Documentação do driver NVIDIA na documentação da NVIDIA

Configuração do agente de contêiner do Amazon ECS

Aplica-se a: instâncias do EC2

O agente de contêiner do Amazon ECS oferece suporte a diversas opções de configuração, a maioria das quais você define por meio de variáveis de ambiente.

Se a instância de contêiner tiver sido executada com a variante do Linux da AMI otimizada para Amazon ECS, será possível definir essas variáveis de ambiente no arquivo `/etc/ecs/ecs.config` e reiniciar o agente. Também é possível gravar essas variáveis de configuração nas instâncias de contêiner com os dados de usuário do Amazon EC2 no momento de inicialização. Para obter mais informações, consulte Inicialização de instâncias de contêiner do Linux no Amazon ECS para transmitir dados.

Se a instância de contêiner tiver sido iniciada com a variante do Windows da AMI otimizada para Amazon ECS, será possível definir essas variáveis de ambiente com o comando SetEnvironmentVariable do PowerShell e, em seguida, reiniciar o agente. Para obter mais informações, consulte Executar comandos ao executar uma instância do EC2 com entrada de dados do usuário no Guia do usuário do Amazon EC2 e em the section called “Inicialização de instâncias de contêiner”.

Se você está iniciando o Amazon ECS manualmente, o agente de contêiner (para AMIs não otimizadas para Amazon ECS), pode usar essas variáveis de ambiente no comando docker run usado para iniciar o agente. Use essas variáveis com a sintaxe `--env=`*`VARIABLE_NAME`*`=`*`VARIABLE_VALUE`*. Para informações delicadas, como credenciais de autenticação para repositórios privados, você deve armazenar suas variáveis de ambiente do agente em um arquivo e passá-las todas ao mesmo tempo com a opção `--env-file` *`path_to_env_file`*. É possível usar os comandos a seguir para adicionar as variáveis.

```
sudo systemctl stop ecs
```

```
sudo vi /etc/ecs/ecs.config
# And add the environment variables with VARIABLE_NAME=VARIABLE_VALUE format.
sudo systemctl start ecs
```

Executar o agente do Amazon ECS com o namespace de PID do host

Por padrão, o agente do Amazon ECS é executado com seu próprio namespace de PID. Nas configurações a seguir, você poderá configurar o agente do Amazon ECS para ser executado com o namespace de PID do host:

- O modo de imposição do SELinux está habilitado.
- A política de segurança SELinux do Docker está definida como verdadeira.

Você pode configurar esse comportamento definindo a variável de ambiente `ECS_AGENT_PID_NAMESPACE_HOST` como `true` em seu arquivo `/etc/ecs/ecs.config`. Quando essa variável estiver habilitada, `ecs-init` iniciará o contêiner do agente do Amazon ECS com o namespace de PID do host (`--pid=host`), permitindo que o agente faça seu bootstrap adequadamente em ambientes que impõem o SELinux. Esse recurso está disponível na versão `1.94.0` e posterior do agente do Amazon ECS.

Para habilitar esse recurso, adicione a seguinte linha ao seu arquivo `/etc/ecs/ecs.config`:

```
ECS_AGENT_PID_NAMESPACE_HOST=true
```

Após fazer essa alteração, reinicie o agente do Amazon ECS para que a alteração entre em vigor:

```
sudo systemctl restart ecs
```

Os seguintes recursos não funcionarão quando o modo de imposição do SELinux estiver habilitado e a política de segurança do Docker estiver definida como verdadeira, mesmo quando `ECS_AGENT_PID_NAMESPACE_HOST=true` estiver definido.

- Amazon ECS Exec
- Anexação de tarefa do Amazon EBS
- Service Connect
- FireLens para o Amazon ECS

Parâmetros disponíveis

Para obter informações sobre os parâmetros de configuração do agente de contêiner do Amazon ECS, consulte Agente de contêiner do Amazon ECS no GitHub..

Armazenamento da configuração da instância de contêiner do Amazon ECS no Amazon S3

A configuração do agente de contêiner do Amazon ECS é controlada com a variável de ambiente. As variantes do Linux da AMI otimizada para Amazon ECS procuram essas variáveis em `/etc/ecs/ecs.config` quando o agente de contêiner é iniciado e configuram o agente adequadamente. Variáveis de ambiente não sensíveis, como `ECS_CLUSTER`, podem ser transmitidas para a instância de contêiner na inicialização com os dados de usuário do Amazon EC2 e gravadas nesse arquivo sem consequências. Contudo, outras informações confidenciais, como suas credenciais da AWS ou a variável `ECS_ENGINE_AUTH_DATA`, nunca devem ser passadas para uma instância em dados de usuário ou ser gravadas em `/etc/ecs/ecs.config` de forma a permitir que sejam exibidas em um arquivo `.bash_history`.

Armazenar informações de configuração em um bucket privado no Amazon S3 e conceder acesso somente leitura à função do IAM de instância de contêiner é uma maneira segura e prática de permitir a configuração da instância de contêiner na inicialização. É possível armazenar uma cópia do seu arquivo `ecs.config` em um bucket privado. É possível usar os dados do usuário do Amazon EC2 para instalar a AWS CLI e copiar as informações de configuração em `/etc/ecs/ecs.config` quando a instância for iniciada.

Para armazenar um arquivo **`ecs.config`** no Amazon S3

1. Você deve conceder permissões para ter acesso somente leitura ao Amazon S3 ao perfil de instância de contêiner (ecsInstanceRole). É possível fazer isso ao atribuir AmazonS3ReadOnlyAccess ao perfil `ecsInstanceRole`. Para obter informações sobre como anexar uma política a um perfil, consulte Atualizar permissões para um perfil no Guia do usuário do AWS Identity and Access Management.
2. Crie um arquivo `ecs.config` com variáveis de configuração válidas do agente Amazon ECS usando o formato a seguir. Este exemplo configura a autenticação de registro privado. Para obter mais informações, consulte Uso de imagens de contêiner que não são da AWS no Amazon ECS.

```
ECS_ENGINE_AUTH_TYPE=dockercfg
```

```
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"auth":"zq212MzEXAMPLE7o6T25Dk0i","email":"email@example.com"}}
```

> **Note**
>
> Para obter uma lista completa das variáveis de configuração do agente do Amazon ECS disponíveis, consulte Agente de contêiner do Amazon ECS no GitHub.

3. Para armazenar o arquivo de configuração, crie um bucket privado no Amazon S3. Para obter mais informações, consulte Criar um bucket no Guia do usuário do Amazon Simple Storage Service.
4. Faça upload do arquivo `ecs.config` no bucket do S3. Para obter mais informações, consulte Fazer upload de objetos no Guia do usuário do Amazon Simple Storage Service.

Para carregar um arquivo **`ecs.config`** do Amazon S3 na inicialização

1. Execute os procedimentos anteriores nesta seção para permitir acesso somente leitura do Amazon S3 às suas instâncias de contêiner e armazene um arquivo `ecs.config` em um bucket do S3 privado.
2. Inicie novas instâncias de contêiner e use o script de exemplo apresentado a seguir nos dados do usuário do EC2. O script instala a AWS CLI e copia o arquivo de configuração para `/etc/ecs/ecs.config`. Para obter mais informações, consulte Iniciar uma instância de contêiner do Linux do Amazon ECS.

```
#!/bin/bash
yum install -y aws-cli
aws s3 cp s3://your_bucket_name/ecs.config /etc/ecs/ecs.config
```

Instalar o agente de contêiner do Amazon ECS

Se desejar registrar uma instância do Amazon EC2 com o cluster do Amazon ECS e essa instância não estiver usando uma AMI baseada na AMI otimizada para o Amazon ECS, você poderá instalar o agente de contêiner do Amazon ECS manualmente usando o procedimento apresentado a seguir. Para fazer isso, você pode efetuar o download do agente de um dos buckets regionais do Amazon S3 ou do Amazon Elastic Container Registry Public. Se você efetuar o download de um dos buckets

regionais do Amazon S3, como opção, poderá verificar a validade do arquivo do agente de contêiner usando a assinatura PGP.

> **Note**
>
> As unidades `systemd` para os serviços do Amazon ECS e do Docker têm uma diretiva para aguardar a conclusão de `cloud-init` antes de iniciar ambos os serviços. O processo `cloud-init` só será considerado concluído quando a execução dos dados do usuário do Amazon EC2 estiver concluída. Portanto, iniciar o Amazon ECS ou o Docker por meio dos dados de usuário do Amazon EC2 pode causar um deadlock. Para iniciar o agente de contêiner usando dados de usuário do Amazon EC2, é possível usar `systemctl enable --now --no-block ecs.service`.

Instalar o agente de contêiner do Amazon ECS em uma instância que não seja do EC2 do Amazon Linux

Para instalar o agente de contêiner do Amazon ECS em uma instância do Amazon EC2, é possível efetuar o download do agente de um dos buckets regionais do Amazon S3 e instalá-lo.

> **Note**
>
> Ao usar uma AMI que não seja do Amazon Linux, sua instância do Amazon EC2 requer suporte do `cgroupfs` para o driver `cgroup` para que o agente do Amazon ECS ofereça suporte a limites de recursos em nível de tarefa. Para obter mais informações, consulte Amazon ECS agent on GitHub (Agente do Amazon ECS no GitHub).

Os arquivos do agente de contêiner do Amazon ECS mais recentes, por região, para cada arquitetura de sistema, são listados a seguir para referência.

Região	Nome da região	Arquivos de deb init do Amazon ECS	Arquivos de rpm init do Amazon ECS
us-east-2	Leste dos EUA (Ohio)	Amazon ECS amd64 init (amd64)	Amazon ECS x86_64 init (x86_64)

Região	Nome da região	Arquivos de deb init do Amazon ECS	Arquivos de rpm init do Amazon ECS
		Amazon ECS arm64 init (arm64)	Amazon ECS aarch64 init (aarch64)
us-east-1	Leste dos EUA (Norte da Virgínia)	Amazon ECS amd64 init (amd64) Amazon ECS arm64 init (arm64)	Amazon ECS x86_64 init (x86_64) Amazon ECS aarch64 init (aarch64)
us-west-1	Oeste dos EUA (Norte da Califórnia)	Amazon ECS amd64 init (amd64) Amazon ECS arm64 init (arm64)	Amazon ECS x86_64 init (x86_64) Amazon ECS aarch64 init (aarch64)
us-west-2	Oeste dos EUA (Oregon)	Amazon ECS amd64 init (amd64) Amazon ECS arm64 init (arm64)	Amazon ECS x86_64 init (x86_64) Amazon ECS aarch64 init (aarch64)
ap-east-1	Ásia-Pacífico (Hong Kong)	Amazon ECS amd64 init (amd64) Amazon ECS arm64 init (arm64)	Amazon ECS x86_64 init (x86_64) Amazon ECS aarch64 init (aarch64)
ap-northeast-1	Ásia-Pacífico (Tóquio)	Amazon ECS amd64 init (amd64) Amazon ECS arm64 init (arm64)	Amazon ECS x86_64 init (x86_64) Amazon ECS aarch64 init (aarch64)

Região	Nome da região	Arquivos de deb init do Amazon ECS	Arquivos de rpm init do Amazon ECS
ap-northeast-2	Ásia-Pacífico (Seul)	Amazon ECS amd64 init (amd64) Amazon ECS arm64 init (arm64)	Amazon ECS x86_64 init (x86_64) Amazon ECS aarch64 init (aarch64)
ap-south-1	Ásia-Pacífico (Mumbai)	Amazon ECS amd64 init (amd64) Amazon ECS arm64 init (arm64)	Amazon ECS x86_64 init (x86_64) Amazon ECS aarch64 init (aarch64)
ap-southeast-1	Ásia-Pacífico (Singapura)	Amazon ECS amd64 init (amd64) Amazon ECS arm64 init (arm64)	Amazon ECS x86_64 init (x86_64) Amazon ECS aarch64 init (aarch64)
ap-southeast-2	Ásia-Pacífico (Sydney)	Amazon ECS amd64 init (amd64) Amazon ECS arm64 init (arm64)	Amazon ECS x86_64 init (x86_64) Amazon ECS aarch64 init (aarch64)
ca-central-1	Canadá (Central)	Amazon ECS amd64 init (amd64) Amazon ECS arm64 init (arm64)	Amazon ECS x86_64 init (x86_64) Amazon ECS aarch64 init (aarch64)
eu-central-1	Europa (Frankfurt)	Amazon ECS amd64 init (amd64) Amazon ECS arm64 init (arm64)	Amazon ECS x86_64 init (x86_64) Amazon ECS aarch64 init (aarch64)

Região	Nome da região	Arquivos de deb init do Amazon ECS	Arquivos de rpm init do Amazon ECS
eu-west-1	Europa (Irlanda)	Amazon ECS amd64 init (amd64) Amazon ECS arm64 init (arm64)	Amazon ECS x86_64 init (x86_64) Amazon ECS aarch64 init (aarch64)
eu-west-2	Europa (Londres)	Amazon ECS amd64 init (amd64) Amazon ECS arm64 init (arm64)	Amazon ECS x86_64 init (x86_64) Amazon ECS aarch64 init (aarch64)
eu-west-3	Europa (Paris)	Amazon ECS amd64 init (amd64) Amazon ECS arm64 init (arm64)	Amazon ECS x86_64 init (x86_64) Amazon ECS aarch64 init (aarch64)
sa-east-1	América do Sul (São Paulo)	Amazon ECS amd64 init (amd64) Amazon ECS arm64 init (arm64)	Amazon ECS x86_64 init Amazon ECS aarch64 init (aarch64)
us-gov-east-1	AWS GovCloud (Leste dos EUA)	Amazon ECS amd64 init (amd64) Amazon ECS arm64 init (arm64)	Amazon ECS x86_64 init (x86_64) Amazon ECS aarch64 init (aarch64)
us-gov-west-1	AWS GovCloud (Oeste dos EUA)	Amazon ECS amd64 init (amd64) Amazon ECS arm64 init (arm64)	Amazon ECS x86_64 init (x86_64) Amazon ECS aarch64 init (aarch64)

Para instalar o agente de contêiner do Amazon ECS em uma instância do Amazon EC2 usando uma AMI que não seja do Amazon Linux

1. Inicie uma instância do Amazon EC2 com uma função do IAM que permita acesso ao Amazon ECS. Para obter mais informações, consulte Função do IAM de instância de contêiner do Amazon ECS.
2. Conecte-se à sua instância.
3. Instale a versão mais recente do Docker na instância.
4. Verifique a versão do Docker para verificar se seu sistema atende à exigência de versão mínima.

   > **Note**
   >
   > A versão mínima do Docker para métricas confiáveis é a versão Docker `v20.10.13` e posteriores, que está incluída na AMI otimizada para o Amazon ECS `20220607` e posteriores.
   > As versões `1.20.0` e posteriores do agente do Amazon ECS descontinuaram o suporte para as versões do Docker anteriores à `18.01.0`.

   ```
   docker --version
   ```

5. Baixe o arquivo de agente do Amazon ECS apropriado para seu sistema operacional e arquitetura de sistema e instale-o.

   Para arquiteturas `deb`:

   ```
   ubuntu:~$ curl -O https://s3.us-west-2.amazonaws.com/amazon-ecs-agent-us-west-2/amazon-ecs-init-latest.amd64.deb
   ubuntu:~$ sudo dpkg -i amazon-ecs-init-latest.amd64.deb
   ```

   Para arquiteturas `rpm`:

   ```
   fedora:~$ curl -O https://s3.us-west-2.amazonaws.com/amazon-ecs-agent-us-west-2/amazon-ecs-init-latest.x86_64.rpm
   fedora:~$ sudo yum localinstall -y amazon-ecs-init-latest.x86_64.rpm
   ```

6. Edite o arquivo `/lib/systemd/system/ecs.service` e adicione a linha apresentada a seguir no final da seção `[Unit]`.

```
After=cloud-final.service
```

7. (Opcional) Para registrar a instância com um cluster diferente do cluster `default`, edite o arquive `/etc/ecs/ecs.config` e adicione o seguinte conteúdo. O exemplo seguinte especifica o cluster `MyCluster`.

```
ECS_CLUSTER=MyCluster
```

   Para obter mais informações sobre essas e outras opções de runtime de agente, consulte Configuração do agente de contêiner do Amazon ECS.

   > **Note**
   >
   > É possível, opcionalmente, armazenar suas variáveis de ambiente do agente no Amazon S3 (que podem ser baixadas para as instâncias de contêiner no momento da inicialização usando os dados de usuário do Amazon EC2). Isso é recomendado para informações confidenciais, como credenciais de autenticação para repositórios privados. Para obter mais informações, consulte Armazenamento da configuração da instância de contêiner do Amazon ECS no Amazon S3 e Uso de imagens de contêiner que não são da AWS no Amazon ECS.

8. Inicie o serviço `ecs`.

```
ubuntu:~$ sudo systemctl start ecs
```

Execução do agente do Amazon ECS com o modo de rede de host

Durante a execução do agente de contêiner do Amazon ECS, `ecs-init` criará o contêiner do agente de contêiner com o modo de rede de `host`. Esse é o único modo de rede com suporte para o contêiner do agente de contêiner.

Isso permite que você bloqueie o acesso ao Endpoint de serviço de metadados da instância do Amazon EC2 (`http://169.254.169.254`) para os contêineres iniciados pelo agente de contêiner. Isso garante que os contêineres não possam acessar as credenciais de função do IAM do perfil da

instância de contêiner e impõe que as tarefas usem apenas as credenciais de função de tarefa do IAM. Para obter mais informações, consulte Perfil do IAM para tarefas do Amazon ECS.

Isso também permite que o agente de contêiner não dispute conexões e tráfego de rede na ponte `docker0`.

Parâmetros de configuração do log do agente de contêiner do Amazon ECS

O agente de contêiner do Amazon ECS armazena logs nas instâncias de contêiner.

Para o agente de contêiner versão 1.36.0 e posteriores, por padrão, os logs estão localizados em `/var/log/ecs/ecs-agent.log` nas instâncias do Linux e em `C:\ProgramData\Amazon\ECS\log\ecs-agent.log` nas instâncias do Windows.

Para o agente de contêiner versão 1.35.0 e posteriores, por padrão, os logs estão localizados em `/var/log/ecs/ecs-agent.log.`*`timestamp`* nas instâncias do Linux e em `C:\ProgramData\Amazon\ECS\log\ecs-agent.log.`*`timestamp`* nas instâncias do Windows.

Por padrão, os logs do agente são rotacionados de hora em hora com o máximo de 24 logs armazenados.

Veja a seguir as variáveis de configuração do agente de contêiner que podem ser usadas para alterar o comportamento padrão de log do agente. Para obter informações detalhadas sobre todos os parâmetros de configuração disponíveis, consulte Configuração do agente de contêiner do Amazon ECS o README do agente do Amazon ECS no GitHub.

Para o agente de contêiner versão 1.36.0 e posteriores, veja a seguir um arquivo de log de exemplo quando o formato `logfmt` é usado.

```
level=info time=2019-12-12T23:43:29Z msg="Loading configuration" module=agent.go
level=info time=2019-12-12T23:43:29Z msg="Image excluded from cleanup: amazon/amazon-
ecs-agent:latest" module=parse.go
level=info time=2019-12-12T23:43:29Z msg="Image excluded from cleanup: amazon/amazon-
ecs-pause:0.1.0" module=parse.go
level=info time=2019-12-12T23:43:29Z msg="Amazon ECS agent Version: 1.36.0, Commit:
 ca640387" module=agent.go
level=info time=2019-12-12T23:43:29Z msg="Creating root ecs cgroup: /ecs"
 module=init_linux.go
level=info time=2019-12-12T23:43:29Z msg="Creating cgroup /ecs"
 module=cgroup_controller_linux.go
level=info time=2019-12-12T23:43:29Z msg="Loading state!" module=statemanager.go
```

```
level=info time=2019-12-12T23:43:29Z msg="Event stream ContainerChange start
 listening..." module=eventstream.go
level=info time=2019-12-12T23:43:29Z msg="Restored cluster 'auto-robc'" module=agent.go
level=info time=2019-12-12T23:43:29Z msg="Restored from checkpoint file. I
 am running as 'arn:aws:ecs:us-west-2:0123456789:container-instance/auto-
robc/3330a8a91d15464ea30662d5840164cd' in cluster 'auto-robc'" module=agent.go
```

Veja a seguir um arquivo de log de exemplo quando o formato JSON é usado.

```
{"time": "2019-11-07T22:52:02Z", "level": "info", "msg": "Starting Amazon Elastic
 Container Service Agent", "module": "engine.go"}
```

Para agentes de contêiner versões 1.35.0 e anteriores, veja a seguir o formato do arquivo de log.

```
2016-08-15T15:54:41Z [INFO] Starting Agent: Amazon ECS Agent - v1.12.0 (895f3c1)
2016-08-15T15:54:41Z [INFO] Loading configuration
2016-08-15T15:54:41Z [WARN] Invalid value for task cleanup duration, will be overridden
 to 3h0m0s, parsed value 0, minimum threshold 1m0s
2016-08-15T15:54:41Z [INFO] Checkpointing is enabled. Attempting to load state
2016-08-15T15:54:41Z [INFO] Loading state! module="statemanager"
2016-08-15T15:54:41Z [INFO] Detected Docker versions [1.17 1.18 1.19 1.20 1.21 1.22]
2016-08-15T15:54:41Z [INFO] Registering Instance with ECS
2016-08-15T15:54:41Z [INFO] Registered! module="api client"
```

Configuração de instâncias de contêiner do Amazon ECS para imagens do Docker privadas

O agente de contêiner do Amazon ECS pode realizar a autenticação com registros privados usando a autenticação básica. Ao ativar a autenticação de registro privado, você pode utilizar imagens de docker privadas nas definições de tarefa. Esse recurso só é compatível com tarefas que usam o tipo de inicialização do EC2.

Outro método para habilitar a autenticação de registro privado usa o AWS Secrets Manager para armazenar suas credenciais de registro privado de forma segura e, então, referenciá-las em sua definição de contêiner. Isso permite que suas tarefas usem imagens de repositórios privados. Esse método oferece suporte a tarefas que usam os tipos de inicialização do EC2 ou do Fargate. Para obter mais informações, consulte Uso de imagens de contêiner que não são da AWS no Amazon ECS.

O agente de contêiner do Amazon ECS procura duas variáveis de ambiente ao ser iniciado:

- `ECS_ENGINE_AUTH_TYPE`, que especifica o tipo de dados de autenticação que está sendo enviado.
- `ECS_ENGINE_AUTH_DATA`, que contém as credenciais de autenticação em si.

As variantes do Linux da AMI otimizada para Amazon ECS procuram essas variáveis no arquivo `/etc/ecs/ecs.config` quando a instância de contêiner é inicializada, e cada vez que o serviço é iniciado (com o comando sudo start ecs). As AMIs que não são otimizadas para Amazon ECS devem armazenar essas variáveis de ambiente em um arquivo e transmiti-las com a opção `--env-file` *`path_to_env_file`* para o comando docker run que inicia o agente de contêiner.

> **Important**
>
> Não recomendamos que você injete essas variáveis de ambiente de autenticação na inicialização da instância com os dados de usuário do Amazon EC2 ou as transmita com a opção `--env` para o comando docker run. Esses métodos não são adequados para dados confidenciais, como credenciais de autenticação. Para obter informações sobre como adicionar credenciais de autenticação com segurança às instâncias de contêiner, consulte Armazenamento da configuração da instância de contêiner do Amazon ECS no Amazon S3.

Formatos de autenticação

Há dois formatos disponíveis para a autenticação de registro privado, `dockercfg` e `docker`.

Formato de autenticação dockercfg

O formato `dockercfg` usa as informações de autenticação armazenadas no arquivo de configuração criado quando você executa o comando docker login. É possível criar esse arquivo executando docker login no sistema local e inserindo o nome de usuário, a senha e o endereço de e-mail do registro. Também é possível fazer login em uma instância de contêiner e executar o comando daí. Dependendo de sua versão do Docker, esse arquivo é salvo como `~/.dockercfg` ou `~/.docker/config.json`.

```
cat ~/.docker/config.json
```

Saída:

```
{
```

```
  "auths": {
    "https://index.docker.io/v1/": {
      "auth": "zq212MzEXAMPLE7o6T25Dk0i"
    }
  }
}
```

> **Important**
>
> As versões mais recentes do Docker criam um arquivo de configuração como mostrado acima, com um objeto `auths` externo. O agente do Amazon ECS oferece suporte somente aos dados de autenticação `dockercfg` que estão no formato abaixo, sem o objeto `auths`. Se tiver o utilitário jq instalado, será possível extrair esses dados com o comando a seguir: cat ~/.docker/config.json | jq .auths

```
cat ~/.docker/config.json | jq .auths
```

Saída:

```
{
  "https://index.docker.io/v1/": {
    "auth": "zq212MzEXAMPLE7o6T25Dk0i",
    "email": "email@example.com"
  }
}
```

No exemplo acima, as variáveis de ambiente a seguir devem ser adicionadas ao arquivo de variável de ambiente (`/etc/ecs/ecs.config` para a AMI otimizada para o Amazon ECS) que o agente de contêiner do Amazon ECS carrega no runtime. Se você não estiver usando a AMI otimizada para Amazon ECS e estiver iniciando o agente manualmente com docker run, especifique o arquivo de variável de ambiente com a opção `--env-file` *`path_to_env_file`* ao iniciar o agente.

```
ECS_ENGINE_AUTH_TYPE=dockercfg
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"auth":"zq212MzEXAMPLE7o6T25Dk0i","email":"email@example.com"}}
```

É possível configurar vários registros privados com a sintaxe a seguir:

```
ECS_ENGINE_AUTH_TYPE=dockercfg
ECS_ENGINE_AUTH_DATA={"repo.example-01.com":
{"auth":"zq212MzEXAMPLE7o6T25Dk0i","email":"email@example-01.com"},"repo.example-02.com":
{"auth":"fQ172MzEXAMPLEoF7225DU0j","email":"email@example-02.com"}}
```

Formato de autenticação docker

O formato `docker` usa uma representação JSON do servidor de registro com que o agente deve se autenticar. Ele também inclui os parâmetros de autenticação exigidos por esse registro (como nome de usuário, senha e endereço de e-mail dessa conta). Para uma conta do Docker Hub, a representação JSON terá a seguinte aparência:

```
{
  "https://index.docker.io/v1/": {
    "username": "my_name",
    "password": "my_password",
    "email": "email@example.com"
  }
}
```

Nesse exemplo as variáveis de ambiente a seguir devem ser adicionadas ao arquivo de variável de ambiente (`/etc/ecs/ecs.config` para a AMI otimizada para o Amazon ECS) que o agente de contêiner do Amazon ECS carrega no runtime. Se você não estiver usando a AMI otimizada para Amazon ECS e estiver iniciando o agente manualmente com docker run, especifique o arquivo de variável de ambiente com a opção `--env-file` *`path_to_env_file`* ao iniciar o agente.

```
ECS_ENGINE_AUTH_TYPE=docker
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"username":"my_name","password":"my_password","email":"email@example.com"}}
```

É possível configurar vários registros privados com a sintaxe a seguir:

```
ECS_ENGINE_AUTH_TYPE=docker
ECS_ENGINE_AUTH_DATA={"repo.example-01.com":
{"username":"my_name","password":"my_password","email":"email@example-01.com"},"repo.example-02
{"username":"another_name","password":"another_password","email":"email@example-02.com"}}
```

Procedimento

Use o procedimento a seguir para ativar registros privados para as instâncias de contêiner.

Para habilitar registros privados na AMI otimizada para Amazon ECS

1. Faça login em sua instância de contêiner usando SSH.
2. Abra o arquivo `/etc/ecs/ecs.config` e adicione os valores `ECS_ENGINE_AUTH_TYPE` e `ECS_ENGINE_AUTH_DATA` para o seu registro e conta:

   ```
   sudo vi /etc/ecs/ecs.config
   ```

   Este exemplo autentica uma conta de usuário do Docker Hub:

   ```
   ECS_ENGINE_AUTH_TYPE=docker
   ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
   {"username":"my_name","password":"my_password","email":"email@example.com"}}
   ```

3. Verifique se o seu agente usa a variável de ambiente `ECS_DATADIR` para salvar seu estado:

   ```
   docker inspect ecs-agent | grep ECS_DATADIR
   ```

   Saída:

   ```
   "ECS_DATADIR=/data",
   ```

   > **Important**
   >
   > Se o comando anterior não retornar a variável de ambiente `ECS_DATADIR`, você deverá interromper todas as tarefas em execução nessa instância de contêiner antes de interromper o agente. Agentes mais novos com a variável de ambiente `ECS_DATADIR` salvam seu estado, e você pode interrompê-los e iniciá-los enquanto as tarefas são executadas sem problemas. Para obter mais informações, consulte Atualizar o agente de contêiner do Amazon ECS.

4. Interrompa o serviço `ecs`:

   ```
   sudo stop ecs
   ```

   Saída:

```
ecs stop/waiting
```

5. Reinicie o serviço `ecs`.

   - Para a AMI do Amazon Linux 2 otimizada para o Amazon ECS:

     ```
     sudo systemctl restart ecs
     ```

   - Para a AMI do Amazon Linux otimizada para o Amazon ECS:

     ```
     sudo stop ecs && sudo start ecs
     ```

6. (Opcional) É possível verificar se o agente está em execução e consultar algumas informações sobre sua nova instância de contêiner consultando a operação da API de introspecção do agente. Para obter mais informações, consulte the section called “Introspecção de contêiner”.

   ```
   curl http://localhost:51678/v1/metadata
   ```

Limpeza automática de tarefas e de imagens do Amazon ECS

Sempre que uma tarefa é colocada em uma instância de contêiner, o agente de contêiner do Amazon ECS verifica se as imagens referenciadas na tarefa são as mais recentes da etiqueta especificada no repositório. Caso contrário, o comportamento padrão permite que o agente execute pull das imagens de seus respectivos repositórios. Se você atualizar frequentemente as imagens em suas tarefas e serviços, o armazenamento de instâncias do contêiner poderá encher rapidamente com imagens do Docker que você não está mais usando e que provavelmente nunca mais usará. Por exemplo, você pode usar um pipeline de integração contínua e implantação contínua (CI/CD).

> **Note**
>
> O comportamento da extração de imagens do agente do Amazon ECS pode ser personalizado por meio do parâmetro `ECS_IMAGE_PULL_BEHAVIOR`. Para obter mais informações, consulte Configuração do agente de contêiner do Amazon ECS.

Da mesma forma, os contêineres que pertencem a tarefas interrompidas também podem consumir armazenamento de instâncias de contêiner com informações de log, volumes de dados

e outros artefatos. Esses artefatos são úteis para os contêineres de depuração que pararam inesperadamente, mas grande parte desse armazenamento pode ser liberado com segurança após um período.

Por padrão, o agente de contêiner do Amazon ECS limpa automaticamente tarefas e imagens do Docker interrompidas que não estão sendo usadas por qualquer tarefa nas instâncias de contêiner.

> **Note**
>
> O recurso de limpeza de imagem automatizado requer pelo menos a versão 1.13.0 do agente de contêiner do Amazon ECS. Para atualizar o agente para a versão mais recente, consulte Atualizar o agente de contêiner do Amazon ECS.

As seguintes variáveis de configuração de agente estão disponíveis para ajustar sua experiência de limpeza automatizada de tarefas e imagens. Para mais informações sobre como definir essas variáveis nas instâncias de contêiner, consulte Configuração do agente de contêiner do Amazon ECS.

`ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION`

O tempo padrão de espera para excluir contêineres de uma tarefa interrompida. Se o valor for definido para menos de 1 segundo, ele será ignorado. Por padrão, este parâmetro é definido para 3 horas, mas é possível reduzir esse período para até 1 minuto, se isso for necessário para sua aplicação.

> **Note**
>
> O processo de limpeza de imagem não pode excluir uma imagem, contanto que haja um contêiner que faça referência a ele. Depois que os contêineres são removidos, todas as imagens não referenciadas se tornam candidatas à limpeza com base nos parâmetros de configuração da limpeza de imagens.

`ECS_DISABLE_IMAGE_CLEANUP`

Se você definir esta variável como `true`, a limpeza automatizada de imagem será desativada na instância de contêiner e nenhuma imagem será removida automaticamente.

`ECS_IMAGE_CLEANUP_INTERVAL`

Esta variável especifica a frequência em que o processo de limpeza automatizada de imagem deve procurar as imagens a serem excluídas. O padrão é a cada 30 minutos, mas você pode reduzir esse período para até 10 minutos, a fim de remover as imagens com mais frequência.

`ECS_IMAGE_MINIMUM_CLEANUP_AGE`

Essa variável especifica o intervalo de tempo mínimo entre o momento em que uma imagem foi obtida e o momento em que ela pode se tornar candidata à remoção. Ela é usada para evitar a limpeza de imagens que acabaram de ser obtidas. O valor padrão é 1 hora.

`ECS_NUM_IMAGES_DELETE_PER_CYCLE`

Esta variável especifica quantas imagens podem ser removidas em um único ciclo de limpeza. O valor padrão é 5 e o valor mínimo é 1.

Quando o agente de contêiner do Amazon ECS estiver em execução e a limpeza automatizada de imagem não estiver desativada, o agente verificará imagens do Docker que não são referenciadas por contêineres interrompidos ou em execução com uma frequência determinada pela variável `ECS_IMAGE_CLEANUP_INTERVAL`. Se as imagens não utilizadas forem localizadas e forem mais antigas do que o tempo mínimo de limpeza especificado pela variável `ECS_IMAGE_MINIMUM_CLEANUP_AGE`, o agente removerá até o número máximo de imagens especificadas com a variável `ECS_NUM_IMAGES_DELETE_PER_CYCLE`. As imagens referenciadas menos recentes são excluídas primeiro. Depois que as imagens forem removidas, o agente esperará até o intervalo seguinte e repetirá o processo novamente.

Programação de contêineres no Amazon ECS

O Amazon Elastic Container Service (Amazon ECS) é um sistema de estado compartilhado e de simultaneidade otimista que oferece recursos de agendamento flexíveis para as workloads em contêineres. Os programadores do Amazon ECS utilizam as mesmas informações de estado de cluster que as da API do Amazon ECS para tomar decisões apropriadas de posicionamento.

O Amazon ECS oferece um programador de serviços para tarefas e aplicações de longa duração. Ele também fornece a capacidade de executar tarefas autônomas ou programadas para trabalhos em lote ou tarefas de execução única. É possível especificar as estratégias e restrições de posicionamento de tarefas para executar tarefas que melhor atendam às suas necessidades. Por exemplo, você pode especificar se as tarefas são executadas em várias zonas de disponibilidade ou em uma única zona de disponibilidade. E, opcionalmente, você pode integrar tarefas com seus próprios programadores personalizados ou de terceiros.

Opção	Quando usar	Mais informações
Serviço	O programador de serviços é adequado para serviços e aplicações sem estado de longa duração. O programador de serviços, opcionalmente, também se certifica de que as tarefas estejam registradas em um balanceador de carga do Elastic Load Balancing. É possível atualizar os serviços mantidos pelo programador de serviços. Isso pode incluir a implantação de uma nova definição de tarefa ou a alteração do número de tarefas desejadas que estão sendo executadas. Por padrão, o programador de serviços distribui tarefas	Serviços do Amazon ECS

Opção	Quando usar	Mais informações
	em várias zonas de disponibilidade. Entretanto, você pode usar estratégias e limitações de posicionamento de tarefas para personalizar decisões de posicionamento de tarefas.	
Tarefa autônoma	Uma tarefa independente é adequada para processos como trabalhos em lote que realizam o trabalho e, em seguida, são interrompidos. Por exemplo, você pode ter uma chamada de processo `RunTask` quando o trabalho entra em uma fila. A tarefa extrai o trabalho da fila, realiza o trabalho e acaba saindo. Ao usar o `RunTask`, você pode permitir que a estratégia de posicionamento da tarefa padrão distribua tarefas aleatoriamente pelo cluster. Isso minimiza as probabilidades de uma única instância obter um número desproporcional de tarefas.	Tarefas autônomas do Amazon ECS

Opção	Quando usar	Mais informações
Tarefas programadas	Uma tarefa programada é adequada quando há necessidade de executar tarefas em intervalos definidos no seu cluster. É possível usar o Agendador do EventBridge para criar uma programação. É possível executar tarefas para uma operação de backup ou uma varredura de log. O programa que você criar do programador do EventBridge pode executar uma ou mais tarefas em seu cluster em horários especificados. Seu evento programado pode ser definido para um intervalo específico (executar a cada *N* minutos, horas ou dias). Caso contrário, para agendamento mais complicado, você pode usar uma expressão do `cron`.	Uso do Agendador do Amazon EventBridge para programar tarefas do Amazon ECS

Opções de computação

Com o Amazon ECS, é possível especificar a infraestrutura em que as tarefas ou os serviços serão executados. Você pode usar uma estratégia do provedor de capacidade ou um tipo de inicialização.

Para o Fargate, os provedores de capacidade são o Fargate e o Fargate Spot. Para o EC2, o provedor de capacidade é o grupo do Auto Scaling com as instâncias de contêiner registradas.

A estratégia do provedor de capacidade distribui as tarefas entre os provedores de capacidade associados ao seu cluster.

Somente provedores de capacidade que já estejam associados a um cluster e tenham status de `UPDATING` ou `ACTIVE` podem ser usados em uma estratégia de provedor de capacidade. Ao criar um cluster, é possível associar um provedor de capacidade a um cluster.

Em uma estratégia de provedor de capacidade, o valor opcional da base designa o número mínimo de tarefas que serão executadas em um provedor de capacidade especificado. Somente um provedor de capacidade em uma estratégia de provedor de capacidade pode ter uma base definida.

O valor do weight (peso) designa a porcentagem relativa do número total de tarefas inicializadas que usam o provedor de capacidade especificado. Considere o seguinte exemplo. Você tem uma estratégia que contém dois provedores de capacidade e ambos têm um peso de 1. Quando o percentual da base é atingido, as tarefas se dividem igualmente entre os dois provedores de capacidade. Com base nessa mesma lógica, imagine que você especifica um peso de 1 para o capacityProviderA e um peso de 4 para o capacityProviderB. Em seguida, para cada tarefa executada usando capacityProviderA, existem quatro tarefas que usam o capacityProviderB.

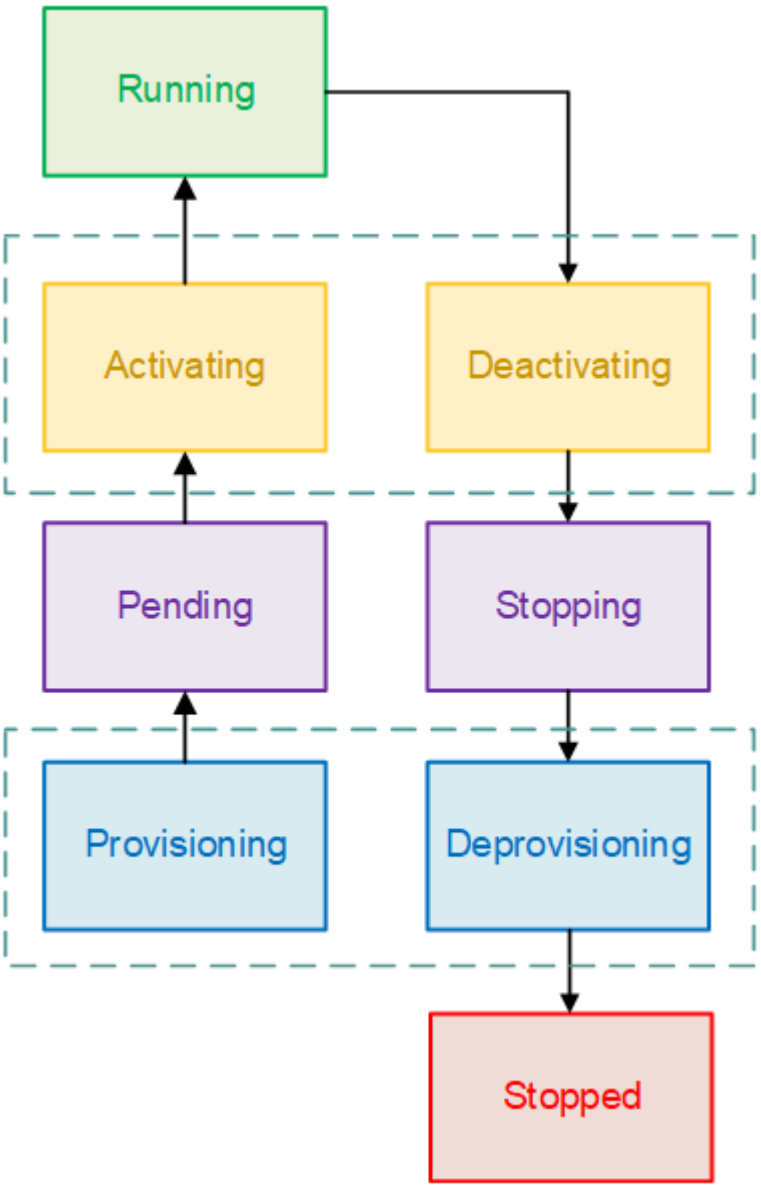
O tipo de inicialização inicia as tarefas diretamente no Fargate ou nas instâncias do Amazon EC2 que foram registradas manualmente em seus clusters.

Ciclo de vida de tarefas do Amazon ECS

Quando uma tarefa é iniciada, manualmente ou como parte de um serviço, ela pode passar por vários estados até terminar sozinha ou ser interrompida manualmente. Algumas tarefas devem ser executadas como trabalhos em lotes que avançam naturalmente de `PENDING` para `RUNNING` até `STOPPED`. Outras tarefas, que podem fazer parte de um serviço, devem continuar sendo executadas indefinidamente ou ser aumentadas e diminuídas conforme necessário.

Quando são solicitadas alterações no status da tarefa, como interromper uma tarefa ou atualizar a contagem desejada de um serviço para aumentá-la ou diminuí-la, o agente de contêiner do Amazon ECS rastreia estas alterações como o último status conhecido (`lastStatus`) da tarefa e o status desejado (`desiredStatus`) da tarefa. É possível ver o último status conhecido e o status desejado de uma tarefa no console ou descrevendo a tarefa com a API ou a AWS CLI.

O fluxograma abaixo mostra o fluxo do ciclo de vida da tarefa.



Estados do ciclo de vida

A seguir estão as descrições de cada um dos estados do ciclo de vida da tarefa.

PROVISIONING

O Amazon ECS precisa executar etapas adicionais antes que a tarefa seja iniciada. Por exemplo, para as tarefas que usam o modo de rede `awsvpc`, a interface de rede elástica precisa ser provisionada.

PENDING

Esse é um estado de transição em que o Amazon ECS está aguardando o agente de contêiner para executar ações adicionais. Uma tarefa permanecerá no estado pendente até que haja recursos disponíveis para ela.

ACTIVATING

Este é um estado de transição onde o Amazon ECS precisa executar etapas adicionais depois que a tarefa é iniciada, mas antes que ela possa fazer a transição para o estado `RUNNING`. Esse é o estado em que o Amazon ECS extrai as imagens do contêiner, cria os contêineres, configura a rede de tarefas, registra os grupos de destino do balanceador de carga e configura a descoberta de serviços.

RUNNING (Em execução)

A tarefa está sendo executada com êxito.

DEACTIVATING

Este é um estado de transição onde o Amazon ECS precisa executar etapas adicionais antes que a tarefa seja interrompida. Por exemplo, para tarefas que fazem parte de um serviço que está configurado para usar grupos de destino de Elastic Load Balancing, o cancelamento do registro do grupo de destino ocorre durante esse estado.

STOPPING

Esse é um estado de transição em que o Amazon ECS está aguardando o agente de contêiner para executar ações adicionais.

Para contêineres Linux, o agente de contêiner enviará o sinal `SIGTERM` para notificar que a aplicação precisa ser finalizada ou desligada, em seguida, enviará um `SIGKILL` após aguardar a duração de `StopTimeout` estabelecida na definição de tarefa.

DEPROVISIONING

O Amazon ECS precisa executar etapas adicionais depois que a tarefa é interrompida, mas antes que ela faça a transição para o estado `STOPPED`. Por exemplo, para as tarefas que usam o modo de rede `awsvpc`, a interface de rede elástica precisa ser desanexada e excluída.

STOPPED

A tarefa foi interrompida com êxito.

Se a tarefa foi interrompida devido a um erro, consulte Visualizar erros de tarefa interrompida do Amazon ECS.

EXCLUÍDA

Esse é um estado de transição quando uma tarefa é interrompida. Esse estado não é exibido no console, mas é exibido no `describe-tasks`.

Como o Amazon ECS posiciona tarefas em instâncias de contêineres

Você pode usar o posicionamento de tarefas para configurar o Amazon ECS para colocar suas tarefas em instâncias de contêiner que atendam a determinados critérios, por exemplo, uma zona de disponibilidade ou um tipo de instância.

Veja estes componentes de posicionamento de tarefas:

- Estratégia de posicionamento de tarefas: o algoritmo que seleciona instâncias de contêiner para posicionamento de tarefas ou encerramento de tarefas. Por exemplo, o Amazon ECS pode selecionar instâncias de contêiner aleatoriamente ou pode selecionar instâncias de contêiner de modo que as tarefas sejam distribuídas uniformemente em um grupo de instâncias.
- Grupo de tarefas: um grupo de tarefas relacionadas, por exemplo, tarefas de banco de dados.
- Restrição de posicionamento de tarefas: essas regras devem ser atendidas para posicionar uma tarefa em uma instância de contêiner. Se a restrição não for atendida, a tarefa não será posicionada e permanecerá no estado `PENDING`. Por exemplo, é possível usar uma restrição para posicionar tarefas somente em um tipo de instância específico.

O Amazon ECS tem algoritmos diferentes para os tipos de execução.

Tipo de inicialização do EC2

Para tarefas que usam o tipo de execução do EC2, o Amazon ECS deve determinar onde posicionar a tarefa com base nos requisitos especificados na definição da tarefa, como CPU e memória. Do mesmo modo, quando você reduz proporcionalmente a contagem de tarefas, o Amazon ECS deve determinar que tarefas serão concluídas. É possível aplicar estratégias e limitações de posicionamento de tarefas para personalizar como o Amazon ECS posiciona e finaliza tarefas.

As estratégias padrão de posicionamento de tarefas dependem de você executar as tarefas manualmente (tarefas autônomas) ou em um serviço. Para tarefas executadas como parte de um serviço Amazon ECS, a estratégia de colocação de tarefas será `spread` usando `attribute:ecs.availability-zone`. Não existe uma restrição padrão de colocação de tarefas para tarefas que não estão em serviços. Para obter mais informações, consulte Programação de contêineres no Amazon ECS.

> **Note**
>
> As estratégias de posicionamento de tarefas têm como base o melhor esforço. O Amazon ECS ainda tenta posicionar tarefas mesmo quando a opção de posicionamento ideal está indisponível. No entanto, as restrições de posicionamento de tarefas são vinculativas e podem impedir o posicionamento delas.

É possível usar estratégias e restrições de posicionamento de tarefas em conjunto. Por exemplo, você pode usar uma estratégia de posicionamento de tarefas e uma restrição de posicionamento de tarefas para distribuir tarefas entre zonas de disponibilidade e tarefas de compartimento com base na memória dentro de cada zona de disponibilidade, mas apenas para instâncias G2.

Quando o Amazon ECS posiciona as tarefas, ele usa o processo a seguir para selecionar as instâncias de contêiner:

1. Identificar instâncias de contêiner que atendam aos requisitos de CPU, GPU, memória e porta na definição de tarefa.
2. Identificar instâncias de contêiner que atendam às limitações de posicionamento de tarefas.
3. Identificar instâncias de contêiner que atendam às estratégias de posicionamento de tarefas.
4. Selecionar instâncias de contêiner para o posicionamento de tarefas.

Tipo de inicialização do Fargate

Não há suporte para estratégias e restrições de posicionamento de tarefas para as tarefas que usam o tipo de inicialização do Fargate. O Fargate fará o possível para distribuir tarefas em zonas de disponibilidade acessíveis. Se o provedor de capacidade incluir o Fargate e o Fargate Spot, o comportamento da distribuição será independente para cada provedor de capacidade.

Uso de estratégias para definir o posicionamento de tarefas do Amazon ECS

Para tarefas que usam o tipo de execução do EC2, o Amazon ECS deve determinar onde posicionar a tarefa com base nos requisitos especificados na definição da tarefa, como CPU e memória. Do mesmo modo, quando você reduz proporcionalmente a contagem de tarefas, o Amazon ECS deve determinar que tarefas serão concluídas. É possível aplicar estratégias e limitações de posicionamento de tarefas para personalizar como o Amazon ECS posiciona e finaliza tarefas.

As estratégias padrão de posicionamento de tarefas dependem de você executar as tarefas manualmente (tarefas autônomas) ou em um serviço. Para tarefas executadas como parte de um serviço Amazon ECS, a estratégia de colocação de tarefas será `spread` usando `attribute:ecs.availability-zone`. Não existe uma restrição padrão de colocação de tarefas para tarefas que não estão em serviços. Para obter mais informações, consulte Programação de contêineres no Amazon ECS.

> **Note**
>
> As estratégias de posicionamento de tarefas têm como base o melhor esforço. O Amazon ECS ainda tenta posicionar tarefas mesmo quando a opção de posicionamento ideal está indisponível. No entanto, as restrições de posicionamento de tarefas são vinculativas e podem impedir o posicionamento delas.

É possível usar estratégias e restrições de posicionamento de tarefas em conjunto. Por exemplo, você pode usar uma estratégia de posicionamento de tarefas e uma restrição de posicionamento de tarefas para distribuir tarefas entre zonas de disponibilidade e tarefas de compartimento com base na memória dentro de cada zona de disponibilidade, mas apenas para instâncias G2.

Quando o Amazon ECS posiciona as tarefas, ele usa o processo a seguir para selecionar as instâncias de contêiner:

1. Identificar instâncias de contêiner que atendam aos requisitos de CPU, GPU, memória e porta na definição de tarefa.
2. Identificar instâncias de contêiner que atendam às limitações de posicionamento de tarefas.
3. Identificar instâncias de contêiner que atendam às estratégias de posicionamento de tarefas.
4. Selecionar instâncias de contêiner para o posicionamento de tarefas.

Você especifica estratégias de posicionamento de tarefas na definição do serviço ou na definição da tarefa usando o parâmetro `placementStrategy`.

```
"placementStrategy": [
    {
        "field": "The field to apply the placement strategy against",
        "type": "The placement strategy to use"
    }
]
```

Você pode especificar as estratégias ao executar uma tarefa (RunTask), criar um serviço (CreateService) ou atualizar um serviço existente (UpdateService).

A tabela a seguir descreve os tipos e campos disponíveis.

type	Valores de campo válidos	
`binpack` As tarefas são colocadas em instâncias de contêiner para deixar a menor quantidade de CPU ou memória não utilizada . Essa estratégia minimiza o número de instâncias de contêiner em uso. Quando essa estratégia é usada e uma ação de redução de escala na horizontal é executada, o Amazon ECS encerra as tarefas. Ele faz isso com base na quantidade de recursos que são deixados na instância de contêiner após o encerramento da tarefa. A instância de contêiner que tiver os recursos mais disponíveis após o encerrame	• cpu • memória	

type	Valores de campo válidos	
nto da tarefa fará com que essa tarefa seja encerrada.		
`random` As tarefas são colocadas aleatoriamente.	Não usado	
`spread` As tarefas são colocadas uniformemente com base no valor especificado. As tarefas de serviço são distribuídas com base nas tarefas desse serviço. Tarefas autônomas são distribuídas com base nas tarefas do mesmo grupo de tarefas. Para obter mais informações sobre grupos de tarefas, consulte Tarefas do Amazon ECS relacionadas a grupos . Quando a estratégia `spread` for usada e uma ação de redução for executada, o Amazon ECS selecionará tarefas para encerrar que mantenham um equilíbrio entre as Zonas de disponibilidade. Dentro de uma zona de disponibilidade, as tarefas são selecionadas aleatoriamente.	• `instanceId` (ou `host`, que tem o mesmo efeito) • qualquer plataforma ou atributo personalizado aplicado a uma instância de contêiner, como `attribute:ecs.availability-zone`	

As estratégias de posicionamento de tarefas também podem ser atualizadas para os serviços existentes. Para obter mais informações, consulte Como o Amazon ECS posiciona tarefas em instâncias de contêineres.

É possível criar uma estratégia de posicionamento de tarefas que use várias estratégias criando matrizes de estratégias na ordem em que você desejar que elas sejam executadas. Por exemplo, se você quiser distribuir tarefas entre zonas de disponibilidade e, em seguida, agrupar tarefas com base na memória dentro de cada zona de disponibilidade, especifique a estratégia de zona de disponibilidade seguida pela estratégia de memória. Para obter exemplos de estratégias, consulte Exemplo de estratégias de posicionamento de tarefas do Amazon ECS.

Exemplo de estratégias de posicionamento de tarefas do Amazon ECS

É possível especificar estratégias de posicionamento de tarefas com as ações a seguir: CreateService, UpdateService e RunTask.

Exemplos

- Distribuir tarefas uniformemente entre zonas de disponibilidade
- Distribuir tarefas uniformemente entre todas as instâncias
- Tarefas de agrupamento baseadas na memória
- Posicionar tarefas aleatoriamente
- Distribui tarefas uniformemente em zonas de disponibilidade e, em seguida, distribui as tarefas uniformemente entre as instâncias em cada zona de disponibilidade
- Distribui tarefas uniformemente em zonas de disponibilidade e, em seguida, agrupa as tarefas com base na memória em cada zona de disponibilidade
- Distribui as tarefas uniformemente entre as instâncias e, em seguida, agrupa as tarefas com base na memória

Distribuir tarefas uniformemente entre zonas de disponibilidade

A estratégia a seguir distribui tarefas uniformemente em zonas de disponibilidade.

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    }
```

```
]
```

Distribuir tarefas uniformemente entre todas as instâncias

A estratégia a seguir distribui tarefas uniformemente em todas as instâncias.

```
"placementStrategy": [
    {
        "field": "instanceId",
        "type": "spread"
    }
]
```

Tarefas de agrupamento baseadas na memória

O pacote de estratégias a seguir agrupa tarefas com base na memória.

```
"placementStrategy": [
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Posicionar tarefas aleatoriamente

A estratégia a seguir posiciona tarefas aleatoriamente.

```
"placementStrategy": [
    {
        "type": "random"
    }
]
```

Distribui tarefas uniformemente em zonas de disponibilidade e, em seguida, distribui as tarefas uniformemente entre as instâncias em cada zona de disponibilidade

A estratégia a seguir distribui tarefas uniformemente em zonas de disponibilidade, em seguida, distribui as tarefas uniformemente entre as instâncias em cada Zona de disponibilidade.

```
"placementStrategy": [
    {
```

```
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    },
    {
        "field": "instanceId",
        "type": "spread"
    }
]
```

Distribui tarefas uniformemente em zonas de disponibilidade e, em seguida, agrupa as tarefas com base na memória em cada zona de disponibilidade

A estratégia a seguir distribui tarefas uniformemente em zonas de disponibilidade, em seguida agrupa as tarefas com base na memória em cada Zona de disponibilidade.

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    },
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Distribui as tarefas uniformemente entre as instâncias e, em seguida, agrupa as tarefas com base na memória

A estratégia a seguir distribui as tarefas uniformemente em todas as instâncias e, em seguida, agrupa as tarefas com base na memória de cada instância.

```
"placementStrategy": [
    {
        "field": "instanceId",
        "type": "spread"
    },
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Tarefas do Amazon ECS relacionadas a grupos

É possível identificar um conjunto de tarefas relacionadas e posicioná-las em um grupo de tarefas. Todas as tarefas com o mesmo nome de grupo de tarefas são consideradas um conjunto quando for usada a estratégia de posicionamento de tarefas `spread`. Por exemplo, suponha que você está executando diferentes aplicações em um cluster, como bases de dados e servidores da Web. Para garantir que os bancos de dados fiquem equilibrados nas zonas de disponibilidade, adicione-os a um grupo de tarefas denominado `databases` e use a estratégia de posicionamento de tarefas `spread`. Para obter mais informações, consulte Uso de estratégias para definir o posicionamento de tarefas do Amazon ECS.

Grupos de tarefas também podem ser usados como uma restrição de posicionamento de tarefas. Quando você especifica um grupo de tarefas na restrição `memberOf`, as tarefas são enviadas somente para instâncias de contêiner que executam tarefas no grupo de tarefas especificado. Para obter um exemplo, consulte Exemplo de restrições de posicionamento de tarefas do Amazon ECS.

Por padrão, as tarefas autônomas usarão o nome da família de definição de tarefa (por exemplo, `family:my-task-definition`) como o nome do grupo de tarefas, se não for especificado um nome de grupo de tarefas personalizado. As tarefas iniciadas como parte de um serviço usam o nome do serviço como nome do grupo de tarefas e não podem ser alteradas.

Os requisitos do grupo de tarefas a seguir se aplicam.

- O nome de um grupo de tarefas deve ser 255 caracteres ou menos.
- Cada tarefa pode estar em exatamente um grupo.
- Após iniciar uma tarefa, você não poderá modificar o grupo de tarefas dela.

Definição de quais instâncias de contêiner o Amazon ECS usa em tarefas

Uma restrição no posicionamento de tarefas é uma regra sobre uma instância de contêiner que o Amazon ECS usa para determinar se a tarefa tem permissão para ser executada na instância. Pelo menos uma instância de contêiner deve corresponder à restrição. Se não houver instâncias que correspondam à restrição, a tarefa permanecerá em um estado `PENDING`. Ao criar um novo serviço ou atualizar um existente, é possível especificar restrições de posicionamento de tarefas para as tarefas do serviço.

Você pode especificar restrições de posicionamento de tarefas na definição do serviço, na definição de tarefa ou na tarefa usando o parâmetro `placementConstraint`.

```
"placementConstraints": [
    {
        "expression": "The expression that defines the task placement constraints",
        "type": "The placement constraint type to use"
    }
]
```

A tabela a seguir descreve como usar os parâmetros.

Constraint type	Pode ser especificado quando
`distinctInstance` Coloque cada tarefa ativa em uma instância de contêiner diferente. O Amazon ECS analisa o status desejado das tarefas para a colocação da tarefa. Por exemplo, se o status desejado da tarefa existente for `STOPPED` (mas o último não for), uma nova tarefa de entrada poderá ser colocada na mesma instância, apesar da restrição de colocação `distinctInstance` . Portanto, você pode ver duas tarefas com o último status de `RUNNING` na mesma instância. **Important** Recomendamos que os clientes que buscam um forte isolament	• Execução de uma tarefa RunTask • Criação de um serviço CreateService,

Constraint type	Pode ser especificado quando	
o para as tarefas usem o Fargate. O Fargate executa cada tarefa em um ambiente de virtualiz ação de hardware. Isso garante que essas workloads em contêineres não compartilhem interface s de rede, armazenam ento temporário do Fargate, CPU ou memória com outras tarefas. Para obter mais informações, consulte Security Overview of AWS Fargate.		
`memberOf` Posicionar tarefas em instâncias de contêiner que atendam a uma expressão.	• Execução de uma tarefa RunTask • Criação de um serviço CreateService, • Criação de uma definição de tarefa RegisterTaskDefini tion • Criação de revisão de uma definição de tarefa RegisterTaskDefinition • Atualização de um serviço UpdateService	

Ao usar o tipo de restrição `memberOf`, você pode criar uma expressão usando a linguagem de consulta do cluster, que define as instâncias de contêiner nas quais o Amazon ECS pode posicionar tarefas. A expressão é uma forma de agrupar as instâncias de contêiner por atributos. A expressão vai no parâmetro `expression` de `placementConstraint`.

Atributos de instância de contêiner do Amazon ECS

É possível adicionar metadados personalizados, conhecidos como atributos, às instâncias de contêiner. Cada atributo tem um nome e um valor de string opcional. É possível usar os atributos integrados fornecidos pelo Amazon ECS ou definir atributos personalizados.

As seções a seguir contêm exemplos de atributos internos, opcionais e personalizados.

Atributos integrados

O Amazon ECS aplica automaticamente os seguintes atributos às instâncias de contêiner.

`ecs.ami-id`

A ID do AMI usado para executar a instância. Um exemplo de valor do atributo é `ami-1234abcd`.

`ecs.availability-zone`

A zona de disponibilidade para a instância. Um exemplo de valor do atributo é `us-east-1a`.

`ecs.instance-type`

O tipo de instância para a instância. Um exemplo de valor do atributo é `g2.2xlarge`.

`ecs.os-type`

O sistema operacional para a instância. Os valores possíveis para esse atributo são `linux` e `windows`.

`ecs.os-family`

A versão do sistema operacional para a instância.

Para instâncias de Linux, o valor válido é `LINUX`. Para instâncias de Windows, o ECS define o valor no formato `WINDOWS_SERVER_<OS_Release>_<FULL or CORE>`. Os valores válidos são `WINDOWS_SERVER_2022_FULL`, `WINDOWS_SERVER_2022_CORE`, `WINDOWS_SERVER_20H2_CORE`, `WINDOWS_SERVER_2019_FULL`, `WINDOWS_SERVER_2019_CORE` e `WINDOWS_SERVER_2016_FULL`.

Isso é importante para contêineres do Windows e Windows containers on AWS Fargate, pois porque a versão do sistema operacional de cada contêiner do Windows deve corresponder à do host. Se a versão Windows da imagem do contêiner for diferente da do host, o contêiner não será iniciado. Para obter mais informações, consulte Compatibilidade de versão do contêiner do Windows no site de documentação da Microsoft.

Caso seu cluster execute várias versões do Windows, é possível garantir que uma tarefa seja colocada em uma instância do EC2 em execução na mesma versão usando a restrição de posicionamento: `memberOf(attribute:ecs.os-family == WINDOWS_SERVER_<OS_Release>_<FULL or CORE>)`. Para obter mais informações, consulte the section called “Recuperação de metadados da AMI do Windows otimizada para o Amazon ECS”.

`ecs.cpu-architecture`

A arquitetura da CPU para a instância. São exemplos de valores do atributo `x86_64` e `arm64`.

`ecs.vpc-id`

A VPC na qual a instância foi executada. Um exemplo de valor do atributo é `vpc-1234abcd`.

`ecs.subnet-id`

A sub-rede que a instância está usando. Um exemplo de valor do atributo é `subnet-1234abcd`.

Atributos opcionais

O Amazon ECS pode adicionar os seguintes atributos às instâncias de contêiner.

`ecs.awsvpc-trunk-id`

Se esse atributo existir, a instância terá uma interface de rede de tronco. Para obter mais informações, consulte Aumento das interfaces de rede de instâncias de contêiner do Linux no Amazon ECS.

`ecs.outpost-arn`

Se esse atributo existir, ele conterá o nome de recurso da Amazon (ARN) do Outpost. Para obter mais informações, consulte the section called “Amazon Elastic Container Service no AWS Outposts”.

`ecs.capability.external`

Se esse atributo existir, a instância será identificada como uma instância externa. Para obter mais informações, consulte Clusters do Amazon ECS para o tipo de inicialização externa.

Atributos personalizados

É possível aplicar atributos personalizados às instâncias de contêiner. Por exemplo, você pode definir um atributo com o nome “pilha” e um valor de “prod”.

Ao especificar atributos personalizados, considere o descrito a seguir.

- O `name` deve conter entre 1 e 128 caracteres e o nome pode conter letras (maiúsculas e minúsculas), números, hifens, sublinhados, barras “/”, barras “\” ou pontos.
- O `value` deve conter entre 1 e 128 caracteres e pode conter letras (maiúsculas e minúsculas), números, hifens, sublinhados, pontos, arrobas (@), barras “/”, barras “\” dois pontos (:) ou espaços. O valor não pode conter espaço em branco no começo ou no fim.

Criação de expressões para definir instâncias de contêiner em tarefas do Amazon ECS

Consultas ao cluster são expressões que permitem agrupar objetos. Por exemplo, você pode agrupar instâncias de contêiner por atributos, como Zona de disponibilidade, tipo de instância ou metadados personalizados. Para obter mais informações, consulte Atributos de instância de contêiner do Amazon ECS.

Depois de definir um grupo de instâncias de contêiner, você pode personalizar o Amazon ECS para posicionar tarefas em instâncias de contêiner baseadas no grupo. Para ter mais informações, consulte Execução de uma aplicação como uma tarefa do Amazon ECS e Criação de uma implantação de atualização contínua do Amazon ECS. Também é possível aplicar um filtro de grupo ao listar instâncias de contêiner.

Sintaxe da expressão

As expressões têm a seguinte sintaxe:

```
subject operator [argument]
```

Sujeito

O atributo ou o campo a ser avaliado.

`agentConnected`

Selecione instâncias de contêiner pelo status da conexão do agente de contêiner do Amazon ECS. É possível usar esse filtro para pesquisar instâncias com agentes de contêiner desconectados.

Operadores válidos: equals (==), not_equals (!=), in, not_in (!in), matches (=~), not_matches (!~)

`agentVersion`

Selecione instâncias de contêiner pela versão do agente de contêiner do Amazon ECS. É possível usar esse filtro para encontrar instâncias que estejam executando versões desatualizadas do agente de contêiner do Amazon ECS.

Operadores válidos: equals (==), not_equals (!=), greater_than (>), greater_than_equal (>=), less_than (<), less_than_equal (<=)

`attribute:`*`attribute-name`*

Selecione instâncias de contêiner por atributo. Para obter mais informações, consulte Atributos de instância de contêiner do Amazon ECS.

`ec2InstanceId`

Selecione instâncias de contêiner pelo ID da instância do Amazon EC2.

Operadores válidos: equals (==), not_equals (!=), in, not_in (!in), matches (=~), not_matches (!~)

`registeredAt`

Selecione instâncias de contêiner na data de registro da instância de contêiner. É possível usar esse filtro para encontrar instâncias recém-registradas ou instâncias muito antigas.

Operadores válidos: equals (==), not_equals (!=), greater_than (>), greater_than_equal (>=), less_than (<), less_than_equal (<=)

Formatos de data válidos: 2018-06-18T22:28:28+00:00, 2018-06-18T22:28:28Z, 2018-06-18T22:28:28, 2018-06-18

`runningTasksCount`

Selecione instâncias de contêiner pelo número de tarefas em execução. É possível usar esse filtro para encontrar instâncias vazias ou quase vazias (algumas tarefas em execução).

Operadores válidos: equals (==), not_equals (!=), greater_than (>), greater_than_equal (>=), less_than (<), less_than_equal (<=)

`task:group`

Selecione instâncias de contêiner por grupo de tarefas. Para obter mais informações, consulte Tarefas do Amazon ECS relacionadas a grupos .

Operador

O operador de comparação. Os operadores a seguir são aceitos.

Operador	Descrição
==, equals	Igualdade de strings
!=, not_equals	Desigualdade de strings
>, greater_than	Maior que
>=, greater_than_equal	Maior ou igual a
<, less_than	Menor que
<=, less_than_equal	Menor ou igual a
exists	O sujeito existe
!exists, not_exists	O sujeito não existe
in	Valor na lista de argumentos
!in, not_in	Valor fora da lista de argumentos
=~, matches	Correspondência de padrão
!~, not_matches	Divergência de padrão

> **Note**
>
> Uma expressão individual não pode conter parênteses. No entanto, os parênteses podem ser usados para especificar precedência em expressões compostas.

Argumento

Para muitos operadores, o argumento é um valor literal.

Os operadores `in` e `not_in` esperam uma lista de argumentos como argumento. Você especifica uma lista de argumentos da seguinte forma:

```
[argument1, argument2, ..., argumentN]
```

Os operadores matches e not_matches um argumento que se conforme à sintaxe de expressões regulares em Java. Para mais informações, consulte java.util.regex.Pattern.

Expressões compostas

É possível combinar expressões usando os operadores booleanos a seguir:

- &&, e
- ||, ou
- !, not

É possível especificar a precedência usando parênteses:

```
(expression1 or expression2) and expression3
```

Exemplo de expressões

Veja a seguir exemplos de expressões.

Exemplo: igualdade de strings

A expressão a seguir seleciona instâncias com o tipo de instância especificado.

```
attribute:ecs.instance-type == t2.small
```

Exemplo: lista de argumentos

A expressão a seguir seleciona instâncias na Zona de disponibilidade us-east-1a ou us-east-1b.

```
attribute:ecs.availability-zone in [us-east-1a, us-east-1b]
```

Exemplo: expressão composta

A expressão a seguir seleciona as instâncias G2 que não estiverem na zona de disponibilidade us-east-1d.

```
attribute:ecs.instance-type =~ g2.* and attribute:ecs.availability-zone != us-east-1d
```

Exemplo: afinidade de tarefas

A expressão a seguir seleciona as instâncias que são tarefas de hospedagem no grupo `service:production`.

```
task:group == service:production
```

Exemplo: antiafinidade de tarefas

A expressão a seguir seleciona as instâncias que não são tarefas de hospedagem no grupo de bancos de dados.

```
not(task:group == database)
```

Exemplo: contagem de tarefas em execução

A expressão a seguir seleciona as instâncias que estão executando apenas uma tarefa.

```
runningTasksCount == 1
```

Exemplo: versão do agente de contêiner do Amazon ECS

A expressão a seguir seleciona instâncias que estão executando uma versão do agente de contêiner abaixo de 1.14.5.

```
agentVersion < 1.14.5
```

Exemplo: tempo de registro da instância

A expressão a seguir seleciona instâncias que foram registradas antes de 13 de fevereiro de 2018.

```
registeredAt < 2018-02-13
```

Exemplo: ID de instância do Amazon EC2

A expressão a seguir seleciona instâncias com os IDs de instância do Amazon EC2 a seguir.

```
ec2InstanceId in ['i-abcd1234', 'i-wxyx7890']
```

Exemplo de restrições de posicionamento de tarefas do Amazon ECS

Os exemplos de restrição de colocação da tarefa são os seguintes.

Esse exemplo usa a restrição `memberOf` para colocar tarefas em instâncias t2. Ele pode ser especificado com as seguintes ações: CreateService, UpdateService, RegisterTaskDefinition e RunTask.

```
"placementConstraints": [
    {
        "expression": "attribute:ecs.instance-type =~ t2.*",
        "type": "memberOf"
    }
]
```

O exemplo usa a restrição `memberOf` para posicionar tarefas de réplica em instâncias com tarefas no grupo de tarefas `daemon-service` de serviço de daemon, respeitando quaisquer estratégias de posicionamento de tarefas que também estejam especificadas. Essa restrição garante que as tarefas do serviço de daemon sejam colocadas na instância do EC2 antes das tarefas do serviço de réplica.

Substitua `daemon-service` pelo nome do serviço de daemon.

```
"placementConstraints": [
    {
        "expression": "task:group == service:daemon-service",
        "type": "memberOf"
    }
]
```

O exemplo usa a restrição `memberOf` para posicionar tarefas em instâncias com outras tarefas no grupo de tarefas `databases`, respeitando quaisquer estratégias de posicionamento de tarefas que também estejam especificadas. Para obter mais informações sobre grupos de tarefas, consulte Tarefas do Amazon ECS relacionadas a grupos . Ele pode ser especificado com as seguintes ações: CreateService, UpdateService, RegisterTaskDefinition e RunTask.

```
"placementConstraints": [
    {
        "expression": "task:group == databases",
        "type": "memberOf"
    }
]
```

A limitação `distinctInstance` posiciona cada tarefa no grupo em uma instância diferente. Ele pode ser especificado com as seguintes ações: CreateService, UpdateService e RunTask

O Amazon ECS analisa o status desejado das tarefas para a colocação da tarefa. Por exemplo, se o status desejado da tarefa existente for `STOPPED` (mas o último não for), uma nova tarefa de entrada poderá ser colocada na mesma instância, apesar da restrição de colocação `distinctInstance`. Portanto, você pode ver duas tarefas com o último status de `RUNNING` na mesma instância.

```
"placementConstraints": [
    {
        "type": "distinctInstance"
    }
]
```

Tarefas autônomas do Amazon ECS

Você pode executar a aplicação como tarefa quando tiver uma aplicação que executa algum trabalho e depois interrompe, por exemplo, um processo em lote. Se quiser executar uma tarefa uma vez, você pode usar o console, a AWS CLI, as APIs ou os SDKs.

Se precisar executar a aplicação em um cronograma baseado em intervalos, em cron ou em um cronograma único, você pode criá-lo usando o Agendador do EventBridge.

Fluxo de trabalho de tarefas

Ao executar tarefas do Amazon ECS (tarefas autônomas ou de serviços do Amazon ECS), uma tarefa é criada e inicialmente movida para o estado `PROVISIONING`. Quando uma tarefa está no

estado `PROVISIONING`, nem a tarefa nem os contêineres existem porque o Amazon ECS precisa encontrar capacidade computacional para posicionar a tarefa.

O Amazon ECS seleciona a capacidade computacional apropriada para a tarefa com base no tipo de execução ou na configuração do provedor de capacidade. Você pode usar provedores de capacidade e estratégias de provedores de capacidade com os tipos de execução do Fargate e do Amazon EC2. Com o Fargate, você não precisa pensar em provisionar, configurar e escalar a capacidade do cluster. O Fargate cuida de todo o gerenciamento da infraestrutura das tarefas. No tipo de execução do EC2, você pode gerenciar a capacidade de cluster registrando instâncias do Amazon EC2 no cluster ou usar o ajuste de escala automático do cluster para simplificar o gerenciamento da capacidade computacional. O ajuste de escala automático do cluster se encarrega de escalar dinamicamente a capacidade do cluster, para que você possa se concentrar na execução de tarefas. O Amazon ECS determina onde posicionar a tarefa, com base nos requisitos especificados na definição de tarefa, como CPU e memória, bem como nas restrições e estratégias de posicionamento. Para obter mais informações, consulte Como o Amazon ECS posiciona tarefas em instâncias de contêineres.

Se você usar um provedor de capacidade com ajuste de escala gerenciado habilitado, as tarefas que não puderem ser iniciadas devido à falta de capacidade computacional serão transferidas para o estado `PROVISIONING` em vez de falharem imediatamente. Depois de encontrar a capacidade de posicionamento da tarefa, o Amazon ECS provisiona os anexos necessários, por exemplo, as interfaces de rede elástica (ENI) para tarefas no modo `awsvpc`. A ferramenta usa o agente de contêiner do Amazon ECS para extrair as imagens do contêiner e iniciá-los. Depois que o provisionamento é concluído e os contêineres relevantes são executados, o Amazon ECS muda a tarefa para o estado `RUNNING`. Para mais informações sobre os estados das tarefas, consulte Ciclo de vida de tarefas do Amazon ECS.

Execução de uma aplicação como uma tarefa do Amazon ECS

Você pode criar uma tarefa para um processo único usando o AWS Management Console.

Para criar uma tarefa autônoma (AWS Management Console)

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. O console do Amazon ECS permite criar uma tarefa autônoma na página de detalhes do cluster ou na lista de revisão da definição de tarefas. Use as etapas a seguir para criar a tarefa autônoma, dependendo da página de recursos escolhida.

Para iniciar um serviço em	Etapas
Página de detalhes do cluster	a. Na página Clusters, selecione o cluster no qual o serviço será criado. b. Na guia Tarefas, escolha Executar tarefa.
A página de revisão da definição de tarefa	a. Na página Definições de tarefas, escolha a família de definições de tarefas para exibir as revisões dessa família. b. Selecione a revisão que deseja usar. c. No menu Implantar, escolha Executar tarefa.

3. Em Cluster existente, escolha o cluster.

   Escolha Criar cluster para executar a tarefa em um novo cluster

4. Escolha como suas tarefas serão distribuídas em toda a infraestrutura do cluster. Em Configuração de computação escolha a sua opção. Para usar uma estratégia de provedor de capacidade, configure os provedores de capacidade no nível do cluster.

   Caso não tenha configurado o cluster para usar um provedor de capacidade, use um tipo de execução.

Método de distribuição	Etapas
Estratégia de provedor de capacidade	a. Na seção Compute options (Opções de computação), selecione Capacity provider strategy (Estratégia do provedor de capacidade).

Método de distribuição	Etapas
	b. Escolha uma estratégia: • Para usar a estratégia de provedor de capacidade padrão do cluster, escolha Use cluster default (Usar padrão de cluster). • Se o cluster não tiver uma estratégia de provedor de capacidade padrão ou, para usar uma estratégia personalizada, escolha Use custom (Usar personalizada), Add capacity provider strategy (Adicionar estratégia de provedor de capacidade) e defina a estratégia de provedor de capacidade personalizada especificando um Provedor de capacidade, uma Base e um Peso. **Note** Para usar um provedor de capacidade em uma estratégia, o provedor de capacidade deve

Método de distribuição	Etapas
	estar associado ao cluster.
Tipo de inicialização	a. Na seção Compute options (Opções de computação), selecione Launch type (Tipo de inicialização). b. Em Launch type (Tipo de inicialização), escolha um tipo de inicialização. c. (Opcional) Quando o tipo de inicialização do Fargate for especific ado, em Platform version (Versão da plataforma), especifique a versão de plataforma a ser usada. Se não for especificada uma versão da plataform a, a versão da plataforma `LATEST` será usada.

5. Em Configuração de implantação, faça o seguinte:

   a. Em Definição de tarefa, insira a definição de tarefa.

   > **Important**
   >
   > O console valida a seleção para garantir que a família de definição de tarefa e a revisão selecionadas sejam compatíveis com a configuração de computação definida.

   b. Em Desired tasks (Tarefas desejadas), insira o número de tarefas que serão iniciadas.

   c. Em Grupo de tarefas, digite o nome do grupo de tarefas.

6. Se a definição de tarefa usar o modo de rede `awsvpc`, expanda Networking (Redes). Use as etapas a seguir para especificar uma configuração personalizada.

   a. Em VPC, selecione a VPC a ser usada.

   b. Em Subnets (Sub-redes), selecione uma ou mais sub-redes na VPC que o programador de tarefas levará em consideração ao posicionar as tarefas.

   c. Em Grupo de segurança, é possível escolher um grupo de segurança existente ou criar outro. Para usar um grupo de segurança existente, escolha o grupo de segurança e vá para a próxima etapa. Para criar um novo grupo de segurança, escolha Create a new security group (Criar um novo grupo de segurança). Você deve especificar o nome de um grupo de segurança, uma descrição e, em seguida, adicionar uma ou mais regras de entrada para o grupo de segurança.

   d. Em Public IP (IP público), escolha se um endereço IP público deve ou não ser atribuído automaticamente à interface de rede elástica (ENI) da tarefa.

      Tarefas do AWS Fargate podem receber um endereço IP público quando são executadas em uma sub-rede pública para terem uma rota para a Internet. As tarefas do EC2 não podem ser atribuídas a um IP público usando esse campo. Para obter mais informações, consulte as opções de rede de tarefas do Amazon ECS para o tipo de lançamento do Fargate e Alocar uma interface de rede para uma tarefa do Amazon ECS.

7. Caso a tarefa use um volume de dados compatível com a configuração na implantação, você pode configurar o volume expandindo Volume.

   O nome e o tipo do volume são configurados ao criar uma revisão de definição de tarefa e não podem ser alterados ao executar uma tarefa autônoma. Para atualizar o nome e o tipo do volume, você deve criar uma revisão de definição de tarefa e executar uma tarefa usando a nova revisão.

Para configurar esse tipo de volume	Faça o seguinte	
Amazon EBS	a. Em Tipo de volume do EBS, escolha o tipo de volume do EBS que você deseja anexar à tarefa.	

Para configurar esse tipo de volume	Faça o seguinte	
	b. Em Tamanho (GiB), insira um valor válido para o tamanho do volume em gibibytes (GiB). Você pode especificar um tamanho de volume mínimo de 1 GiB e máximo de 16.384 GiB. Esse valor é obrigatório, a menos que você forneça um ID de snapshot. c. Em IOPS, insira o número máximo de operações de entrada e saída (IOPS) que o volume deve oferecer. Esse valor é configurável somente em tipos de volumes `io1`, `io2` e `gp3`. d. Em Throughput (MiB/s), insira o throughput que o volume deve fornecer, em mebibytes por segundo (MiBps ou MiB/s). Esse valor é configurável somente no tipo de volume `gp3`. e. Em ID do snapshot, escolha um snapshot de volume existente do Amazon EBS ou insira o ARN de um snapshot se quiser criar um volume	

Para configurar esse tipo de volume	Faça o seguinte	
	usando um snapshot. Você também pode criar um volume vazio sem escolher ou inserir um ID de snapshot. f. Se você especificar um ID de snapshot, poderá especificar uma Taxa de inicialização do volume (MiB/s). Insira um valor entre 100 e 300, em MiB/s, que determinará a rapidez com que os dados são carregados do snapshot especificado usando o ID do snapshot para a criação do volume. g. Em Política de encerramento, desmarque a caixa de seleção se quiser que o volume configurado para anexação à tarefa seja preservado após o encerramento dela. Por padrão, os volumes do EBS anexados às tarefas são excluídos quando a tarefa é encerrada. h. Em Tipo de sistema de arquivos, escolha o tipo de sistema de arquivos que será usado para armazenamento e	

Para configurar esse tipo de volume	Faça o seguinte	
	recuperação de dados no volume. Você pode escolher o sistema operacional padrão ou um tipo específico de sistema de arquivos. O padrão no Linux é XFS. Em volumes criados de um snapshot, você deve especificar o mesmo tipo de sistema de arquivos que o volume estava usando quando o snapshot foi criado. Se houver uma incompatibilidade no tipo do sistema de arquivos, a tarefa não será iniciada. i. Em Perfil de infraestrutura, escolha um perfil do IAM com as permissões necessárias para permitir que o Amazon ECS gerencie volumes do Amazon EBS em tarefas. Você pode anexar a política gerenciada `AmazonECSInfrastructureRolePolicyForVolumes` ao perfil ou usar a política como guia para criar e anexar sua própria política com permissões que atendam às suas	

Para configurar esse tipo de volume	Faça o seguinte	
	necessidades específicas. Para obter mais informaçõ es sobre as permissõe s necessárias, consulte Perfil do IAM de infraestr utura do Amazon ECS. j. Em Criptografia, escolha Padrão se deseja usar a criptografia do Amazon EBS por meio das configurações padrão. Se a conta tiver a Criptografia por padrão configurada, o volume será criptogra fado com a chave do AWS Key Management Service (AWS KMS) especific ada na configuração. Caso escolha Padrão e a criptografia padrão do Amazon EBS não estiver ativada, o volume não será criptografado. Caso escolha Personali zada, você pode especific ar uma AWS KMS key de sua escolha para criptogra fia de volume. Caso escolha Nenhum, o volume não será criptogra fado, a menos que você tenha a criptografia por	

Para configurar esse tipo de volume	Faça o seguinte	
	padrão configurada ou crie um volume usando um snapshot criptografado. k. Caso tenha escolhido Personalizada em Criptografia, você deve especificar a AWS KMS key que deseja usar. Em Chave do KMS, escolha uma AWS KMS key ou insira um ARN de chave. Se você optar por criptografar o volume usando uma chave simétrica gerenciada pelo cliente, verifique se tem as permissões corretas definidas na política do AWS KMS key. Para obter mais informações, consulte Data encryption for Amazon EBS volumes. l. (Opcional) Em Tags, você pode adicionar tags ao volume do Amazon EBS propagando tags na definição de tarefa ou fornecendo suas próprias tags. Se quiser propagar tags na definição de tarefa, escolha Definição de	

Para configurar esse tipo de volume	Faça o seguinte	
	tarefa em Propagar tags em. Caso escolha Não propagar ou se não selecionar um valor, as tags não são propagadas. Se quiser fornecer suas próprias tags, escolha Adicionar tag e forneça a chave e o valor de cada tag adicionada. Para obter mais informaçõ es sobre marcação de volumes do Amazon EBS, consulte Tagging Amazon EBS volumes.	

8. (Opcional) Para usar uma estratégia de posicionamento de tarefas diferente da padrão, expanda Task Placement (Posicionamento de tarefas) e escolha uma das opções a seguir.

   Para obter mais informações, consulte Como o Amazon ECS posiciona tarefas em instâncias de contêineres.

   - Distribuição balanceada de AZ: distribua tarefas por zonas de disponibilidade e entre instâncias de contêiner na zona de disponibilidade.
   - BinPack balanceado de AZ: distribua tarefas por zonas de disponibilidade e entre instâncias de contêiner com a menor memória disponível.
   - BinPack: distribua tarefas com base na menor quantidade disponível de CPU ou memória.
   - Uma tarefa por host: posicione, no máximo, uma tarefa do serviço em cada instância de contêiner.
   - Personalizado: defina sua própria estratégia de posicionamento de tarefas.

Se você escolheu Custom (Personalizado), defina o algoritmo de modo a posicionar as tarefas e as regras que serão consideradas durante o posicionamento de tarefas.

- Em Strategy (Estratégia), em Type (Tipo) e Field (Campo), escolha o algoritmo e a entidade a serem usados com o algoritmo.

  É possível adicionar no máximo cinco estratégias.

- Em Restrição, para Tipo e Expressão, escolha a regra e o atributo para a restrição.

  Por exemplo, para definir a restrição de modo a posicionar tarefas em instâncias T2, em Expression (Expressão), insira attribute:ecs.instance-type =~ t2.*.

  É possível adicionar no máximo dez restrições.

9. (Opcional) Para substituir o perfil do IAM da tarefa ou o perfil de execução da tarefa especificado na definição de tarefa, expanda Task overrides (Substituições de tarefa) e realize as seguintes etapas:

   a. Em Perfil da tarefa, escolha um perfil do IAM para essa tarefa. Para obter mais informações, consulte Perfil do IAM para tarefas do Amazon ECS.

      Somente as funções com o relacionamento de confiança `ecs-tasks.amazonaws.com` são exibidas. Para obter instruções sobre a criação de uma função do IAM para as tarefas, consulte Criar o perfil do IAM de tarefa.

   b. Em Perfil de execução da tarefa, escolha um perfil de execução da tarefa. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.

10. (Opcional) Para substituir os comandos do contêiner e as variáveis de ambiente, expanda Container Overrides (Substituições de contêiner) e expanda o contêiner.

    - Para enviar um comando para o contêiner que não seja o comando de definição de tarefa, em Substituição de comando, insira o comando do Docker.
    - Para adicionar uma variável de ambiente, use Add environment variable (Adicionar variável de ambiente). Em Key (Chave), insira o nome da variável de ambiente. Em Value (Valor), insira um valor de string para seu valor de ambiente (sem as aspas duplas (" ")).

      A AWS delimita a string com aspas duplas (" ") e a transmite ao contêiner no seguinte formato:

```
MY_ENV_VAR="This variable contains a string."
```

11. (Opcional) Para ajudar a identificar a tarefa, expanda a seção Tags (Etiquetas) e configure suas etiquetas.

    Para que o Amazon ECS marque automaticamente todas as tarefas recém-iniciadas com o nome do cluster e as tags de definição de tarefa, selecione Turn on Amazon ECS managed tags (Ativar tags gerenciadas pelo Amazon ECS) e depois Definições de tarefas.

    Adicione ou remova uma tag.

    - [Adicionar uma etiqueta] Escolha Add tag (Adicionar etiqueta) e faça o seguinte:
      - Em Chave, insira o nome da chave.
      - Em Valor insira o valor da chave.
    - [Remover uma tag] Ao lado da tag, escolha Remove tag (Remover tag).
12. Escolha Criar.

Uso do Agendador do Amazon EventBridge para programar tarefas do Amazon ECS

O Agendador do EventBridge é um programador com tecnologia sem servidor que permite criar, executar e gerenciar tarefas de um serviço gerenciado central. Ele fornece uma funcionalidade de programação única e recorrente, independente das regras e barramentos de eventos. O Agendador do EventBridge é altamente personalizável e oferece escalabilidade aprimorada em relação às regras programadas do EventBridge, com um conjunto mais amplo de operações da API de destino e serviços da AWS. O Agendador do EventBridge fornece as programações a seguir que podem ser configuradas para suas tarefas no console do Agendador do EventBridge:

- Baseada em taxa
- Baseado em cron

  É possível configurar programações baseadas em cron em qualquer fuso horário.
- Programações únicas

  É possível configurar programações únicas em qualquer fuso horário.

Você pode programar o Amazon ECS usando o Agendador do Amazon EventBridge.

Embora seja possível criar uma tarefa programada no console do Amazon ECS, no momento, o console do Agendador do EventBridge fornece mais funcionalidades.

Conclua as etapas a seguir antes de programar uma tarefa:

1. Use o console da VPC para obter os IDs de sub-rede em que as tarefas são executadas e os IDs do grupo de segurança para as sub-redes. Para obter mais informações, consulte Sub-redes para sua VPC e Controlar o tráfego para seus recursos da AWS usando grupos de segurança no Guia do usuário da Amazon VPC.
2. Configure o perfil de execução do Agendador do EventBridge. Para obter mais informações, consulte Configurar o perfil de execução no Guia do usuário do Agendador do Amazon EventBridge.
3. Se você desejar usar uma estratégia de provedor de capacidade para executar a tarefa, você deverá ter um provedor de capacidade associado ao cluster.

Para criar uma nova programação usando o console

1. Abra o console do Agendador do Amazon EventBridge em https://console.aws.amazon.com/scheduler/home.
2. Na página Programações, clique em Criar programação.
3. Na página Especificar detalhes da programação, na seção Nome e descrição da programação, faça o seguinte:

   a. Em Nome da programação, insira um nome para a programação. Por exemplo, .**`MyTestSchedule`**

   b. (Opcional) Em Descrição, insira a descrição da programação. Por exemplo, .**`TestSchedule`**

   c. Em Grupo de agendamentos, escolha um grupo de agendamento. Se você não tiver um grupo, escolha padrão. Para criar um grupo de programação, escolha criar sua própria programação.

   Para adicionar tags a grupos de programação, você usa os grupos de programação.
4. Escolha as opções de programação.

Ocorrência	Fazer isso...	
Programação única A programação única invoca o destino somente uma vez na data e hora que você especificar.	Em Data e hora, faça o seguinte: • Insira uma data válida no formato `YYYY/MM/DD` . • Insira um carimbo de data/hora no formato de 24 horas `hh:mm`. • Em Fuso horário, escolha o fuso horário.	

Ocorrência	Fazer isso...
Programação recorrente A programação recorrente e invoca o destino em uma taxa especificada por você usando uma expressão cron ou rate.	a. Em Tipo de programação, siga um dos procedimentos a seguir. • Para usar uma expressão cron para definir a programação, escolha Programação baseada em cron e insira a expressão cron. • Para usar uma expressão rate para definir a programação, escolha Programação baseada em rate e insira a expressão rate. Para obter mais informações sobre as expressões cron e rate, consulte Tipos de programação no Agendador do EventBridge no Guia do usuário do Agendador do Amazon EventBridge. b. Em Janela de tempo flexível, escolha Desativar para desativar a opção ou escolha uma das janelas de tempo predefinidas. Por exemplo, se você escolher 15 minutos e definir uma programação

Ocorrência	Fazer isso...
	recorrente para invocar o destino uma vez a cada hora, a programação será executada em até 15 minutos após o início de cada hora.

5. (Opcional) Se você escolher Programação recorrente na etapa anterior, na seção Período, faça o seguinte:

   a. Em Fuso horário, escolha um fuso horário.
   b. Em Data e hora de início, insira uma data válida no formato `YYYY/MM/DD` e, em seguida, especifique um carimbo de data/hora no formato de 24 horas `hh:mm`.
   c. Para Data e hora de término, insira uma data válida no formato `YYYY/MM/DD` e, em seguida, especifique um carimbo de data/hora no formato 24 horas `hh:mm`.
6. Escolha Próximo.
7. Na página Selecionar destino, faça o seguinte:

   a. Escolha Todas as APIs e, na caixa de pesquisa, digite ECS.
   b. Selecione Amazon ECS.
   c. Na caixa de pesquisa, insira RunTask e, em seguida, escolha RunTask.
   d. Em Cluster do ECS, escolha o cluster.
   e. Em Tarefa do ECS, escolha a definição de tarefa a ser usada para a tarefa.
   f. Escolha como suas tarefas serão distribuídas em toda a infraestrutura do cluster. Expanda Opções de computação e, em seguida, escolha uma das opções a seguir

Opção de computação	Etapas
Estratégia de provedor de capacidade	a. Escolha Estratégia de provedor de capacidade. b. Escolha uma estratégia: • Para usar a estratégia de provedor de capacidade padrão do

Opção de computação	Etapas
	cluster, escolha Usar padrão do cluster. • Para usar uma estratégia personalizada, escolha Usar personalizada. Em seguida, insira a Base, o Provedor de capacidade e o Peso. Para o EC2, o provedor de capacidade é o grupo do Auto Scaling.
Tipo de inicialização	a. Escolha Tipo de inicialização. b. Em Launch type (Tipo de inicialização), escolha um tipo de inicialização. c. Quando o tipo de inicialização do Fargate é especificado, em Versão da plataforma, especifique a versão da plataforma a ser usada.

g. Em Sub-redes, insira os IDs de sub-rede nas quais executar a tarefa.

h. Em Grupos de segurança, insira as IDs do grupo de segurança da sub-rede.

i. (Opcional) Para usar uma estratégia de posicionamento de tarefas diferente da padrão, expanda Restrição de posicionamento e insira as restrições.

   Para obter mais informações, consulte Como o Amazon ECS posiciona tarefas em instâncias de contêineres.

j. (Opcional) Para ajudar a identificar as tarefas, em Tags, configure suas tags.

Para que o Amazon ECS atribua tags automaticamente a todas as tarefas recém-iniciadas com tags de definição de tarefa, selecione Habilitar tags gerenciadas pelo Amazon ECS.

8. Escolha Próximo.
9. Na página Configurações, faça o seguinte:

   a. Para ativar a programação, em Estado da programação, mude para Ativar programação.

   b. Para configurar uma política de novas tentativas para a programação, em Política de novas tentativas e fila de mensagens não entregues (DLQ), faça o seguinte:

      - Mude para Tentar novamente.
      - Em Tempo de retenção máximo do evento, insira a(s) hora(s) e o(s) minuto(s) máximo(s) que o Agendador do EventBridge deverá manter um evento não processado.
      - O período máximo é de 24 horas.
      - Em Máximo de tentativas, insira o número máximo de vezes que o Agendador do EventBridge tentará executar a programação se o destino retornar um erro.

        O valor máximo é 185 tentativas.

      Com as políticas de novas tentativas, se a programação não conseguir invocar o destino, o Agendador do EventBridge tentará executar novamente a programação. Se configurado, você deve definir o tempo máximo de retenção e as novas tentativas da programação.

   c. Escolha onde o Agendador do EventBridge armazena os eventos não entregues.

Opção Fila de mensagens não entregues (DLQ)	Fazer isso...	
Não armazene	Selecione Nenhum.	
Armazenar o evento na mesma Conta da AWS em que você está criando a programação	a. Escolha Selecionar uma fila do Amazon SQS na minha Conta da AWS como uma DLQ. b. Escolha o nome do recurso da Amazon	

Opção Fila de mensagens não entregues (DLQ)	Fazer isso...	
	(ARN) da fila do Amazon SQS.	
Armazenar o evento em uma Conta da AWS diferente de onde você está criando a programação	a. Escolha Especificar uma fila do Amazon SQS em outras Contas da AWS como uma DLQ. b. Insira o nome do recurso da Amazon (ARN) da fila do Amazon SQS.	

d. Para usar uma chave gerenciada pelo cliente para criptografar a entrada de destino, em Criptografia, escolha Personalizar as configurações de criptografia (avançado).

   Se você escolher essa opção, insira o ARN da chave do KMS existente ou escolha Criar AWS KMS key para navegar até o console do AWS KMS. Para obter mais informações sobre como o Agendador do EventBridge criptografa os dados em repouso, consulte Criptografia em repouso no Guia do usuário do Agendador do Amazon EventBridge.

e. Em Permissões, escolha Usar perfil existente e selecione o perfil.

   Para que o Agendador do EventBridge crie um novo perfil de execução para você, escolha Criar novo perfil para esta programação. Depois, insira um nome em Nome do perfil. Se você escolher essa opção, o Agendador do EventBridge anexará as permissões necessárias para o destino de exemplo ao perfil.

10. Escolha Próximo.
11. Na página Revisar e criar programação, revise os detalhes da programação. Em cada seção, escolha Editar para voltar a essa etapa e editar seus detalhes.
12. Clique em Criar programação.

    Você pode ver a lista com as programações novas e existentes na página Programações. Na coluna Status, verifique se a nova programação está Ativada.

Próximas etapas

É possível usar o console do Agendador do EventBridge ou a AWS CLI para gerenciar a programação. Para obter mais informações, consulte Gerenciamento de uma programação no Guia do usuário do Agendador do Amazon EventBridge.

Interrupção de uma tarefa do Amazon ECS

Caso não precise mais manter uma tarefa autônoma em execução, é possível interrompê-la. O console do Amazon ECS facilita a interrupção de uma ou mais tarefas.

Você não pode reiniciar uma tarefa autônoma que foi interrompida.

Caso queira interromper um serviço, consulte Exclusão de um serviço do Amazon ECS usando o console.

Para interromper uma tarefa autônoma (AWS Management Console)

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Clusters.
3. Na página Clusters, escolha o cluster para navegar até a página de detalhes do cluster.
4. Na página de detalhes do cluster, escolha a guia Tarefas.
5. Você pode filtrar tarefas por tipo de execução usando a lista Filtrar tipo de execução.

Tarefas para interromper	Etapas
Uma ou mais	a. Selecione as tarefas e escolha Interromper, Interromper selecionado. b. Na Página de confirmação da interrupção de tarefas, escolha Interromper
Todos	**Important** Se você optar por interromper

Tarefas para interromper	Etapas
	todas as tarefas usando o console, o Amazon ECS interrompe todas as tarefas autônomas e aquelas que fazem parte de um serviço. Portanto, recomenda mos cautela ao usar essa opção. a. Escolha Parar, Parar todos. b. Na Página de confirmaç ão da interrupção de tarefas, insira Interromper todas as tarefas e escolha Interromper.

Serviços do Amazon ECS

É possível usar um serviço do Amazon ECS para executar e manter simultaneamente um número especificado de instâncias de uma definição de tarefa em um cluster do Amazon ECS. Se uma das suas tarefas falhar ou for interrompida, o programador de serviços do Amazon ECS executará outra instância da sua definição de tarefa para substituí-la. Isso ajuda a manter o número desejado de tarefas no serviço.

Você também pode executar o serviço atrás de um balanceador de carga. O load balancer distribui o tráfego entre as tarefas associadas ao serviço.

Recomendamos usar o programador de serviços para serviços e aplicações sem estado de longa execução. O programador de serviços garante que a estratégia de programação que você especifica seja seguida e reprograma as tarefas em caso de falha. Por exemplo, se a infraestrutura subjacente apresentar falha, o programador de serviços reprogramará uma tarefa. É possível usar estratégias e

restrições de posicionamento de tarefas para personalizar como o programador posiciona e finaliza tarefas. Se uma tarefa em um serviço é interrompida, o programador inicia uma tarefa nova para substituí-la. Esse processo continua até que o serviço atinja o número desejado de tarefas com base na estratégia de programação usada pelo serviço. A estratégia de programação do serviço também é chamada de tipo de serviço.

O programador de serviços também substitui tarefas consideradas não íntegras após uma falha na verificação de integridade do contêiner ou na verificação de integridade do grupo-alvo do balanceador de carga. Essa substituição depende dos parâmetros de definição do serviço `maximumPercent` e `desiredCount`. Se uma tarefa for marcada como não íntegra, o programador de serviços iniciará primeiro uma tarefa de substituição. Então, acontece o seguinte.

- Se o status de integridade da tarefa substituta for `HEALTHY`, o agendador do serviço interromperá a tarefa que não está íntegra
- Se a tarefa de substituição tiver um status de integridade de `UNHEALTHY`, o programador interromperá a tarefa de substituição não íntegra ou a tarefa não íntegra existente para que a contagem total de tarefas seja igual a `desiredCount`.

Se o parâmetro `maximumPercent` limitar o programador a iniciar uma tarefa de substituição primeiro, o programador interromperá aleatoriamente uma tarefa não íntegra, uma de cada vez, para liberar capacidade e, em seguida, iniciará uma tarefa de substituição. O processo de iniciar e parar continua até que todas as tarefas não íntegras sejam substituídas por tarefas íntegras. Depois que todas as tarefas não íntegras forem substituídas e somente as tarefas íntegras estiverem em execução, se a contagem total de tarefas exceder a `desiredCount`, as tarefas íntegras serão interrompidas aleatoriamente até que a contagem total de tarefas seja igual a `desiredCount`. Para obter mais informações sobre `maximumPercent` e `desiredCount`, consulte Parâmetros de definição de serviço.

O programador de serviços inclui uma lógica que regula a frequência com que as tarefas são reiniciadas, caso elas falhem repetidamente ao tentar iniciar. Se uma tarefa for interrompida sem ter entrado em um estado `RUNNING`, o programador de serviços começará a retardar as tentativas de inicialização e enviará uma mensagem de evento de serviço. Esse comportamento impede que recursos desnecessários sejam usados para tarefas com falha antes que você possa resolver o problema. Depois que o serviço for atualizado, o programador de serviços continuará com seu comportamento de programação normal. Para obter mais informações, consulte Lógica de controle de utilização do serviço do Amazon ECS e Visualizar mensagens de eventos de serviço do Amazon ECS.

Opção de computação de infraestrutura

Há duas opções de computação que distribuem as tarefas.

- Uma estratégia de provedor de capacidade faz com que o Amazon ECS distribua as tarefas em um ou entre vários provedores de capacidade.

  Para capacity provider strategy (estratégia de provedor de capacidade), por padrão, o console seleciona uma opção de computação. Veja, a seguir, a descrição da ordem que o console usa para selecionar um padrão:

  - Se o cluster tiver uma estratégia padrão de provedor de capacidade definida, ela será selecionada.
  - Se o cluster não tiver uma estratégia de provedor de capacidade padrão definida, mas você tiver os provedores de capacidade do Fargate adicionados ao cluster, será selecionada uma estratégia de provedor de capacidade personalizada que use o provedor de capacidade do `FARGATE`.
  - Se o cluster não tiver uma estratégia de provedor de capacidade padrão definida, mas você tiver um ou mais provedores de capacidade do grupo do Auto Scaling adicionados ao cluster, será selecionada a opção Usar personalizada (Avançado) e a estratégia precisará ser definida manualmente.
  - Se o cluster não tiver uma estratégia de provedor de capacidade padrão definida e nenhum provedor de capacidade for adicionado ao cluster, será selecionado o tipo de inicialização do Fargate.
- Um tipo de execução faz com que o Amazon ECS inicie as tarefas diretamente no Fargate ou nas instâncias do EC2 registradas nos seus clusters.

  Por padrão, o serviço começa nas sub-redes da VPC do cluster.

Autoescalabilidade do serviço

Ajuste de escala automático é a capacidade de aumentar ou diminuir o número de tarefas desejadas no serviço do Amazon ECS. O Amazon ECS utiliza o serviço do Application Auto Scaling para fornecer essa funcionalidade.

Para obter mais informações, consulte Como escalar automaticamente o serviço do Amazon ECS.

Balanceamento de carga do serviço

Os serviços do Amazon ECS hospedados no AWS Fargate oferecem suporte a Application Load Balancers, Network Load Balancers e Gateway Load Balancers. Use a tabela apresentada a seguir para descobrir qual tipo de balanceador de carga usar.

Tipo de balanceador de carga	Uso para estes casos	
Application Load Balancer	Roteamento do tráfego HTTP/HTTPS (ou da camada 7). Os application load balancers oferecem vários recursos que os tornam atrativos para uso com serviços do Amazon ECS: • Cada serviço pode atender a tráfego de vários load balancers e expor várias portas de balanceamento de carga especificando vários grupos de destino. • Eles são compatíveis com tarefas hospedadas em instâncias do Fargate e do EC2. • Os application load balancers permitem que os contêineres usem mapeamento de porta host dinâmico (de modo que várias tarefas do mesmo serviço sejam permitidas por instância de contêiner). • Os Application Load Balancers oferecem suporte	

Tipo de balanceador de carga	Uso para estes casos
	a regras de roteamento e prioridade com base no caminho (de modo que vários serviços possam usar a mesma porta de listener em um único Application Load Balancer).
Network Load Balancer	Roteamento do tráfego TCP ou UDP (ou da camada 4).
Gateway Load Balancer	Roteamento do tráfego TCP ou UDP (ou da camada 4). Use dispositivos virtuais, como firewalls, sistemas de detecção e prevenção de intrusão e sistemas de inspeção profunda de pacotes.

Para obter mais informações, consulte Uso do balanceamento de carga para distribuir o tráfego de serviço do Amazon ECS.

Serviços de interconexão

Se você precisar de uma aplicação para se conectar a outras aplicações executadas nos serviços do Amazon ECS, o Amazon ECS oferecerá as maneiras abaixo de fazer isso sem um balanceador de carga:

- Service Connect: permite comunicações entre serviços com descoberta automática usando nomes curtos e portas padrão.
- Descoberta de serviços: a descoberta de serviços usa o Route 53 para criar um namespace para seu serviço, o que permite que ele seja descoberto por meio do DNS.
- Amazon VPC Lattice: é um serviço de rede de aplicações totalmente gerenciado que você usa para conectar, proteger e monitorar seus serviços em várias contas e VPCs. Há um custo associado a ele.

Para obter mais informações, consulte Interconexão de serviços do Amazon ECS.

Controladores e estratégias de implantação de serviços do Amazon ECS

Antes de implantar seu serviço, determine as opções para implantá-lo e os recursos que o serviço utiliza.

Estratégia de programação

Há duas estratégias de programador de serviços disponíveis:

- `REPLICA`: a estratégia de programação de réplica coloca e mantém o número desejado de tarefas no seu cluster. Por padrão, o programador de serviço distribui tarefas por zonas de disponibilidade. É possível usar estratégias e limitações de posicionamento de tarefas para personalizar decisões de posicionamento de tarefa. Para obter mais informações, consulte Estratégia de agendamento de réplicas.
- `DAEMON`: a estratégia de programação do daemon implanta exatamente uma tarefa em cada instância de contêiner ativa que atenda a todas as restrições de posicionamento de tarefas que você especifica no seu cluster. Ao usar essa estratégia, não há necessidade de especificar um número desejado de tarefas, uma estratégia de posicionamento de tarefas ou usar políticas de Auto Scaling do serviço. Para obter mais informações, consulte Estratégia de agendamento de daemon.

> **Note**
>
> As tarefas do Fargate não são compatíveis com a estratégia de programação do `DAEMON`.

Estratégia de agendamento de réplicas

A estratégia de programação de réplica posiciona e mantém o número desejado de tarefas no seu cluster.

Para um serviço que execute tarefas no Fargate, quando o programador de serviços iniciar novas tarefas ou interromper tarefas em execução, o programador de serviços tentará manter o equilíbrio entre as zonas de disponibilidade da melhor forma. Não há necessidade de especificar estratégias ou restrições de posicionamento de tarefas.

Ao criar um serviço que executa tarefas em instâncias do EC2, você pode opcionalmente especificar estratégias e restrições de posicionamento de tarefas para personalizar decisões de posicionamento

de tarefas. Se nenhuma estratégia ou restrições de posicionamento de tarefas for especificada, por padrão, o programador do serviço distribuirá as tarefas entre zonas de disponibilidade. O programador de serviço usa a seguinte lógica:

- Determina quais instâncias de contêiner no cluster podem oferecer suporte à definição de tarefa de serviço (por exemplo, atributos necessários de CPU, memória, portas e de instância de contêiner).
- Determina quais instâncias de contêiner atendem a qualquer restrição de posicionamento definida para o serviço.
- Quando você tiver um serviço de réplica que dependa de um serviço de daemon (por exemplo, uma tarefa de roteador de log de daemon que precise ser executada antes que as tarefas possam usar o registro em log), crie uma restrição de posicionamento de tarefas que garanta que as tarefas do serviço de daemon sejam colocadas na instância do EC2 antes das tarefas do serviço de réplica. Para obter mais informações, consulte Exemplo de restrições de posicionamento de tarefas do Amazon ECS.
- Quando houver uma estratégia de posicionamento definida, use essa estratégia para selecionar uma instância entre os candidatos restantes.
- Quando não houver uma estratégia de posicionamento definida, use a lógica a seguir para equilibrar as tarefas entre as zonas de disponibilidade no cluster:
  - Ordena as instâncias de contêiner válidas. Dá prioridade a instâncias que têm o menor número de tarefas em execução nesse serviço na respectiva zona de disponibilidade. Por exemplo, se a zona A tiver uma tarefa de serviço em execução e as zonas B e C tiverem nenhuma, as instâncias de contêiner válidas nas zonas B ou C serão consideradas ideais para a colocação.
  - Coloca a nova tarefa de serviço em uma instância de contêiner válida de uma zona de disponibilidade ideal de acordo com as etapas anteriores. Favorece instâncias de contêiner com o menor número de tarefas em execução para esse serviço.

Recomendamos usar o recurso de rebalanceamento de serviços ao usar a estratégia `REPLICA`, pois isso ajuda a garantir a alta disponibilidade do seu serviço.

Estratégia de agendamento de daemon

A estratégia de agendamento do daemon implanta exatamente uma tarefa em cada instância de contêiner ativa que atende a todas as restrições de posicionamento de tarefas especificado no seu cluster. O agendador de serviço avalia as restrições de atribuição das tarefas em execução e interrompe as que não atendem às restrições. Ao usar essa estratégia, não há necessidade de

especificar um número desejado de tarefas e uma estratégia de atribuição de tarefas, nem de usar políticas do Service Auto Scaling.

O Amazon ECS reserva recursos de computação de instância de contêiner, incluindo CPU, memória e interfaces de rede para as tarefas do daemon. Quando você inicia um serviço do daemon em um cluster com outros serviços de réplica, o Amazon ECS prioriza a tarefa do daemon. Isso significa que a tarefa do daemon é a primeira a ser iniciada nas instâncias e a última a ser interrompida após a interrupção de todas as tarefas de réplica. Essa estratégia garante que os recursos não sejam usados por tarefas de réplica pendentes e estejam disponíveis para as tarefas do daemon.

O programador de serviço do daemon não posiciona quaisquer tarefas em instâncias que possuem status `DRAINING`. Se uma instância de contêiner fizer a transição para o status `DRAINING`, as tarefas de daemon nela contidas serão interrompidas. O programador de serviço também monitora quando novas instâncias de contêiner são adicionadas ao seu cluster e adiciona as tarefas de daemon a elas.

Ao especificar uma configuração de implantação, o valor do parâmetro `maximumPercent` deverá ser `100` (especificado como uma porcentagem), que é o valor padrão usado se nenhum valor for definido. O valor padrão do parâmetro `minimumHealthyPercent` é `0` (especificado como porcentagem).

Você precisará reiniciar o serviço quando alterar as restrições de posicionamento do serviço do daemon. O Amazon ECS atualiza dinamicamente os recursos reservados em instâncias qualificadas para a tarefa do daemon. Para instâncias existentes, o programador tenta posicionar a tarefa na instância.

Uma nova implantação será iniciada quando houver uma alteração no tamanho da tarefa ou na reserva de recursos do contêiner na definição de tarefa. Uma nova implantação também começa ao atualizar um serviço ou definir uma revisão diferente da definição da tarefa. O Amazon ECS pega as reservas de CPU e memória atualizadas para o daemon e, em seguida, bloqueia esta capacidade para a tarefa do daemon.

Se não houver recursos suficientes para qualquer um dos casos acima, ocorrerá o seguinte:

- O posicionamento da tarefa apresentará falha.
- Será gerado um evento do CloudWatch.
- O Amazon ECS continuará tentando programar a tarefa na instância, aguardando a disponibilização dos recursos.

- O Amazon ECS liberará todas as instâncias reservadas que não atendam mais aos critérios de restrição de posicionamento e interromperá as tarefas correspondentes do daemon.

A estratégia de programação do daemon pode ser usada nos seguintes casos:

- Execução de containers de aplicações
- Execução de contêineres de suporte para tarefas de registro, monitoramento e rastreamento

As tarefas que usam o tipo de execução do Fargate ou os tipos de controlador de implantação `CODE_DEPLOY` ou `EXTERNAL` não são compatíveis com a estratégia de programação do daemon.

Quando o programador de serviços parar de executar tarefas, ele tentará manter o equilíbrio entre as zonas de disponibilidade do cluster. O programador usa a seguinte lógica:

- Se houver uma estratégia de posicionamento definida, use essa estratégia para selecionar quais tarefas devem ser encerradas. Por exemplo, se um serviço tiver uma estratégia de distribuição de zona de disponibilidade definida, será selecionada uma tarefa que deixa as tarefas restantes com a melhor distribuição.
- Se não houver nenhuma estratégia de posicionamento definida, uso a lógica a seguir para manter o equilíbrio em seu cluster entre as zonas de disponibilidade:
  - Ordene as instâncias de contêiner válidas. Dê prioridade a instâncias que têm o maior número de tarefas em execução nesse serviço na respectiva zona de disponibilidade. Por exemplo, se a zona A tiver uma tarefa de serviço em execução e as zonas B e C tiverem duas, as instâncias de contêiner nas zonas B ou C serão consideradas ideais para encerramento.
  - Interrompa a tarefa em uma instância de contêiner de uma zona de disponibilidade ideal de acordo com as etapas anteriores. Favoreça instâncias de contêiner com o maior número de tarefas em execução para esse serviço.

Controladores de implantação

O controlador de implantação é o mecanismo que determina como as tarefas são implantadas para o seu serviço. As opções válidas são:

- ECS

  Ao criar um serviço que usa o controlador de implantação do `ECS`, você pode escolher entre as seguintes estratégias de implantação:

- `ROLLING`: quando você cria um serviço que usa a estratégia de implantação com atualização contínua (`ROLLING`), o agendador de serviços do Amazon ECS substitui as tarefas em execução no momento por novas tarefas. O número de tarefas que o Amazon ECS adiciona ou remove do serviço durante uma atualização contínua é controlado pela configuração de implantação do serviço.

  Implantações de atualizações contínuas são mais adequadas para os seguintes cenários:

  - Atualizações graduais do serviço: você precisa atualizar seu serviço de forma incremental sem colocar todo o serviço offline de uma só vez.
  - Requisitos de recursos limitados: você deseja evitar os custos adicionais de recursos da execução simultânea de dois ambientes completos (conforme exigido pelas implantações azul/verde).
  - Tempo de implantação aceitável: sua aplicação pode tolerar um processo de implantação mais longo, pois as atualizações contínuas substituem as tarefas uma a uma.
  - Não há necessidade de reversão instantânea: seu serviço pode tolerar um processo de reversão que leva minutos em vez de segundos.
  - Processo de implantação simples: você prefere uma abordagem de implantação direta sem a complexidade de gerenciar vários ambientes, grupos de destino e receptores.
  - Sem necessidade de balanceador de carga: seu serviço não usa nem exige um balanceador de carga, o Application Load Balancer, o Network Load Balancer ou o Service Connect (que são necessários para implantações azul/verde).
  - Aplicações com estado: sua aplicação mantém um estado que dificulta a execução de dois ambientes paralelos.
  - Sensibilidade ao custo: você deseja minimizar os custos de implantação ao não executar ambientes duplicados durante a implantação.

  As atualizações contínuas são a estratégia de implantação padrão para serviços e fornecem um equilíbrio entre a segurança de implantação e a eficiência de recursos para muitos cenários comuns de aplicações.

- `BLUE_GREEN`: uma estratégia de implantação azul/verde (`BLUE_GREEN`) é uma metodologia de lançamento que reduz o tempo de inatividade e o risco ao executar dois ambientes de produção idênticos chamados azul e verde. Com as implantações azul/verde do Amazon ECS, você pode validar novas revisões de serviços antes de direcionar o tráfego de produção para elas. Essa abordagem fornece uma maneira mais segura de implantar alterações com a capacidade de revertê-las rapidamente, se necessário.

As implantações azul/verde do Amazon ECS são mais adequadas para os seguintes cenários:

- Validação de serviço: quando você precisa validar novas revisões de serviço antes de direcionar o tráfego de produção para elas
- Tempo de inatividade zero: quando seu serviço exige implantações com tempo de inatividade zero
- Reversão instantânea: quando você precisa da capacidade de reverter rapidamente se forem detectados problemas
- Requisito de balanceador de carga: quando seu serviço usa o Application Load Balancer, o Network Load Balancer ou o Service Connect

- Externo

  Use um controlador de implantação de terceiros.

- Implantação azul/verde (fornecido pelo AWS CodeDeploy)

  O CodeDeploy instala uma versão atualizada da aplicação como um novo conjunto de tarefas de substituição e redireciona o tráfego de produção do conjunto de tarefas original para o conjunto de tarefas de substituição. O conjunto de tarefas original é encerrado após uma implantação bem-sucedida. Use este controlador de implantação para verificar uma nova implantação de um serviço antes de enviar tráfego de produção para ele.

Detecção de falhas na implantação do Amazon ECS

O Amazon ECS fornece dois métodos para detectar falhas de implantação:

- Disjuntor de implantação
- Alarmes do CloudWatch

Ambos os métodos podem ser configurados para reverter automaticamente as implantações com falha para o último estado válido conhecido.

Considere o seguinte:

- Ambos os métodos oferecem suporte apenas à implantação de atualizações contínuas e aos tipos de implantação azul/verde.
- Quando os dois métodos são usados, qualquer um deles pode causar uma falha na implantação.

- O método de reversão requer uma implantação anterior no estado COMPLETED.
- Os eventos do EventBridge são gerados para alterações no estado de implantação.

Como o disjuntor de implantação do Amazon ECS realiza a detecção de falhas

O disjuntor de implantação é o mecanismo de atualização contínua que determina se as tarefas atingem um estado estacionário. O disjuntor de implantação tem uma opção que reverterá automaticamente uma implantação com falha para a implantação que estiver no estado `COMPLETED`.

Quando uma implantação de serviço muda de estado, o Amazon ECS envia um evento de alteração de estado de implantação de serviço para o EventBridge. Isso fornece uma maneira programática de monitorar o status das implantações de serviço. Para obter mais informações, consulte Eventos de alteração no estado da implantação do serviço do Amazon ECS. Recomendamos que você crie e monitore uma regra do EventBridge com um `eventName` de `SERVICE_DEPLOYMENT_FAILED` para que você possa realizar uma ação manual para iniciar sua implantação. Para obter mais informações, consulte Introdução ao EventBridge no Guia do usuário do Amazon EventBridge.

Quando o disjuntor de implantação determina que uma implantação falhou, ele procura a implantação mais recente que estiver em um estado `COMPLETED`. Essa será a implantação que ele usará como implantação de reversão. Quando a reversão começa, a implantação muda de `COMPLETED` para `IN_PROGRESS`. Isso significa que a implantação não está qualificada para outra reversão até atingir o estado `COMPLETED`. Quando o disjuntor de implantação não encontra uma implantação que esteja em um estado `COMPLETED`, o disjuntor não inicia novas tarefas e a implantação é paralisada.

Ao criar um serviço, o programador acompanha as tarefas que falharam na execução em dois estágios.

- Estágio 1: o programador monitora as tarefas para ver se elas mudam para o estado RUNNING.
  - Sucesso: a implantação tem uma chance de mudar para o estado COMPLETED porque há mais de uma tarefa que passou para o estado RUNNING. O critério de falha é ignorado e o disjuntor passa para o estágio 2.
  - Falha: há tarefas consecutivas que não mudaram para o estado RUNNING e a implantação pode passar para o estado FAILED.
- Estágio 2: a implantação entra nesse estágio quando há pelo menos uma tarefa no estado RUNNING. O disjuntor analisa as verificações de integridade das tarefas na implantação atual que está sendo avaliada. As verificações de integridade validadas são Elastic Load Balancing,

verificações de integridade de serviços do AWS Cloud Map e verificações de integridade de contêineres.

- Sucesso: há pelo menos uma tarefa em execução com verificações de integridade concluídas.
- Falha: as tarefas que foram substituídas devido a falhas na verificação de integridade atingiram o limite de falhas.

Considere o seguinte quando usar o método do disjuntor de implantação em um serviço. O EventBridge gera a regra.

- A resposta `DescribeServices` fornece um insight sobre o estado de uma implantação, o `rolloutState` e o `rolloutStateReason`. Quando uma nova implantação é iniciada, a implantação começa no estado `IN_PROGRESS`. Quando o serviço atinge um estado estacionário, o estado da implantação passa a ser `COMPLETED`. Se o serviço não conseguir alcançar um estado estacionário e o disjuntor estiver ativado, a implantação passará para o estado `FAILED`. Uma implantação em um estado `FAILED` não iniciará qualquer nova tarefa.
- Além dos eventos de alteração de estado de implantação do serviço que o Amazon ECS envia para implantações que foram iniciadas e concluídas, o Amazon ECS também envia um evento quando uma implantação com disjuntor ativado apresenta falha. Esses eventos fornecem detalhes sobre o motivo da falha de uma implantação ou se uma implantação foi iniciada devido a uma reversão. Para obter mais informações, consulte Eventos de alteração no estado da implantação do serviço do Amazon ECS.
- Se uma nova implantação for iniciada porque uma implantação anterior apresentou falha e a reversão ocorreu, o campo `reason` do evento de alteração de estado de implantação do serviço indicará que a implantação foi iniciada devido a uma reversão.
- O disjuntor de implantação só é compatível com serviços do Amazon ECS que usam o controlador de implantação de atualização contínua (`ECS`).
- É necessário usar o console do Amazon ECS ou a AWS CLI quando utilizar o disjuntor de implantação com a opção CloudWatch. Para obter mais informações, consulte the section called "Criar um serviço usando parâmetros definidos" e create-service na Referência da AWS Command Line Interface.

Os seguinte exemplo de `create-service` da AWS CLI mostra como criar um serviço do Linux quando o disjuntor de implantação é usado com reversão.

```
aws ecs create-service \
```

```
    --service-name MyService \
    --deployment-controller type=ECS \
    --desired-count 3 \
    --deployment-configuration "deploymentCircuitBreaker={enable=true,rollback=true}"
\
    --task-definition sample-fargate:1 \
    --launch-type FARGATE \
    --platform-family LINUX \
    --platform-version 1.4.0 \
    --network-configuration
"awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp=EN
```

Exemplo:

A implantação 1 está em um estado `COMPLETED`.

A implantação 2 não pode ser iniciada, então o disjuntor reverte para a implantação 1. A implantação 1 faz a transição para o estado `IN_PROGRESS`.

A implantação 3 é iniciada e não há nenhuma implantação no estado `COMPLETED`, portanto, a implantação 3 não pode reverter ou iniciar tarefas.

Limite de falha

O disjuntor de implantação calcula o valor limite e, em seguida, usa o valor para determinar quando mover a implantação para um estado `FAILED`.

O disjuntor de implantação tem um limite mínimo de 3 e um limite máximo de 200, e usa os valores na fórmula a seguir para determinar a falha de implantação.

```
Minimum threshold <= 0.5 * desired task count => maximum threshold
```

Quando o resultado do cálculo for maior que o mínimo de 3, mas menor que o máximo de 200, o limite de falha é definido como o limite calculado (arredondado para cima).

> **Note**
>
> Você não pode alterar nenhum dos valores de limite.

Há dois estágios para a verificação do status da implantação.

1. O disjuntor de implantação monitora as tarefas que fazem parte da implantação e verifica se há tarefas que estão no estado `RUNNING`. O programador ignora os critérios de falha quando uma tarefa na implantação atual está no estado `RUNNING` e prossegue para o próximo estágio. Quando as tarefas não conseguem alcançar no estado `RUNNING`, o disjuntor de implantação aumenta a contagem de falhas em um. Quando a contagem de falhas é igual ao limite, a implantação é marcada como `FAILED`.
2. Esse estágio inicia quando há uma ou mais tarefas no estado `RUNNING`. O disjuntor de implantação executa verificações de integridade nos seguintes recursos para as tarefas na implantação atual:
   - Load balancers Elastic Load Balancing
   - Serviço da AWS Cloud Map
   - Verificações de integridade do contêiner do Amazon ECS

   Quando uma verificação de integridade falha para a tarefa, o disjuntor de implantação aumenta a contagem de falhas em um. Quando a contagem de falhas é igual ao limite, a implantação é marcada como `FAILED`.

A tabela a seguir oferece alguns exemplos.

Contagem de tarefas desejada	Cálculo	Limite
1	`3 <= 0.5 * 1 => 200`	3 (o valor calculado é menor que o mínimo)
25	`3 <= 0.5 * 25 => 200`	13 (o valor é arredondado para cima)
400	`3 <= 0.5 * 400 => 200`	200
800	`3 <= 0.5 * 800 => 200`	200 (o valor calculado é maior do que o máximo)

Por exemplo, quando o limite é 3, o disjuntor começa com a contagem de falhas definida em 0. Quando uma tarefa não atinge o estado `RUNNING`, o disjuntor de implantação aumenta a contagem de falhas em um. Quando a contagem de falhas é igual a 3, a implantação é marcada como `FAILED`.

Para obter exemplos adicionais sobre como usar a opção de reversão, consulte Announcing Amazon ECS deployment circuit breaker (Anunciar o disjuntor de implantação do Amazon ECS).

Como os alarmes do CloudWatch realizam a detecção de falhas de implantação do Amazon ECS

É possível configurar o Amazon ECS para definir que a implantação falhou quando ele detectar que um alarme especificado do CloudWatch entrou no estado de `ALARM`.

Opcionalmente, você pode definir a configuração para reverter uma implantação com falha para a última implantação concluída.

O seguinte exemplo de `create-service` da AWS CLI mostra como criar um serviço do Linux quando os alarmes de implantação são usados com a opção de reversão.

```
aws ecs create-service \
     --service-name MyService \
     --deployment-controller type=ECS \
     --desired-count 3 \
     --deployment-configuration
 "alarms={alarmNames=[alarm1Name,alarm2Name],enable=true,rollback=true}" \
     --task-definition sample-fargate:1 \
     --launch-type FARGATE \
     --platform-family LINUX \
     --platform-version 1.4.0 \
     --network-configuration
 "awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp=EN
```

Considere o seguinte quando usar o método de alarmes do Amazon CloudWatch em um serviço.

- A duração em que as revisões de serviço azul e verde são executadas simultaneamente após a mudança do tráfego de produção. O Amazon ECS calcula esse período com base na configuração do alarme associada à implantação. Você não pode definir esse valor.
- O parâmetro de solicitação `deploymentConfiguration` agora contém o tipo de dados `alarms`. É possível especificar os nomes dos alarmes, se deve usar o método e se deve iniciar uma reversão quando os alarmes indicarem uma falha na implantação. Para obter mais informações, consulte CreateService na Referência de API do Amazon Elastic Container Service.
- A resposta `DescribeServices` fornece um insight sobre o estado de uma implantação, o `rolloutState` e o `rolloutStateReason`. Quando uma nova implantação é iniciada, seu estado começa como `IN_PROGRESS`. Quando o serviço atinge um estado estacionário e o tempo de incorporação está concluído, o estado da implantação passa a ser `COMPLETED`. Se o serviço

não conseguir alcançar um estado estacionário e o alarme entrar no estado `ALARM`, a implantação passará para o estado `FAILED`. Uma implantação em um estado `FAILED` não iniciará qualquer nova tarefa.

- Além dos eventos de alteração de estado de implantação do serviço que o Amazon ECS envia para implantações que foram iniciadas e concluídas, o Amazon ECS também envia um evento quando uma implantação que usa alarmes apresenta falha. Esses eventos fornecem detalhes sobre o motivo da falha de uma implantação ou se uma implantação foi iniciada devido a uma reversão. Para obter mais informações, consulte Eventos de alteração no estado da implantação do serviço do Amazon ECS.
- Se uma nova implantação for iniciada porque uma implantação anterior tiver apresentado falha e a reversão tiver sido ativada, o campo `reason` do evento de alteração de estado de implantação do serviço indicará que a implantação foi iniciada devido a uma reversão.
- Se você usar o disjuntor de implantação e os alarmes do Amazon CloudWatch para detectar falhas, qualquer um deles poderá iniciar uma falha de implantação assim que os critérios de qualquer um dos métodos forem atendidos. Uma reversão ocorre quando você usa a opção de reversão para o método que iniciou a falha de implantação.
- Os alarmes do Amazon CloudWatch só são compatíveis com serviços do Amazon ECS que usam o controlador de implantação de atualização contínua (`ECS`).
- É possível configurar essa opção usando o console do Amazon ECS ou a AWS CLI. Para obter mais informações, consulte the section called “Criar um serviço usando parâmetros definidos” e create-service na Referência da AWS Command Line Interface.
- É possível observar que o status da implantação permanece `IN_PROGRESS` por um período prolongado. A razão para isso é que o Amazon ECS não altera o status até que ele tenha excluído a implantação ativa e isso não acontece até depois do tempo de incorporação. Dependendo da configuração do alarme, a implantação aparentemente pode demorar alguns minutos a mais do que quando você não usa alarmes (mesmo que o novo conjunto de tarefas primárias tenha um aumento na escala verticalmente e a antiga implantação tenha uma redução na escala verticalmente). Se você usa os tempos limite do CloudFormation, considere aumentar os tempos limite. Para obter mais informações, consulte Criar condições de espera em um modelo no Guia do usuário do AWS CloudFormation.
- O Amazon ECS chama `DescribeAlarms` para pesquisar os alarmes. As chamadas para `DescribeAlarms` são contabilizadas nas cotas de serviço do CloudWatch associadas à sua conta. Se você tiver outros serviços da AWS que chamem `DescribeAlarms`, pode haver um impacto na pesquisa dos alarmes pelo Amazon ECS. Por exemplo, se outro serviço fizer chamadas para `DescribeAlarms` suficientes para atingir a cota, esse serviço será submetido

a controle de utilização e o Amazon ECS também será submetido a controle de utilização e não poderá sondar alarmes. Se um alarme for gerado durante o período de controle de utilização, o Amazon ECS poderá perder o alarme e a reversão poderá não ocorrer. Não há outro impacto na implantação. Para obter mais informações sobre cotas de serviço do CloudWatch, consulte Cotas de serviço do CloudWatch no Guia do usuário do CloudWatch.

- Se um alarme estiver no estado `ALARM` no início de uma implantação, o Amazon ECS não monitorará os alarmes durante a implantação (o Amazon ECS ignora a configuração do alarme). Esse comportamento aborda o caso em que você deseja iniciar uma nova implantação para corrigir uma falha de implantação inicial.

Alarmes recomendados

Recomendamos que você use as seguintes métricas de alarme:

- Se você usa um Application Load Balancer, use as métricas `HTTPCode_ELB_5XX_Count` e`HTTPCode_ELB_4XX_Count` do Application Load Balancer. Essas métricas verificam picos de HTTP. Para obter mais informações sobre as métricas do Application Load Balancer, consulte CloudWatch metrics for your Application Load Balancer (Métricas do CloudWatch para seu Application Load Balancer) no Guia do usuário para Application Load Balancers.
- Se você tiver uma aplicação existente, use as métricas `CPUUtilization` e `MemoryUtilization`. Essas métricas verificam a porcentagem de CPU e memória que o cluster ou serviço usa. Para obter mais informações, consulte the section called “Considerações”.
- Se você usa filas do Amazon Simple Queue Service nas suas tarefas, use a métrica `ApproximateNumberOfMessagesNotVisible` do Amazon SQS. Essa métrica verifica o número de mensagens na fila que estão atrasadas e indisponíveis para leitura imediata. Para obter mais informações sobre métricas do Amazon SQS, consulte Métricas disponíveis do CloudWatch para o Amazon SQS no Guia do desenvolvedor do Amazon Simple Queue Service.

Ganchos do ciclo de vida para implantações de serviços do Amazon ECS

Quando uma implantação é iniciada, ela passa pelos estágios do ciclo de vida. Esses estágios podem estar em estados como IN_PROGRESS ou SUCCESSFUL. Você pode usar ganchos doe ciclo de vida, que são funções do Lambda que o Amazon ECS executa em seu nome em estágios específicos do ciclo de vida. As funções podem ser uma das seguintes:

- Uma API assíncrona que valida a verificação de integridade em até 15 minutos.

- Uma API de polling que inicia outro processo assíncrono que avalia a conclusão do gancho do ciclo de vida.

Depois que a função terminar de ser executada, ela deve retornar a `hookStatus` para que a implantação continue. Se `hookStatus` não for retornado, ou se a função falhar, a implantação será revertida. Confira abaixo os valores de `hookStatus`:

- `SUCCEEDED`: a implantação continua até o próximo estágio do ciclo de vida
- `FAILED`: a implantação será revertida para a última implantação bem-sucedida.
- `IN_PROGRESS`: o Amazon ECS executa a função novamente após um curto período. Por padrão, é um intervalo de 30 segundos, no entanto, esse valor é personalizável retornando um `callBackDelaySeconds` junto com `hookStatus`.

O exemplo a seguir mostra como retornar um `hookStatus` com um atraso de retorno de chamada personalizado. Neste exemplo, o Amazon ECS tentaria novamente esse gancho em 60 segundos, em vez dos 30 segundos padrão:

```
{
    "hookStatus": "IN_PROGRESS",
    "callBackDelaySeconds": 60
}
```

Quando ocorre uma reversão, o Amazon ECS executa os ganchos do ciclo de vida para os seguintes estágios do ciclo de vida:

- PRODUCTION_TRAFFIC_SHIFT
- TEST_TRAFFIC_SHIFT

Cargas úteis do ciclo de vida

Quando você configura ganchos do ciclo de vida para suas implantações de serviços do ECS, o Amazon ECS invoca esses ganchos em estágios específicos do processo de implantação. Cada estágio do ciclo de vida fornece uma carga útil JSON com informações sobre o estado atual da implantação. Este documento descreve a estrutura da carga útil para cada estágio do ciclo de vida.

Estrutura de carga útil comum

Todas as cargas úteis do estágio do ciclo de vida incluem os seguintes campos comuns:

- `serviceArn`: o nome do recurso da Amazon (ARN) do serviço.
- `targetServiceRevisionArn`: o ARN da revisão do serviço de destino que está sendo implantada.
- `testTrafficWeights`: um mapa dos ARNs de revisão de serviço com suas porcentagens de peso de tráfego de teste correspondentes.
- `productionTrafficWeights`: um mapa dos ARNs de revisão de serviço com suas porcentagens de peso de tráfego de produção correspondentes.

Cargas úteis do estágio do ciclo de vida

RECONCILE_SERVICE

Este estágio ocorre no início do processo de implantação, quando o serviço está sendo reconciliado. Veja a seguir um exemplo de carga útil para esse estágio do ciclo de vida.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/01275892":
 100,
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/78652123": 0
  },
  "productionTrafficWeights": {
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/01275892":
 100,
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/78652123": 0
  }
}
```

Expectativas nesta fase:

- O conjunto de tarefas primárias está na escala de 0%

PRE_SCALE_UP

Este estágio ocorre antes que as novas tarefas tenha a escala aumentada verticalmente. Veja a seguir um exemplo de carga útil para esse estágio do ciclo de vida.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {},
  "productionTrafficWeights": {}
}
```

Expectativas nesta fase:

- As tarefas de revisão do serviço verde estão em escala de 0%

POST_SCALE_UP

Este estágio ocorre após as novas tarefas terem tido a escala aumentada verticalmente e estarem íntegras. Veja a seguir um exemplo de carga útil para esse estágio do ciclo de vida.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {},
  "productionTrafficWeights": {}
}
```

Expectativas nesta fase:

- As tarefas de revisão do serviço verde estão em escala de 100%
- As tarefas na revisão do serviço verde estão íntegras

TEST_TRAFFIC_SHIFT

Este estágio ocorre quando o tráfego de teste está sendo transferido para as tarefas de revisão do serviço verde.

Veja a seguir um exemplo de carga útil para esse estágio do ciclo de vida.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/01275892":
 100,
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/78652123": 0
  },
  "productionTrafficWeights": {}
}
```

Expectativas nesta fase:

- O tráfego de teste está em processo de migração para as tarefas de revisão do serviço verde.

POST_TEST_TRAFFIC_SHIFT

Este estágio ocorre após o tráfego de teste ter sido totalmente transferido para as novas tarefas.

Veja a seguir um exemplo de carga útil para esse estágio do ciclo de vida.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {},
  "productionTrafficWeights": {}
}
```

Expectativas nesta fase:

- 100% do tráfego de teste foi direcionado para as tarefas de revisão do serviço verde.

PRODUCTION_TRAFFIC_SHIFT

Este estágio ocorre quando o tráfego de produção está sendo transferido para as tarefas de revisão do serviço verde.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {},
  "productionTrafficWeights": {
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/01275892":
 100,
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/78652123": 0
  }
}
```

Expectativas nesta fase:

- O tráfego de produção está em processo de migração para a revisão do serviço verde.

POST_PRODUCTION_TRAFFIC_SHIFT

Este estágio ocorre após o tráfego de produção ter sido totalmente transferido para as tarefas de revisão do serviço verde.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {},
  "productionTrafficWeights": {}
}
```

Expectativas nesta fase:

- 100% do tráfego de produção foi direcionado para as tarefas de revisão do serviço verde.

Categorias de estágio do ciclo de vida

Os estágios do ciclo de vida se encaixam em duas categorias:

1. Estágios de invocação única: estes estágios são invocados somente uma vez durante a implantação de um serviço:
   - PRE_SCALE_UP
   - POST_SCALE_UP
   - POST_TEST_TRAFFIC_SHIFT
   - POST_PRODUCTION_TRAFFIC_SHIFT
2. Estágios de invocação recorrente: estes estágios podem ser invocados várias vezes durante a implantação de um serviço, por exemplo, quando ocorre uma operação de reversão:
   - TEST_TRAFFIC_SHIFT
   - PRODUCTION_TRAFFIC_SHIFT

Status de implantação durante ganchos do ciclo de vida

Enquanto os ganchos do ciclo de vida estiverem em execução, o status de implantação será `IN_PROGRESS` para todos os estágios do ciclo de vida.

Estágio do ciclo de vida	Status da implantação
RECONCILE_SERVICE	IN_PROGRESS
PRE_SCALE_UP	IN_PROGRESS
POST_SCALE_UP	IN_PROGRESS
TEST_TRAFFIC_SHIFT	IN_PROGRESS
POST_TEST_TRAFFIC_SHIFT	IN_PROGRESS
PRODUCTION_TRAFFIC_SHIFT	IN_PROGRESS
POST_PRODUCTION_TRAFFIC_SHIFT	IN_PROGRESS

Parar implantações de serviços do Amazon ECS

É possível interromper manualmente uma implantação quando uma implantação com falha não é detectada pelo disjuntor ou pelos alarmes do CloudWatch. Os seguintes tipos de paradas estão disponíveis:

- Reversão: essa opção reverte a implantação do serviço para a revisão de serviço anterior.

  Você poderá usar essa opção mesmo que não tenha configurado a implantação do serviço para a opção de reversão.

É possível interromper uma implantação que esta em qualquer um dos estados apresentados a seguir. Para obter mais informações sobre estados de implantações de serviços, consulte Visualize o histórico de serviços usando implantações de serviços do Amazon ECS.

- PENDING: a implantação do serviço passa para o estado ROLLBACK_REQUESTED e, em seguida, a operação de reversão é iniciada.
- IN_PROGRESS: a implantação do serviço passa para o estado ROLLBACK_REQUESTED e, em seguida, a operação de reversão é iniciada.
- STOP_REQUESTED: a implantação do serviço continua a parar.
- ROLLBACK_REQUESTED: a implantação do serviço continua a operação de reversão.
- ROLLBACK_IN_PROGRESS: a implantação do serviço continua a operação de reversão.

Procedimento

Antes de começar, configure as permissões necessárias para visualizar as implantações de serviços. Para obter mais informações, consulte Permissões necessárias para visualizar implantações de serviço do Amazon ECS.

Amazon ECS Console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, escolha o cluster.
3. Na página de detalhes do cluster, na seção Serviços, escolha o serviço.

   A página de detalhes do serviço é exibida.
4. Na página de detalhes da implantação, escolha Implantações.

A página de implantações é exibida.

5. Em Implantação em andamento, escolha Reverter. Em seguida, na janela janela de confirmação, escolha Reverter.

AWS CLI

1. Execute `list-service-deployments` para recuperar o ARN da implantação do serviço.

   Substitua os valores das *`user-input`* pelos seus.

   ```
   aws ecs list-service-deployments --cluster cluster-name --service service-name
   ```

   Observe o `serviceDeploymentArn` da implantação que deseja interromper.

   ```
   {
       "serviceDeployments": [
           {
               "serviceDeploymentArn": "arn:aws:ecs:us-west-2:123456789012:service-
   deployment/cluster-name/service-name/NCWGC2ZR-taawPAYrIaU5",
               "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/cluster-
   name/service-name",
               "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/cluster-
   name",
               "targetServiceRevisionArn": "arn:aws:ecs:us-
   west-2:123456789012:service-revision/cluster-name/service-
   name/4980306466373577095",
               "status": "SUCCESSFUL"
           }
       ]
   }
   ```

2. Executar `stop-service-deployments`. Use o `serviceDeploymentArn` que foi devolvido de `list-service-deployments`.

   Substitua os valores das *`user-input`* pelos seus.

   ```
   aws ecs stop-service-deployment --service-deployment-arn
    arn:aws:ecs:region:123456789012:service-deployment/cluster-name/service-
   name/NCWGC2ZR-taawPAYrIaU5 --stop-type ROLLBACK
   ```

Próximas etapas

Decida quais alterações precisam ser feitas no serviço e, em seguida, atualize-o. Para obter mais informações, consulte Atualizar um serviço do Amazon ECS.

Implantação de serviços do Amazon ECS por meio da substituição de tarefas

Ao criar um serviço que usa o tipo de implantação atualização cumulativa (ECS), o agendador de serviços do Amazon ECS substitui as tarefas que estão em execução por novas tarefas. O número de tarefas que o Amazon ECS adiciona ou remove do serviço durante uma atualização contínua é controlado pela configuração de implantação do serviço.

O Amazon ECS usa os seguintes parâmetros para determinar o número de tarefas:

- O `minimumHealthyPercent` representa o limite inferior do número de tarefas que devem estar sendo executadas para um serviço durante uma implantação ou quando uma instância de contêiner está sendo drenada, como uma porcentagem do número desejado de tarefas para o serviço. Esse valor é arredondado para cima. Por exemplo, se a porcentagem mínima de integridade é `50` e a contagem de tarefas desejadas é quatro, o programador pode interromper duas tarefas existentes antes de iniciar duas novas tarefas. Da mesma forma, se a porcentagem mínima de integridade é 75% e a contagem de tarefas desejada é dois, o programador não pode parar quaisquer tarefas porque o valor resultante também é dois.

  Se as tarefas se tornarem não íntegras, o agendador de serviço do Amazon ECS primeiro iniciará as tarefas substitutas e manterá as tarefas com `minimumHealthyPercent` até que as tarefas substitutas se tornem íntegras. À medida que as tarefas substitutas forem iniciadas e se tornarem íntegras, as tarefas não íntegras serão gradualmente interrompidas.
- O `maximumPercent` representa o limite superior do número de tarefas que devem estar sendo executadas para um serviço durante uma implantação ou quando uma instância de contêiner está sendo drenada, como uma porcentagem do número desejado de tarefas para o serviço. Esse valor é arredondado para baixo. Por exemplo, se a porcentagem máxima de integridade for `200` e a contagem de tarefas desejadas for quatro, o programador poderá iniciar quatro novas tarefas antes de interromper quatro tarefas existentes. Da mesma forma, se a porcentagem máxima de integridade é `125` e a contagem de tarefas desejada é três, o programador não pode iniciar quaisquer tarefas porque o valor resultante também é três.

> **Important**
>
> Ao definir um percentual mínimo ou um percentual máximo de integridade, você deve garantir que o programador possa interromper ou iniciar pelo menos uma tarefa quando uma implantação for iniciada. Se seu serviço tiver uma implantação travada devido a uma configuração de implantação inválida, será enviada uma mensagem de evento de serviço. Para obter mais informações, consulte O serviço (*service-name*) não conseguiu interromper ou iniciar tarefas durante uma implantação devido à configuração de implantação do serviço. Atualize o valor minimumHealthyPercent ou maximumPercent e tente novamente..

As implantações contínuas têm dois métodos que fornecem uma maneira de identificar rapidamente quando uma implantação de serviço falhou:

- the section called " Como o disjuntor de implantação realiza a detecção de falhas"
- the section called "Como os alarmes do CloudWatch realizam a detecção de falhas de implantação"

Os métodos podem ser usados separadamente ou em conjunto. Quando ambos os métodos são usados, a implantação é definida como falha assim que os critérios de falha de qualquer um dos métodos de falha são satisfeitos.

Siga as diretrizes a seguir para ajudar a determinar qual método será usado:

- Disjuntor: use este método se quiser interromper uma implantação quando as tarefas não puderem ser iniciadas.
- Alarmes do CloudWatch: use este método quando quiser interromper uma implantação com base nas métricas da aplicação.

Ambos os métodos permitem a reversão para a revisão de serviço anterior.

Resolução da imagem do contêiner

Por padrão, o Amazon ECS resolve tags de imagem de contêiner especificadas na definição da tarefa para resumos de imagens de contêiner. Se você criar um serviço que executa e mantém uma única tarefa, essa tarefa será usada para estabelecer resumos de imagem para as imagens no contêiner. Se você criar um serviço que executa e mantém várias tarefas, a primeira tarefa iniciada

pelo agendador de serviços durante a implantação será usada para estabelecer resumos de imagens para os contêineres nas tarefas.

Se três ou mais tentativas de estabelecer os resumos de imagens no contêiner falharem, a implantação continuará sem a resolução do resumo da imagem. Se o disjuntor de implantação estiver habilitado, a implantação também falhará e será revertida.

Depois que os resumos de imagens no contêiner forem estabelecidos, o Amazon ECS usará os resumos para iniciar qualquer outra tarefa desejada e para qualquer atualização futura do serviço. Isso faz com que todas as tarefas em um serviço sempre executem imagens de contêiner idênticas, o que resultará na consistência de versões do seu software.

É possível configurar esse comportamento para cada contêiner em sua tarefa usando o parâmetro `versionConsistency` na definição do contêiner. Para obter mais informações, consulte `versionConsistency`.

Note

- Versões do Amazon ECS Agent anteriores à `1.31.0` não oferecem suporte à resolução de resumos de imagens. As versões do agente `1.31.0` a `1.69.0` oferecem suporte à resolução de resumos de imagens somente para imagens enviadas aos repositórios do Amazon ECR. As versões do agente `1.70.0` ou superiores oferecem suporte à resolução de resumos de imagens para todas as imagens.
- A versão mínima da plataforma Linux para o Fargate para resolução de resumos de imagens é `1.3.0`. A versão mínima da plataforma Windows para o Fargate para resolução de resumos de imagens é `1.0.0`.
- O Amazon ECS não captura resumos de contêineres auxiliares gerenciados pelo Amazon ECS, como o agente de segurança do Amazon GuardDuty ou o proxy do Service Connect.
- Para reduzir a latência potencial associada à resolução de imagens de contêineres em serviços com várias tarefas, execute o agente do Amazon ECS versão `1.83.0` ou superior em instâncias de contêiner do EC2. Para evitar latência potencial, especifique os resumos de imagens do contêiner na definição da tarefa.
- Se você criar um serviço com uma contagem de tarefas desejada igual a zero, o Amazon ECS não poderá estabelecer resumos do contêiner até que você acione outra implantação do serviço com uma contagem de tarefas desejada maior que zero.
- Para estabelecer resumos de imagens atualizados, é possível forçar uma nova implantação. Os resumos atualizados serão usados para iniciar novas tarefas e não

afetará as tarefas já em execução. Para obter mais informações sobre forçar novas implantações, consulte forceNewDeployment na referência da API do Amazon ECS.

- Ao usar provedores de capacidade do EC2, se não houver capacidade suficiente para iniciar uma tarefa durante a implantação inicial, a consistência da versão do software poderá falhar. Para garantir que a consistência da versão seja mantida mesmo quando a capacidade é limitada, defina explicitamente `versionConsistency: true` na configuração do contêiner da definição de tarefas em vez de confiar no comportamento padrão. Isso faz com que o Amazon ECS espere até que a capacidade esteja disponível antes de continuar com a implantação.

Práticas recomendadas para parâmetros de serviço do Amazon ECS

Para garantir que não haja tempo de inatividade da aplicação, o processo de implantação é o seguinte:

1. Inicie os novos contêineres de aplicações enquanto mantém os contêineres existentes em execução.
2. Verifique se os novos contêineres estão íntegros.
3. Interrompa os contêineres antigos.

Dependendo da configuração de implantação e da quantidade de espaço livre e não reservado no cluster, podem ser necessárias várias rodadas para concluir a substituição de todas as tarefas antigas por novas.

Há duas opções de configuração do serviço que podem ser usadas para modificar o número:

- `minimumHealthyPercent`: 100% (padrão)

  O limite inferior do número de tarefas do serviço que devem permanecer no estado `RUNNING` durante uma implantação. Essa é uma porcentagem da `desiredCount` arredondada para o número inteiro superior mais próximo. Esse parâmetro permite implantar sem usar a capacidade adicional de cluster.
- `maximumPercent`: 200% (padrão)

  O limite superior do número de tarefas do serviço que são permitidas no estado `RUNNING` ou `PENDING` durante uma implantação. Essa é uma porcentagem da `desiredCount` arredondada para o número inteiro inferior mais próximo.

Exemplo: opções de configuração padrão

Considere o serviço a seguir, que tem seis tarefas, implantadas em um cluster com espaço para oito tarefas no total. As opções de configuração padrão do serviço não permitem que a implantação fique abaixo de 100% das seis tarefas desejadas.

O processo de implantação é o seguinte:

1. O objetivo é substituir as seis tarefas.
2. O programador inicia duas novas tarefas porque as configurações padrão exigem que haja seis tarefas em execução.

   Agora há seis tarefas existentes e duas tarefas novas.
3. O programador interrompe duas das tarefas existentes.

   Agora há quatro tarefas existentes e duas novas.
4. O programador inicia duas tarefas novas adicionais.

   Agora há quatro tarefas existentes e quatro tarefas novas.
5. O programador encerra duas das tarefas existentes.

   Agora há duas tarefas existentes e quatro novas.
6. O programador inicia duas tarefas novas adicionais.

   Agora há duas tarefas existentes e seis tarefas novas.
7. O programador encerra as duas tarefas existentes restantes.

   Agora há seis tarefas novas.

No exemplo acima, ao usar os valores padrão das opções, há uma espera de 2,5 minutos para cada nova tarefa iniciada. Além disso, o balanceador de carga pode ter que esperar cinco minutos para que a tarefa antiga seja encerrada.

Exemplo: Modificar **`minimumHealthyPercent`**

Você pode acelerar a implantação definindo o valor do `minimumHealthyPercent` como 50%.

Considere o serviço a seguir, que tem seis tarefas, implantadas em um cluster com espaço para oito tarefas no total. O processo de implantação é o seguinte:

1. O objetivo é substituir seis tarefas.
2. O programador interrompe três das tarefas existentes.

   Ainda há três tarefas existentes em execução que atendem ao valor de `minimumHealthyPercent`.
3. O programador inicia cinco tarefas novas.

   Há três tarefas existentes e cinco tarefas novas.
4. O programador interrompe as três tarefas existentes restantes.

   Há cinco novas tarefas
5. O programador inicia as tarefas novas finais.

   Há seis tarefas novas.

Exemplo: modificar o espaço livre do cluster

Você também pode adicionar mais espaço livre para executar tarefas adicionais.

Considere o serviço a seguir, que tem seis tarefas, implantadas em um cluster com espaço para dez tarefas no total. O processo de implantação é o seguinte:

1. O objetivo é substituir as tarefas existentes.
2. O programador interrompe três das tarefas existentes.

   Há três tarefas existentes.
3. O programador inicia seis tarefas novas.

   Há três tarefas existentes e seis tarefas novas.
4. O programador interrompe as três tarefas existentes.

   Há seis tarefas novas.

Recomendações

Use os valores a seguir nas opções de configuração do serviço quando as tarefas estiverem ociosas por algum tempo e não tiverem uma alta taxa de utilização.

- `minimumHealthyPercent`: 50%

- `maximumPercent`: 200%

Criação de uma implantação de atualização contínua do Amazon ECS

Crie um serviço para executar e manter simultaneamente um número especificado de instâncias de uma definição de tarefa em um cluster. Se uma das suas tarefas falhar ou for interrompida, o programador de serviços do Amazon ECS executará outra instância da sua definição de tarefa para substituí-la. Isso ajuda a manter o número desejado de tarefas no serviço.

Escolha os seguintes parâmetros de configuração antes de criar um serviço:

- Há duas opções de computação que distribuem as tarefas.
  - Uma estratégia de provedor de capacidade faz com que o Amazon ECS distribua as tarefas em um ou entre vários provedores de capacidade.
  - Um tipo de execução faz com que o Amazon ECS inicie as tarefas diretamente no Fargate ou nas instâncias do EC2 registradas nos seus clusters.
- Definições de tarefa que usam o modo de rede `awsvpc` ou serviços configurados para usar um balanceador de carga devem ter uma configuração de rede. Por padrão, o console seleciona a Amazon VPC padrão juntamente com todas as sub-redes e o grupo de segurança padrão na Amazon VPC padrão.
- A estratégia padrão de posicionamento de tarefas distribui as tarefas uniformemente nas zonas de disponibilidade.

  Recomendamos usar o rebalanceamento de zonas de disponibilidade para ajudar a garantir a alta disponibilidade do seu serviço. Para obter mais informações, consulte Balancear um serviço do Amazon ECS entre zonas de disponibilidade.
- Quando você usa o Launch Type (Tipo de inicialização) para a implantação do serviço, por padrão, o serviço começa nas sub-redes da VPC do cluster.
- Para capacity provider strategy (estratégia de provedor de capacidade), por padrão, o console seleciona uma opção de computação. Veja, a seguir, a descrição da ordem que o console usa para selecionar um padrão:
  - Se o cluster tiver uma estratégia padrão de provedor de capacidade definida, ela será selecionada.
  - Se o cluster não tiver uma estratégia de provedor de capacidade padrão definida, mas você tiver os provedores de capacidade do Fargate adicionados ao cluster, será selecionada uma

estratégia de provedor de capacidade personalizada que use o provedor de capacidade do FARGATE.

- Se o cluster não tiver uma estratégia de provedor de capacidade padrão definida, mas você tiver um ou mais provedores de capacidade do grupo do Auto Scaling adicionados ao cluster, será selecionada a opção Usar personalizada (Avançado) e a estratégia precisará ser definida manualmente.
- Se o cluster não tiver uma estratégia de provedor de capacidade padrão definida e nenhum provedor de capacidade for adicionado ao cluster, será selecionado o tipo de inicialização do Fargate.

- As opções padrão de detecção de falhas de implantação são usar a opção Disjuntor de implantação do Amazon ECS com a opção Reverter em caso de falhas.

  Para obter mais informações, consulte Como o disjuntor de implantação do Amazon ECS realiza a detecção de falhas.
- Se você quiser usar a opção de implantação azul/verde, determine como o CodeDeploy moverá as aplicações. As seguintes opções estão disponíveis:
  - CodeDeployDefault.ECSAllAtOnce: desloca todo o tráfego para o contêiner atualizado do Amazon ECS de uma só vez
  - CodeDeployDefault.ECSLinear10PercentEvery1Minutes: desloca 10% do tráfego a cada minuto até que todo o tráfego seja deslocado.
  - CodeDeployDefault.ECSLinear10PercentEvery3Minutes: desloca 10% do tráfego a cada 3 minutos até que todo o tráfego seja deslocado.
  - CodeDeployDefault.ECSCanary10Percent5Minutes: desloca 10% do tráfego no primeiro incremento. Os 90 por cento restantes são implantados cinco minutos depois.
  - CodeDeployDefault.ECSCanary10Percent15Minutes: desloca 10% do tráfego no primeiro incremento. Os 90 por cento restantes são implantados 15 minutos depois.
- Decida se você deseja que o Amazon ECS aumente ou diminua automaticamente o número de tarefas desejadas no serviço. Para mais informações, consulte Como escalar automaticamente o serviço do Amazon ECS.
- Se você precisar que uma aplicação se conecte a outras aplicações executadas no Amazon ECS, determine a opção mais adequada à sua arquitetura. Para obter mais informações, consulte Interconexão de serviços do Amazon ECS.
- Quando você cria um serviço que usa o disjuntor do Amazon ECS, o Amazon ECS cria uma implantação e uma revisão do serviço. Esses recursos permitem que você visualize informações

detalhadas sobre o histórico de serviços. Para obter mais informações, consulte Visualize o histórico de serviços usando implantações de serviços do Amazon ECS.

Para obter informações sobre como criar um serviço usando a AWS CLI, consulte create-service na Referência da AWS Command Line Interface.

Para obter informações sobre como criar um serviço usando o AWS CloudFormation, consulte AWS::ECS::Service no Guia do usuário do AWS CloudFormation.

Criar um serviço com as opções padrão

É possível usar o console para criar e implantar um serviço rapidamente. O serviço conta com a seguinte configuração:

- É implantado na VPC e nas sub-redes associadas ao cluster
- Implanta uma tarefa
- Usa a implantação contínua
- Usa a estratégia de provedor de capacidade com o provedor de capacidade padrão
- Usa o disjuntor de implantação para detectar falhas e define a opção de reverter automaticamente a implantação em caso de falha

Para implantar um serviço usando os parâmetros padrão, siga estas etapas.

Para criar um serviço (console do Amazon ECS)

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página de navegação, escolha Clusters.
3. Na página Clusters, selecione o cluster no qual o serviço será criado.
4. Na guia Services (Serviços), escolha Create (Criar).

   A página Criar pilha é exibida.
5. Em Detalhes do serviço, faça o seguinte:

   a. Em Definição de tarefa, insira a família e a revisão das definições de tarefa a serem usadas.

   b. Em Service name (Nome do serviço), insira um nome para o serviço.
6. Em Configuração de implantação, faça o seguinte:

- Em Desired tasks (Tarefas desejadas), insira o número de tarefas que serão iniciadas e mantidas no serviço.

7. (Opcional) Para ajudar a identificar seu serviço e tarefas, expanda a seção Tags (Etiquetas) e configure suas etiquetas.

   Para que o Amazon ECS marque automaticamente todas as tarefas recém-iniciadas com o nome do cluster e as tags de definição de tarefa, selecione Turn on Amazon ECS managed tags (Ativar tags gerenciadas pelo Amazon ECS) e depois Definições de tarefas.

   Para que o Amazon ECS marque automaticamente todas as tarefas recém-iniciadas com o nome do cluster e as tags de serviços, selecione Turn on Amazon ECS managed tags (Ativar tags gerenciadas pelo Amazon ECS) e depois Definições de tarefas.

   Adicione ou remova uma tag.

   - [Adicionar uma etiqueta] Escolha Add tag (Adicionar etiqueta) e faça o seguinte:
     - Em Chave, insira o nome da chave.
     - Em Valor insira o valor da chave.
   - [Remover uma tag] Ao lado da tag, escolha Remove tag (Remover tag).

Criar um serviço usando parâmetros definidos

Para criar um serviço usando os parâmetros definidos, siga estas etapas.

Para criar um serviço (console do Amazon ECS)

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Determine o recurso no qual você inicia o serviço.

Para iniciar um serviço em	Etapas
Clusters	a. Na página Clusters, selecione o cluster no qual o serviço será criado.

Para iniciar um serviço em	Etapas
	b. Na guia Services (Serviços), escolha Create (Criar).
Definição de tarefa	a. Na página Definições de tarefas, selecione o botão de opções ao lado da definição da tarefa. b. No menu Implantar, escolha Criar serviço.

A página Criar pilha é exibida.

3. Em Detalhes do serviço, faça o seguinte:

   a. Em Definição de tarefa, insira a definição de tarefa a ser usada. Em seguida, em Revisão, escolha a revisão a ser usada.

   b. Em Service name (Nome do serviço), insira um nome para o serviço.

4. Em Cluster existente, escolha o cluster.

   Escolha Criar cluster para executar a tarefa em um novo cluster

5. Escolha como suas tarefas serão distribuídas em toda a infraestrutura do cluster. Expanda Configuração de computação e escolha sua opção.

Opção de computação	Etapas
Estratégia de provedor de capacidade	a. Em Opções de computaçã o, escolha Estratégia do provedor de capacidade. b. Escolha uma estratégia: • Para usar a estratégi a de provedor de capacidade padrão do cluster, escolha Use

Opção de computação	Etapas
	cluster default (Usar padrão de cluster). • Se o seu cluster não tiver uma estratégia de provedor de capacidade padrão ou, para usar uma estratégia personalizada, escolha Usar personalizada, Adicionar estratégia de provedor de capacidade e, em seguida, defina sua estratégia de provedor de capacidade personalizada especificando uma Base, um Provedor de capacidade e um Peso. **Note** Para usar um provedor de capacidade em uma estratégia, o provedor de capacidade deve estar associado ao cluster.

Opção de computação	Etapas
Tipo de inicialização	a. Na seção Compute options (Opções de computação), selecione Launch type (Tipo de inicialização). b. Em Launch type (Tipo de inicialização), escolha um tipo de inicialização. c. (Opcional) Quando o tipo de inicialização do Fargate for especific ado, em Platform version (Versão da plataforma), especifique a versão de plataforma a ser usada. Se não for especificada uma versão da plataform a, a versão da plataforma `LATEST` será usada.

6. Em Configuração de implantação, faça o seguinte:

   a. Em Service type (Tipo de serviço), escolha a estratégia de programação de serviços.

      - Para o programador implantar exatamente uma tarefa em cada instância de contêiner ativa que atende a todas as restrições de posicionamento de tarefas, escolha Daemon.
      - Para que o programador posicione e mantenha o número desejado de tarefas no cluster, escolha Replica (Réplica).

   b. Caso escolha Replica (Réplica), em Desired tasks (Tarefas desejadas), especifique o número de tarefas que serão iniciadas e mantidas no serviço.

   c. Se você escolheu Replica, para que o Amazon ECS monitore a distribuição de tarefas entre as zonas de disponibilidade e as redistribua quando houver um desequilíbrio, em Rebalanceamento do serviço em zonas de disponibilidade, selecione Rebalanceamento do serviço em zonas de disponibilidade.

d. Em Período de carência de verificação de integridade, insira quanto tempo (em segundos) após o início de uma tarefa o agendador do serviço ignorará as verificações de integridade que forem não íntegras para o Elastic Load Balancing, a VPC Lattice e o contêiner. Se você não especificar um valor de período de tolerância de verificação de integridade, o valor padrão 0 será usado.

e. Determine o tipo de implantação do seu serviço. Expanda Opções de implantação e especifique os parâmetros a seguir.

Tipo de implantação	Etapas	
Atualização contínua	a. Em Min running tasks (Mínimo de tarefas em execução), insira o limite inferior do número de tarefas do serviço que devem permanecer no estado `RUNNING` durante uma implantação, como uma porcentagem do número desejado de tarefas (arredondado para o número inteiro superior mais próximo). Para obter mais informações, consulte Configuração da implantação. b. Em Max running tasks (Máximo de tarefas em execução), insira o limite superior do número de tarefas do serviço que devem permanecer no estado `RUNNING` ou `PENDING` durante uma implantação, como uma porcentagem do número desejado de tarefas (arredondado para o número inteiro inferior mais próximo).	

Tipo de implantação	Etapas	
Implantação azul-verde	a. Em Configuração da implantação, escolha como o CodeDeploy roteará o tráfego de produção para o conjunto de tarefas de substituição durante uma implantação. b. Em Perfil de serviço do CodeDeploy, escolha o perfil do IAM que o serviço usa para fazer solicitações de API aos Serviços da AWS autorizados.	

f. Para configurar como o Amazon ECS detectará e lidará com falhas de implantação, expanda Deployment failure detection(Detecção de falhas de implantação) e escolha suas opções.

   i. Para interromper uma implantação quando as tarefas não podem ser iniciadas, selecione Use the Amazon ECS deployment circuit breaker) (Usar o disjuntor de implantação do Amazon ECS.

      Para que o software reverta automaticamente a implantação para o último estado de implantação concluído quando o disjuntor de implantação definir a implantação para um estado de falha, selecione Reverter em caso de falha.

   ii. Para interromper uma implantação com base nas métricas da aplicação, selecione Usar alarmes do CloudWatch. Em seguida, em Nomes de alarmes do CloudWatch, escolha os alarmes. Para criar um alarme, acesse o console do CloudWatch.

      Para que o software reverta automaticamente a implantação para o último estado de implantação concluído quando um alarme do CloudWatch definir a implantação para um estado de falha, selecione Reverter em caso de falha.

7. Se a definição de tarefa usar o modo de rede `awsvpc`, você poderá especificar uma configuração de rede personalizada, expandir Rede e fazer o seguinte:

   a. Em VPC, selecione a VPC a ser usada.

   b. Em Subnets (Sub-redes), selecione uma ou mais sub-redes na VPC que o programador de tarefas levará em consideração ao posicionar as tarefas.

   c. Em Security group (Grupo de segurança), você pode selecionar um grupo de segurança existente ou criar outro. Para usar um grupo de segurança existente, selecione o grupo de segurança e vá para a próxima etapa. Para criar um novo grupo de segurança, escolha Create a new security group (Criar um novo grupo de segurança). Você deve especificar o nome de um grupo de segurança, uma descrição e, em seguida, adicionar uma ou mais regras de entrada para o grupo de segurança.

   d. Em Public IP (IP público), escolha se um endereço IP público deve ou não ser atribuído automaticamente à interface de rede elástica (ENI) da tarefa.

      Tarefas do AWS Fargate podem receber um endereço IP público quando são executadas em uma sub-rede pública para terem uma rota para a Internet. As tarefas do EC2 não podem ser atribuídas a um IP público usando esse campo. Para obter mais informações, consulte as opções de rede de tarefas do Amazon ECS para o tipo de lançamento do Fargate e Alocar uma interface de rede para uma tarefa do Amazon ECS.

8. (Opcional) Para interconectar o serviço usando o Service Connect, expanda Service Connect e especifique o seguinte:

   a. Selecione Ativar o Service Connect.

   b. Em Service Connect configuration (Configuração do Service Connect), especifique o modo cliente.

      - Se seu serviço executa uma aplicação cliente de rede que só precisa se conectar a outros serviços no namespace, escolha Somente no lado do cliente.
      - Se seu serviço executa uma aplicação de rede ou de serviço Web e precisa fornecer endpoints para esse serviço e se conectar a outros serviços no namespace, escolha Client and server (Cliente e servidor).

   c. Para usar um namespace que não seja o namespace padrão do cluster, em Namespace, escolha o namespace do serviço.

   d. (Opcional) Especifique uma configuração de log. Selecione Usar conjunto de logs. A opção padrão envia os logs do contêiner ao CloudWatch Logs. As outras opções de driver de log

são configuradas usando o AWS FireLens. Para obter mais informações, consulte Envio de logs do Amazon ECS para um serviço da AWS ou para uma AWS Partner.

Veja a seguir a descrição de cada destino de log de contêiner mais detalhadamente.

- Amazon CloudWatch: configura a tarefa para enviar logs de contêiner ao CloudWatch Logs. São fornecidas as opções de driver de log padrão, que criam um grupo de logs do CloudWatch em seu nome. Para especificar um nome de grupo de logs diferente, altere os valores da opção de driver.
- Amazon Data Firehose: configura a tarefa para enviar logs de contêiner ao Firehose. São fornecidas as opções de driver de log padrão, que enviam logs para um fluxo de entrega do Firehose. Para especificar um nome de fluxo de entrega diferente, altere os valores da opção de driver.
- Amazon Kinesis Data Streams: configura a tarefa para enviar logs de contêiner ao Kinesis Data Streams. São fornecidas as opções de driver de log padrão, que enviam logs a um fluxo do Kinesis Data Streams. Para especificar um nome de transmissão diferente, altere os valores da opção de driver.
- Amazon OpenSearch Service: configura a tarefa para enviar logs de contêiner a um domínio do OpenSearch Service. As opções de driver de log devem ser fornecidas.
- Amazon S3: configura a tarefa para enviar logs de contêiner a um bucket do Amazon S3. As opções de driver de log padrão são fornecidas, mas você deve especificar um nome de bucket válido do Amazon S3.

9. (Opcional) Para interconectar o serviço usando a descoberta de serviços, expanda Descoberta de serviços e faça o seguinte:

   a. Selecione Usar descoberta de serviços.

   b. Para usar um novo namespace, escolha Criar um namespace em Configurar namespace e forneça um nome e uma descrição para o namespace. Para usar um namespace existente, escolha Selecionar um namespace existente e escolha o namespace desejado.

   c. Forneça as informações de serviço da descoberta de serviços, como nome e descrição do serviço.

   d. Para que o Amazon ECS realize verificações de integridade periódicas em nível de contêiner, selecione Habilitar propagação de integridade de tarefas do Amazon ECS.

   e. Em DNS record type (Tipo de registro DNS), selecione o tipo de registro DNS a ser criado para o serviço. A descoberta de serviços do Amazon ECS só é compatível com registros A e SRV, dependendo do modo de rede que a definição de tarefa especifica. Para obter

informações sobre esses tipos de registro, consulte Tipos de registro DNS compatíveis no Guia do desenvolvedor do Amazon Route 53.

- Se a definição de tarefa que a sua tarefa de serviço especifica usar o modo de rede `bridge` ou `host`, somente haverá suporte para os registros do tipo SRV. Escolha o nome do contêiner e a combinação de portas a serem associadas ao registro.
- Se a definição de tarefa que a sua tarefa de serviço especifica usar o modo de rede `awsvpc`, selecione o tipo de registro A ou SRV. Caso escolha A, vá para a próxima etapa. Se você escolher SRV, especifique a porta na qual o serviço pode ser encontrado ou o nome do contêiner e a combinação de portas a serem associadas ao registro.

Em TTL, insira a duração, em segundos, do tempo em que um conjunto de registros é armazenado em cache pelos resolvedores de DNS e pelos navegadores da Web.

10. (Opcional) Para interconectar o serviço usando o VPC Lattice, expanda VPC Lattice e faça o seguinte:

    a. Selecione Ativar o VPC Lattice

    b. Em Função de infraestrutura, escolha o perfil de infraestrutura.

    Se você não criou um perfil, escolha Criar perfil de infraestrutura.

    c. Em Grupos de destino, escolha um ou mais grupos de destino. Escolha pelo menos um grupo de destino e, no máximo, cinco. Escolha Adicionar grupo de destino para adicionar mais grupos de destino. Escolha o Nome da porta, o Protocolo e a Porta para cada grupo de destino escolhido.

    Para excluir um grupo de destino, escolha Remover.

    > **Note**
    >
    > - Se desejar adicionar grupos de destino existentes, será necessário usar a AWS CLI. Para obter instruções sobre como adicionar grupos de destino usando o AWS CLI, consulte register-targets na Referência do AWS Command Line Interface.
    > - Embora um serviço do VPC Lattice possa ter vários grupos de destino, cada grupo de destino só pode ser adicionado a um serviço.

d. Para concluir a configuração do VPC Lattice, inclua seus novos grupos de destino na ação padrão do receptor ou nas regras de um serviço existente do VPC Lattice no console do VPC Lattice. Para obter mais informações, consulte Regras de receptor para o serviço do VPC Lattice.

11. (Opcional) Para configurar um balanceador de carga para o serviço, expanda Load balancing (Balanceamento de carga).

    Selecione o balanceador de carga.

Para usar esse balanceador de carga	Faça o seguinte	
Application Load Balancer	a. Em Load balancer type (Tipo de balanceador de carga), selecione Application Load Balancer. b. Escolha Create a new load balancer (Criar um novo balanceador de carga) para criar um novo Application Load Balancer ou Use an existing load balancer (Usar um balanceador de carga existente) para selecionar um Application Load Balancer existente. c. Em Load balancer (Balanceador de carga), insira um nome exclusivo. d. Em Choose container to load balance (Escolher contêiner para balancear a carga), escolha o contêiner que hospeda o serviço.	

Para usar esse balanceador de carga	Faça o seguinte	
	e. Em Listener (Receptor), insira uma porta e um protocolo para que o Application Load Balancer receba solicitações de conexão. Por padrão, o balanceador de carga será configurado para usar a porta 80 e HTTP. f. Em Target group name (Nome do grupo de destino), insira um nome e um protocolo para o grupo de destino ao qual o Application Load Balancer encaminhará as solicitações. Por padrão, o grupo de destino encaminha solicitações ao primeiro contêiner especificado na definição de tarefa. g. Em Atraso no cancelamento de registro, insira o número de segundos para que o balanceador de carga altere o estado de destino para `UNUSED`. O padrão é trezentos segundos. h. Em Health check path (Caminho da verificação de integridade), insira um caminho existente no	

Para usar esse balanceador de carga	Faça o seguinte	
	contêiner para o qual o Application Load Balancer enviará periodicamente solicitações para verificar a integridade da conexão entre o Application Load Balancer e o contêiner. O padrão é o diretório raiz (/).	

Para usar esse balanceador de carga	Faça o seguinte
Network Load Balancer	a. Em Load balancer type (Tipo de balanceador de carga), selecione Network Load Balancer. b. Em Load Balancer (Balanceador de carga), escolha um Network Load Balancer existente. c. Em Choose container to load balance (Escolher contêiner para balancear a carga), escolha o contêiner que hospeda o serviço. d. Em Target group name (Nome do grupo de destino), insira um nome e um protocolo para o grupo de destino ao qual o Network Load Balancer encaminhará solicitaç ões. Por padrão, o grupo de destino encaminha solicitações ao primeiro contêiner especificado na definição de tarefa. e. Em Atraso no cancelame nto de registro, insira o número de segundos para que o balanceador de carga altere o estado de destino para `UNUSED`.

Para usar esse balanceador de carga	Faça o seguinte	
	O padrão é trezentos segundos. f. Em Health check path (Caminho da verificação de integridade), insira um caminho existente no contêiner para o qual o Network Load Balancer envia periodicamente solicitações para verificar a integridade da conexão entre o Application Load Balancer e o contêiner. O padrão é o diretório raiz (/).	

12. (Opcional) Para configurar o ajuste de escala automático do serviço, expanda Ajuste de escala automático do serviço e especifique os parâmetros a seguir. Para usar o ajuste de escala automático preditivo, que utiliza os dados de carga anteriores dos fluxos de tráfego, configure-o depois de criar o serviço. Para obter mais informações, consulte Usar padrões históricos para escalar os serviços do Amazon ECS com escalabilidade preditiva.

    a. Para usar a escalabilidade automática do serviço, selecione Service auto scaling (Escalabilidade automática do serviço).

    b. Em Número mínimo de tarefas, insira o limite inferior do número de tarefas a serem usadas pelo ajuste de escala automático. A contagem desejada não será inferior a essa contagem.

    c. Em Número máximo de tarefas, insira o limite superior do número de tarefas a serem usadas pelo ajuste de escala automático. A contagem desejada não ultrapassará essa contagem.

    d. Escolha o tipo de política. Em Tipo de política de ajuste de escala, escolha uma das opções a seguir.

Para usar este tipo de política	Faça o seguinte	
Monitoramento do objetivo	a. Em Scaling policy type (Tipo de política de escalabilidade), escolha Target tracking (Rastreamento de destino). b. Em Policy name (Nome da política), insira o nome da política. c. Em Métrica de serviço do ECS, selecione uma das métricas a seguir. • ECSServiceAverageCPUUtilization: utilização média da CPU pelo serviço. • ECSServiceAverageMemoryUtilization: utilização média da memória pelo serviço. • ALBRequestCountPerTarget: número de solicitações concluídas por destino em um grupo de destino do Application Load Balancer. d. Em Target value (Valor de destino), insira o valor que o serviço manterá	

Para usar este tipo de política	Faça o seguinte	
	para a métrica seleciona da. e. Em Aumentar a escala horizontalmente no período de espera, insira a quantidade de tempo, em segundos, após uma atividade de aumento na escala (adicionar tarefas) que deve ser concluída antes que outra atividade de aumento na escala possa ser iniciada. f. Em Reduzir a escala horizontalmente no período de espera, insira a quantidade de tempo, em segundos, após uma atividade de redução na escala (remover tarefas) que deve ser concluída antes que outra atividade de redução na escala possa ser iniciada. g. Para impedir que a política realize uma atividade de redução da escala na horizontal, selecione Turn off scale-in (Desativar a redução da escala na horizontal).	

Para usar este tipo de política	Faça o seguinte	
	h. • (Opcional) Selecione Desativar redução horizontal de escala se quiser que a política de ajuste de escala aumente a escala horizontalmente para ampliar o tráfego, mas não precisar que ela seja reduzida quando o tráfego diminuir.	

Para usar este tipo de política	Faça o seguinte	
Escalabilidade em etapas	a. Em Scaling policy type (Tipo de política de escalabilidade), escolha Step scaling (Escalabilidade em etapas). b. Em Nome da política, insira o nome da política. c. Em Alarm name (Nome do alarme), insira um nome exclusivo para o alarme. d. Em Métrica de serviço do Amazon ECS, escolha a métrica a ser usada para o alarme. e. Em Estatística, escolha a estatística de alarme. f. Em Período, escolha o período do alarme. g. Em Condição de alarme, escolha como comparar a métrica selecionada com o limite definido. h. Em Limite para comparar métricas e Período de avaliação para iniciar o alarme, insira o limite usado para o alarme e por quanto tempo avaliar o limite. i. Em Ações de escalabilidade, faça o seguinte:	

Para usar este tipo de política	Faça o seguinte	
	• Em Ação, selecione se deseja adicionar , remover ou definir uma contagem desejada específica para o serviço. • Se você decidir adicionar ou remover tarefas, em Valor, insira o número de tarefas (ou o percentual de tarefas existentes) para adicionar ou remover quando a ação de ajuste de escala for iniciada. Se você optar por definir a contagem desejada, insira o número de tarefas. Em Tipo, selecione se o Valor é um valor inteiro ou percentual da contagem desejada existente. • Em Limite inferior e Limite superior, insira o limite inferior e o limite superior do seu ajuste de escalabil idade de etapas. Por padrão, o limite inferior de uma política	

Para usar este tipo de política	Faça o seguinte	
	de adição é o limite do alarme e o limite superior é mais (+) infinito. Por padrão, o limite superior de uma política de remoção é o limite do alarme e o limite inferior é menos (-) infinito. • (Opcional) Adicione outras opções de escalabilidade. Escolha Adicionar nova ação de escalabilidade e repita as etapas de Ações de escalabilidade. • Em Período de espera, insira a quantidade de tempo, em segundos, para aguardar que uma atividade de ajuste de escala anterior entre em vigor. Para uma política de adição, esse é o momento após uma atividade de aumento horizontal de escala em que a política de ajuste de escala bloqueia as atividades de redução	

Para usar este tipo de política	Faça o seguinte	
	horizontal de escala e limita quantas tarefas podem ser ampliadas por vez. Para uma política de remoção, esse é o momento após uma atividade de redução horizontal de escala que precisa ser concluída antes que outras atividades do mesmo tipo possam iniciar.	

13. (Opcional) Para usar uma estratégia de posicionamento de tarefas diferente da padrão, expanda Task Placement (Posicionamento de tarefas) e escolha uma das opções a seguir.

    Para obter mais informações, consulte Como o Amazon ECS posiciona tarefas em instâncias de contêineres.

    - Distribuição balanceada de AZ: distribua tarefas por zonas de disponibilidade e entre instâncias de contêiner na zona de disponibilidade.
    - BinPack balanceado de AZ: distribua tarefas por zonas de disponibilidade e entre instâncias de contêiner com a menor memória disponível.
    - BinPack: distribua tarefas com base na menor quantidade disponível de CPU ou memória.
    - Uma tarefa por host: posicione, no máximo, uma tarefa do serviço em cada instância de contêiner.
    - Personalizado: defina sua própria estratégia de posicionamento de tarefas.

    Se você escolheu Custom (Personalizado), defina o algoritmo de modo a posicionar as tarefas e as regras que serão consideradas durante o posicionamento de tarefas.

    - Em Strategy (Estratégia), em Type (Tipo) e Field (Campo), escolha o algoritmo e a entidade a serem usados com o algoritmo.

É possível adicionar no máximo cinco estratégias.

- Em Restrição, para Tipo e Expressão, escolha a regra e o atributo para a restrição.

  Por exemplo, para definir a restrição de modo a posicionar tarefas em instâncias T2, em Expression (Expressão), insira attribute:ecs.instance-type =~ t2.*.

  É possível adicionar no máximo dez restrições.

14. Caso a tarefa use um volume de dados compatível com a configuração na implantação, você pode configurar o volume expandindo Volume.

    O nome e o tipo do volume são configurados ao criar uma revisão de definição de tarefa e não podem ser alterados ao criar um serviço. Para atualizar o nome e o tipo do volume, você deve criar uma revisão de definição de tarefa e criar um serviço usando a nova revisão.

Para configurar esse tipo de volume	Faça o seguinte
Amazon EBS	a. Em Tipo de volume do EBS, escolha o tipo de volume do EBS que você deseja anexar à tarefa. b. Em Tamanho (GiB), insira um valor válido para o tamanho do volume em gibibytes (GiB). Você pode especificar um tamanho de volume mínimo de 1 GiB e máximo de 16.384 GiB. Esse valor é obrigatório, a menos que você forneça um ID de snapshot. c. Em IOPS, insira o número máximo de operações de entrada e saída (IOPS) que o volume deve

Para configurar esse tipo de volume	Faça o seguinte	
	oferecer. Esse valor é configurável somente em tipos de volumes `io1`, `io2` e `gp3`. d. Em Throughput (MiB/s), insira o throughput que o volume deve fornecer, em mebibytes por segundo (MiBps ou MiB/s). Esse valor é configurável somente no tipo de volume `gp3`. e. Em ID do snapshot, escolha um snapshot de volume existente do Amazon EBS ou insira o ARN de um snapshot se quiser criar um volume usando um snapshot. Você também pode criar um volume vazio sem escolher ou inserir um ID de snapshot. f. Se você especificar um ID de snapshot, poderá especificar uma Taxa de inicialização do volume (MiB/s). Insira um valor entre 100 e 300, em MiB/s, que determinará a rapidez com que os dados são carregados do snapshot especificado	

Para configurar esse tipo de volume	Faça o seguinte	
	usando o ID do snapshot para a criação do volume. g. Em Tipo de sistema de arquivos, escolha o tipo de sistema de arquivos que será usado para armazenamento e recuperação de dados no volume. Você pode escolher o sistema operacional padrão ou um tipo específico de sistema de arquivos. O padrão no Linux é XFS. Em volumes criados de um snapshot, você deve especificar o mesmo tipo de sistema de arquivos que o volume estava usando quando o snapshot foi criado. Se houver uma incompatibilidade no tipo do sistema de arquivos, a tarefa não será iniciada. h. Em Perfil de infraestrutura, escolha um perfil do IAM com as permissões necessárias para permitir que o Amazon ECS gerencie volumes do Amazon EBS em tarefas. Você pode anexar a política gerenciada	

Para configurar esse tipo de volume	Faça o seguinte	
	`AmazonECSInfrastructureRolePolicyForVolumes` ao perfil ou usar a política como guia para criar e anexar sua própria política com permissões que atendam às suas necessidades específicas. Para obter mais informações sobre as permissões necessárias, consulte Perfil do IAM de infraestrutura do Amazon ECS. i. Em Criptografia, escolha Padrão se deseja usar a criptografia do Amazon EBS por meio das configurações padrão. Se a conta tiver a Criptografia por padrão configurada, o volume será criptografado com a chave do AWS Key Management Service (AWS KMS) especificada na configuração. Caso escolha Padrão e a criptografia padrão do Amazon EBS não estiver ativada, o volume não será criptografado. Caso escolha Personalizada, você pode especific	

Para configurar esse tipo de volume	Faça o seguinte	
	ar uma AWS KMS key de sua escolha para criptografia de volume. Caso escolha Nenhum, o volume não será criptografado, a menos que você tenha a criptografia por padrão configurada ou crie um volume usando um snapshot criptografado. j. Caso tenha escolhido Personalizada em Criptografia, você deve especificar a AWS KMS key que deseja usar. Em Chave do KMS, escolha uma AWS KMS key ou insira um ARN de chave. Se você optar por criptografar o volume usando uma chave simétrica gerenciada pelo cliente, verifique se tem as permissões corretas definidas na política do AWS KMS key. Para obter mais informações, consulte Data encryption for Amazon EBS volumes. k. (Opcional) Em Tags, você pode adicionar tags ao volume do Amazon	

Para configurar esse tipo de volume	Faça o seguinte	
	EBS propagando tags da definição de tarefa ou do serviço ou fornecendo suas próprias tags. Se quiser propagar tags na definição de tarefa, escolha Definição de tarefa em Propagar tags em. Se quiser propagar tags do serviço, escolha Serviço em Propagar tags de. Caso escolha Não propagar ou se não selecionar um valor, as tags não são propagadas. Se quiser fornecer suas próprias tags, escolha Adicionar tag e forneça a chave e o valor de cada tag adicionada. Para obter mais informaçõ es sobre marcação de volumes do Amazon EBS, consulte Tagging Amazon EBS volumes.	

15. (Opcional) Para ajudar a identificar seu serviço e tarefas, expanda a seção Tags (Etiquetas) e configure suas etiquetas.

    Para que o Amazon ECS atribua tags automaticamente a todas as tarefas recém-iniciadas com o nome do cluster e as tags de definição de tarefa, selecione Ativar tags gerenciadas pelo Amazon ECS e, para Propagar tags de, escolha Definições de tarefas.

Para que o Amazon ECS atribua tags automaticamente a todas as tarefas recém-iniciadas com o nome do cluster e as tags de serviços, selecione Ativar tags gerenciadas pelo Amazon ECS e, para Propagar tags de, escolha Serviço.

Adicione ou remova uma tag.

- [Adicionar uma etiqueta] Escolha Add tag (Adicionar etiqueta) e faça o seguinte:
  - Em Chave, insira o nome da chave.
  - Em Valor insira o valor da chave.
- [Remover uma tag] Ao lado da tag, escolha Remove tag (Remover tag).

16. Selecione Create (Criar).

Próximas etapas

A seguir, estão as ações adicionais após criar um serviço.

- Configure o ajuste de escala automático preditivo, que utiliza os dados de carga anteriores dos fluxos de tráfego. Para obter mais informações, consulte Usar padrões históricos para escalar os serviços do Amazon ECS com escalabilidade preditiva.
- Acompanhe sua implantação e visualize seu histórico de serviços para os disjuntores do Amazon ECS. Para obter mais informações, consulte Visualize o histórico de serviços usando implantações de serviços do Amazon ECS.

Implantações azuis/verdes do Amazon ECS

Uma implantação azul/verde é uma metodologia de liberação que reduz o tempo de inatividade e o risco ao executar dois ambientes de produção idênticos chamados azul e verde. Com as implantações azul/verde do Amazon ECS, você pode validar novas revisões de serviços antes de direcionar o tráfego de produção para elas. Essa abordagem fornece uma maneira mais segura de implantar alterações com a capacidade de revertê-las rapidamente, se necessário.

Benefícios

O uso de implantações azul/verde oferece os seguintes benefícios:

- Reduz o risco por meio de testes com o tráfego de produção antes de mudar para a produção. Você pode validar a nova implantação com tráfego de teste antes de direcionar o tráfego de produção para ela.
- Implementações sem tempo de inatividade. O ambiente de produção permanece disponível durante todo o processo de implantação, garantindo a disponibilidade contínua do serviço.
- Fácil reversão se forem detectados problemas. Se surgirem problemas com a implantação verde, você poderá reverter rapidamente para a implantação azul sem interrupção prolongada do serviço.
- Ambiente de teste controlado. O ambiente verde fornece um espaço isolado para testar novos recursos com padrões reais de tráfego antes da implantação completa.
- Processo de implantação previsível. A abordagem estruturada com estágios de ciclo de vida definidos torna as implantações mais consistentes e confiáveis.
- Validação automatizada por meio de ganchos do ciclo de vida. Você pode implementar testes automatizados em vários estágios da implantação para verificar a funcionalidade.

Terminologia

Confira abaixo os termos de implantação azul/verde do Amazon ECS:

- Tempo de incorporação: a duração em que as revisões de serviço azul e verde são executadas simultaneamente após a mudança do tráfego de produção.
- Implantação azul: a revisão atual do serviço de produção que você deseja substituir.
- Implantação verde: a nova revisão do serviço que você deseja implantar.
- Estágios do ciclo de vida: uma série de eventos na operação de implantação, como “após a mudança no tráfego de produção”.
- Gancho do ciclo de vida: uma função do Lambda que verifica a implantação em um estágio específico do ciclo de vida.
- Receptor: um recurso do Elastic Load Balancing que verifica solicitações de conexão usando o protocolo e a porta que você configurou. As regras que você define para um receptor determinam como o Amazon ECS roteia solicitações para seus destinos registrados.
- Regra: um recurso do Elastic Load Balancing associado a um receptor. Uma regra define como as solicitações são roteadas e consiste em uma ação, condição e prioridade.
- Grupo de destino: um recurso do Elastic Load Balancing usado para rotear solicitações para um ou mais destinos registrados (por exemplo, instâncias do EC2). Ao criar um listener, especifique um grupo de destino para a ação padrão dele. O tráfego é encaminhado para o grupo de destino especificado na regra do listener.

- Mudança de tráfego: o processo que o Amazon ECS usa para transferir o tráfego da implantação azul para a implantação verde. Para implantações azul/verde do Amazon ECS, todo o tráfego é transferido do serviço azul para o serviço verde de uma só vez.

Considerações

Considere o seguinte ao escolher um tipo de implantação:

- Uso de recursos: as implantações azul/verde executam temporariamente as revisões do serviço azul e verde simultaneamente, o que pode dobrar o uso de recursos durante as implantações.
- Monitoramento da implantação: as implantações azul/verde fornecem informações mais detalhadas sobre o status da implantação, permitindo que você monitore cada estágio do processo de implantação.
- Reversão: as implantações azul/verde facilitam a reversão para a versão anterior se forem detectados problemas, pois a revisão azul é mantida em execução até que o tempo de incorporação expire.

Fluxo de trabalho de implantações de serviços azul/verde do Amazon ECS

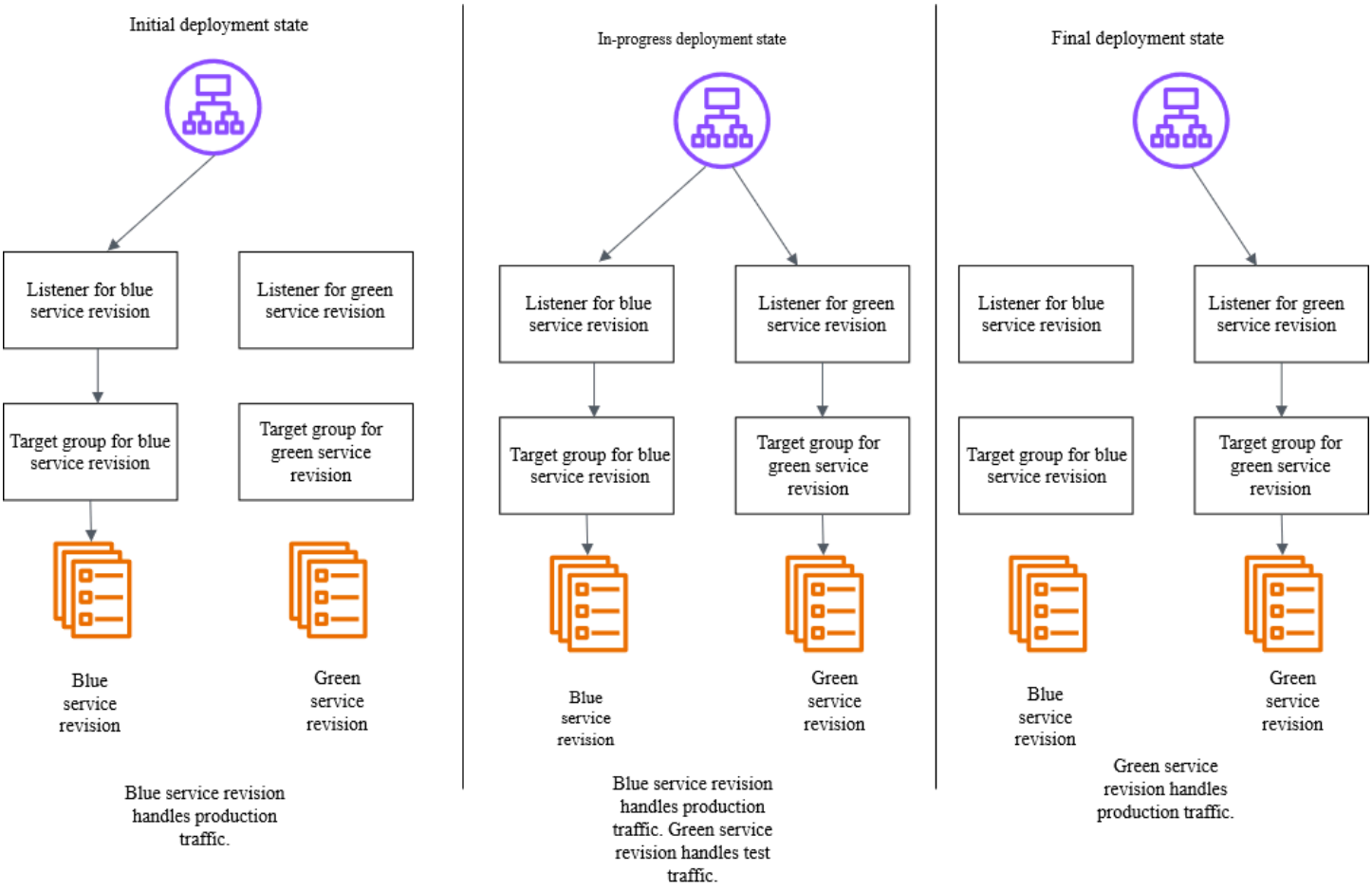
O processo de implantação azul/verde do Amazon ECS segue uma abordagem estruturada com seis fases distintas que garantem atualizações de aplicações seguras e confiáveis. Cada fase tem um propósito específico na validação e transição da aplicação da versão atual (azul) para a nova versão (verde).

1. Fase de preparação: crie o ambiente verde junto com o ambiente azul existente. Isso inclui o provisionamento de novas revisões de serviços e a preparação de grupos de destino.
2. Fase de implantação: implante a nova revisão do serviço no ambiente verde. O Amazon ECS lança novas tarefas usando a revisão atualizada do serviço, enquanto o ambiente azul continua atendendo ao tráfego de produção.
3. Fase de teste: valide o ambiente verde usando o roteamento de tráfego de teste. O Application Load Balancer direciona as solicitações de teste para o ambiente verde, enquanto o tráfego de produção permanece no azul.
4. Fase de mudança de tráfego: mude o tráfego de produção do azul para o verde com base na sua estratégia de implantação configurada. Essa fase inclui pontos de verificação de monitoramento e validação.

5. Fase de monitoramento: monitore a integridade da aplicação, as métricas de performance e os estados dos alarmes durante o tempo de incorporação. Uma operação de reversão é iniciada quando problemas são detectados.
6. Fase de conclusão: finalize a implantação encerrando o ambiente azul ou mantendo-o para possíveis cenários de reversão, dependendo da sua configuração.

Fluxo de trabalho

O diagrama a seguir ilustra o fluxo de trabalho abrangente de implantação azul/verde, mostrando a interação entre o Amazon ECS e o Application Load Balancer:



Esse fluxo de trabalho de implantação aprimorado inclui as seguintes etapas detalhadas:

1. Estado inicial: o serviço azul (produção atual) gerencia 100% do tráfego de produção. O Application Load Balancer tem um único receptor com regras que direcionam todas as solicitações para o grupo de destino azul contendo tarefas azuis íntegras.

2. Provisionamento de ambiente verde: o Amazon ECS cria novas tarefas usando a definição atualizada de tarefas. Essas tarefas são registradas em um novo grupo de destino verde, mas inicialmente não recebem tráfego.
3. Validação da verificação de integridade: o Application Load Balancer realiza verificações de integridade em tarefas verdes. Somente quando as tarefas verdes passam pelas verificações de integridade, a implantação vai para a próxima fase.
4. Roteamento de tráfego de teste: se configurado, as regras de receptor do Application Load Balancer direcionam padrões de tráfego específicos (como solicitações com cabeçalhos de teste) para o ambiente verde para validação, enquanto o tráfego de produção permanece no azul. Isso é controlado pelo mesmo receptor que manipula o tráfego de produção, usando regras diferentes com base nos atributos da solicitação.
5. Mudança de tráfego de produção: com base na configuração de implantação, o tráfego muda de azul para verde. Nas implantações azul/verde do ECS, esta é uma mudança imediata (tudo de uma vez), em que 100% do tráfego é movido do ambiente azul para o verde. O Application Load Balancer usa um único receptor com regras de receptor que controlam a distribuição do tráfego entre os grupos de destino azul e verde com base nos pesos.
6. Monitoramento e validação: durante toda a mudança de tráfego, o Amazon ECS monitora as métricas, os estados dos alarmes e a integridade da implantação do CloudWatch. Os gatilhos de reversão automática serão ativados se forem detectados problemas.
7. Período de tempo de incorporação: a duração em que as revisões de serviço azul e verde são executadas simultaneamente após a mudança do tráfego de produção.
8. Encerramento do ambiente azul: após a mudança e a validação bem-sucedidas do tráfego, o ambiente azul é encerrado para liberar recursos do cluster, ou é mantido para capacidade de reversão rápida.
9. Estado final: o ambiente verde torna-se o novo ambiente de produção, lidando com 100% do tráfego. A implantação está marcada como bem-sucedida.

Estágios do ciclo de vida de implantação

O processo de implantação azul/verde avança em estágios distintos do ciclo de vida (uma série de eventos na operação de implantação, como “após a mudança do tráfego de produção”), cada um com responsabilidades específicas e pontos de verificação de validação. A compreensão desses estágios ajuda você a monitorar o andamento da implantação e solucionar problemas de forma eficaz.

Cada estágio do ciclo de vida pode durar até 24 horas. Recomendamos que o valor permaneça abaixo da marca de 24 horas. Isso ocorre porque os processos assíncronos precisam de tempo para acionar os ganchos. O sistema atinge o tempo limite, falha na implantação e, em seguida, inicia uma reversão após um estágio atingir 24 horas. As implantações do AWS CloudFormation têm restrições adicionais de tempo limite. Embora o limite de estágio de 24 horas permaneça em vigor, o AWS CloudFormation impõe um limite de 36 horas em toda a implantação. A implantação do AWS CloudFormation falha e, em seguida, iniciará uma reversão se o processo não for concluído em até 36 horas.

Estágios do ciclo de vida	Descrição	Use este estágio para o gancho do ciclo de vida?
RECONCILE_SERVICE	Este estágio só acontece quando você inicia uma nova implantação de serviço com mais de uma revisão de serviço em um estado ACTIVE.	Sim
PRE_SCALE_UP	A revisão do serviço verde ainda não foi iniciada. A revisão do serviço azul está processando 100% do tráfego de produção. Não há tráfego de teste.	Sim
SCALE_UP	O momento em que a revisão do serviço verde aumenta a escala verticalmente até 100% e inicia novas tarefas. A revisão do serviço verde não está recebendo nenhum tráfego neste momento.	Não
POST_SCALE_UP	A revisão do serviço verde foi iniciada. A revisão do serviço azul está processan do 100% do tráfego de produção. Não há tráfego de teste.	Sim
TEST_TRAFFIC_SHIFT	As revisões do serviço azul e verde estão em execução. A revisão do serviço azul processa 100% do tráfego de produção.	Sim

Estágios do ciclo de vida	Descrição	Use este estágio para o gancho do ciclo de vida?
	A revisão do serviço verde está migrando de 0% para 100% do tráfego de teste.	
POST_TEST_TRAFFIC_SHIFT	A mudança de tráfego de teste foi concluída. A revisão do serviço verde processa 100% do tráfego de teste.	Sim
PRODUCTION_TRAFFIC_SHIFT	O tráfego de produção está mudando para a revisão do serviço verde. A revisão do serviço verde está migrando de 0% para 100% do tráfego de produção.	Sim
POST_PRODUCTION_TRAFFIC_SHIFT	A mudança do tráfego de produção está concluída.	Sim
BAKE_TIME	A duração em que as revisões de serviço azul e verde são executadas simultaneamente.	Não
CLEAN_UP	A revisão do serviço azul teve a escala reduzida verticalmente por completo para 0 tarefa em execução. A revisão do serviço verde agora é a revisão do serviço de produção após esse estágio.	Não

Cada estágio do ciclo de vida inclui pontos de verificação de validação integrados que devem ser aprovados antes de prosseguir para a próxima etapa. Se alguma validação falhar, a implantação poderá ser revertida automaticamente para manter a disponibilidade e a confiabilidade do serviço.

Recursos necessários para implantações azul/verde do Amazon ECS

Para usar uma implantação azul/verde com mudança de tráfego gerenciada, seu serviço deve usar um dos seguintes recursos:

- Elastic Load Balancing

- Service Connect

Os serviços que não usam a Descoberta de serviços, o Service Connect, o VPC Lattice ou o Elastic Load Balancing também podem usar implantações azul/verde, mas não obtêm nenhum dos benefícios de transferência de tráfego gerenciado.

A lista abaixo fornece uma visão geral de alto nível do que você precisa configurar para implantações azul/verde do Amazon ECS:

- Seu serviço usa o Application Load Balancer, o Network Load Balancer ou o Service Connect. Configure os recursos apropriados.
  - Application Load Balancer: para obter mais informações, consulte Recursos do Application Load Balancer para implantações azul/verde.
  - Network Load Balancer: para obter mais informações, consulte Recursos do Network Load Balancer para implantações azul/verde do Amazon ECS.
  - Service Connect: para obter mais informações, consulte Recursos do Service Connect para implantações azul/verde do Amazon ECS.
- Defina o controlador de implantação do serviço para `ECS`.
- Configure a estratégia de implantação como `blue/green` na sua definição de serviço.
- Opcionalmente, configure parâmetros adicionais, como:
  - Tempo de incorporação para a nova implantação
  - Alarmes do CloudWatch para reversão automática
  - Ganchos do ciclo de vida de implantação para testes (são funções do Lambda que são executadas em estágios de implantação especificados)

Práticas recomendadas

Siga estas práticas recomendadas para implantações azul/verde do Amazon ECS:

- Configure as verificações de integridade apropriadas que reflitam com precisão a integridade da sua aplicação.
- Defina um tempo de incorporação que permita testes suficientes da implantação verde.
- Implemente alarmes do CloudWatch para detectar problemas automaticamente e acionar reversões.
- Use ganchos do ciclo de vida para realizar testes automatizados em cada estágio da implantação.

- Garanta que sua aplicação possa processar ambientes azul e verde em execução simultânea.
- Planeje uma capacidade de cluster suficiente para processar ambos os ambientes durante a implantação.
- Teste seus procedimentos de reversão antes de implementá-los na produção.

Recursos do Application Load Balancer para implantações azul/verde

Para usar Application Load Balancers com implantações azul/verde do Amazon ECS, você precisa configurar recursos específicos que permitam o roteamento de tráfego entre as revisões de serviço azul e verde.

Limitações

As seguintes limitações se aplicam atualmente às implantações azul/verde com Application Load Balancers:

- Não há suporte para implantações azul/verde ao usar receptores do Application Load Balancer que têm vários tipos de ação configurados com ações sem roteamento. Se as regras do receptor do Application Load Balancer contiverem uma combinação de ações com roteamento e sem roteamento (como redirecionamento, resposta fixa ou autenticação), a implantação falhará.

Grupos de destino

Para implantações azul/verde com o Elastic Load Balancing, você precisa criar dois grupos de destino:

- Um grupo de destino principal para a revisão do serviço azul (tráfego de produção atual)
- Um grupo de destino alternativo para a revisão do serviço verde (nova versão)

Ambos os grupos de destino devem ser configurados com as seguintes configurações:

- Tipo de destino: `IP` (para Fargate ou EC2 com modo de rede `awsvpc`)
- Protocolo: `HTTP` (ou o protocolo que sua aplicação usa)
- Porta: a porta que sua aplicação escuta (normalmente `80` para HTTP)
- VPC: a mesma VPC das suas tarefas do Amazon ECS
- Configurações de verificação de integridade: configuradas para verificar adequadamente a integridade da sua aplicação

Durante uma implantação azul/verde, o Amazon ECS registra automaticamente as tarefas com o grupo de destino apropriado com base no estágio de implantação.

Example Criar grupos de destino para um Application Load Balancer

Os seguintes comandos da CLI criam dois grupos de destino para uso com um Application Load Balancer em uma implantação azul/verde:

```
aws elbv2 create-target-group \
    --name blue-target-group \
    --protocol HTTP \
    --port 80 \
    --vpc-id vpc-abcd1234 \
    --target-type ip \
    --health-check-path / \
    --health-check-protocol HTTP \
    --health-check-interval-seconds 30 \
    --health-check-timeout-seconds 5 \
    --healthy-threshold-count 2 \
    --unhealthy-threshold-count 2

aws elbv2 create-target-group \
    --name green-target-group \
    --protocol HTTP \
    --port 80 \
    --vpc-id vpc-abcd1234 \
    --target-type ip \
    --health-check-path / \
    --health-check-protocol HTTP \
    --health-check-interval-seconds 30 \
    --health-check-timeout-seconds 5 \
    --healthy-threshold-count 2 \
    --unhealthy-threshold-count 2
```

Application Load Balancer

É necessário criar um Application Load Balancer com a seguinte configuração:

- Esquema: voltado para a internet ou interno, dependendo de suas necessidades
- Tipo de endereço IP: IPv4
- VPC: a mesma VPC das suas tarefas do Amazon ECS
- Sub-redes: pelo menos duas sub-redes em diferentes zonas de disponibilidade

- Grupos de segurança: um grupo de segurança que permite tráfego nas portas do receptor

O grupo de segurança anexado ao Application Load Balancer deve ter uma regra de saída que permita o tráfego para o grupo de segurança anexado às suas tarefas do Amazon ECS.

Example Criar um Application Load Balancer

O seguinte comando da CLI cria um Application Load Balancer para ser usado em uma implantação azul/verde:

```
aws elbv2 create-load-balancer \
    --name my-application-load-balancer \
    --type application \
    --security-groups sg-abcd1234 \
    --subnets subnet-12345678 subnet-87654321
```

Receptores e regras

Para implantações azul/verde, você precisa configurar receptores em seu Application Load Balancer:

- Receptor de produção: processa o tráfego de produção (normalmente na porta 80 ou 443)
  - Inicialmente encaminha o tráfego para o grupo de destino principal (revisão de serviço azul)
  - Após a implantação, encaminha o tráfego para o grupo de destino alternativo (revisão de serviço verde)
- Receptor de teste (opcional): processa o tráfego de teste para validar a revisão do serviço verde antes de mudar o tráfego de produção
  - Pode ser configurado em outra porta (por exemplo, 8080 ou 8443)
  - Encaminha o tráfego para o grupo de destino alternativo (revisão de serviço verde) durante o teste

Durante uma implantação azul/verde, o Amazon ECS atualiza automaticamente as regras do receptor para direcionar o tráfego para o grupo de destino apropriado com base no estágio da implantação.

Example Criação de um receptor de produção

O seguinte comando da CLI cria um receptor de produção na porta 80 que encaminha o tráfego para o grupo de destino primário (azul):

```
aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:region:123456789012:loadbalancer/
app/my-application-load-balancer/abcdef123456 \
    --protocol HTTP \
    --port 80 \
    --default-actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
blue-target-group/abcdef123456
```

Example Criação de um receptor de teste

O seguinte comando da CLI cria um receptor de teste na porta 8080 que encaminha o tráfego para o grupo de destino alternativo (verde):

```
aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:region:123456789012:loadbalancer/
app/my-application-load-balancer/abcdef123456 \
    --protocol HTTP \
    --port 8080 \
    --default-actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
green-target-group/ghijkl789012
```

Example Criação de uma regra de receptor para roteamento baseado em caminho

O seguinte comando da CLI cria uma regra que encaminha o tráfego de um caminho específico para o grupo de destino verde para teste:

```
aws elbv2 create-rule \
    --listener-arn arn:aws:elasticloadbalancing:region:123456789012:listener/app/my-
application-load-balancer/abcdef123456/ghijkl789012 \
    --priority 10 \
    --conditions Field=path-pattern,Values='/test/*' \
    --actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
green-target-group/ghijkl789012
```

Example Criação de uma regra de receptor para roteamento baseado em cabeçalho

O seguinte comando da CLI cria uma regra que encaminha o tráfego com um cabeçalho específico para o grupo de destino verde para teste:

```
aws elbv2 create-rule \
    --listener-arn arn:aws:elasticloadbalancing:region:123456789012:listener/app/my-
application-load-balancer/abcdef123456/ghijkl789012 \
    --priority 20 \
    --conditions Field=http-header,HttpHeaderConfig='{Name=X-
Environment,Values=[test]}' \
    --actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
green-target-group/ghijkl789012
```

Configuração de serviço

Você deve ter permissões para permitir que o Amazon ECS gerencie recursos do balanceador de carga nos clusters em seu nome. Para obter mais informações, consulte Perfil do IAM da infraestrutura do Amazon ECS para balanceadores de carga.

Ao criar ou atualizar um serviço do Amazon ECS para implantações azul/verde com o Elastic Load Balancing, você precisa especificar a configuração a seguir.

Substitua os valores das *user-input* pelos seus.

Os principais componentes dessa configuração são:

- `targetGroupArn`: o ARN do grupo de destino principal (revisão de serviço azul).
- `alternateTargetGroupArn`: o ARN do grupo de destino alternativo (revisão de serviço verde).
- `productionListenerRule`: o ARN da regra do receptor para tráfego de produção.
- `roleArn`: o ARN do perfil que permite ao Amazon ECS gerenciar recursos do Elastic Load Balancing.
- `strategy`: defina como `BLUE_GREEN` para habilitar implantações azul/verde.
- `bakeTimeInMinutes`: a duração em que as revisões de serviço azul e verde são executadas simultaneamente após a mudança do tráfego de produção.
- `TestListenerRule`: o ARN da regra do receptor para tráfego de teste. Esse parâmetro é opcional.

```
{
    "loadBalancers": [
        {
```

```
            "targetGroupArn":
 "arn:aws:elasticloadbalancing:region:123456789012:targetgroup/primary-target-group/
abcdef123456",
            "containerName": "container-name",
            "containerPort": 80,
            "advancedConfiguration": {
                "alternateTargetGroupArn":
 "arn:aws:elasticloadbalancing:region:account-id:targetgroup/alternate-target-group/
ghijkl789012",
                "productionListenerRule": "arn:aws:elasticloadbalancing:region:account-
id:listener-rule/app/load-balancer-name/abcdef123456/listener/ghijkl789012/rule/
mnopqr345678",
                "roleArn": "arn:aws:iam::123456789012:role/ecs-elb-role"
            }
        }
    ],
    "deploymentConfiguration": {
        "strategy": "BLUE_GREEN",
        "maximumPercent": 200,
        "minimumHealthyPercent": 100,
        "bakeTimeInMinutes": 5
    }
}
```

Fluxo de tráfego durante a implantação

Durante uma implantação azul/verde com o Elastic Load Balancing, o tráfego flui pelo sistema da seguinte forma:

1. Estado inicial: todo o tráfego de produção é roteado para o grupo de destino principal (revisão de serviço azul).
2. Implantação de revisão de serviço verde: o Amazon ECS implanta as novas tarefas e as registra no grupo de destino alternativo.
3. Tráfego de teste: se um receptor de teste estiver configurado, o tráfego de teste será roteado para o grupo de destino alternativo para validar a revisão do serviço verde.
4. Mudança de tráfego de produção: o Amazon ECS atualiza a regra de receptor de produção para rotear o tráfego para o grupo de destino alternativo (revisão de serviço verde).
5. Tempo de incorporação: a duração em que as revisões de serviço azul e verde são executadas simultaneamente após a mudança do tráfego de produção.
6. Conclusão: após uma implantação bem-sucedida, a revisão do serviço azul é encerrada.

Se forem detectados problemas durante a implantação, o Amazon ECS pode reverter automaticamente roteando o tráfego de volta para o grupo de destino principal (revisão de serviço azul).

Recursos do Network Load Balancer para implantações azul/verde do Amazon ECS

Para usar um Network Load Balancer com implantações azul/verde do Amazon ECS, você precisa configurar recursos específicos que permitam o roteamento de tráfego entre as revisões de serviço azul e verde. Esta seção explica os componentes necessários e suas configurações.

Quando sua configuração inclui um Network Load Balancer, o Amazon ECS adiciona um atraso de dez minutos aos seguintes estágios do ciclo de vida:

- PRE_SCALE_UP
- TEST_TRAFFIC_SHIFT
- PRODUCTION_TRAFFIC_SHIFT

Esse atraso é responsável por problemas de tempo do Network Load Balancer que podem causar uma incompatibilidade entre os pesos de tráfego configurados e o roteamento real do tráfego no plano de dados.

Grupos de destino

Para implantações azul/verde com um Network Load Balancer, você precisa criar dois grupos de destino:

- Um grupo de destino principal para a revisão do serviço azul (tráfego de produção atual)
- Um grupo de destino alternativo para a revisão do serviço verde (nova revisão do serviço)

Ambos os grupos de destino devem ser configurados com as seguintes configurações:

- Tipo de destino: `ip` (para Fargate ou EC2 com modo de rede `awsvpc`)
- Protocolo: `TCP` (ou o protocolo que sua aplicação usa)
- Porta: a porta que sua aplicação escuta (normalmente `80` para HTTP)
- VPC: a mesma VPC das suas tarefas do Amazon ECS
- Configurações de verificação de integridade: configuradas para verificar adequadamente a integridade da sua aplicação

Para verificações de integridade do TCP, o Network Load Balancer estabelece uma conexão TCP com o destino. Se a conexão for bem-sucedida, o destino será considerado íntegro.

Para verificações de integridade de HTTP/HTTPS, o Network Load Balancer envia uma solicitação HTTP/HTTPS ao destino e verifica a resposta.

Durante uma implantação azul/verde, o Amazon ECS registra automaticamente as tarefas com o grupo de destino apropriado com base no estágio de implantação.

Example Criação de grupos de destino para o Network Load Balancer

Os seguintes comandos da AWS CLI criam dois grupos de destino para uso com um Network Load Balancer em uma implantação azul/verde:

```
aws elbv2 create-target-group \
    --name blue-target-group \
    --protocol TCP \
    --port 80 \
    --vpc-id vpc-abcd1234 \
    --target-type ip \
    --health-check-protocol TCP

aws elbv2 create-target-group \
    --name green-target-group \
    --protocol TCP \
    --port 80 \
    --vpc-id vpc-abcd1234 \
    --target-type ip \
    --health-check-protocol TCP
```

Network Load Balancer

É necessário criar um Network Load Balancer com a seguinte configuração:

- Esquema: voltado para a internet ou interno, dependendo de suas necessidades
- Tipo de endereço IP: IPv4
- VPC: a mesma VPC das suas tarefas do Amazon ECS
- Sub-redes: pelo menos duas sub-redes em diferentes zonas de disponibilidade

Diferentemente dos Application Load Balancers, os Network Load Balancers operam na camada de transporte (camada 4) e não usam grupos de segurança. Em vez disso, você precisa garantir que os grupos de segurança associados às tarefas do Amazon ECS permitam o tráfego do Network Load Balancer nas portas do receptor.

Example Criar um Network Load Balancer

O seguinte comando da AWS CLI cria um Network Load Balancer para ser usado em uma implantação azul/verde:

```
aws elbv2 create-load-balancer \
    --name my-network-load-balancer \
    --type network \
    --subnets subnet-12345678 subnet-87654321
```

Considerações sobre o uso do NLB com implantações azul/verde

Ao usar um Network Load Balancer para implantações azul/verde, considere o seguinte:

- Operação da camada 4: os Network Load Balancers operam na camada de transporte (camada 4) e não inspecionam o conteúdo da camada da aplicação (camada 7). Isso significa que você não pode usar cabeçalhos ou caminhos HTTP para decisões de roteamento.
- Verificações de integridade: as verificações de integridade do Network Load Balancer estão limitadas aos protocolos TCP, HTTP ou HTTPS. Para verificações de integridade do TCP, o Network Load Balancer apenas verifica se a conexão pode ser estabelecida.
- Preservação da conexão: os Network Load Balancers preservam o endereço IP de origem do cliente, o que pode ser útil para fins de segurança e registro em log.
- Endereços IP estáticos: os Network Load Balancers fornecem endereços IP estáticos para cada sub-rede, o que pode ser útil para listas de permissões ou quando os clientes precisam se conectar a um endereço IP fixo.
- Tráfego de teste: como os Network Load Balancers não são compatíveis com o roteamento baseado em conteúdo, o tráfego de teste deve ser enviado para outra porta do tráfego de produção.

Receptores e regras

Para implantações azul/verde com um Network Load Balancer, você precisa configurar receptores:

- Receptor de produção: processa o tráfego de produção (normalmente na porta 80 ou 443)
  - Inicialmente encaminha o tráfego para o grupo de destino principal (revisão de serviço azul)
  - Após a implantação, encaminha o tráfego para o grupo de destino alternativo (revisão de serviço verde)
- Receptor de teste (opcional): processa o tráfego de teste para validar a revisão do serviço verde antes de mudar o tráfego de produção
  - Pode ser configurado em outra porta (por exemplo, 8080 ou 8443)
  - Encaminha o tráfego para o grupo de destino alternativo (revisão de serviço verde) durante o teste

Diferentemente dos Application Load Balancers, os Network Load Balancers não são compatíveis com regras de roteamento baseadas em conteúdo. Em vez disso, o tráfego é roteado com base na porta e no protocolo do receptor.

Os seguintes comandos da AWS CLI criam receptores de produção e teste para um Network Load Balancer:

Substitua os valores das *user-input* pelos seus.

```
aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:region:123456789012:loadbalancer/
net/my-network-lb/1234567890123456 \
    --protocol TCP \
    --port 80 \
    --default-actions Type=forward,
 TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/blue-
target-group/1234567890123456

aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:region:123456789012:loadbalancer/
net/my-network-lb/1234567890123456 \
    --protocol TCP \
    --port 8080 \
    --default-actions Type=forward,
 TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/green-
target-group/1234567890123456
```

Configuração de serviço

Você deve ter permissões para permitir que o Amazon ECS gerencie recursos do balanceador de carga nos clusters em seu nome. Para obter mais informações, consulte Perfil do IAM da infraestrutura do Amazon ECS para balanceadores de carga.

Ao criar ou atualizar um serviço do Amazon ECS para implantações azul/verde com um Network Load Balancer, você precisa especificar a seguinte configuração:

Substitua os valores das *user-input* pelos seus.

Os principais componentes dessa configuração são:

- `targetGroupArn`: o ARN do grupo de destino principal (revisão de serviço azul)
- `alternateTargetGroupArn`: o ARN do grupo de destino alternativo (revisão de serviço verde)
- `productionListenerArn`: o ARN do receptor para tráfego de produção
- `testListenerArn`: (opcional) o ARN do receptor para tráfego de teste
- `roleArn`: o ARN do perfil que permite ao Amazon ECS gerenciar recursos do Network Load Balancer
- `strategy`: defina como `BLUE_GREEN` para habilitar implantações em azul/verde
- `bakeTimeInMinutes`: o tempo de espera após a implantação da revisão do serviço verde antes de mudar o tráfego de produção

```
{
    "loadBalancers": [
        {
            "targetGroupArn":
 "arn:aws:elasticloadbalancing:region:123456789012:targetgroup/blue-target-
group/1234567890123456",
            "containerName": "container-name",
            "containerPort": 80,
            "advancedConfiguration": {
                "alternateTargetGroupArn":
 "arn:aws:elasticloadbalancing:region:123456789012:targetgroup/green-target-
group/1234567890123456",
                "productionListenerArn":
 "arn:aws:elasticloadbalancing:region:123456789012:listener/net/my-network-
lb/1234567890123456/1234567890123456",
```

```
                "testListenerArn":
 "arn:aws:elasticloadbalancing:region:123456789012:listener/net/my-network-
lb/1234567890123456/2345678901234567",
                "roleArn": "arn:aws:iam::123456789012:role/ecs-nlb-role"
            }
        }
    ],
    "deploymentConfiguration": {
        "strategy": "BLUE_GREEN",
        "maximumPercent": 200,
        "minimumHealthyPercent": 100,
        "bakeTimeInMinutes": 5
    }
}
```

Fluxo de tráfego durante a implantação

Durante uma implantação azul/verde com um Network Load Balancer, o tráfego flui pelo sistema da seguinte forma:

1. Estado inicial: todo o tráfego de produção é roteado para o grupo de destino principal (revisão de serviço azul).
2. Implantação de revisão de serviço verde: o Amazon ECS implanta as novas tarefas e as registra no grupo de destino alternativo.
3. Tráfego de teste: se um receptor de teste estiver configurado, o tráfego de teste será roteado para o grupo de destino alternativo para validar a revisão do serviço verde.
4. Mudança de tráfego de produção: o Amazon ECS atualiza o receptor de produção para rotear o tráfego para o grupo de destino alternativo (revisão de serviço verde).
5. Tempo de incorporação: a duração em que as revisões de serviço azul e verde são executadas simultaneamente após a mudança do tráfego de produção.
6. Conclusão: após uma implantação bem-sucedida, a revisão do serviço azul é encerrada.

Se forem detectados problemas durante a implantação, o Amazon ECS pode reverter automaticamente roteando o tráfego de volta para o grupo de destino principal (revisão de serviço azul).

Recursos do Service Connect para implantações azul/verde do Amazon ECS

Ao usar o Service Connect com implantações azul/verde, você precisa configurar componentes específicos para permitir o roteamento de tráfego adequado entre as revisões de serviço azul e verde. Esta seção explica os componentes necessários e suas configurações.

Visão geral da arquitetura

O Service Connect cria recursos de descoberta de serviços e de malha de serviços por meio de um proxy sidecar gerenciado que é injetado automaticamente em suas tarefas do Amazon ECS. Esses proxies processam decisões de roteamento, novas tentativas e coleta de métricas, enquanto o AWS Cloud Map fornece o backend do registro de serviços. Quando você implanta um serviço com o Service Connect habilitado, o serviço se registra no AWS Cloud Map, e os serviços do cliente o descobrem por meio do namespace.

Em uma implementação padrão do Service Connect, os serviços do cliente se conectam aos nomes lógicos dos serviços, e o proxy sidecar gerencia o roteamento para as instâncias reais do serviço. Com implantações azul/verde, esse modelo é estendido para incluir roteamento de tráfego de teste por meio da configuração `testTrafficRules`.

Durante uma implantação azul/verde, os seguintes componentes-chave funcionam juntos:

- Proxy do Service Connect: todo o tráfego entre os serviços passa pelo proxy do Service Connect, que toma decisões de roteamento com base na configuração.
- Registro do AWS Cloud Map: as implantações azul e verde são registradas com o AWS Cloud Map, mas a implantação verde é inicialmente registrada como um endpoint de “teste”.
- Roteamento de tráfego de teste: `testTrafficRules` na configuração do Service Connect determina como identificar e rotear o tráfego de teste para a implantação verde. Isso é feito por meio do roteamento baseado em cabeçalho, em que cabeçalhos HTTP específicos nas solicitações direcionam o tráfego para a revisão de teste. Por padrão, o Service Connect reconhece o cabeçalho `x-amzn-ecs-blue-green-test` dos protocolos baseados em HTTP quando nenhuma regra personalizada é especificada.
- Configuração do cliente: todos os clientes no namespace recebem automaticamente as rotas de produção e teste, mas somente as solicitações que correspondam às regras de teste serão enviadas para a implantação verde.

O que torna essa abordagem eficaz é que ela lida com a complexidade da descoberta de serviços durante as transições. À medida que o tráfego muda da implantação azul para a verde, todos os

mecanismos de conectividade e descoberta são atualizados automaticamente. Não há necessidade de atualizar registros DNS, reconfigurar balanceadores de carga ou implantar alterações de descoberta de serviços separadamente, pois a malha de serviços cuida de tudo isso.

Roteamento e teste de tráfego

O Service Connect fornece recursos avançados de roteamento de tráfego para implantações azul/verde, incluindo roteamento baseado em cabeçalho e configuração de alias de cliente para cenários de teste.

Teste as regras do cabeçalho de tráfego

Durante implantações em azul/verde, você pode configurar regras de cabeçalho de tráfego de teste para rotear solicitações específicas para a revisão de serviço verde (nova) para fins de teste. Isso permite que você valide a nova versão com tráfego controlado antes de concluir a implantação.

O Service Connect usa roteamento baseado em cabeçalho para identificar o tráfego de teste. Por padrão, o Service Connect reconhece o cabeçalho `x-amzn-ecs-blue-green-test` dos protocolos baseados em HTTP quando nenhuma regra personalizada é especificada. Quando esse cabeçalho estiver presente em uma solicitação, o proxy do Service Connect roteará automaticamente a solicitação para a implantação verde para teste.

As regras de cabeçalho de tráfego de teste permitem que você:

- Encaminhe solicitações com cabeçalhos específicos para a revisão do serviço verde
- Teste novas funcionalidades com um subconjunto do tráfego
- Valide o comportamento do serviço antes da substituição total do tráfego
- Implemente estratégias de teste de canário
- Realize testes de integração em um ambiente semelhante ao de produção

O mecanismo de roteamento baseado em cabeçalho funciona perfeitamente com sua arquitetura de aplicação existente. Os serviços do cliente não precisam estar cientes do processo de implantação azul/verde. Eles simplesmente incluem os cabeçalhos apropriados ao enviar solicitações de teste, e o proxy do Service Connect processa a lógica de roteamento automaticamente.

Para obter mais informações sobre como configurar as regras de cabeçalho de tráfego de teste, consulte ServiceConnectTestTrafficHeaderRulesna Referência de APIs do Amazon Elastic Container Service.

Regras de correspondência de cabeçalho

As regras de correspondência de cabeçalho definem os critérios para rotear o tráfego de teste durante implantações azul/verde. Você pode configurar várias condições de correspondência para controlar com precisão quais solicitações são encaminhadas para a revisão do serviço verde.

A correspondência de cabeçalhos é compatível com:

- Correspondência exata do valor do cabeçalho
- Verificação de presença de cabeçalho
- Correspondência baseada em padrões
- Várias combinações de cabeçalhos

Exemplos de casos de uso incluem o encaminhamento de solicitações com strings específicas do agente do usuário, versões de API ou sinalizadores de recursos para o serviço verde para testes.

Para obter mais informações sobre como configurar a correspondência de cabeçalhos, consulte ServiceConnectTestTrafficHeaderMatchRulesna Referência de APIs do Amazon Elastic Container Service.

Aliases de cliente para implantações azul/verde

Os aliases de cliente fornecem endpoints de DNS estáveis para serviços durante implantações azul/verde. Eles permitem o roteamento contínuo do tráfego entre as revisões de serviço azul e verde sem exigir que as aplicações do cliente alterem seus endpoints de conexão.

Durante uma implantação azul/verde, os aliases de cliente:

- Mantêm os nomes de DNS consistentes para conexões de clientes
- Habilitam a mudança automática de tráfego entre as revisões de serviço
- Suportam estratégias graduais de migração de tráfego
- Fornecem recursos de reversão redirecionando o tráfego para a revisão azul

Você pode configurar vários aliases de cliente para portas ou protocolos diferentes, permitindo que arquiteturas de serviços complexas mantenham a conectividade durante as implantações.

Para obter mais informações sobre a configuração de aliases de cliente, consulte ServiceConnectClientAlias na Referência de APIs do Amazon Elastic Container Service.

Práticas recomendadas para roteamento de tráfego

Ao implementar o roteamento de tráfego para implantações azul/verde com o Service Connect, considere as seguintes práticas recomendadas:

- Comece com testes baseados em cabeçalhos: use regras de cabeçalho de tráfego de teste para validar o serviço verde com tráfego controlado antes de mudar todo o tráfego.
- Configure verificações de integridade: garanta que os serviços azul e verde tenham verificações de integridade apropriadas configuradas para evitar o roteamento de tráfego para instâncias não íntegras.
- Monitore as métricas de serviço: acompanhe os principais indicadores de performance de ambas as revisões de serviço durante a implantação para identificar problemas antecipadamente.
- Planeje a estratégia de reversão: configure aliases de cliente e regras de roteamento para permitir a rápida reversão para o serviço azul se forem detectados problemas.
- Teste a lógica de correspondência de cabeçalhos: valide suas regras de correspondência de cabeçalhos em um ambiente não de produção antes de aplicá-las às implantações de produção.

Fluxo de trabalho de implantação azul/verde do Service Connect

Entender como o Service Connect gerencia o processo de implantação azul/verde ajuda você a implementar e solucionar problemas de suas implantações de forma eficaz. O fluxo de trabalho a seguir mostra como os diferentes componentes interagem durante cada fase da implantação.

Fases da implantação

A implantação azul/verde do Service Connect avança em várias fases distintas:

1. Estado inicial: o serviço azul processa 100% do tráfego de produção. Todos os serviços de cliente no namespace se conectam ao serviço azul por meio do nome lógico do serviço configurado no Service Connect.
2. Registro de serviço verde: quando a implantação verde é iniciada, ela é registrada com o AWS Cloud Map como um endpoint de “teste”. O proxy do Service Connect nos serviços do cliente recebe automaticamente as configurações de rota de produção e teste.
3. Roteamento de tráfego de teste: as solicitações contendo os cabeçalhos de tráfego de teste (como `x-amzn-ecs-blue-green-test`) são automaticamente roteadas para o serviço verde pelo proxy do Service Connect. O tráfego de produção continua fluindo para o serviço azul.

4. Preparação da mudança de tráfego: após o teste bem-sucedido, o processo de implantação se prepara para a mudança do tráfego de produção. Tanto o serviço azul quanto o verde permanecem registrados e íntegros.
5. Mudança de tráfego de produção: a configuração do Service Connect é atualizada para direcionar o tráfego de produção para o serviço verde. Isso acontece automaticamente sem exigir atualizações do serviço do cliente ou alterações no DNS.
6. Período de tempo de incorporação: a duração em que as revisões de serviço azul e verde são executadas simultaneamente após a mudança do tráfego de produção.
7. Cancelamento do registro do serviço azul: após a validação e a mudança de tráfego bem-sucedidas, o registro do serviço azul é cancelado do AWS Cloud Map e encerrado, concluindo a implantação.

Comportamento do proxy do Service Connect

O proxy do Service Connect desempenha um papel crucial no gerenciamento do tráfego durante implantações azul/verde. Compreender seu comportamento ajuda você a projetar estratégias eficazes de teste e implantação.

Principais comportamentos do proxy durante implantações azul/verde:

- Descoberta automática de rotas: o proxy descobre automaticamente as rotas de produção e teste do AWS Cloud Map sem exigir a reinicialização da aplicação ou alterações na configuração.
- Roteamento baseado em cabeçalho: o proxy examina os cabeçalhos das solicitações recebidas e encaminha o tráfego para a revisão de serviço apropriada com base nas regras de tráfego de teste configuradas.
- Integração da verificação de integridade: o proxy direciona o tráfego somente para instâncias de serviço íntegras, excluindo automaticamente tarefas não íntegras do grupo de roteamento.
- Repetição e interrupção de circuito: o proxy fornece lógica de repetição integrada e recursos de interrupção de circuito, melhorando a resiliência durante as implantações.
- Coleta de métricas: o proxy coleta métricas detalhadas para o serviço azul e o verde, permitindo um monitoramento abrangente durante as implantações.

Atualizações da descoberta de serviços

Uma das principais vantagens de usar o Service Connect para implantações azul/verde é o processamento automático das atualizações de descoberta de serviços. As implantações tradicionais

azul/verde geralmente exigem atualizações complexas de DNS ou a reconfiguração do balanceador de carga, mas o Service Connect gerencia essas alterações de forma transparente.

Durante uma implantação, o Service Connect processa:

- Atualizações de namespace: o namespace do Service Connect inclui automaticamente endpoints do serviço azul e do verde, com as regras de roteamento apropriadas.
- Configuração do cliente: todos os serviços do cliente no namespace recebem automaticamente informações de roteamento atualizadas sem exigir reinicializações ou reimplantação.
- Transição gradual: as atualizações da descoberta de serviços acontecem de forma gradual e segura, garantindo que não haja interrupções nas solicitações em andamento.
- Suporte à reversão: se uma reversão for necessária, o Service Connect poderá reverter rapidamente as configurações da descoberta de serviços para direcionar o tráfego de volta para o serviço azul.

Criação de implantações azul/verde do Amazon ECS

Ao usar implantações azul/verde do Amazon ECS, você pode fazer e testar alterações no serviço antes de implementá-las em um ambiente de produção.

Pré-requisitos

Execute as operações a seguir antes de iniciar uma implantação azul/verde.

1. Configurar as permissões apropriadas.
   - Para obter informações sobre as permissões do Elastic Load Balancing, consulte Perfil do IAM da infraestrutura do Amazon ECS para balanceadores de carga.
   - Para obter informações sobre as permissões do Lambda, consulte Permissões necessárias para funções do Lambda em implantações azul/verde do Amazon ECS
2. As implantações azul/verde do Amazon ECS exigem que seu serviço use um dos recursos a seguir. Configure os recursos apropriados.
   - Application Load Balancer: para obter mais informações, consulte Recursos do Application Load Balancer para implantações azul/verde.
   - Network Load Balancer: para obter mais informações, consulte Recursos do Network Load Balancer para implantações azul/verde do Amazon ECS.
   - Service Connect: para obter mais informações, consulte Recursos do Service Connect para implantações azul/verde do Amazon ECS.

3. Crie uma regra para rotear o tráfego para a revisão do serviço verde. Para obter mais informações, consulte Listener rules no Guia do usuário do Network Load Balancer.
4. Crie um grupo de destino para a revisão do serviço verde. Quando você usa o modo de rede `awsvpc` para suas tarefas, o tipo de destino deve ser `ip`. Para obter informações sobre grupos de destino, consulte Target groups no Guia do usuário do Network Load Balancer.
5. Decida se deseja executar as funções do Lambda para os eventos do ciclo de vida.
   - Antes de aumentar a escala verticalmente
   - Depois de aumentar a escala verticalmente
   - Mudança do tráfego de teste
   - Depois da mudança do tráfego de teste
   - Mudança do tráfego de produção
   - Depois da mudança do tráfego de produção

   Crie funções do Lambda para cada evento do ciclo de vida. Para obter mais informações, consulte Criar uma função do Lambda com o console no Guia do desenvolvedor do AWS Lambda.

Procedimento

Você pode usar o console ou a AWS CLI para criar um serviço azul/verde do Amazon ECS.

Console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Determine o recurso no qual você inicia o serviço.

Para iniciar um serviço em	Etapas	
Clusters	a. Na página Clusters, selecione o cluster no qual o serviço será criado. A página de detalhes do cluster é exibida.	

Para iniciar um serviço em	Etapas
	b. Na guia Services (Serviços), escolha Create (Criar).
Definição de tarefa	a. Na página Definições de tarefa, selecione a definição de tarefa. b. No menu Implantar, escolha Criar serviço.

A página Criar serviço é exibida.

3. Em Detalhes do serviço, faça o seguinte:

   a. Em Família de definições de tarefas, escolha a definição de tarefa a ser usada. Em seguida, em Revisão da definição de tarefa, insira a revisão a ser usada.

   b. Em Service name (Nome do serviço), insira um nome para o serviço.

4. Para executar o serviço em um cluster existente, em Cluster existente, escolha o cluster. Para executar o serviço em um novo cluster, escolha Criar cluster

5. Escolha como suas tarefas serão distribuídas em toda a infraestrutura do cluster. Expanda Configuração de computação e escolha sua opção.

Opção de computação	Etapas
Estratégia de provedor de capacidade	a. Em Opções de computação, escolha Estratégia do provedor de capacidade. b. Escolha uma estratégia: • Para usar a estratégia de provedor de capacidade padrão do cluster, escolha Use

Opção de computação	Etapas	
	cluster default (Usar padrão de cluster). • Se o seu cluster não tiver uma estratégia de provedor de capacidade padrão ou, para usar uma estratégia personalizada, escolha Usar personalizada, Adicionar estratégia de provedor de capacidade e, em seguida, defina sua estratégia de provedor de capacidade personalizada especificando uma Base, um Provedor de capacidade e um Peso. **Note** Para usar um provedor de capacidade em uma estratégia, o provedor de capacidade deve estar associado ao cluster.	

Opção de computação	Etapas	
Tipo de inicialização	a. Na seção Compute options (Opções de computação), selecione Launch type (Tipo de inicialização). b. Em Launch type (Tipo de inicialização), escolha um tipo de inicialização. c. (Opcional) Quando o tipo de inicialização do Fargate for especific ado, em Platform version (Versão da plataforma), especifique a versão de plataforma a ser usada. Se não for especificada uma versão da plataform a, a versão da plataforma `LATEST` será usada.	

6. Em Configuração de implantação, faça o seguinte:

   a. Em Tipo de serviço, escolha Réplica.

   b. Em Desired tasks (Tarefas desejadas), insira o número de tarefas que serão iniciadas e mantidas no serviço.

   c. Para que o Amazon ECS monitore a distribuição de tarefas entre as zonas de disponibilidade e as redistribua quando houver um desequilíbrio, em Rebalanceamento do serviço em zonas de disponibilidade, selecione Rebalanceamento do serviço em zonas de disponibilidade.

   d. Em Período de carência da verificação de integridade, insira quanto tempo (em segundos) após o início de uma tarefa o agendador do serviço ignorará as verificações de integridade que forem não íntegras para o Elastic Load Balancing, o VPC Lattice e o contêiner. Se você não especificar um valor de período de tolerância de verificação de integridade, o valor padrão 0 será usado.

7. a. Em Tempo de incorporação, insira o número de minutos em que as revisões do serviço azul e do verde serão executadas simultaneamente antes que a revisão azul seja encerrada. Isso possibilita um tempo para a verificação e testes.

   b. (Opcional) Execute as funções do Lambda para serem executadas em estágios específicos da implantação. Em Ganchos do ciclo de vida de implantação, selecione os estágios para executar os ganchos do ciclo de vida.

      Para adicionar um gancho do ciclo de vida:

      a. Escolha Adicionar.

      b. Em Função do Lambda, insira o nome da função ou o ARN.

      c. Em Perfil selecione o perfil do IAM que tem permissão para invocar a função do Lambda.

      d. Em Estágios do ciclo de vida, selecione os estágios em que a função do Lambda deve ser executada.

8. Para configurar como o Amazon ECS detectará e lidará com falhas de implantação, expanda Deployment failure detection(Detecção de falhas de implantação) e escolha suas opções.

   a. Para interromper uma implantação quando as tarefas não podem ser iniciadas, selecione Use the Amazon ECS deployment circuit breaker) (Usar o disjuntor de implantação do Amazon ECS.

      Para que o software reverta automaticamente a implantação para o último estado de implantação concluído quando o disjuntor de implantação definir a implantação para um estado de falha, selecione Reverter em caso de falha.

   b. Para interromper uma implantação com base nas métricas da aplicação, selecione Usar alarmes do CloudWatch. Em seguida, em Nomes de alarmes do CloudWatch, escolha os alarmes. Para criar um alarme, acesse o console do CloudWatch.

      Para que o software reverta automaticamente a implantação para o último estado de implantação concluído quando um alarme do CloudWatch definir a implantação para um estado de falha, selecione Reverter em caso de falha.

9. (Opcional) Para interconectar o serviço usando o Service Connect, expanda Service Connect e especifique o seguinte:

   a. Selecione Ativar o Service Connect.

b. Em Service Connect configuration (Configuração do Service Connect), especifique o modo cliente.

   - Se seu serviço executa uma aplicação cliente de rede que só precisa se conectar a outros serviços no namespace, escolha Somente no lado do cliente.
   - Se seu serviço executa uma aplicação de rede ou de serviço Web e precisa fornecer endpoints para esse serviço e se conectar a outros serviços no namespace, escolha Client and server (Cliente e servidor).

c. Para usar um namespace que não seja o namespace padrão do cluster, em Namespace, escolha o namespace do serviço.

d. (Opcional) Configure regras de cabeçalho de tráfego de teste para implantações em azul/verde. Em Testar roteamento de tráfego, especifique o seguinte:

   i. Selecione Habilitar regras de cabeçalho de tráfego de teste para rotear solicitações específicas para a revisão do serviço verde durante o teste.

   ii. Em Regras de correspondência de cabeçalhos, configure os critérios para rotear o tráfego de teste:

      - Nome do cabeçalho: insira o nome do cabeçalho HTTP correspondente (por exemplo, `X-Test-Version` ou `User-Agent`).
      - Tipo de correspondência: escolha os critérios de correspondência:
        - Correspondência exata: roteie solicitações em que o valor do cabeçalho corresponda exatamente ao valor especificado
        - Cabeçalho presente: roteie solicitações que contenham o cabeçalho especificado, independentemente do valor
        - Correspondência de padrões: roteie solicitações em que o valor do cabeçalho corresponda a um padrão especificado
      - Valor do cabeçalho (se estiver usando correspondência exata ou correspondência de padrão): insira o valor ou padrão com o qual corresponder.

      Você pode adicionar várias regras de correspondência de cabeçalhos para criar uma lógica de roteamento complexa. As solicitações que correspondam a qualquer uma das regras configuradas serão encaminhadas para a revisão do serviço verde para testes.

iii. Escolha Adicionar regra de cabeçalho para configurar condições adicionais de correspondência de cabeçalho.

> **Note**
>
> As regras de cabeçalho de tráfego de teste permitem que você valide a nova funcionalidade com tráfego controlado antes de concluir a implantação completa. Isso permite que você teste a revisão do serviço verde com solicitações específicas (como as de ferramentas de teste internas ou usuários beta) enquanto mantém o fluxo de tráfego normal para a revisão do serviço azul.

e. (Opcional) Especifique uma configuração de log. Selecione Usar conjunto de logs. A opção padrão envia os logs do contêiner ao CloudWatch Logs. As outras opções de driver de log são configuradas usando o AWS FireLens. Para obter mais informações, consulte Envio de logs do Amazon ECS para um serviço da AWS ou para uma AWS Partner.

Veja a seguir a descrição de cada destino de log de contêiner mais detalhadamente.

- Amazon CloudWatch: configura a tarefa para enviar logs de contêiner ao CloudWatch Logs. São fornecidas as opções de driver de log padrão, que criam um grupo de logs do CloudWatch em seu nome. Para especificar um nome de grupo de logs diferente, altere os valores da opção de driver.
- Amazon Data Firehose: configura a tarefa para enviar logs de contêiner ao Firehose. São fornecidas as opções de driver de log padrão, que enviam logs para um fluxo de entrega do Firehose. Para especificar um nome de fluxo de entrega diferente, altere os valores da opção de driver.
- Amazon Kinesis Data Streams: configura a tarefa para enviar logs de contêiner ao Kinesis Data Streams. São fornecidas as opções de driver de log padrão, que enviam logs a um fluxo do Kinesis Data Streams. Para especificar um nome de transmissão diferente, altere os valores da opção de driver.
- Amazon OpenSearch Service: configura a tarefa para enviar logs de contêiner a um domínio do OpenSearch Service. As opções de driver de log devem ser fornecidas.
- Amazon S3: configura a tarefa para enviar logs de contêiner a um bucket do Amazon S3. As opções de driver de log padrão são fornecidas, mas você deve especificar um nome de bucket válido do Amazon S3.

10. (Opcional) Configure o balanceamento de carga para implantação azul/verde.

Tipo do Elastic Load Balancing	Etapas	
Application Load Balancer	a. Em Tipo de balancead or de carga, escolha Application Load Balancer. b. Escolha Create a new load balancer (Criar um novo balanceador de carga) para criar um novo Application Load Balancer ou Use an existing load balancer (Usar um balanceador de carga existente) para selecionar um Applicati on Load Balancer existente. c. Em Contêiner, escolha o contêiner que hospeda o serviço. d. Em Load balancer (Balanceador de carga), insira um nome exclusivo . e. Em Listener (Receptor), insira uma porta e um protocolo para que o Application Load Balancer receba solicitaç ões de conexão. Por padrão, o balanceador de carga será configura	

Tipo do Elastic Load Balancing	Etapas	
	do para usar a porta 80 e HTTP. • Em Regra de produção, insira a Ordem de avaliação e o Padrão de caminho para a regra. Essa regra é para seu tráfego de revisão do serviço de produção (azul). • Em Regra de teste, insira a Ordem de avaliação e o Padrão de caminho para a regra. Essa regra é para o tráfego de revisão do serviço de teste (verde). f. Em Grupo de destino, configure o seguinte: • Em Target group name (Nome do grupo de destino), insira um nome e um protocolo para o grupo de destino ao qual o Application Load Balancer encaminhará as solicitações.	

Tipo do Elastic Load Balancing	Etapas	
	• Em Protocolo, escolha o protocolo para o grupo de destino ao qual o Application Load Balancer encaminhará as solicitações. Por padrão, o grupo de destino encaminha solicitações ao primeiro contêiner especificado na definição de tarefa. • Em Atraso no cancelamento de registro, insira o número de segundos para que o balanceador de carga altere o estado de destino para `UNUSED`. O padrão é trezentos segundos. • Em Health check path (Caminho da verificação de integridade), insira um caminho existente no contêiner para o qual o Application Load Balancer enviará periodicamente solicitações para verificar a integridade da	

Tipo do Elastic Load Balancing	Etapas	
	conexão entre o Application Load Balancer e o contêiner. O padrão é o diretório raiz (/). • Em Nome do grupo alternativo, insira o nome do grupo de destino para a revisão do serviço de teste (verde).	

Tipo do Elastic Load Balancing	Etapas	
Network Load Balancer	a. Em Load balancer type (Tipo de balanceador de carga), selecione Network Load Balancer. b. Em Load Balancer (Balanceador de carga), escolha um Network Load Balancer existente. c. Em Choose container to load balance (Escolher contêiner para balancear a carga), escolha o contêiner que hospeda o serviço. d. Em Receptor de produção, escolha a Porta do receptor de produção e o Protocolo do receptor de produção. Este é o receptor do tráfego de revisão do serviço de produção (azul). e. Em Receptor de teste, escolha a Porta do receptor de teste e o Protocolo do receptor de teste. Este é o receptor do tráfego de revisão do serviço de teste (verde).	

Tipo do Elastic Load Balancing	Etapas	
	f. Em Grupo de destino, configure o seguinte: • Em Target group name (Nome do grupo de destino), insira um nome e um protocolo para o grupo de destino ao qual o Network Load Balancer encaminhará solicitações. • Em Protocolo, escolha o protocolo para o grupo de destino ao qual o Network Load Balancer encaminha rá as solicitações. Por padrão, o grupo de destino encaminha solicitações ao primeiro contêiner especificado na definição de tarefa. • Em Atraso no cancelamento de registro, insira o número de segundos para que o balancead or de carga altere o estado de destino para `UNUSED`. O padrão é trezentos segundos.	

Tipo do Elastic Load Balancing	Etapas	
	• Em Health check path (Caminho da verificação de integridade), insira um caminho existente no contêiner para o qual o Application Load Balancer enviará periodicamente solicitações para verificar a integridade da conexão entre o Application Load Balancer e o contêiner. O padrão é o diretório raiz (/). • Em Nome do grupo alternativo, insira o nome do grupo de destino para a revisão do serviço de teste (verde).	

11. (Opcional) Para ajudar a identificar seu serviço e tarefas, expanda a seção Tags (Etiquetas) e configure suas etiquetas.

    Para que o Amazon ECS atribua tags automaticamente a todas as tarefas recém-iniciadas com o nome do cluster e as tags de definição de tarefa, selecione Ativar tags gerenciadas pelo Amazon ECS e, para Propagar tags de, escolha Definições de tarefas.

    Para que o Amazon ECS atribua tags automaticamente a todas as tarefas recém-iniciadas com o nome do cluster e as tags de serviços, selecione Ativar tags gerenciadas pelo Amazon ECS e, para Propagar tags de, escolha Serviço.

    Adicione ou remova uma tag.

- [Adicionar uma etiqueta] Escolha Add tag (Adicionar etiqueta) e faça o seguinte:
  - Em Chave, insira o nome da chave.
  - Em Valor insira o valor da chave.
- [Remover uma tag] Ao lado da tag, escolha Remove tag (Remover tag).

12. Escolha Criar.

AWS CLI

1. Crie um arquivo chamado `service-definition.json` com o conteúdo a seguir.

   Substitua os valores das *`user-input`* pelos seus.

```
{
  "serviceName": "myBlueGreenService",
  "cluster": "arn:aws:ecs:us-west-2:123456789012:cluster/sample-fargate-
cluster",
  "taskDefinition": "sample-fargate:1",
  "desiredCount": 5,
  "launchType": "FARGATE",
  "networkConfiguration": {
    "awsvpcConfiguration": {
      "subnets": [
        "subnet-09ce6e74c116a2299",
        "subnet-00bb3bd7a73526788",
        "subnet-0048a611aaec65477"
      ],
      "securityGroups": [
        "sg-09d45005497daa123"
      ],
      "assignPublicIp": "ENABLED"
    }
  },
  "deploymentController": {
    "type": "ECS"
  },
  "deploymentConfiguration": {
    "strategy": "BLUE_GREEN",
    "maximumPercent": 200,
    "minimumHealthyPercent": 100,
    "bakeTimeInMinutes": 2,
```

```
    "alarms": {
      "alarmNames": [
        "myAlarm"
      ],
      "rollback": true,
      "enable": true
    },
    "lifecycleHooks": [
      {
        "hookTargetArn": "arn:aws:lambda:us-
west-2:7123456789012:function:checkExample",
        "roleArn": "arn:aws:iam::123456789012:role/ECSLifecycleHookInvoke",
        "lifecycleStages": [
          "PRE_SCALE_UP"
        ]
      }
    ]
  },
  "loadBalancers": [
    {
      "targetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/blue-target-group/54402ff563af1197",
      "containerName": "fargate-app",
      "containerPort": 80,
      "advancedConfiguration": {
        "alternateTargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/green-target-group/cad10a56f5843199",
        "productionListenerRule": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener-rule/app/my-blue-green-
demo/32e0e4f946c3c05b/9cfa8c482e204f7d/831dbaf72edb911",
        "roleArn": "arn:aws:iam::123456789012:role/LoadBalancerManagementforECS"
      }
    }
  ]
}
```

2. Executar `create-service`.

   Substitua os valores das *`user-input`* pelos seus.

```
aws ecs create-service --cli-input-json file://service-definition.json
```

Como alternativa, você pode usar o exemplo a seguir, que cria um serviço de implantação azul/verde com uma configuração de balanceador de carga:

```
aws ecs create-service \
   --cluster "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster" \
   --service-name "blue-green-example-service" \
   --task-definition "nginxServer:1" \
   --launch-type "FARGATE" \
   --network-configuration
 "awsvpcConfiguration={subnets=[subnet-12345,subnet-67890,subnet-abcdef,subnet-
fedcba],securityGroups=[sg-12345],assignPublicIp=ENABLED}" \
   --desired-count 3 \
   --deployment-controller "type=ECS" \
   --deployment-configuration
 "strategy=BLUE_GREEN,maximumPercent=200,minimumHealthyPercent=100,bakeTimeInMinutes=0"
 \
   --load-balancers "targetGroupArn=arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/MyBGtg1/
abcdef1234567890,containerName=nginx,containerPort=80,advancedConfiguration={alternateTa
west-2:123456789012:targetgroup/
MyBGtg2/0987654321fedcba,productionListenerRule=arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener-rule/app/
MyLB/1234567890abcdef/1234567890abcdef,roleArn=arn:aws:iam::123456789012:role/
ELBManagementRole}"
```

Próximas etapas

- Atualize o serviço para iniciar a implantação. Para obter mais informações, consulte Atualizar um serviço do Amazon ECS.
- Monitore o processo de implantação para garantir que ele siga o padrão azul/verde:
  - A revisão do serviço verde é criada e tem a escala aumentada verticalmente
  - O tráfego de teste é roteado para a revisão verde (se configurado)
  - O tráfego de produção é mudado para a revisão verde
  - Após o tempo de incorporação, a revisão azul é encerrada

Implantação de serviços do Amazon ECS usando um controlador de terceiros

O tipo de implantação externa permite usar qualquer controlador de implantação de terceiros para ter controle total do processo de implantação em um serviço do Amazon ECS. Os detalhes do seu serviço são gerenciados por ações da API de gerenciamento de serviços (`CreateService`, `UpdateService` e `DeleteService`) ou ações da API de gerenciamento de conjuntos de tarefas (`CreateTaskSet`, `UpdateTaskSet`, `UpdateServicePrimaryTaskSet` e `DeleteTaskSet`). Cada ação de API gerencia um subconjunto dos parâmetros de definição de serviço.

A ação de API `UpdateService` atualiza os parâmetros de contagem desejada e período de carência da verificação de integridade para um serviço. Se o tipo de inicialização, a versão da plataforma, os detalhes do load balancer, a configuração de rede ou definição de tarefa precisarem ser atualizados, você deverá criar um novo conjunto de tarefas.

A ação de API `UpdateTaskSet` atualiza apenas o parâmetro de escala para um conjunto de tarefas.

A ação de API `UpdateServicePrimaryTaskSet` modifica qual conjunto de tarefas em um serviço é o conjunto de tarefas principal. Quando você chama a ação de API `DescribeServices`, ela retorna todos os campos especificados para um conjunto de tarefas principal. Se o conjunto de tarefas principal de um serviço for atualizado, qualquer valor de parâmetro de conjunto de tarefas existente no novo conjunto de tarefas principal que for diferente do conjunto de tarefas principal antigo em um serviço será atualizado para o novo valor quando um novo conjunto de tarefas principal for definido. Se nenhum conjunto de tarefas principal for definido para um serviço, durante a descrição do serviço, os campos do conjunto de tarefas serão nulos.

Considerações sobre a implantação externa

Considere o seguinte ao usar o tipo de implantação externa:

- Os tipos de load balancer compatíveis são um Application Load Balancer ou um Network Load Balancer.
- O tipo de inicialização do Fargate ou os tipos de controladores de implantação `EXTERNAL` não são compatíveis com a estratégia de programação do `DAEMON`.
- A integração do Application AutoScaling com o Amazon ECS é compatível apenas com o serviço do Amazon ECS como destino. A dimensão escalável compatível com o Amazon ECS é `ecs:service:DesiredCount`: a contagem de tarefas de um serviço do Amazon ECS. Não há integração direta entre o Application AutoScaling e os conjuntos de tarefas do Amazon ECS. Os conjuntos de tarefas do Amazon ECS calculam a `ComputedDesiredCount` com base no serviço `DesiredCount` do Amazon ECS.

Fluxo de trabalho de implantações externas

Veja a seguir o fluxo de trabalho básico para gerenciar uma implantação externa no Amazon ECS.

Para gerenciar um serviço do Amazon ECS usando um controlador de implantação externa

1. Crie um serviço do Amazon ECS. O único parâmetro obrigatório é o nome do serviço. É possível especificar os parâmetros a seguir ao criar um serviço usando um controlador de implantação externo. Todos os outros parâmetros de serviço são especificados durante a criação de um conjunto de tarefas no serviço.

   `serviceName`

   Tipo: string

   Obrigatório: Sim

   O nome do serviço. São permitidos até 255 letras (caixa alta e baixa), números, hífens e sublinhados. Os nomes dos serviços devem ser exclusivos em um cluster, mas é possível ter serviços com nomes semelhantes em vários clusters em uma ou várias regiões.

   `desiredCount`

   O número de instanciações da definição de tarefa do conjunto de tarefas especificado para posicionar e manter em execução no serviço.

   `deploymentConfiguration`

   Parâmetros opcionais de implantação que controlam quantas tarefas são executadas durante uma implantação e a ordem de interrupção e início de tarefas.

   `tags`

   Tipo: matriz de objetos

   Obrigatório: Não

   Os metadados que você aplica ao serviço para ajudá-lo a categorizá-los e organizá-los. Cada tag consiste de uma chave e um valor opcional, que podem ser definidos. Quando um serviço é excluído, as tags também são excluídas. Um máximo de 50 tags podem ser aplicadas ao serviço. Para obter mais informações, consulte Marcação de recursos do Amazon ECS.

`key`

Tipo: String

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Obrigatório: Não

Uma parte de um par de chave/valor que compõe uma tag. Uma chave é um rótulo geral que age como uma categoria para valores de tag mais específicos.

`value`

Tipo: String

Restrições de tamanho: o tamanho mínimo é 0. O tamanho máximo é 256.

Obrigatório: Não

A parte opcional de um par de chave/valor que compõe uma tag. Um valor atua como um descritor dentro de uma categoria de tag (chave).

`enableECSManagedTags`

Especifica se devem ser usadas as tags gerenciadas do Amazon ECS para as tarefas no serviço. Para obter mais informações, consulte Usar etiquetas para faturamento.

`propagateTags`

Tipo: Sequência

Valores válidos: `TASK_DEFINITION` | `SERVICE`

Obrigatório: Não

Especifica se deve copiar as tags da definição de tarefa ou do serviço para as tarefas no serviço. Se nenhum valor for especificado, as tags não serão copiadas. As tags só podem ser copiadas para tarefas no serviço durante a criação do serviço. Para adicionar etiquetas a uma tarefa após a criação do serviço ou da tarefa, use a ação de API `TagResource`.

`schedulingStrategy`

A estratégia de programação para usar. Os serviços que usam um controlador de implantação externo oferecem suporte apenas à estratégia de programação de `REPLICA`.

`placementConstraints`

Um array de objetos de restrição de posicionamento para usar em tarefas no serviço. É possível especificar um máximo de 10 restrições por tarefa. Esse limite inclui restrições na definição de tarefa e naquelas especificadas no runtime. Se você estiver usando o tipo de inicialização do Fargate, não haverá suporte para restrições de posicionamento de tarefas.

`placementStrategy`

A estratégia de posicionamento de objetos para usar em tarefas no serviço. É possível especificar um máximo de quatro regras de estratégia por serviço.

Veja a seguir uma definição de serviço de exemplo para a criação de um serviço usando um controlador de implantação externo.

```
{
    "cluster": "",
    "serviceName": "",
    "desiredCount": 0,
    "role": "",
    "deploymentConfiguration": {
        "maximumPercent": 0,
        "minimumHealthyPercent": 0
    },
    "placementConstraints": [
        {
            "type": "distinctInstance",
            "expression": ""
        }
    ],
    "placementStrategy": [
        {
            "type": "binpack",
            "field": ""
        }
    ],
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "EXTERNAL"
    },
    "tags": [
        {
```

```
            "key": "",
            "value": ""
        }
    ],
    "enableECSManagedTags": true,
    "propagateTags": "TASK_DEFINITION"
}
```

2. Crie um conjunto de tarefas inicial. O conjunto de tarefas contém os seguintes detalhes sobre o serviço:

   `taskDefinition`

   A definição de tarefa para as tarefas do conjunto de tarefas usarem.

   `launchType`

   Tipo: String

   Valores válidos: `EC2` | `FARGATE` | `EXTERNAL`

   Obrigatório: Não

   O tipo de inicialização na qual executar seu serviço. Se não for especificado um tipo de inicialização, o padrão `capacityProviderStrategy` será usado. Para obter mais informações, consulte Tipos de inicialização do Amazon ECS.

   Se um `launchType` for especificado, o parâmetro `capacityProviderStrategy` deverá ser omitido.

   `platformVersion`

   Tipo: String

   Obrigatório: Não

   A versão da plataforma na qual suas tarefas no serviço estão em execução. Uma versão da plataforma é especificada apenas para tarefas que usam o tipo de inicialização do Fargate. Se não for especificada, a versão mais recente (`LATEST`) será usada como padrão.

   As versões da plataforma do AWS Fargate são usadas para fazer referência a um ambiente de runtime específico para a infraestrutura de tarefas do Fargate. Ao especificar a versão da plataforma `LATEST` quando estiver executando uma tarefa ou criando um serviço, você

obtém a versão de plataforma mais atual disponível para suas tarefas. Ao escalar seu serviço, essas tarefas recebem a versão de plataforma especificada na implantação atual do serviço. Para obter mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.

> **Note**
>
> Versões de plataforma não são especificadas para tarefas que usam o tipo de inicialização do EC2.

`loadBalancers`

Um objeto do load balancer que representa o load balancer para uso no serviço. Quando for usado um controlador de implantação externa, somente Application Load Balancers e Network Load Balancers são compatíveis. Se você estiver usando um Application Load Balancer, somente um grupo de destino do Application Load Balancer será permitido por conjunto de tarefas.

O trecho a seguir mostra o exemplo de um objeto `loadBalancer` a ser usado.

```
"loadBalancers": [
        {
            "targetGroupArn": "",
            "containerName": "",
            "containerPort": 0
        }
]
```

> **Note**
>
> Ao especificar um objeto `loadBalancer`, é necessário especificar um `targetGroupArn` e omitir os parâmetros `loadBalancerName`.

`networkConfiguration`

A configuração de rede para o serviço. Esse parâmetro é necessário para definições de tarefa que usam o modo de rede `awsvpc` para receber sua própria interface de rede elástica,

e não tem suporte para outros modos de rede. Para obter mais informações sobre redes para o tipo de execução do Fargate, consulte Opções de redes de tarefas do Amazon ECS para o tipo de inicialização do Fargate.

`serviceRegistries`

Os detalhes dos registros de descoberta de serviços a serem atribuídos ao serviço. Para obter mais informações, consulte Uso da descoberta de serviços para conectar serviços do Amazon ECS com nomes DNS.

`scale`

Uma porcentagem de ponto flutuante do número desejado de tarefas para posicionar e manter em execução no conjunto de tarefas. O valor é especificado como uma porcentagem total de `desiredCount` de um serviço. Os valores aceitos são números entre 0 e 100.

Veja a seguir um exemplo de JSON para criar um conjunto de tarefas para um controlador de implantação externo.

```
{
    "service": "",
    "cluster": "",
    "externalId": "",
    "taskDefinition": "",
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": [
                ""
            ],
            "securityGroups": [
                ""
            ],
            "assignPublicIp": "DISABLED"
        }
    },
    "loadBalancers": [
        {
            "targetGroupArn": "",
            "containerName": "",
            "containerPort": 0
        }
    ],
```

```
    "serviceRegistries": [
        {
            "registryArn": "",
            "port": 0,
            "containerName": "",
            "containerPort": 0
        }
    ],
    "launchType": "EC2",
    "capacityProviderStrategy": [
        {
            "capacityProvider": "",
            "weight": 0,
            "base": 0
        }
    ],
    "platformVersion": "",
    "scale": {
        "value": null,
        "unit": "PERCENT"
    },
    "clientToken": ""
}
```

3. Quando forem necessárias alterações no serviço, use a ação de API `UpdateService`, `CreateTaskSet` ou `UpdateTaskSet`, dependendo de quais parâmetros você estiver atualizando. Se você tiver criado um conjunto de tarefas, use o parâmetro `scale` para cada tarefa em um serviço para determinar quantas tarefas devem ser mantidas em execução no serviço. Por exemplo, se você tiver um serviço que contenha `tasksetA` e criar um `tasksetB`, poderá testar a validade de `tasksetB` antes de querer fazer a transição do tráfego de produção para ele. Seria possível definir `scale` para os dois conjuntos de tarefas como `100`, e quando estivesse pronto para fazer a transição de todo o tráfego de produção para `tasksetB`, poderia atualizar `scale` para `tasksetA` como `0` para reduzi-lo.

Implantações azul/verde do CodeDeploy para o Amazon ECS

Recomendamos que você use a implantação azul/verde do Amazon ECS. Para obter mais informações, consulte Criação de implantações azul/verde do Amazon ECS .

O tipo de implantação azul/verde usa o modelo de implantação azul/verde controlado pelo CodeDeploy. Use este tipo de implantação para verificar uma nova implantação de um serviço

antes de enviar tráfego de produção para ele. Para obter mais informações, consulte O que é o CodeDeploy? no Guia do usuário do AWS CodeDeploy. Valide o estado de um serviço do Amazon ECS antes da implantação

Existem três maneiras pelas quais o tráfego pode mudar durante uma implantação azul/verde:

- Canário: o tráfego é deslocado em dois incrementos. É possível escolher entre opções de canary predefinidas que especificam a porcentagem de tráfego deslocada para a definição da tarefa atualizada no primeiro incremento e o intervalo, em minutos, antes que o tráfego restante seja deslocado no segundo incremento.
- Linear: o tráfego é deslocado em incrementos iguais com um número igual de minutos entre cada incremento. É possível escolher entre opções lineares predefinidas que especificam a porcentagem de tráfego deslocado em cada incremento e o número de minutos entre cada incremento.
- Tudo de uma vez: todo o tráfego é deslocado do conjunto de tarefas original para o conjunto de tarefas atualizado de uma só vez.

Veja a seguir os componentes do CodeDeploy que o Amazon ECS usa quando um serviço utiliza o tipo de implantação azul/verde:

Aplicação CodeDeploy

Uma coleção de recursos do CodeDeploy. Isso consiste em um ou mais grupos de implantação.

Grupo de implantação do CodeDeploy

As configurações de implantação. Isso consiste no seguinte:

- Cluster e serviço do Amazon ECS
- Informações sobre o grupo de destino e o listener do load balancer
- Estratégia de reversão de implantação
- Configurações de reencaminhamento de tráfego
- Configurações de finalização de revisão original
- Configuração de implantação
- Configuração de alarmes do CloudWatch que pode ser definida para interromper implantações
- Configurações do SNS ou do CloudWatch Events para notificações

Para obter mais informações, consulte Trabalhar com grupos de implantação no Guia do usuário do AWS CodeDeploy.

Configuração de implantação do CodeDeploy

Especifica como o CodeDeploy roteia o tráfego de produção para o conjunto de tarefas de substituição durante uma implantação. As configurações de implantação linear e de canary predefinidas a seguir estão disponíveis. Também é possível criar implantações personalizadas lineares e de canary. Para obter mais informações, consulte Trabalhar com configurações de implantação no Guia do usuário do AWS CodeDeploy.

- CodeDeployDefault.ECSAllAtOnce: desloca todo o tráfego para o contêiner atualizado do Amazon ECS de uma só vez
- CodeDeployDefault.ECSLinear10PercentEvery1Minutes: desloca 10% do tráfego a cada minuto até que todo o tráfego seja deslocado.
- CodeDeployDefault.ECSLinear10PercentEvery3Minutes: desloca 10% do tráfego a cada 3 minutos até que todo o tráfego seja deslocado.
- CodeDeployDefault.ECSCanary10Percent5Minutes: desloca 10% do tráfego no primeiro incremento. Os 90 por cento restantes são implantados cinco minutos depois.
- CodeDeployDefault.ECSCanary10Percent15Minutes: desloca 10% do tráfego no primeiro incremento. Os 90 por cento restantes são implantados 15 minutos depois.

Revisão

Uma revisão é o arquivo de especificação de aplicação do CodeDeploy (arquivo AppSpec). No arquivo AppSpec, você especifica o ARN completo da definição da tarefa e o contêiner e a porta do conjunto de tarefas de substituição em que o tráfego deve ser roteado quando uma nova implantação é criada. O nome do contêiner deve ser um dos nomes de contêineres referenciados em sua definição de tarefa. Se a configuração de rede ou a versão da plataforma tiver sido atualizada na definição de serviço, você também deverá especificar esses detalhes no arquivo AppSpec. Também é possível especificar as funções do Lambda a serem executadas durante os eventos do ciclo de vida da implantação. As funções do Lambda permitem que você execute testes e retorne métricas durante a implantação. Para obter mais informações, consulte Referência do arquivo AppSpec no Guia do usuário do AWS CodeDeploy.

Considerações

Considere o seguinte ao usar o tipo de implantação azul/verde:

- Quando um serviço do Amazon ECS que usa o tipo de implantação azul/verde é criado inicialmente, é criado um conjunto de tarefas do Amazon ECS.
- Você deve configurar o serviço para usar um Application Load Balancer ou um Network Load Balancer. A seguir estão os requisitos do load balancer.
  - Você deve adicionar um listener de produção ao load balancer, que é usado para rotear o tráfego de produção.
  - Um listener de teste opcional pode ser adicionado ao load balancer, que é usado para rotear o tráfego de teste. Se você especificar um listener de teste, o CodeDeploy encaminhará o tráfego de teste para o conjunto de tarefas de substituição durante uma implantação.
  - Os listeners de produção e teste devem pertencer ao mesmo load balancer.
  - É necessário definir um grupo de destino para o load balancer. O grupo de destino roteia o tráfego para o conjunto de tarefas original em um serviço por meio do listener de produção.
  - Quando um Network Load Balancer é usado, somente a configuração de implantação `CodeDeployDefault.ECSAllAtOnce` é permitida.
- Para serviços configurados para usar autoescalabilidade de serviço e o tipo de implantação azul/verde, a autoescalabilidade não é bloqueada durante uma implantação, mas a implantação pode apresentar falha em algumas circunstâncias. Veja a seguir a descrição desse comportamento, com mais detalhes.
  - Se um serviço estiver sendo dimensionado e uma implantação for iniciada, será criado o conjunto de tarefas verde e o CodeDeploy aguardará por até uma hora até que o conjunto de tarefas verde atinja o estado estacionário e não mude qualquer tráfego até que isso aconteça.
  - Se um serviço estiver no processo de implantação azul/verde e ocorrer um evento de escalabilidade, o tráfego continuará a mudar por 5 minutos. Se o serviço não atingir o estado estacionário em até 5 minutos, o CodeDeploy interromperá a implantação e a marcará como falha.
- As tarefas que usam o tipo de inicialização do Fargate ou os tipos de controlador de implantação `CODE_DEPLOY` não são compatíveis com a estratégia de programação `DAEMON`.
- Ao criar inicialmente uma aplicação e um grupo de implantação do CodeDeploy, é necessário especificar o seguinte:
  - É necessário definir dois grupos de destino para o load balancer. Um grupo de destino deverá ser o grupo de destino inicial definido para o balanceador de carga quando o serviço do Amazon ECS for criado. O único requisito do segundo grupo de destino é que ele não pode ser associado a um load balancer diferente do que é usado pelo serviço.

- Quando você cria uma implantação do CodeDeploy para um serviço do Amazon ECS, o CodeDeploy cria um conjunto de tarefas de substituição (ou conjunto de tarefas verde) na implantação. Se você tiver adicionado um listener de teste ao balanceador de carga, o CodeDeploy encaminhará o tráfego de teste para o conjunto de tarefas de substituição. É nesse momento que você pode executar quaisquer testes de validação. O CodeDeploy redireciona o tráfego de produção do conjunto de tarefas original para o conjunto de tarefas de substituição, de acordo com as configurações de redirecionamento de tráfego do grupo de implantação.

Permissões obrigatórias do IAM

As implantações azul/verde podem ser realizadas usando uma combinação das APIs do Amazon ECS e do CodeDeploy. Os usuários devem ter as permissões apropriadas para esses serviços antes de poderem usar as implantações azuis/verdes do Amazon ECS no AWS Management Console ou com a AWS CLI ou os SDKs.

Além das permissões padrão do IAM para criar e atualizar serviços, o Amazon ECS exige as permissões a seguir. Essas permissões foram adicionadas à política `AmazonECS_FullAccess` do IAM. Para obter mais informações, consulte AmazonECS_FullAccess.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codedeploy:CreateApplication",
                "codedeploy:CreateDeployment",
                "codedeploy:CreateDeploymentGroup",
                "codedeploy:GetApplication",
                "codedeploy:GetDeployment",
                "codedeploy:GetDeploymentGroup",
                "codedeploy:ListApplications",
                "codedeploy:ListDeploymentGroups",
                "codedeploy:ListDeployments",
                "codedeploy:StopDeployment",
                "codedeploy:GetDeploymentTarget",
                "codedeploy:ListDeploymentTargets",
                "codedeploy:GetDeploymentConfig",
                "codedeploy:GetApplicationRevision",
                "codedeploy:RegisterApplicationRevision",
                "codedeploy:BatchGetApplicationRevisions",
```

```
                "codedeploy:BatchGetDeploymentGroups",
                "codedeploy:BatchGetDeployments",
                "codedeploy:BatchGetApplications",
                "codedeploy:ListApplicationRevisions",
                "codedeploy:ListDeploymentConfigs",
                "codedeploy:ContinueDeployment",
                "sns:ListTopics",
                "cloudwatch:DescribeAlarms",
                "lambda:ListFunctions"
            ],
            "Resource": ["*"]
        }
    ]
}
```

> **Note**
>
> Além das permissões padrão do Amazon ECS necessárias para executar tarefas e serviços, os usuários do IAM também precisam de permissões `iam:PassRole` para usar perfis do IAM para tarefas.

O CodeDeploy precisa de permissões para chamar APIs do Amazon ECS, modificar o Elastic Load Balancing, invocar funções do Lambda e descrever alarmes do CloudWatch, além de permissões para modificar a contagem desejada do serviço para você. Antes de criar um serviço do Amazon ECS que use o tipo de implantação azul/verde, você deve criar uma função do IAM (`ecsCodeDeployRole`). Para obter mais informações, consulte Função do IAM para CodeDeploy do Amazon ECS.

Os exemplos de política Exemplo de criação de serviço do Amazon ECS e Exemplo de serviço de atualização do Amazon ECS do IAM mostram as permissões que são necessárias para os usuários usarem implantações azuis/verdes do Amazon ECS no AWS Management Console.

Migrar implantações azul/verde do CodeDeploy para implantações azul/verde do Amazon ECS

As implantações azul/verde do CodeDeploy e do Amazon ECS oferecem funcionalidades semelhantes, mas diferem na forma como você as configura e gerencia.

Visão geral da implantação azul/verde do CodeDeploy

Ao criar um serviço do Amazon ECS usando o CodeDeploy, você:

1. Crie um balanceador de carga com um receptor de produção e (opcionalmente) um receptor de teste. Cada receptor é configurado com uma única regra (padrão) que roteia todo o tráfego para um único grupo de destino (o grupo de destino principal).
2. Crie um serviço do Amazon ECS, configurado para usar o receptor e o grupo de destino, com o tipo `deploymentController` definido como `CODE_DEPLOY`. A criação do serviço resulta na criação de um conjunto de tarefas (azul) registrado com o grupo de destino especificado.
3. Crie um grupo de implantação do CodeDeploy (como parte de uma aplicação do CodeDeploy) e configure-o com detalhes do cluster do Amazon ECS, nome do serviço, receptores do balanceador de carga, dois grupos de destino (o grupo de destino principal usado na regra de receptor de produção e um grupo de destino secundário a ser usado para tarefas de substituição), um perfil de serviço (para conceder permissões ao CodeDeploy para manipular o Amazon ECS e recursos do Elastic Load Balancing) e vários parâmetros que controlam o comportamento da implantação.

Com o CodeDeploy, novas versões de um serviço são implantadas usando `CreateDeployment()`, especificando o nome da aplicação do CodeDeploy, o nome do grupo de implantação e um arquivo AppSpec que fornece detalhes da nova revisão e dos ganchos opcionais do ciclo de vida. A implantação do CodeDeploy cria um conjunto de tarefas de substituição (verde) e registra suas tarefas no grupo de destino secundário. Quando isso se torna íntegro, está disponível para teste (opcional) e para produção. Em ambos os casos, o redirecionamento é obtido alterando a respectiva regra do receptor para apontar para o grupo de destino secundário associado ao conjunto de tarefas verde. A reversão é obtida alterando a regra do receptor de produção de volta para o grupo de destino principal.

Visão geral da implantação azul/verde do Amazon ECS

Com implantações azul/verde do Amazon ECS, a configuração de implantação é parte do próprio serviço do Amazon ECS:

1. Você deve pré-configurar o receptor de produção do balanceador de carga com uma regra que inclua dois grupos de destino com pesos de 1 e 0.
2. Você precisa especificar os seguintes recursos, ou atualizar os recursos do serviço:
   - O ARN dessa regra de receptor
   - Os dois grupos de destino
   - Um perfil do IAM para conceder permissão ao Amazon ECS para chamar as APIs do Elastic Load Balancing

- Um perfil opcional do IAM para executar funções do Lambda
- Defina o tipo `deploymentController` como `ECS` e `deploymentConfiguration.strategy` para `BLUE_GREEN`. Isso resulta na criação de uma implantação de serviço (azul) cujas tarefas são registradas no grupo de destino principal.

Com o Amazon ECS azul/verde, uma nova revisão de serviço é criada chamando `UpdateService()` do Amazon ECS, passando os detalhes da nova revisão. A implantação do serviço cria novas tarefas de revisão do serviço (verde) e as registra no grupo de destino secundário. O Amazon ECS processa as operações de redirecionamento e reversão mudando os pesos na regra do receptor.

Principais diferenças de implementação

Embora ambas as abordagens resultem na criação de um conjunto inicial de tarefas, a implementação subjacente difere:

- O CodeDeploy usa um conjunto de tarefas, enquanto o Amazon ECS usa uma revisão de serviço. Os conjuntos de tarefas do Amazon ECS são uma construção mais antiga que foi substituída pelas revisões e implantações de serviços do Amazon ECS. Esse último oferece maior visibilidade do processo de implantação, bem como do histórico de implantação e revisão do serviço.
- Com o CodeDeploy, os ganchos do ciclo de vida são especificados como parte do arquivo AppSpec fornecido a `CreateDeployment()`. Isso significa que os ganchos podem ser alterados de uma implantação para outra. Com o Amazon ECS azul/verde, os ganchos são especificados como parte da configuração do serviço, e qualquer atualização exigiria uma chamada `UpdateService()`.
- Tanto o CodeDeploy quanto o Amazon ECS azul/verde usam o Lambda para implementação de ganchos, mas as entradas e saídas esperadas são diferentes.

  Com o CodeDeploy, a função deve chamar `PutLifecycleEventHookExecutionStatus()` para retornar o status do gancho, que pode ser `SUCCEEDED` ou `FAILED`. Com o Amazon ECS, a resposta do Lambda é usada para indicar o status do gancho.
- O CodeDeploy invoca cada gancho como uma chamada única, e espera que o status final da execução seja retornado em até uma hora. Os ganchos do Amazon ECS são mais flexíveis, pois podem retornar um indicador `IN_PROGRESS`, o que sinaliza que o gancho deve ser invocado de novo repetidamente até que resulte em `SUCCEEDED` ou `FAILED`. Para obter mais informações, consulte Ganchos do ciclo de vida para implantações de serviços do Amazon ECS.

Abordagens de migração

Há três abordagens principais para migrar das implantações azul/verde do CodeDeploy para as implantações azul/verde do Amazon ECS. Cada abordagem tem características diferentes em termos de complexidade, risco, capacidade de reversão e possível tempo de inatividade.

Reutilizar os mesmos recursos do Elastic Load Balancing usados para o CodeDeploy

Você atualiza o serviço existente do Amazon ECS para usar o controlador de implantação do Amazon ECS com a estratégia de implantação azul/verde em vez do controlador de implantação do CodeDeploy. Ao usar essa abordagem, considere o seguinte:

- O procedimento de migração é mais simples porque você está atualizando o controlador de implantação de serviços e a estratégia de implantação existentes do Amazon ECS.
- Não há tempo de inatividade quando configurado e migrado corretamente.
- Uma reversão exige que você reverta a revisão do serviço.
- O risco é alto porque não há uma configuração azul/verde paralela.

Você usa o mesmo receptor de balanceador de carga e grupos de destino usados para o CodeDeploy. Se você estiver usando AWS CloudFormation, consulte Migração de um modelo de implantação azul/verde do CodeDeploy para AWS CloudFormation para um modelo azul/verde do Amazon ECS para AWS CloudFormation.

1. Modifique a regra padrão dos receptores de produção/teste para incluir o grupo de destino alternativo, e defina o peso do grupo de destino principal como 1 e o do grupo de destino alternativo como 0.

   Para o CodeDeploy, os receptores do balanceador de carga anexado ao serviço são configurados com uma única regra (padrão) que roteia todo o tráfego para um único grupo de destino. Para o Amazon ECS azul/verde, os receptores do balanceador de carga devem ser pré-configurados com uma regra que inclua os dois grupos de destino com pesos.
2. Atualize o serviço existente do Amazon ECS chamando a API `UpdateService` e definindo o parâmetro `deploymentController` como `ECS` e o parâmetro `deploymentStrategy` como `BLUE_GREEN`. Especifique os ARNs do grupo de destino, do grupo de destino alternativo, do receptor de produção e de um receptor de teste opcional.
3. Verifique se o serviço funciona conforme o esperado.

4. Exclua a configuração do CodeDeploy para esse serviço do Amazon ECS, pois agora você está usando o Amazon ECS azul/verde.

Novo serviço com um balanceador de carga existente

Essa abordagem usa a estratégia azul/verde para a migração.

Ao usar essa abordagem, considere o seguinte:

- A interrupção é mínima. Ela ocorre somente durante a troca de portas do Elastic Load Balancing.
- Uma reversão exige que você reverta a troca de portas do Elastic Load Balancing.
- O risco é baixo porque há configurações paralelas. Portanto, você pode testar antes da mudança de tráfego.

1. Deixe os receptores, os grupos de destino e o serviço do Amazon ECS para a configuração do CodeDeploy intactos para que você possa facilmente reverter para essa configuração, se necessário.
2. Crie novos grupos de destino e novos receptores (com portas diferentes dos receptores originais) no balanceador de carga existente. Em seguida, crie um novo serviço do Amazon ECS que corresponda ao serviço existente do Amazon ECS, exceto que você usa o `ECS` como controlador de implantação, `BLUE_GREEN` como estratégia de implantação e passa os ARNs para os novos grupos de destino e as novas regras de receptores.
3. Verifique a nova configuração enviando manualmente o tráfego HTTP para o serviço. Se tudo correr bem, troque as portas dos receptores originais e dos novos receptores para rotear o tráfego para a nova configuração.
4. Verifique a nova configuração e, se tudo continuar funcionando conforme o esperado, exclua a configuração do CodeDeploy.

Novo serviço com um balanceador de carga

Como na abordagem anterior, essa abordagem usa a estratégia azul/verde para a migração. A principal diferença é que a mudança da configuração do CodeDeploy para a configuração azul/verde do Amazon ECS ocorre em uma camada de proxy reverso acima do balanceador de carga. Exemplos de implementações para a camada de proxy reverso são o Route 53 e o CloudFront.

Essa abordagem é adequada para clientes que já têm essa camada de proxy reverso e se toda a comunicação com o serviço estiver acontecendo por meio dela (por exemplo, nenhuma comunicação direta no nível do balanceador de carga).

Ao usar essa abordagem, considere o seguinte:

- Isso requer uma camada de proxy reverso.
- O procedimento de migração é mais complexo porque você precisa atualizar o controlador de implantação de serviços e a estratégia de implantação existentes do Amazon ECS.
- A interrupção é mínima. Ela ocorre somente durante a troca de portas do Elastic Load Balancing.
- Uma reversão exige que você reverta as alterações na configuração do proxy.
- O risco é baixo porque há configurações paralelas. Portanto, você pode testar antes da mudança de tráfego.

1. Não modifique a configuração existente do CodeDeploy intacta (balanceador de carga, receptores, grupos de destino, serviço do Amazon ECS e grupo de implantação do CodeDeploy).
2. Crie um novo balanceador de carga, grupos de destino e receptores configurados para implantações azul/verde do Amazon ECS.

   Configure os recursos apropriados.

   - Application Load Balancer: para obter mais informações, consulte Recursos do Application Load Balancer para implantações azul/verde.
   - Network Load Balancer: para obter mais informações, consulte Recursos do Network Load Balancer para implantações azul/verde do Amazon ECS.
3. Crie um novo serviço com o `ECS` como controlador de implantação e `BLUE_GREEN` como estratégia de implantação, apontando para os novos recursos do balanceador de carga.
4. Verifique a nova configuração testando-a por meio do novo balanceador de carga.
5. Atualize a configuração do proxy reverso para rotear o tráfego para o novo balanceador de carga.
6. Observe a nova revisão do serviço e, se tudo continuar funcionando conforme o esperado, exclua a configuração do CodeDeploy.

Próximas etapas

Depois de migrar para implantações azul/verde do Amazon ECS:

- Atualize seus scripts de implantação e pipelines de CI/CD para usar a API `UpdateService` do Amazon ECS em vez da API `CreateDeployment` do CodeDeploy.
- Atualize seu monitoramento e alertas para rastrear as implantações do serviço do Amazon ECS em vez das implantações do CodeDeploy.
- Considere implementar testes automatizados do seu novo processo de implantação para garantir que ele funcione conforme o esperado.

Migração de uma implantação de serviço azul/verde do CodeDeploy para uma do Amazon ECS

Ao usar implantações azul/verde do Amazon ECS, você pode fazer e testar alterações no serviço antes de implementá-las em um ambiente de produção.

Você deve criar novos ganchos do ciclo de vida para a implantação azul/verde do Amazon ECS.

Pré-requisitos

Execute as operações a seguir antes de iniciar uma implantação azul/verde.

1. Substitua o perfil do IAM do CodeDeploy para Amazon ECS pelas seguintes permissões.
   - Para obter informações sobre as permissões do Elastic Load Balancing, consulte Perfil do IAM da infraestrutura do Amazon ECS para balanceadores de carga.
   - Para obter informações sobre as permissões do Lambda, consulte Permissões necessárias para funções do Lambda em implantações azul/verde do Amazon ECS.
2. Desative a automação do CodeDeploy. Para obter mais informações, consulte Working with deployment groups in CodeDeploy no Guia do usuário do CodeDeploy.
3. Certifique-se de ter as informações a seguir da sua implantação azul/verde do CodeDeploy. Você pode reutilizar essas informações para a implantação azul/verde do Amazon ECS:
   - O grupo de destino de produção
   - O receptor de produção
   - A regra de produção
   - O grupo de destino de teste

     Este é o grupo de destino para a revisão do serviço verde.

4. Certifique-se de que seus grupos de destino do Application Load Balancer estejam associados adequadamente às regras de receptor:
   - Se você não estiver usando receptores de teste, os dois grupos de destino (produção e teste) devem ser associados às regras de receptores de produção.
   - Se você estiver usando receptores de teste, um grupo de destino deve estar vinculado às regras de receptores de produção, e o outro grupo de destino deve estar vinculado às regras de receptores de teste.

   Se esse requisito não for atendido, a implantação do serviço falhará com o seguinte erro: `Service deployment rolled back because of invalid networking configuration. Both targetGroup and alternateTargetGroup must be associated with the productionListenerRule or testListenerRule.`
5. Verifique se não há implantações de serviço em andamento para o serviço. Para obter mais informações, consulte Visualize o histórico de serviços usando implantações de serviços do Amazon ECS.
6. As implantações azul/verde do Amazon ECS exigem que seu serviço use um dos recursos a seguir. Configure os recursos apropriados.
   - Application Load Balancer: para obter mais informações, consulte Recursos do Application Load Balancer para implantações azul/verde.
   - Network Load Balancer: para obter mais informações, consulte Recursos do Network Load Balancer para implantações azul/verde do Amazon ECS.
   - Service Connect: para obter mais informações, consulte Recursos do Service Connect para implantações azul/verde do Amazon ECS.
7. Decida se você deseja executar funções do Lambda para os estágios do ciclo de vida na implantação azul/verde do Amazon ECS.
   - Antes de aumentar a escala verticalmente
   - Depois de aumentar a escala verticalmente
   - Mudança do tráfego de teste
   - Depois da mudança do tráfego de teste
   - Mudança do tráfego de produção
   - Depois da mudança do tráfego de produção

   Crie funções do Lambda para cada estágio do ciclo de vida. Para obter mais informações, consulte Criar uma função do Lambda com o console no Guia do desenvolvedor do AWS Lambda.

Para obter mais informações sobre como atualizar o controlador de implantação de um serviço, consulte Atualizar parâmetros de serviço do Amazon ECS.

Procedimento

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, escolha o cluster.

   A página de detalhes do cluster é exibida.
3. Na guia Serviços, escolha o serviço.

   A página de detalhes do serviço é exibida.
4. No banner, escolha Atualizar tipo de controlador de implantação.

   A página Migrar tipo de controlador de implantação é exibida.
5. Expanda Novo e especifique os parâmetros a seguir.

   a. Em Tipo de controlador de implantação, escolha ECS.

   b. Em Estratégia de implantação, escolha Azul/verde.

   c. Em Tempo de incorporação, insira o tempo em que as revisões de serviço azul e verde são executadas.

   d. Para executar funções do Lambda em um estágio do ciclo de vida, em Ganchos do ciclo de vida de implantação, faça o seguinte para cada função exclusiva do Lambda:

      i. Escolha Adicionar.

         Repita o procedimento para cada função exclusiva que você deseja executar.

      ii. Em Função do Lambda, insira o nome da função.

      iii. Em Perfil, escolha o perfil que você criou nos pré-requisitos com as permissões azul/verde.

         Para obter mais informações, consulte Permissões necessárias para funções do Lambda em implantações azul/verde do Amazon ECS.

      iv. Em Estágios do ciclo de vida, selecione os estágios que a função do Lambda executa.

      v. (Opcional) Em Detalhes do gancho, insira um par de chave/valor que forneça informações sobre o gancho.
6. Expanda o Balanceamento de carga e configure o seguinte:

a. Em Perfil, escolha o perfil que você criou nos pré-requisitos com as permissões azul/verde.

   Para obter mais informações, consulte Permissões necessárias para funções do Lambda em implantações azul/verde do Amazon ECS.

b. Em Receptor, escolha o receptor de produção da sua implantação azul/verde do CodeDeploy.

c. Em Regra de produção, escolha a regra de produção da sua implantação azul/verde do CodeDeploy.

d. Em Regra de teste, escolha a regra de teste da sua implantação azul/verde do CodeDeploy.

e. Em Grupo de destino, escolha o grupo de destino de produção em sua implantação azul/verde do CodeDeploy.

f. Em Grupo de destino alternativo, escolha o grupo de destino de teste da sua implantação azul/verde do CodeDeploy.

7. Selecione Atualizar.

Próximas etapas

- Atualize o serviço para iniciar a implantação. Para obter mais informações, consulte Atualizar um serviço do Amazon ECS.
- Monitore o processo de implantação para garantir que ele siga o padrão azul/verde:
  - A revisão do serviço verde é criada e tem a escala aumentada verticalmente
  - O tráfego de teste é roteado para a revisão verde (se configurado)
  - O tráfego de produção é mudado para a revisão verde
  - Após o tempo de incorporação, a revisão azul é encerrada

Migração de um modelo de implantação azul/verde do CodeDeploy para AWS CloudFormation para um modelo azul/verde do Amazon ECS para AWS CloudFormation

Migre um modelo do AWS CloudFormation que usa implantações azul/verde do CodeDeploy para serviços do Amazon ECS para um que use a estratégia de implantação azul/verde nativa do Amazon ECS. A migração segue a abordagem “Reutilize os mesmos recursos do Elastic Load Balancing usados para o CodeDeploy”. Para obter mais informações, consulte a documentação do the section called “Migrar implantações azul/verde do CodeDeploy para implantações azul/verde do Amazon ECS”.

Modelo de origem

Este modelo usa a transformação `AWS::CodeDeployBlueGreen` e o gancho `AWS::CodeDeploy::BlueGreen` para implementar implantações azul/verde para um serviço do Amazon ECS.

Origem

Este é o modelo completo do AWS CloudFormation usando a implantação azul/verde do CodeDeploy. Para obter mais informações, consulte Exemplo de modelo de implantação azul/verde no Guia do usuário do AWS CloudFormation:

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "Vpc": {
      "Type": "AWS::EC2::VPC::Id"
    },
    "Subnet1": {
      "Type": "AWS::EC2::Subnet::Id"
    },
    "Subnet2": {
      "Type": "AWS::EC2::Subnet::Id"
    }
  },
  "Transform": [
    "AWS::CodeDeployBlueGreen"
  ],
  "Hooks": {
    "CodeDeployBlueGreenHook": {
      "Type": "AWS::CodeDeploy::BlueGreen",
      "Properties": {
        "TrafficRoutingConfig": {
          "Type": "TimeBasedCanary",
          "TimeBasedCanary": {
            "StepPercentage": 15,
            "BakeTimeMins": 5
          }
        },
        "Applications": [
          {
            "Target": {
              "Type": "AWS::ECS::Service",
```

```
            "LogicalID": "ECSDemoService"
          },
          "ECSAttributes": {
            "TaskDefinitions": [
              "BlueTaskDefinition",
              "GreenTaskDefinition"
            ],
            "TaskSets": [
              "BlueTaskSet",
              "GreenTaskSet"
            ],
            "TrafficRouting": {
              "ProdTrafficRoute": {
                "Type": "AWS::ElasticLoadBalancingV2::Listener",
                "LogicalID": "ALBListenerProdTraffic"
              },
              "TargetGroups": [
                "ALBTargetGroupBlue",
                "ALBTargetGroupGreen"
              ]
            }
          }
        }
      ]
    }
  }
},
"Resources": {
  "ExampleSecurityGroup": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
      "GroupDescription": "Security group for ec2 access",
      "VpcId": {"Ref": "Vpc"},
      "SecurityGroupIngress": [
        {
          "IpProtocol": "tcp",
          "FromPort": 80,
          "ToPort": 80,
          "CidrIp": "0.0.0.0/0"
        },
        {
          "IpProtocol": "tcp",
          "FromPort": 8080,
          "ToPort": 8080,
```

```
            "CidrIp": "0.0.0.0/0"
          },
          {
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIp": "0.0.0.0/0"
          }
        ]
      }
    },
    "ALBTargetGroupBlue": {
      "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
      "Properties": {
        "HealthCheckIntervalSeconds": 5,
        "HealthCheckPath": "/",
        "HealthCheckPort": "80",
        "HealthCheckProtocol": "HTTP",
        "HealthCheckTimeoutSeconds": 2,
        "HealthyThresholdCount": 2,
        "Matcher": {
          "HttpCode": "200"
        },
        "Port": 80,
        "Protocol": "HTTP",
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "TargetType": "ip",
        "UnhealthyThresholdCount": 4,
        "VpcId": {"Ref": "Vpc"}
      }
    },
    "ALBTargetGroupGreen": {
      "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
      "Properties": {
        "HealthCheckIntervalSeconds": 5,
        "HealthCheckPath": "/",
        "HealthCheckPort": "80",
        "HealthCheckProtocol": "HTTP",
        "HealthCheckTimeoutSeconds": 2,
```

```
        "HealthyThresholdCount": 2,
        "Matcher": {
          "HttpCode": "200"
        },
        "Port": 80,
        "Protocol": "HTTP",
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "TargetType": "ip",
        "UnhealthyThresholdCount": 4,
        "VpcId": {"Ref": "Vpc"}
      }
    },
    "ExampleALB": {
      "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
      "Properties": {
        "Scheme": "internet-facing",
        "SecurityGroups": [
          {"Ref": "ExampleSecurityGroup"}
        ],
        "Subnets": [
          {"Ref": "Subnet1"},
          {"Ref": "Subnet2"}
        ],
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "Type": "application",
        "IpAddressType": "ipv4"
      }
    },
    "ALBListenerProdTraffic": {
      "Type": "AWS::ElasticLoadBalancingV2::Listener",
      "Properties": {
        "DefaultActions": [
          {
            "Type": "forward",
```

```
            "ForwardConfig": {
              "TargetGroups": [
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
                  "Weight": 1
                }
              ]
            }
          }
        ],
        "LoadBalancerArn": {"Ref": "ExampleALB"},
        "Port": 80,
        "Protocol": "HTTP"
      }
    },
    "ALBListenerProdRule": {
      "Type": "AWS::ElasticLoadBalancingV2::ListenerRule",
      "Properties": {
        "Actions": [
          {
            "Type": "forward",
            "ForwardConfig": {
              "TargetGroups": [
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
                  "Weight": 1
                }
              ]
            }
          }
        ],
        "Conditions": [
          {
            "Field": "http-header",
            "HttpHeaderConfig": {
              "HttpHeaderName": "User-Agent",
              "Values": [
                "Mozilla"
              ]
            }
          }
        ],
        "ListenerArn": {"Ref": "ALBListenerProdTraffic"},
        "Priority": 1
```

```
      }
    },
    "ECSTaskExecutionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Sid": "",
              "Effect": "Allow",
              "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
        },
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
        ]
      }
    },
    "BlueTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "ExecutionRoleArn": {"Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]},
        "ContainerDefinitions": [
          {
            "Name": "DemoApp",
            "Image": "nginxdemos/hello:latest",
            "Essential": true,
            "PortMappings": [
              {
                "HostPort": 80,
                "Protocol": "tcp",
                "ContainerPort": 80
              }
            ]
          }
        ],
        "RequiresCompatibilities": [
          "FARGATE"
        ],
```

```
        "NetworkMode": "awsvpc",
        "Cpu": "256",
        "Memory": "512",
        "Family": "ecs-demo"
      }
    },
    "ECSDemoCluster": {
      "Type": "AWS::ECS::Cluster",
      "Properties": {}
    },
    "ECSDemoService": {
      "Type": "AWS::ECS::Service",
      "Properties": {
        "Cluster": {"Ref": "ECSDemoCluster"},
        "DesiredCount": 1,
        "DeploymentController": {
          "Type": "EXTERNAL"
        }
      }
    },
    "BlueTaskSet": {
      "Type": "AWS::ECS::TaskSet",
      "Properties": {
        "Cluster": {"Ref": "ECSDemoCluster"},
        "LaunchType": "FARGATE",
        "NetworkConfiguration": {
          "AwsVpcConfiguration": {
            "AssignPublicIp": "ENABLED",
            "SecurityGroups": [
              {"Ref": "ExampleSecurityGroup"}
            ],
            "Subnets": [
              {"Ref": "Subnet1"},
              {"Ref": "Subnet2"}
            ]
          }
        },
        "PlatformVersion": "1.4.0",
        "Scale": {
          "Unit": "PERCENT",
          "Value": 100
        },
        "Service": {"Ref": "ECSDemoService"},
        "TaskDefinition": {"Ref": "BlueTaskDefinition"},
```

```
        "LoadBalancers": [
          {
            "ContainerName": "DemoApp",
            "ContainerPort": 80,
            "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"}
          }
        ]
      }
    },
    "PrimaryTaskSet": {
      "Type": "AWS::ECS::PrimaryTaskSet",
      "Properties": {
        "Cluster": {"Ref": "ECSDemoCluster"},
        "Service": {"Ref": "ECSDemoService"},
        "TaskSetId": {"Fn::GetAtt": ["BlueTaskSet", "Id"]}
      }
    }
  }
}
```

Etapas da migração

Remover recursos específicos do CodeDeploy

Você não precisa mais das seguintes propriedades:

- A transformação `AWS::CodeDeployBlueGreen`
- O gancho `CodeDeployBlueGreenHook`
- Os recursos `GreenTaskDefinition` e `GreenTaskSet` (estes serão gerenciados pelo Amazon ECS)
- O recurso `PrimaryTaskSet` (o Amazon ECS gerenciará conjuntos de tarefas internamente)

Reconfigurar o receptor do balanceador de carga

Modifique o recurso `ALBListenerProdTraffic` para usar uma ação de encaminhamento com dois grupos de destino:

```
{
  "DefaultActions": [
    {
      "Type": "forward",
```

```
      "ForwardConfig": {
        "TargetGroups": [
          {
            "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
            "Weight": 1
          },
          {
            "TargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
            "Weight": 0
          }
        ]
      }
    }
  ]
}
```

Atualizar as propriedades da implantação

Atualize e adicione o seguinte:

- Altere a propriedade `DeploymentController` de `EXTERNAL` para `ECS`.
- Adicione a propriedade `Strategy` e defina-a como BLUE_GREEN.
- Adicione a propriedade `BakeTimeInMinutes`.

```
{
  "DeploymentConfiguration": {
    "MaximumPercent": 200,
    "MinimumHealthyPercent": 100,
    "DeploymentCircuitBreaker": {
      "Enable": true,
      "Rollback": true
    },
    "BakeTimeInMinutes": 5,
    "Strategy": "BLUE_GREEN"
  }
}
```

- Adicione a configuração do balanceador de carga ao serviço:

```
{
  "LoadBalancers": [
    {
```

```
      "ContainerName": "DemoApp",
      "ContainerPort": 80,
      "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
      "AdvancedConfiguration": {
        "AlternateTargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
        "ProductionListenerRule": {"Ref": "ALBListenerProdRule"},
        "RoleArn": {"Fn::GetAtt": ["ECSInfrastructureRoleForLoadBalancers", "Arn"]}
      }
    }
  ]
}
```

- Adicione a referência de definição de tarefa ao serviço:

```
{
  "TaskDefinition": {"Ref": "BlueTaskDefinition"}
}
```

Criar o perfil AmazonECSInfrastructureRolePolicyForLoadBalancers

Adicione um novo perfil do IAM que permita ao Amazon ECS gerenciar recursos do balanceador de carga:

Substitua *user-input* pelo seu valor.

```
{
  "ECSInfrastructureRoleForLoadBalancers": {
    "Type": "AWS::IAM::Role",
    "Properties": {
      "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Sid": "AllowAccessToECSForInfrastructureManagement",
            "Effect": "Allow",
            "Principal": {
              "Service": "ecs.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
          }
        ]
      },
```

```
      "ManagedPolicyArns": [
        "arn:aws:iam::aws:policy/AmazonECSInfrastructureRolePolicyForLoadBalancers"
      ]
    }
  }
}
```

Testes de recomendações

1. Implante o modelo migrado em um ambiente de não produção.
2. Verifique se o serviço é implantado corretamente com a configuração inicial.
3. Teste uma implantação atualizando a definição da tarefa e observando o processo de implantação azul/verde.
4. Verifique se o tráfego muda corretamente entre as implantações azul e verde.
5. Teste a funcionalidade de reversão forçando uma falha na implantação.

Modelo após a migração

Modelo final

Este é o modelo completo do AWS CloudFormation usando uma implantação azul/verde do Amazon ECS:

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "Vpc": {
      "Type": "AWS::EC2::VPC::Id"
    },
    "Subnet1": {
      "Type": "AWS::EC2::Subnet::Id"
    },
    "Subnet2": {
      "Type": "AWS::EC2::Subnet::Id"
    }
  },
  "Resources": {
    "ExampleSecurityGroup": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "GroupDescription": "Security group for ec2 access",
```

```
        "VpcId": {"Ref": "Vpc"},
        "SecurityGroupIngress": [
          {
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIp": "0.0.0.0/0"
          },
          {
            "IpProtocol": "tcp",
            "FromPort": 8080,
            "ToPort": 8080,
            "CidrIp": "0.0.0.0/0"
          },
          {
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIp": "0.0.0.0/0"
          }
        ]
      }
    },
    "ALBTargetGroupBlue": {
      "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
      "Properties": {
        "HealthCheckIntervalSeconds": 5,
        "HealthCheckPath": "/",
        "HealthCheckPort": "80",
        "HealthCheckProtocol": "HTTP",
        "HealthCheckTimeoutSeconds": 2,
        "HealthyThresholdCount": 2,
        "Matcher": {
          "HttpCode": "200"
        },
        "Port": 80,
        "Protocol": "HTTP",
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "TargetType": "ip",
```

```
        "UnhealthyThresholdCount": 4,
        "VpcId": {"Ref": "Vpc"}
      }
    },
    "ALBTargetGroupGreen": {
      "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
      "Properties": {
        "HealthCheckIntervalSeconds": 5,
        "HealthCheckPath": "/",
        "HealthCheckPort": "80",
        "HealthCheckProtocol": "HTTP",
        "HealthCheckTimeoutSeconds": 2,
        "HealthyThresholdCount": 2,
        "Matcher": {
          "HttpCode": "200"
        },
        "Port": 80,
        "Protocol": "HTTP",
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "TargetType": "ip",
        "UnhealthyThresholdCount": 4,
        "VpcId": {"Ref": "Vpc"}
      }
    },
    "ExampleALB": {
      "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
      "Properties": {
        "Scheme": "internet-facing",
        "SecurityGroups": [
          {"Ref": "ExampleSecurityGroup"}
        ],
        "Subnets": [
          {"Ref": "Subnet1"},
          {"Ref": "Subnet2"}
        ],
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
```

```
        }
      ],
      "Type": "application",
      "IpAddressType": "ipv4"
    }
  },
  "ALBListenerProdTraffic": {
    "Type": "AWS::ElasticLoadBalancingV2::Listener",
    "Properties": {
      "DefaultActions": [
        {
          "Type": "forward",
          "ForwardConfig": {
            "TargetGroups": [
              {
                "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
                "Weight": 1
              },
              {
                "TargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
                "Weight": 0
              }
            ]
          }
        }
      ],
      "LoadBalancerArn": {"Ref": "ExampleALB"},
      "Port": 80,
      "Protocol": "HTTP"
    }
  },
  "ALBListenerProdRule": {
    "Type": "AWS::ElasticLoadBalancingV2::ListenerRule",
    "Properties": {
      "Actions": [
        {
          "Type": "forward",
          "ForwardConfig": {
            "TargetGroups": [
              {
                "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
                "Weight": 1
              },
              {
```

```
                "TargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
                "Weight": 0
              }
            ]
          }
        }
      ],
      "Conditions": [
        {
          "Field": "http-header",
          "HttpHeaderConfig": {
            "HttpHeaderName": "User-Agent",
            "Values": [
              "Mozilla"
            ]
          }
        }
      ],
      "ListenerArn": {"Ref": "ALBListenerProdTraffic"},
      "Priority": 1
    }
  },
  "ECSTaskExecutionRole": {
    "Type": "AWS::IAM::Role",
    "Properties": {
      "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
              "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
          }
        ]
      },
      "ManagedPolicyArns": [
        "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
      ]
    }
  },
  "ECSInfrastructureRoleForLoadBalancers": {
```

```
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Sid": "AllowAccessToECSForInfrastructureManagement",
              "Effect": "Allow",
              "Principal": {
                "Service": "ecs.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
        },
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/AmazonECSInfrastructureRolePolicyForLoadBalancers"
        ]
      }
    },
    "BlueTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "ExecutionRoleArn": {"Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]},
        "ContainerDefinitions": [
          {
            "Name": "DemoApp",
            "Image": "nginxdemos/hello:latest",
            "Essential": true,
            "PortMappings": [
              {
                "HostPort": 80,
                "Protocol": "tcp",
                "ContainerPort": 80
              }
            ]
          }
        ],
        "RequiresCompatibilities": [
          "FARGATE"
        ],
        "NetworkMode": "awsvpc",
        "Cpu": "256",
        "Memory": "512",
```

```
        "Family": "ecs-demo"
      }
    },
    "ECSDemoCluster": {
      "Type": "AWS::ECS::Cluster",
      "Properties": {}
    },
    "ECSDemoService": {
      "Type": "AWS::ECS::Service",
      "Properties": {
        "Cluster": {"Ref": "ECSDemoCluster"},
        "DesiredCount": 1,
        "DeploymentController": {
          "Type": "ECS"
        },
        "DeploymentConfiguration": {
          "MaximumPercent": 200,
          "MinimumHealthyPercent": 100,
          "DeploymentCircuitBreaker": {
            "Enable": true,
            "Rollback": true
          },
          "BakeTimeInMinutes": 5,
          "Strategy": "BLUE_GREEN"
        },
        "NetworkConfiguration": {
          "AwsvpcConfiguration": {
            "AssignPublicIp": "ENABLED",
            "SecurityGroups": [
              {"Ref": "ExampleSecurityGroup"}
            ],
            "Subnets": [
              {"Ref": "Subnet1"},
              {"Ref": "Subnet2"}
            ]
          }
        },
        "LaunchType": "FARGATE",
        "PlatformVersion": "1.4.0",
        "TaskDefinition": {"Ref": "BlueTaskDefinition"},
        "LoadBalancers": [
          {
            "ContainerName": "DemoApp",
            "ContainerPort": 80,
```

```
            "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
            "AdvancedConfiguration": {
              "AlternateTargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
              "ProductionListenerRule": {"Ref": "ALBListenerProdRule"},
              "RoleArn": {"Fn::GetAtt": ["ECSInfrastructureRoleForLoadBalancers",
 "Arn"]}
            }
          }
        ]
      }
    }
  }
}
```

Migração de uma implantação azul/verde do CodeDeploy para uma implantação de serviço de atualização contínua do Amazon ECS

Você pode migrar suas implantações de serviço de uma implantação azul/verde do CodeDeploy para uma implantação de atualização contínua do Amazon ECS. Isso permite que você migre da dependência do CodeDeploy para uma implantação integrada.

O agendador de serviços do Amazon ECS substitui as tarefas em execução no momento por novas tarefas. O número de tarefas que o Amazon ECS adiciona ou remove do serviço durante uma atualização contínua é controlado pela configuração de implantação do serviço.

Pré-requisitos

Execute as operações a seguir antes de iniciar uma implantação azul/verde.

1. Você não precisa mais do perfil do IAM do CodeDeploy para o Amazon ECS.
2. Desative a automação do CodeDeploy. Para obter mais informações, consulte Working with deployment groups in CodeDeploy no Guia do usuário do CodeDeploy.
3. Verifique se não há implantações de serviço em andamento para o serviço. Para obter mais informações, consulte Visualize o histórico de serviços usando implantações de serviços do Amazon ECS.

Para obter mais informações sobre como atualizar o controlador de implantação de um serviço, consulte Atualizar parâmetros de serviço do Amazon ECS.

Procedimento

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, escolha o cluster.

   A página de detalhes do cluster é exibida.
3. Na guia Serviços, escolha o serviço.

   A página de detalhes do serviço é exibida.
4. No banner, escolha Migrar.

   A página Atualizar configuração de implantação é exibida.
5. Expanda Opções de implantação e especifique os parâmetros a seguir.

   a. Em Tipo de controlador de implantação, escolha ECS.

   b. Em Estratégia de implantação, escolha Atualização contínua.

   c. Em Min running tasks (Mínimo de tarefas em execução), insira o limite inferior do número de tarefas do serviço que devem permanecer no estado `RUNNING` durante uma implantação, como uma porcentagem do número desejado de tarefas (arredondado para o número inteiro superior mais próximo). Para obter mais informações, consulte Configuração da implantação.

   d. Em Max running tasks (Máximo de tarefas em execução), insira o limite superior do número de tarefas do serviço que devem permanecer no estado `RUNNING` ou `PENDING` durante uma implantação, como uma porcentagem do número desejado de tarefas (arredondado para o número inteiro inferior mais próximo).
6. Expanda o Balanceamento de carga e configure o seguinte:

   a. Em Perfil, escolha o perfil que você criou nos pré-requisitos com as permissões azul/verde.

      Para obter mais informações, consulte Permissões necessárias para funções do Lambda em implantações azul/verde do Amazon ECS.

   b. Em Receptor, escolha o receptor de produção da sua implantação azul/verde do CodeDeploy.

   c. Em Grupo de destino, escolha o grupo de destino de produção em sua implantação azul/verde do CodeDeploy.
7. Selecione Atualizar.

Próximas etapas

É necessário atualizar o serviço para que a alteração tenha efeito. Para obter mais informações, consulte Atualizar um serviço do Amazon ECS.

Atualizar a estratégia de implantação do Amazon ECS

O Amazon ECS é compatível com várias estratégias de implantação para atualizar seus serviços. Você pode migrar entre essas estratégias com base nos requisitos da aplicação. Este tópico explica como migrar entre implantações contínuas e implantações azul/verde.

Compreensão das estratégias de implantação do Amazon ECS

Antes de migrar entre as estratégias de implantação, é importante entender como cada estratégia funciona e suas principais diferenças:

Implantações contínuas

Em uma implantação contínua, o Amazon ECS substitui a versão atual em execução da sua aplicação por uma nova versão. O agendador de serviços usa os parâmetros de porcentagem mínima e máxima de integridade para determinar a estratégia de implantação.

As implantações contínuas são mais simples de configurar, mas oferecem menos controle sobre o processo de implantação e o roteamento do tráfego.

Implantações azuis/verdes

Em uma implantação azul/verde, o Amazon ECS cria uma nova versão do seu serviço (verde) junto com a versão existente (azul). Isso permite que você verifique a nova versão antes de rotear o tráfego de produção para ela.

As implantações azul/verde fornecem mais controle sobre o processo de implantação, incluindo recursos de mudança de tráfego, testes e reversão.

Práticas recomendadas

Siga estas práticas recomendadas ao migrar entre estratégias de implantação:

- Teste em um ambiente que não seja de produção: sempre teste a atualização em um ambiente de não produção antes de aplicar as alterações nos serviços de produção.

- Plano de reversão: tenha um plano de reversão caso a atualização não funcione conforme o esperado.
- Monitore durante a transição: monitore de perto seu serviço durante e após a migração para garantir que ele continue operando corretamente.
- Atualize a documentação: atualize sua documentação de implantação para refletir a nova estratégia de implantação.
- Considere o impacto do tráfego: entenda como a atualização pode afetar o tráfego do seu serviço e planeje adequadamente.

Atualização da estratégia de implantação da atualização contínua para o Amazon ECS azul/verde

Você pode migrar de uma implantação de atualização contínua para uma implantação azul/verde do Amazon ECS quando quiser fazer e testar alterações no serviço antes de implementá-las em um ambiente de produção.

Pré-requisitos

Antes de migrar seu serviço de implantações contínuas para azul/verde, verifique se você tem o seguinte:

- Aguarde a conclusão de todas as implantações atuais.
- Um serviço existente do Amazon ECS usando a estratégia de implantação contínua.
- Se você tiver várias revisões de serviço entregando tráfego, o Amazon ECS tentará consolidar o tráfego em uma única revisão durante a migração. Se isso falhar, poderá ser necessário atualizar manualmente o serviço para usar uma única revisão antes de migrar.
- Configurar as permissões apropriadas.
  - Para obter informações sobre as permissões do Elastic Load Balancing, consulte Perfil do IAM da infraestrutura do Amazon ECS para balanceadores de carga.
  - Para obter informações sobre as permissões do Lambda, consulte Permissões necessárias para funções do Lambda em implantações azul/verde do Amazon ECS.
- Dependendo da configuração, você deve executar uma das seguintes opções:
  - Se seu serviço usa o Elastic Load Balancing, atualize-o com a nova “advancedConfiguration” e inicie uma implantação contínua.
  - Se seu serviço usa o Service Connect, atualize-o e inicie uma implantação contínua.

  - Se o seu serviço tanto o Elastic Load Balancing como o Service Connect, execute as duas etapas acima (você pode usar uma única solicitação UpdateService).
  - Se o seu serviço não usar nenhuma das opções acima, nenhuma operação adicional será necessária.
- As implantações azul/verde do Amazon ECS exigem que seu serviço use um dos recursos a seguir. Configure os recursos apropriados.
  - Application Load Balancer: para obter mais informações, consulte Recursos do Application Load Balancer para implantações azul/verde.
  - Network Load Balancer: para obter mais informações, consulte Recursos do Network Load Balancer para implantações azul/verde do Amazon ECS.
  - Service Connect: para obter mais informações, consulte Recursos do Service Connect para implantações azul/verde do Amazon ECS.

Procedimento

1. Abra o console do Amazon ECS em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Clusters.
3. Na página Clusters, escolha o cluster que contém o serviço que você deseja migrar.

   A página de detalhes do cluster será exibida.
4. Na página Detalhes do cluster, escolha a guia Serviços.
5. Escolha o serviço e depois Atualizar.

   A página Atualizar serviço é exibida
6. Expanda Opções de implantação e especifique o seguinte:
7. Em Estratégia de implantação, escolha Azul/verde.
8. Defina as configurações da implantação azul/verde:

   a. Em Tempo de incorporação, insira o número de minutos em que as revisões do serviço azul e do verde serão executadas simultaneamente antes que a revisão azul seja encerrada.

      Isso possibilita um tempo para a verificação e testes.

   b. (Opcional) Configure as funções do Lambda para serem executadas em estágios específicos da implantação. Em Ganchos do ciclo de vida de implantação, configure as funções do Lambda para os seguintes estágios:

- Pré-aumento vertical da escala: é executado antes do aumento vertical da escala da revisão do serviço verde
- Pós-aumento vertical da escala: é executado após do aumento vertical da escala da revisão do serviço verde
- Teste de mudança de tráfego: é executado durante o roteamento de tráfego de teste para a revisão do serviço verde
- Mudança de tráfego pós-teste: é executado após o tráfego de teste ser roteado para a revisão do serviço verde
- Mudança de tráfego de produção: é executada durante o roteamento do tráfego de produção para a revisão do serviço verde
- Mudança de tráfego pós-produção: é executada após o tráfego de produção ser roteado para a revisão do serviço verde

Para adicionar um gancho do ciclo de vida:

a. Escolha Adicionar.
b. Em Função do Lambda, insira o nome da função ou o ARN.
c. Em Perfil, escolha o perfil do IAM que tem permissão para invocar a função do Lambda.
d. Em Estágios do ciclo de vida, selecione os estágios em que a função do Lambda deve ser executada.
e. Opcional: em Detalhes do gancho, insira pares de chave/valor para fornecer informações adicionais ao gancho.

9. Defina as configurações do balanceador de carga:

a. Em Balanceamento de carga, verifique se o serviço está configurado para usar um balanceador de carga.
b. Em Grupo de destino, escolha o grupo de destino principal para seu ambiente de produção (azul).
c. Em Grupo de destino alternativo, escolha o grupo de destino para seu ambiente de teste (verde).
d. Em Regra de receptor de produção, escolha a regra de receptor para rotear o tráfego de produção.

e. Opcional: em Testar regra de receptor, escolha uma regra de receptor para rotear o tráfego de teste para seu ambiente verde.

f. Em Perfil, escolha o perfil do IAM que permite que o Amazon ECS gerencie seu balanceador de carga.

10. Reveja as alterações da configuração e escolha Atualizar.

Próximas etapas

- Atualize o serviço para iniciar a implantação. Para obter mais informações, consulte Atualizar um serviço do Amazon ECS.
- Monitore o processo de implantação para garantir que ele siga o padrão azul/verde:
  - A revisão do serviço verde é criada e tem a escala aumentada verticalmente
  - O tráfego de teste é roteado para a revisão verde (se configurado)
  - O tráfego de produção é mudado para a revisão verde
  - Após o tempo de incorporação, a revisão azul é encerrada

Atualização da estratégia de implantação azul/verde do Amazon ECS para a atualização contínua

É possível migrar uma implantação azul/verde para uma implantação de atualização contínua.

Ao migrar para implantações contínuas, tenha em mente as seguintes considerações:

- Processamento de tráfego: com implantações contínuas, novas tarefas começam a receber tráfego assim que passam pelas verificações de integridade. Não há uma fase de teste separada, como nas implantações azul/verde.
- Eficiência de recursos: as implantações contínuas normalmente usam menos recursos do que as implantações azul/verde porque substituem as tarefas de forma incremental em vez de criarem um ambiente totalmente duplicado.
- Complexidade de reversão: as implantações contínuas tornam as reversões mais complexas em comparação com as implantações azul/verdes. Se você precisar reverter, deverá iniciar uma nova implantação com a definição da tarefa anterior.
- Velocidade de implantação: as implantações contínuas podem levar mais tempo para serem concluídas do que as implantações azul/verdes, especialmente para serviços com muitas tarefas.

- Configuração do balanceador de carga: sua configuração atual do balanceador de carga continuará funcionando com implantações contínuas, mas o comportamento de mudança de tráfego será diferente.

Pré-requisitos

Antes de migrar seu serviço de implantações azul/verde para contínuas, verifique se você tem o seguinte:

- Um serviço existente do Amazon ECS usando a estratégia de implantação azul/verde
- Não há implantações contínuas para o serviço (aguarde a conclusão de todas as implantações atuais)
- Uma compreensão clara de como seu serviço se comportará com implantações contínuas

> **Note**
>
> Você não poderá migrar um serviço para uma implantação contínua se ele tiver uma implantação em andamento. Aguarde que todas as implantações atuais sejam concluídas antes de continuar.

Procedimento de migração

Siga estas etapas para migrar seu serviço do Amazon ECS de azul/verde para implantações contínuas:

1. Abra o console do Amazon ECS em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Clusters.
3. Na página Clusters, escolha o cluster que contém o serviço que você deseja migrar.
4. Na página Detalhes do cluster, escolha a guia Serviços.
5. Selecione o serviço que você deseja migrar e escolha Atualizar.
6. Na página Atualizar serviço, navegue até a seção Opções de implantação e expanda-a, se necessário.
7. Em Estratégia de implantação, escolha Atualização contínua.
8. Defina as configurações de implantação contínua:

a. Em Porcentagem de integridade mínima, insira a porcentagem mínima de tarefas que seu serviço deve manter no estado `RUNNING` durante uma implantação. Esse valor é especificado como uma porcentagem do número de tarefas desejadas para o serviço.

b. Em Porcentagem máxima, insira a porcentagem máxima de tarefas que são permitidas no estado `RUNNING` ou `PENDING` durante uma implantação. Esse valor é especificado como uma porcentagem do número de tarefas desejadas para o serviço.

9. Opcional: em Detecção de falhas de implantação, configure como o Amazon ECS detectará e lidará com falhas de implantação:

a. Para habilitar o disjuntor de implantação, escolha Usar o disjuntor de implantação de implantação.

b. Para reverter automaticamente implantações com falha, escolha Reverter em caso de falha.

10. Revise suas alterações de configuração e escolha Atualizar para salvá-las e migrar o serviço para uma implantação contínua.

O Amazon ECS atualizará sua configuração de serviço para usar a estratégia de implantação contínua. Na próxima vez que você atualizar o serviço, ele usará o processo de implantação contínua.

> **Note**
>
> Quando você migra da implantação azul/verde para a contínua, o Amazon ECS lida com a transição da seguinte forma:
>
> 1. Identificando a revisão atual do serviço ativo que está entregando tráfego.
> 2. Mantendo a configuração existente do balanceador de carga, mas alterando a forma como as novas implantações são processadas.
> 3. Preparando o serviço para futuras implantações contínuas.

Próximas etapas

- Atualize o serviço para iniciar a implantação. Para obter mais informações, consulte Atualizar um serviço do Amazon ECS.

Solucionar problemas de atualizações da estratégia de implantação do Amazon ECS

Esta seção fornece soluções para problemas comuns que você pode encontrar ao migrar estratégias de implantação.

Várias revisões de serviços ou conjuntos de tarefas

Os problemas a seguir estão relacionados à presença de várias revisões de serviço para uma implantação.

Vários conjuntos de tarefas ao atualizar o controlador de implantação do ECS

A mensagem de erro: `Updating the deployment controller is not supported when there are multiple tasksets in the service. Please ensure your service has only one taskset and try again.`

Solução: esse erro ocorre ao tentar alterar o tipo de controlador de implantação de um serviço com vários conjuntos de tarefas ativos. Para resolver esse problema do controlador de implantação `CODE_DEPLOY` ou `EXTERNAL`:

1. Verifique os conjuntos de tarefas atuais:

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].taskSets"
```

2. Aguarde a conclusão de todas as implantações em andamento.
3. Force uma nova implantação para limpar os conjuntos de tarefas:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --force-new-deployment
```

4. Se necessário, exclua manualmente os conjuntos de tarefas extras:

```
aws ecs delete-task-set --cluster your-cluster-name --service your-service-name
 --task-set task-set-id
```

5. Depois que restar apenas um conjunto de tarefas, tente atualizar novamente o controlador de implantação.

Para obter mais informações, consulte the section called “Controladores e estratégias de implantação de serviços”.

O conjunto de tarefas primárias está ausente ao atualizar o controlador de implantação do ECS

A mensagem de erro: `Updating the deployment controller requires a primary taskset in the service. Please ensure your service has a primary taskset and try again.`

Solução: esse erro ocorre ao tentar alterar o tipo de controlador de implantação de um serviço que não tem um conjunto de tarefas primárias. Para resolver esse problema:

1. Verifique o status do serviço e os conjuntos de tarefas.). Caso exista um conjunto de tarefas no serviço, ele deverá ser marcado como `ACTIVE`.

   ```
   aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].taskSets[*].[status,id]
   ```

   Se não houver conjuntos de tarefas no estado `ACTIVE`, migre a implantação. Para obter mais informações, consulte Abordagens de migração.

2. Se o serviço não tiver tarefas em execução, implante pelo menos uma tarefa atualizando o serviço:

   ```
   aws ecs update-service-primary-task-set --cluster your-cluster-name --service your-service-name --primary-task-set your-taskset-id
   ```

   Isso marcará o conjunto de tarefas (anteriormente `ACTIVE`) no serviço como status `PRIMARY`.

3. Aguarde até que a tarefa alcance um estado de execução estável. Você pode verificar o status com:

   ```
   aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].deployments"
   ```

4. Depois que o serviço tiver uma tarefa principal definida com tarefas em execução, tente atualizar novamente o controlador de implantação.

Para obter mais informações, consulte the section called “Controladores e estratégias de implantação de serviços”.

Incompatibilidade entre o tipo de detecção de falhas da implantação e o controlador de implantação

Os problemas a seguir estão relacionados a uma incompatibilidade entre o tipo de detecção de falha de implantação e o controlador de implantação.

Disjuntor de implantação com um controlador não ECS

A mensagem de erro: `Deployment circuit breaker feature is only supported with ECS deployment controller. Update to ECS deployment controller and try again.`

Solução: esse erro ocorre ao tentar habilitar o recurso de disjuntor de implantação em um serviço que não está usando o controlador de implantação do ECS. O disjuntor de implantação é compatível apenas com o controlador de implantação do ECS.

1. Verifique seu controlador de implantação atual do serviço:

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].deploymentController"
```

2. Atualize seu serviço para usar o controlador de implantação do ECS:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

3. Depois que o serviço estiver usando o controlador de implantação do ECS, habilite o disjuntor de implantação:

```
aws ecs update-service --cluster your-cluster-name --
service your-service-name --deployment-configuration
 "deploymentCircuitBreaker={enable=true,rollback=true}"
```

Para obter mais informações, consulte the section called “ Como o disjuntor de implantação realiza a detecção de falhas”.

Reversão baseada em alarme com controlador não ECS

A mensagem de erro: `Alarm based rollback feature is only supported with ECS deployment controller. Update to ECS deployment controller and try again.`

Solução: esse erro ocorre ao tentar configurar a reversão baseada em alarme em um serviço que não está usando o controlador de implantação do ECS. O recurso de reversão baseado em alarme é compatível apenas com o controlador de implantação do ECS.

1. Verifique seu controlador de implantação atual do serviço:

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].deploymentController"
```

2. Atualize seu serviço para usar o controlador de implantação do ECS:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

3. Depois que o serviço estiver usando o controlador de implantação do ECS, configure a reversão baseada em alarme:

```
aws ecs update-service --cluster your-cluster-name --services your-service-name --deployment-configuration "alarms={alarmNames=[your-alarm-name],enable=true,rollback=true}"
```

Para obter mais informações, consulte the section called “Como os alarmes do CloudWatch realizam a detecção de falhas de implantação”.

Incompatibilidade entre o Service Connect e o controlador de implantação

Os problemas a seguir estão relacionados a uma incompatibilidade entre o Service Connect e o controlador de implantação.

Controlador EXTERNAL com o Service Connect

A mensagem de erro: `The EXTERNAL deployment controller type is not supported for services using Service Connect.`

Solução: esse erro ocorre ao tentar usar o controlador de implantação EXTERNAL com um serviço que tenha o Service Connect habilitado. O controlador EXTERNAL não é compatível com o Service Connect.

1. Verifique se o seu serviço tem o Service Connect habilitado:

```
aws ecs describe-services --cluster your-cluster-name --services your-service-
name --query "services[0].serviceConnectConfiguration"
```

2. Se você precisar usar o controlador de implantação EXTERNAL, desabilite o Service Connect atualizando seu serviço:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --service-connect-configuration "{}"
```

3. Como alternativa, se você precisar usar o Service Connect, use o controlador de implantação do ECS:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

Para obter mais informações, consulte the section called “Controladores e estratégias de implantação de serviços”.

Service Connect com controlador não ECS

A mensagem de erro: `Service Connect feature is only supported with ECS (rolling update) deployment controller. Update to ECS deployment controller and try again.`

Solução: esse erro ocorre ao tentar habilitar o Service Connect que não está usando o controlador de implantação do ECS. O recurso Service Connect é compatível apenas com o controlador de implantação do ECS.

1. Verifique seu controlador de implantação atual do serviço:

```
aws ecs describe-services --cluster your-cluster-name --services your-service-
name --query "services[0].deploymentController"
```

2. Atualize seu serviço para usar o controlador de implantação do ECS:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

3. Quando o serviço estiver usando o controlador de implantação do ECS, habilite o Service Connect:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --service-connect-configuration "enabled=true,namespace=your-namespace"
```

Para obter mais informações, consulte the section called “Controladores e estratégias de implantação de serviços”.

Incompatibilidade entre o tipo de controlador e a estratégia de agendamento

Os problemas a seguir estão relacionados a uma incompatibilidade entre o tipo de controlador e a estratégia de agendamento.

Controlador `CODE_DEPLOY` com estratégia de agendamento `DAEMON`

A mensagem de erro: `The CODE_DEPLOY deployment controller type is not supported for services using the DAEMON scheduling strategy.`

Solução: esse erro ocorre ao tentar usar o controlador de implantação CODE_DEPLOY com um serviço que usa a estratégia de agendamento `DAEMON`. O controlador `CODE_DEPLOY` é compatível apenas com a estratégia de agendamento `REPLICA`.

1. Verifique a estratégia de agendamento atual do seu serviço:

   ```
   aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].schedulingStrategy"
   ```

2. Se você precisar de implantações azul/verde, altere o serviço para usar a estratégia de agendamento `REPLICA`:

   ```
   aws ecs update-service --cluster your-cluster-name --service your-service-name
    --scheduling-strategy REPLICA
   ```

3. Como alternativa, se você precisar usar a estratégia de agendamento `DAEMON`, use o controlador de implantação do `ECS`:

   ```
   aws ecs update-service --cluster your-cluster-name --service your-service-name
    --deployment-controller type=ECS
   ```

Para obter mais informações, consulte the section called “Controladores e estratégias de implantação de serviços”.

Controlador EXTERNAL com estratégia de agendamento DAEMON

A mensagem de erro: `The EXTERNAL deployment controller type is not supported for services using the DAEMON scheduling strategy.`

Solução: esse erro ocorre ao tentar usar o controlador de implantação EXTERNAL com um serviço do ECS que usa a estratégia de agendamento DAEMON. O controlador EXTERNAL é compatível apenas com a estratégia de agendamento REPLICA.

1. Verifique a estratégia de agendamento atual do seu serviço:

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].schedulingStrategy"
```

2. Se você precisar usar o controlador de implantação `EXTERNAL`, altere o serviço para usar a estratégia de agendamento `REPLICA`:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --scheduling-strategy REPLICA
```

3. Como alternativa, se você precisar usar a estratégia de agendamento `DAEMON`, use o controlador de implantação do `ECS`:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

Para obter mais informações, consulte the section called “Controladores e estratégias de implantação de serviços”.

Registros de serviços com tipo de inicialização externa

A mensagem de erro: `Service registries are not supported for external launch type.`

Solução: esse erro ocorre ao tentar configurar a descoberta de serviços (registros de serviços) para um serviço que usa o tipo de inicialização `EXTERNAL`. A descoberta de serviços não é compatível com o tipo de inicialização `EXTERNAL`.

1. Verifique o tipo de inicialização atual do seu serviço:

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].launchType"
```

2. Se você precisar da descoberta de serviços, altere o serviço para usar o tipo de inicialização do EC2 ou do FARGATE:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --launch-type FARGATE
```

3. Como alternativa, se você precisar usar o tipo de inicialização EXTERNAL, remova a configuração do registro do serviço:

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --service-registries "[]"
```

Para obter mais informações, consulte the section called “Controladores e estratégias de implantação de serviços”.

Reverter uma atualização do controlador de implantação

Se você decidir que deseja retornar ao controlador de implantação anterior, realize uma das seguintes ações:

- Se você usou o AWS CloudFormation, poderá usar o modelo anterior para criar uma nova pilha. Para obter mais informações, consulte Criar uma pilha no console do CloudFormation no Guia do usuário do AWS CloudFormation.
- Se você usou o console do Amazon ECS ou a AWS CLI, poderá atualizar o serviço. Para obter mais informações, consulte the section called “Atualizar um serviço”.

  Se você usar o comando update-service, use a opção `--deployment-controller` e defina-a para o controlador de implantação anterior.

Visualize o histórico de serviços usando implantações de serviços do Amazon ECS

As implantações de serviços fornecem uma visão abrangente das suas implantações. As implantações de serviços fornecem as seguintes informações sobre o serviço:

- A configuração da workload atualmente implantada (a revisão do serviço de origem)
- A configuração da workload em implantação (a revisão do serviço de destino)
- O status da implantação
- O número de tarefas com falha que a interrupção do circuito detectou
- Os alarmes do CloudWatch que estão em alarme
- Quando a implantação do serviço foi iniciada e concluída
- Os detalhes de uma reversão, caso tenha ocorrido

Para obter informações sobre as propriedades de implantação do serviço, consulte Propriedades incluídas na implantação de serviço do Amazon ECS.

As implantações de serviços são somente leitura e cada uma tem um ID exclusivo.

Há três estágios de implantação de um serviço:

Estágio	Definição	Estados associados
Pendente	Uma implantação de serviço foi criada, mas ainda não foi iniciada	PENDING
Contínuo	Uma implantação de serviço está em andamento	• IN_PROGRESS • STOP_REQUISITADO • ROLLBACK_REQUESTED • ROLLBACK_IN_PROGRESS
Concluído	A implantação de um serviço foi concluída (com ou sem êxito)	• SUCCESSFUL • STOPPED • ROLLBACK_SUCCESSFUL

Estágio	Definição	Estados associados
		• ROLLBACK_FAILED

Você usa implantações de serviços para entender o ciclo de vida do seu serviço e determinar se há alguma medida a ser tomada. Por exemplo, se uma reversão ocorreu, talvez seja necessário investigar a implantação do serviço e observar os eventos do serviço.

Você pode ver o histórico mais recente de 90 dias das implantações criadas em ou após 25 de outubro de 2024 usando o console, a API e a AWS CLI.

É possível interromper uma implantação que não foi concluída. Para obter mais informações, consulte Parar implantações de serviços do Amazon ECS.

Ciclo de vida de implantação de serviços

O Amazon ECS cria uma nova implantação de serviço automaticamente quando qualquer uma das seguintes ações acontece:

- Um usuário cria um serviço.
- Um usuário atualiza o serviço e usa a opção de forçar nova implantação.
- Um usuário atualiza uma ou mais propriedades do serviço que exigem uma implantação.

Enquanto a implantação está em andamento, o Amazon ECS atualiza as seguintes propriedades de implantação do serviço para refletir o progresso da implantação:

- O estado
- O número de tarefas em execução

  O número de tarefas em execução indicado na revisão do serviço pode não ser igual ao número real de tarefas em execução. Esse número representa o número de tarefas em execução quando a implantação foi concluída. Por exemplo, se você iniciou tarefas independentemente da implantação do serviço, essas tarefas não serão incluídas na contagem de tarefas em execução para a revisão do serviço.
- Detecção de falha no disjuntor:
  - O número de tarefas que falharam ao iniciar
- Detecção de falhas de alarme do CloudWatch

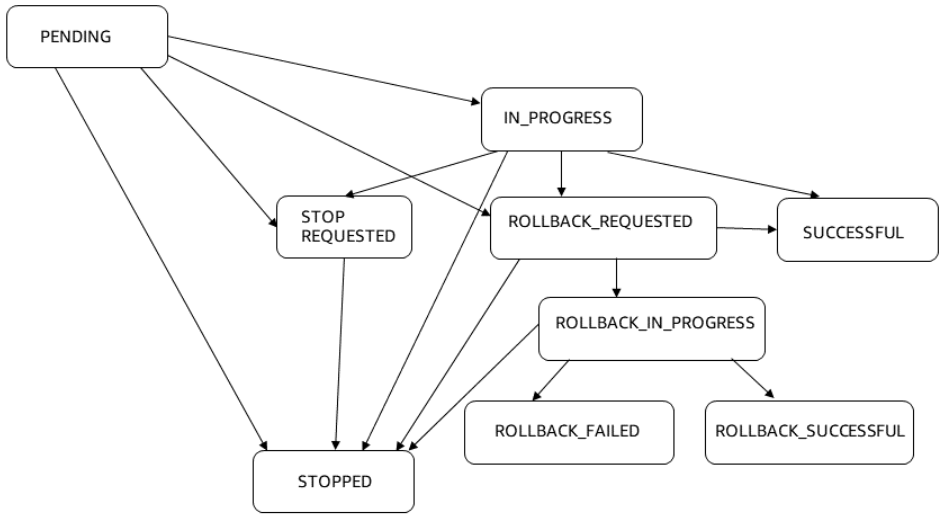
  - Os alarmes que estão ativos
- Informações de reversão:
  - A hora de início
  - O motivo da reversão
  - O ARN da revisão de serviço usada para a reversão
- O motivo do status

O Amazon ECS exclui a implantação do serviço quando você exclui um serviço.

Estados da implantação de serviços

A implantação de um serviço começa no estado `PENDING`.

A ilustração a seguir mostra os estados de implantação de um serviço que podem ocorrer após o estado `PENDING`: `IN_PROGRESS`, `ROLLBACK_REQUESTED`, `SUCCESSFUL`, `STOP_REQUESTED`, `ROLLBACK_IN_PROGRESSS`, `ROLLBACK_FAILED`, `ROLLBACK_SUCCESSFUL` e `STOPPED`.



As informações a seguir fornecem detalhes sobre os estados de implantação de um serviço:

- `PENDING`: a implantação de serviço foi criada, mas ainda não foi iniciada.

O estado pode avançar para `IN_PROGRESS`, `ROLLBACK_REQUESTED`, `STOP_REQUESTED` ou `STOPPED`.

- `IN_PROGRESS`: a implantação do serviço está em andamento.

  O estado pode avançar para `SUCCESSFUL`, `STOP_REQUESTED`, `ROLLBACK_REQUESTED`, `ROLLBACK_IN_PROGRESS` ou `STOPPED`.
- `STOP_REQUESTED`: o estado de implantação do serviço avança para `STOP_REQUESTED` quando qualquer uma das seguintes situações acontece:
  - Um usuário inicia uma novaa implantação de serviço.
  - A opção de reversão não está em uso para o mecanismo de detecção de falhas (disjuntor ou baseado em alarme) e o serviço não atinge o estado `SUCCESSFUL`.

  O estado avança para `STOPPED`.
- `ROLLBACK_REQUESTED`: o estado de implantação do serviço muda para `ROLLBACK_REQUESTED` quando um usuário solicita uma reversão via console, API ou CLI.

  O estado pode avançar para `SUCCESSFUL`, `ROLLBACK_IN_PROGRESS` e `STOPPED`.
- `SUCCESSFUL`: o estado de implantação do serviço avança para `SUCCESSFUL` quando a implantação do serviço é concluída com êxito.
- `ROLLBACK_IN_PROGRESS`: o estado de implantação do serviço avança para `ROLLBACK_IN_PROGRESS` quando a opção de reversão está em uso para o mecanismo de detecção de falhas (disjuntor ou baseado em alarme) e o serviço falha.

  O estado avança para `ROLLBACK_SUCCESSFUL` ou `ROLLBACK_FAILED`.

Propriedades incluídas na implantação de serviço do Amazon ECS

As propriedades a seguir estão incluídas em uma implantação de serviço.

Propriedade	Descrição	
ARN da implantação do serviço	O ARN da implantação do serviço.	
Nome de recurso da Amazon (ARN) do serviço	O ARN do serviço para esta implantação de serviço.	

Propriedade	Descrição
ARN do cluster	O ARN do cluster que hospeda o serviço.
Hora de criação da implantação do serviço	A hora em que a implantação do serviço foi criada.
Hora de início da implantação do serviço	A hora em que a implantação do serviço começou.
Hora de término da implantação do serviço	A hora em que a implantação do serviço terminou.
Hora de interrupção da implantação do serviço	A hora em que a implantação do serviço foi interrompida.
Hora de atualização da implantação do serviço	A hora da última atualização da implantação do serviço.
Revisões de serviços de origem	As revisões de serviços atualmente em execução. Para obter mais informações sobre as propriedades incluídas, consulte Propriedades incluídas na revisão de serviço do Amazon ECS.

Propriedade	Descrição
Configuração de implantação	Os parâmetros de implantação, incluindo a configuração do disjuntor e os alarmes. • O limite inferior do número de tarefas que devem estar em execução para um serviço durante uma implantação ou quando uma instância de contêiner está sendo drenada, como uma porcentagem do número desejado de tarefas para o serviço. • O limite superior do número de tarefas que devem estar em execução para um serviço durante uma implantação ou quando uma instância de contêiner está sendo drenada, como uma porcentagem do número desejado de tarefas para o serviço. • A configuração do disjuntor. • Os alarmes usados para detecção de falhas.

Propriedade	Descrição
Revisão do serviço de destino	A revisão do serviço a ser implantado. Depois que a implantação for concluída com êxito, a revisão do serviço de destino será a revisão do serviço em execução.
Status da implantação de serviços	O estado da implantação de serviços. Os valores válidos são PENDING, SUCCESSFUL, STOPPED, STOP_REQU ESTED, STOP_IN_P ROGRESS, IN_PROGRESS, ROLLBACK_IN_PROGRESS, ROLLBACK_SUCCESSFUL, e ROLLBACK_FAILED.
Informações sobre status de implantação de serviços	Informações sobre por que a implantação do serviço está no status atual. Por exemplo, o disjuntor detectou uma falha.
Informações de reversão	As opções de reversão usadas pela implantação do serviço quando a implantação falha.
Opções de disjuntores de implantação de serviços	O disjuntor que determina que a implantação de um serviço falhou.

Propriedade	Descrição	
Alarmes do CloudWatch para a implantação do serviço	Os alarmes do CloudWatch que determinam quando a implantação de um serviço falha.	

Permissões necessárias para visualizar implantações de serviço do Amazon ECS

Ao seguir a prática recomendada de conceder privilégio mínimo, é necessário adicionar permissões adicionais para visualizar implantações de serviço no console.

É necessário ter acesso às seguintes ações:

- ListServiceDeployments
- DescribeServiceDeployments
- DescribeServiceRevisions

É necessário ter acesso aos seguintes recursos:

- Serviço
- Implantação de serviços
- Revisão de serviços

O exemplo de política a seguir contém as permissões necessárias e limita as ações a um serviço especificado.

Substitua `account`, `cluster-name` e `service-name` por seus próprios valores.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "ecs:ListServiceDeployments",
              "ecs:DescribeServiceDeployments",
              "ecs:DescribeServiceRevisions"
```

```
            ],
            "Resource": [
             "arn:aws:ecs:us-east-1:123456789012:service/cluster-name/service-name",
             "arn:aws:ecs:us-east-1:123456789012:service-deployment/cluster-name/
service-name/*",
             "arn:aws:ecs:us-east-1:123456789012:service-revision/cluster-name/service-
name/*"
            ]
        }
    ]
}
```

Visualizar implantações de serviços do Amazon ECS

Você pode ver o histórico mais recente de 90 dias das implantações criadas em ou após 25 de outubro de 2024. As implantações de serviços podem estar em qualquer um dos seguintes estados:

- Em andamento
- Pendente
- Concluído

Você pode usar essas informações para determinar se precisa atualizar a forma como o serviço está sendo implantado ou a revisão do serviço. Para obter mais informações sobre as propriedades incluídas, consulte Propriedades incluídas na implantação de serviço do Amazon ECS.

Antes de começar, configure as permissões necessárias para visualizar as implantações de serviços. Para obter mais informações, consulte Permissões necessárias para visualizar implantações de serviço do Amazon ECS.

Amazon ECS Console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, escolha o cluster.
3. Na página de detalhes do cluster, na seção Serviços, escolha o serviço.

   A página de detalhes do serviço é exibida.
4. Na página de detalhes da implantação, escolha Implantações.
5. Escolha a implantação do serviço a ser visualizada.

Para visualizar implantaç ões de serviços para esse tipo de implantação	Faça o seguinte
Implantação em andamento	Em Implantações em andamento, escolha o ID da implantação.
Última implantação	Em Última implantação, escolha o ID da implantaç ão.
Implantações concluídas	Em Implantações de serviços, escolha o ID da implantação.

A página de detalhes da implantação do serviço é exibida.

6. (Opcional) Compare as revisões do serviço para visualizar as diferenças.

   Em Revisões do serviço, escolha Comparar revisões e selecione 2 revisões para comparar.

   As revisões do serviço são exibidas lado a lado com as diferenças destacadas.

AWS CLI

1. Execute `list-service-deployments` para recuperar o ARN da implantação do serviço.

   Substitua as variáveis por seus próprios valores.

   ```
   aws ecs list-service-deployments --cluster cluster-name --service service-name
   ```

   Anote o serviceDeploymentArn para a implantação que deseja visualizar.

   ```
   {
       "serviceDeployments": [
           {
   ```

```
            "serviceDeploymentArn": "arn:aws:ecs:us-west-2:123456789012:service-
deployment/example/sd-example/NCWGC2ZR-taawPAYrIaU5",
            "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/example/
sd-example",
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/example",
            "targetServiceRevisionArn": "arn:aws:ecs:us-
west-2:123456789012:service-revision/example/sd-example/4980306466373577095",
            "status": "SUCCESSFUL"
        }
    ]
}
```

2. Executar `describe-service-deployments`. Use o `serviceDeploymentArn` que foi devolvido de `list-service-deployments`.

   Substitua as variáveis por seus próprios valores.

   ```
   aws ecs describe-service-deployments --service-deployment-arns
    arn:aws:ecs:region:123456789012:service-deployment/cluster-name/service-
   name/NCWGC2ZR-taawPAYrIaU5
   ```

Próximas etapas

Você pode visualizar os detalhes das revisões de serviço na implantação. Para obter mais informações, consulte Visualizar detalhes da revisão de serviço do Amazon ECS.

Revisões de serviços do Amazon ECS

Uma revisão do serviço contém um registro da configuração da workload que o Amazon ECS está tentando implantar. Sempre que você cria ou implanta um serviço, o Amazon ECS cria e captura automaticamente a configuração que você está tentando implantar na revisão do serviço.

As revisões do serviço são somente leitura e têm identificadores exclusivos. Para obter mais informações sobre as propriedades incluídas, consulte Propriedades incluídas na revisão de serviço do Amazon ECS.

As revisões de serviço oferecem os seguintes benefícios:

- Durante a implantação de um serviço, você pode comparar a revisão do serviço atualmente implantada (origem) com a que está sendo implantada (destino).

- Quando você usa a opção de reversão para uma implantação de serviço, o Amazon ECS reverte automaticamente a implantação do serviço até a última revisão de serviço implantada com sucesso.
- As revisões de serviço contêm o registro da configuração da workload em um recurso.

Ciclo de vida de uma revisão de serviço

O Amazon ECS cria automaticamente uma nova revisão de serviço quando você cria um serviço ou atualiza uma propriedade do serviço que inicia uma implantação.

O Amazon ECS não cria uma nova revisão de serviço para uma operação de reversão. O Amazon ECS usa a última revisão de serviço bem-sucedida para a reversão.

A revisão de serviço é imutável.

O Amazon ECS exclui a revisão do serviço quando você exclui um serviço.

Você pode visualizar as revisões de serviço criadas em ou após 25 de outubro de 2024 usando o console, a API e a CLI.

Propriedades incluídas na revisão de serviço do Amazon ECS

As propriedades a seguir estão incluídas em uma revisão de serviço.

Recurso	Descrição
Nome de recurso da Amazon (ARN) do serviço	O ARN que identifica o serviço.
ARN do cluster	O ARN do cluster que hospeda o serviço.
ARN da definição de tarefas	O ARN da definição de tarefa usado pelas tarefas do serviço.
Registros de serviços	Detalhes dos registros de serviço usados para a descoberta de serviços.

Recurso	Descrição	
	• O nome do contêiner O valor do nome do contêiner a ser usado para o serviço de descoberta de serviços. • A porta do contêiner O valor da porta a ser usada para o serviço de descobert a de serviços. Esse valor também é especificado na definição da tarefa. • A porta O valor da porta usado se o serviço de descoberta de serviços especificou um registro de SRV. • O ARN do registro O nome de recurso da Amazon (ARN) do registro de serviço.	

Recurso	Descrição
Provedores de capacidade	Os detalhes de uma estratégia de provedor de capacidade. • O nome do provedor de capacidade O nome do provedor de capacidade usado para o serviço. • O peso do provedor de capacidade O percentual relativo do número total de tarefas executadas que devem usar o provedor de capacidade especificado. • A base do provedor de capacidade O número de tarefas, no mínimo, a serem executadas no provedor de capacidade especificado.
Imagens de contêiner	Os detalhes sobre as imagens de contêineres. • O nome do contêiner • O resumo da imagem • A imagem do contêiner

Recurso	Descrição
Redes	A configuração de rede para o serviço. • Se há um endereço IP público • As sub-redes nas quais as tarefas são executadas • Os grupos de segurança que controlam o tráfego do serviço
Tipo de inicialização	O tipo de execução usado para o serviço.

Propriedades específicas do Fargate	Quando o tipo de execução é Fargate, trata-se de informaçõ es sobre a versão do Fargate.
	• A versão da plataforma e a família da plataforma
	• Armazenamento efêmero
	  • A quantidade de armazenamento temporário a ser alocada para a implantação.
	  • A chave do KMS a ser usada para a criptografia do armazenamento.

Recurso	Descrição	
Volumes do Amazon EBS configurados na implantação	A configuração de um volume especificado na definição da tarefa como um volume configurado no momento da execução. • Criptografado Indica se o volume é criptografado. • Tipo do sistema de arquivos O tipo do sistema de arquivos Linux para o volume. • IOPS O número de operações de E/S por segundo (IOPS). • Chave do KMS • Perfil do IAM O nome ou ARN do perfil do IAM associado ao trabalho. • Tamanho O tamanho do volume do em GiB. • ID do snapshot O ID do snapshot do volume. • Especificações de tag A configuração da tag para o volume.	

Recurso	Descrição
	• Throughput O throughput provisionado para o volume. • Tipo de volume O tipo de volume.
Service Connect	A configuração Service Connect. • A indicação de se o Service Connect está em uso • O namespace . • A lista de serviços interconectados • A configuração de log do Service Connect
Balanceadores de carga do serviço	Os balanceadores de carga que roteiam o tráfego do serviço. • O nome • O nome do contêiner • A porta do contêiner • O ARN do grupo de destino
Monitoramento de runtime	Indica se o monitoramento de runtime está ativado.
Data de Criação	A data em que a revisão do serviço foi criada.
VPC Lattice	A configuração do VPC Lattice para a revisão do serviço.

Visualizar detalhes da revisão de serviço do Amazon ECS

É possível visualizar informações sobre os seguintes tipos de revisão de serviço que foram criadas em ou após 25 de outubro de 2024:

- Origem: a configuração da workload atualmente implantada
- Destino: a configuração da workload em implantação

Amazon ECS Console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, escolha o cluster.
3. Na página de detalhes do cluster, na seção Serviços, escolha o serviço.

   A página de detalhes do serviço é exibida.
4. Na página de detalhes da implantação, escolha Implantações.
5. Escolha a revisão de serviço a ser visualizada.

Para visualizar revisões de serviços para esse tipo de implantação	Faça o seguinte	
Implantações em andamento	Em Implantações contínuas , faça o seguinte: • Para visualizar a revisão do serviço de origem, em Revisões de serviço, escolha o ID da revisão do serviço para o Tipo de revisão de origem. • Para visualizar a revisão do serviço de destino, em Revisões de serviço, escolha o ID da revisão	

Para visualizar revisões de serviços para esse tipo de implantação	Faça o seguinte	
	do serviço para o Tipo de revisão de destino.	
Última implantação	Em Última implantação, escolha a Revisão do serviço de destino.	
Implantações concluídas	Em Implantações de serviços, faça o seguinte: • Em Revisão do serviço de destino, escolha o ID.	

AWS CLI

1. Execute `describe-service-deployments` para recuperar o ARN da revisão de serviço.

   Substitua as variáveis por seus próprios valores.

   ```
   aws ecs describe-service-deployments --service-deployment-arns
    arn:aws:ecs:region:account-id:service/cluster-name/service-name/NCWGC2ZR-
   taawPAYrIaU5
   ```

   Anote o `arn` para `sourceServiceRevisions` ou `targetServiceRevisions`.

   ```
   {
       "serviceDeployments": [
           {
               "serviceDeploymentArn": "arn:aws:ecs:us-west-2:123456789012:service-
   deployment/example/sd-example/NCWGC2ZR-taawPAYrIaU5",
               "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/example/
   sd-example",
               "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/example",
               "updatedAt": "2024-09-10T16:49:35.572000+00:00",
               "sourceServiceRevision": {
   ```

```
                "arn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
example/sd-example/4980306466373578954",
                "requestedTaskCount": 0,
                "runningTaskCount": 0,
                "pendingTaskCount": 0
            },
            "targetServiceRevision": {
                "arn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
example/sd-example/4980306466373577095",
                "requestedTaskCount": 0,
                "runningTaskCount": 0,
                "pendingTaskCount": 0
            },
            "status": "IN_PROGRESS",
            "deploymentConfiguration": {
                "deploymentCircuitBreaker": {
                    "enable": false,
                    "rollback": false
                },
                "maximumPercent": 200,
                "minimumHealthyPercent": 100
            }
        }
    ],
    "failures": []
}
```

2. Executar `describe-service-revisions`. Use o `arn` que foi devolvido de `describe-service-deployments`.

   Substitua as variáveis por seus próprios valores.

   ```
   aws ecs describe-service-revisions --service-revision-arns
    arn:aws:ecs:region:123456789012:service-revision/cluster-name/service-
   name/4980306466373577095
   ```

Balancear um serviço do Amazon ECS entre zonas de disponibilidade

Para ajudar suas aplicações a alcançar alta disponibilidade, recomendamos configurar seus serviços multitarefas para serem executados em várias zonas de disponibilidade. Para serviços que especificam que sua primeira estratégia de colocação seja distribuída por zonas de disponibilidade, a

AWS faz o possível para distribuir uniformemente as tarefas de serviço nas zonas de disponibilidade disponíveis.

No entanto, pode haver momentos em que o número de tarefas em execução em uma zona de disponibilidade não é o mesmo que em outras zonas de disponibilidade, como após uma interrupção na zona de disponibilidade. Para resolver esse desequilíbrio de tarefas, é possível habilitar o recurso de rebalanceamento de zonas de disponibilidade.

Com o rebalanceamento de zonas de disponibilidade, o Amazon ECS monitora continuamente a distribuição de tarefas entre as zonas de disponibilidade para cada um dos seus serviços. Quando o Amazon ECS detecta uma distribuição desigual de tarefas, ele atua automaticamente para reequilibrar a workload entre as zonas de disponibilidade. Isso envolve o lançamento de novas tarefas em zonas de disponibilidade com menos tarefas em execução e o encerramento de tarefas em zonas de disponibilidade sobrecarregadas.

Essa redistribuição garante que nenhuma zona de disponibilidade se torne um ponto de falha, ajudando a manter a disponibilidade geral de suas aplicações em contêineres. O processo de rebalanceamento automatizado elimina a necessidade de intervenção manual, acelerando o tempo de recuperação após um evento.

Uma visão geral do processo de rebalanceamento de zonas de disponibilidade é:

1. O Amazon ECS começa a monitorar um serviço depois que ele atinge o estado estável e analisa o número de tarefas em execução em cada zona de disponibilidade.
2. O Amazon ECS executa as seguintes operações ao detectar um desequilíbrio no número de tarefas em execução em cada zona de disponibilidade:
   - Envia um evento de serviço indicando que o rebalanceamento de zonas de sisponibilidade está começando.
   - Inicia tarefas em zonas de disponibilidade com o menor número de tarefas em execução
   - Interrompe as tarefas em zonas de disponibilidade com o maior número de tarefas em execução.
   - O agendador espera que as tarefas recém-iniciadas estejam `HEALTHY` e `RUNNING` antes de interromper as tarefas na zona de disponibilidade superdimensionada.
   - Envia um evento de serviço com o resultado do rebalanceamento de zonas de disponibilidade.

Como o Amazon ECS detecta a distribuição desigual de tarefas

O Amazon ECS determina um desequilíbrio no número de tarefas em execução em cada zona de disponibilidade ao dividir a contagem de tarefas desejada do serviço pelo número de zonas de disponibilidade configuradas. Se a contagem de tarefas desejada não for dividida de forma igual, o Amazon ECS distribuirá o restante das tarefas uniformemente entre as zonas de disponibilidade configuradas. Cada zona de disponibilidade precisa ter pelo menos uma tarefa.

Por exemplo, considere um serviço do Amazon ECS com uma contagem desejada de duas tarefas configuradas para duas zonas de disponibilidade. Nesse cenário, a contagem de tarefas desejada será dividida uniformemente. Uma distribuição balanceada seria uma tarefa por zona de disponibilidade. Se houvesse duas tarefas na zona de disponibilidade 1 e nenhuma tarefa na zona de disponibilidade 2, o Amazon ECS iniciaria o rebalanceamento ao iniciar uma tarefa na zona de disponibilidade 2 antes de interromper uma tarefa na zona de disponibilidade 1.

Agora, considere um serviço do Amazon ECS com uma contagem desejada de três tarefas configuradas para duas zonas de disponibilidade. Nesse cenário, a contagem de tarefas desejada não será dividida uniformemente. Uma distribuição balanceada seria uma tarefa na zona de disponibilidade 1 e duas tarefas na zona de disponibilidade 2, porque cada zona de disponibilidade tem pelo menos uma tarefa e a tarefa restante é colocada na zona de disponibilidade 2.

Considere um serviço do Amazon ECS que tenha uma contagem desejada de cinco tarefas configuradas para três zonas de disponibilidade. Nesse cenário, a contagem de tarefas desejada não será dividida uniformemente. Uma distribuição balanceada seria uma tarefa na zona de disponibilidade 1 e duas tarefas cada nas zonas de disponibilidade 2 e 3. Após contabilizar cada zona de disponibilidade com uma tarefa cada, as duas tarefas restantes são distribuídas uniformemente entre as zonas de disponibilidade.

Considerações para configurar o rebalanceamento de zonas de disponibilidade

Ao configurar o rebalanceamento de zonas de disponibilidade, considere o seguinte:

- O rebalanceamento de zonas de disponibilidade oferece suporte aos tipos de execução do Fargate e do EC2. Para o Fargate, o Amazon ECS redistribuirá automaticamente as tarefas nas zonas de disponibilidade disponíveis para manter o equilíbrio. Para o tipo de execução do EC2, o Amazon ECS rebalanceia as tarefas entre as instâncias de contêineres existentes da melhor forma possível, respeitando suas estratégias e restrições de posicionamento definidas. Porém, o Amazon ECS não pode iniciar novas instâncias em zonas de disponibilidade subutilizadas como parte do processo de rebalanceamento, limitando o rebalanceamento às instâncias de contêiner existentes.

- O rebalanceamento de zonas de disponibilidade funciona nas seguintes configurações:
  - Serviços que usam a estratégia `Replica`
  - Os serviços que especificam a zona de disponibilidade se distribuem como a primeira estratégia de posicionamento de tarefas ou não especificam uma estratégia de posicionamento.
- Não é possível usar o rebalanceamento de zonas de disponibilidade com serviços que atendem a qualquer um dos seguintes critérios:
  - Usam a estratégia `Daemon`
  - Usam o tipo de execução `EXTERNAL` (ECS Anywhere)
  - Usam 100% como valor de `maximumPercent`
  - Usam um Classic Load Balancer
  - Usa `attribute:ecs.availability-zone` como uma restrição de posicionamento de tarefas

Estratégias e restrições de posicionamento com rebalanceamento de zonas de disponibilidade

As estratégias de posicionamento determinam como o Amazon ECS seleciona instâncias de contêineres e zonas de disponibilidade para o término da colocação de tarefas. As restrições de posicionamento de tarefas são regras que determinam se uma tarefa pode ser executada em uma instância de contêiner específica.

Para o tipo de execução do EC2, é possível usar estratégias e restrições de posicionamento em conjunto com o rebalanceamento de zonas de disponibilidade. No entanto, para que o rebalanceamento de zonas de disponibilidade funcione, a estratégia de posicionamento de distribuição em zonas de disponibilidade deve ser a primeira estratégia especificada.

O rebalanceamento de zonas de disponibilidade é compatível com várias combinações de estratégias de posicionamento. Por exemplo, você pode criar uma estratégia que primeiro distribua as tarefas uniformemente entre as zonas de disponibilidade e, em seguida, agrupe as tarefas com base na memória dentro de cada zona de disponibilidade. Nesse caso, o rebalanceamento de zonas de disponibilidade funciona porque a estratégia de distribuição em zonas de disponibilidade é especificada primeiro.

É importante observar que o rebalanceamento de zonas de disponibilidade não funcionará se a primeira estratégia na matriz da estratégia de posicionamento não for um componente de distribuição

de zona de disponibilidade. Esse requisito garante que o foco principal da distribuição de tarefas seja manter o equilíbrio entre as zonas de disponibilidade, o que é crucial para a alta disponibilidade.

Para obter mais informações sobre estratégias e restrições de posicionamento de tarefas, consulte Como o Amazon ECS posiciona tarefas em instâncias de contêineres.

Não há suporte para estratégias e restrições de posicionamento de tarefas para as tarefas que usam o tipo de inicialização do Fargate. O Fargate fará o possível para distribuir tarefas em zonas de disponibilidade acessíveis. Se o provedor de capacidade incluir o Fargate e o Fargate Spot, o comportamento da distribuição será independente para cada provedor de capacidade.

A estratégia a seguir distribui tarefas uniformemente em zonas de disponibilidade e, em seguida, agrupa as tarefas com base na memória em cada zona de disponibilidade. O rebalanceamento de zonas de disponibilidade é compatível com o serviço porque a estratégia `spread` é a primeira.

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    },
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Ative o rebalanceamento de zonas de disponibilidade

É necessário habilitar o rebalanceamento de zonas de disponibilidade para serviços novos e existentes.

É possível habilitar e desabilitar rebalanceamento de zonas de disponibilidade usando o console, as APIs ou a AWS CLI.

Tipo de serviço	API	Console	CLI
Existentes	UpdateService	Atualizar um serviço do Amazon ECS	update-service
Novo	CreateService	Criação de uma implantação de	create-service

Tipo de serviço	API	Console	CLI
		atualização contínua do Amazon ECS	

O seguinte exemplo mostra como habilitar o rebalanceamento de serviços ao criar um novo serviço:

```
aws ecs create-service \
    --cluster my-cluster \
    --service-name my-service \
    --task-definition my-task-definition:1 \
    --desired-count 6 \
    --availability-zone-rebalancing ENABLED
```

Monitorar o rebalanceamento de zonas de disponibilidade do Amazon ECS

Você pode verificar se o rebalanceamento de zonas de disponibilidade está habilitado para um serviço no console ou então chamar `describe-services`. O exemplo a seguir pode ser usado para verificar o status com a CLI.

A resposta será `ENABLED` ou `DISABLED`.

```
aws ecs describe-services \
    --services service-name \
    --cluster cluster-name \
    --query services[0].availabilityZoneRebalancing
```

Eventos de serviço

O Amazon ECS envia eventos de ação de serviço para ajudar você a entender o ciclo de vida do reequilíbrio de zonas de disponibilidade.

Event	Cenário	Tipo	Saiba mais
`SERVICE_REBALANCING_STARTED`	O Amazon ECS inicia uma operação de rebalanceamento de zonas de disponibilidade	INFO	O serviço (*service-name*) não está balanceado em AZ com *number-tasks* tarefas

Event	Cenário	Tipo	Saiba mais
			na *Zona de disponibilidade 1*, *number-tasks* na *Zona de disponibilidade 2* e *number-tasks* na *Zona de disponibilidade 3*. Rebalanceamento de AZ em andamento.
SERVICE_REBALANCING_COMPLETED	A operação de rebalanceamento de zonas de disponibilidade é concluída	INFO	O serviço (*service-name*) está balanceado em AZ com *number-tasks* tarefas na *Zona de disponibilidade 1*, *number-tasks* na *Zona de disponibilidade 2* e *number-tasks* na *Zona de disponibilidade 3*.
TASKS_STARTED	O Amazon ECS inicia com êxito tarefas como parte da operação de rebalanceamento das zonas de disponibilidade	INFO	O *service-name* iniciou *number-tasks* tarefas na *Zona de disponibilidade* para rebalancear AZ: *task-ids*.

Event	Cenário	Tipo	Saiba mais
TASKS_STOPPED	O Amazon ECS interrompeu com êxito tarefas como parte da operação de rebalanceamento de zonas de disponibilidade	INFO	O *service-name* interrompeu *number-tasks* tarefas na *Zona de disponibilidade* devido ao rebalanceamento de AZ: *task-id*.
SERVICE_TASK_PLACEMENT_FAILURE	O Amazon ECS falhou ao iniciar uma tarefa como parte da operação de rebalanceamento de zonas de disponibilidade	ERRO	Para o tipo de execução do EC2, consulte O serviço (*service-name*) não conseguiu colocar uma tarefa na *Zona de disponibilidade* porque nenhuma instância de contêiner atendeu a todos os requisitos. Para o tipo de execução do Fargate, consulte O serviço (*service-name*) não conseguiu colocar uma tarefa na *Zona de disponibilidade*.

Event	Cenário	Tipo	Saiba mais
`TASKSET_SCALE_IN_FAILURE_BY_TASK_PROTECTION`	A operação de rebalanceamento de zonas de disponibilidade está bloqueada porque a proteção de tarefas está em uso.	INFO	O serviço (*`service-name`*) não conseguiu fazer o rebalanceamento de AZ porque não foi possível escalar *`task-set-name`* devido a *`reason`*.
`SERVICE_REBALANCING_STOPPED`	A operação de rebalanceamento de zonas de disponibilidade foi interrompida. O Amazon ECS envia eventos adicionais que fornecem mais informações.	INFO	O serviço (*`service-name`*) interrompeu o rebalanceamento de AZ.

Eventos de alteração no estado da tarefa

O Amazon ECS envia um evento de alteração de estado da tarefa (`START`) para cada tarefa que inicia como parte do processo de rebalanceamento.

O Amazon ECS envia um evento de alteração de estado da tarefa (`STOPPED`) para cada tarefa que interrompe como parte do processo de rebalanceamento. O motivo é definido como `Availability-zone rebalancing initiated by (deployment ecs-svc/`*`deployment-id`*`)`.

Para obter mais informações sobre eventos, consulte Eventos de alteração de estado de tarefa do Amazon ECS.

Solução de problemas de rebalanceamento de serviços

Se você encontrar problemas com o rebalanceamento do serviço, considere as seguintes etapas de solução de problemas:

O rebalanceamento não inicia

Verificar se:

- O rebalanceamento de serviço está habilitado para seu serviço
- Seu serviço usa uma configuração compatível (consulte the section called “Considerações para configurar o rebalanceamento de zonas de disponibilidade”)
- Seu serviço atingiu um estado estável

Falhas no posicionamento de tarefas durante o rebalanceamento

Se você vir eventos `SERVICE_TASK_PLACEMENT_FAILURE`:

- Em tipo de inicialização do EC2: verifique se você tem instâncias de contêineres disponíveis na zona de disponibilidade de destino
- Em tipo de inicialização do Fargate: verifique se há restrições de recursos ou cotas de serviço limitando o posicionamento de tarefas
- Revise suas restrições de posicionamento de tarefas para garantir que elas não estejam impedindo a distribuição adequada de tarefas

O rebalanceamento é interrompido inesperadamente

Se você vir eventos `SERVICE_REBALANCING_STOPPED`:

- Verifique a proteção de tarefa que pode estar bloqueando a operação
- Procure implantações simultâneas de serviços que possam interromper o rebalanceamento
- Analise os eventos do serviço para obter informações adicionais sobre por que o rebalanceamento foi interrompido

Práticas recomendadas para rebalanceamento de serviços

Siga estas práticas recomendadas para aproveitar ao máximo o rebalanceamento de serviços:

- Monitore as operações de rebalanceamento: configure os alarmes do CloudWatch para monitorar eventos de serviço relacionados ao rebalanceamento e identificar rapidamente quaisquer problemas.
- Considere o impacto na performance: esteja ciente de que as operações de rebalanceamento podem aumentar temporariamente o uso de recursos à medida que novas tarefas são iniciadas antes que as antigas sejam interrompidas.
- Use a proteção de tarefas estrategicamente: se você tiver tarefas críticas que não devem ser encerradas durante o rebalanceamento, considere usar a proteção de tarefas.

- Planeje a capacidade do EC2: para o tipo de inicialização do EC2, garanta que você tenha instâncias de contêineres suficientes em todas as zonas de disponibilidade para suportar um rebalanceamento eficaz.
- Teste o comportamento de rebalanceamento: antes de confiar no rebalanceamento na produção, teste como seus serviços se comportam durante as operações de rebalanceamento em um ambiente que não seja de produção.

Uso do balanceamento de carga para distribuir o tráfego de serviço do Amazon ECS

O serviço Amazon ECS pode ser configurado opcionalmente para usar o Elastic Load Balancing para distribuir tráfego entre as tarefas do serviço igualmente.

> **Note**
>
> Quando você usa conjuntos de tarefas, todas as tarefas do conjunto devem ser configuradas para usar o Elastic Load Balancing ou para não usar o Elastic Load Balancing.

Os serviços do Amazon ECS hospedados no AWS Fargate oferecem suporte a Application Load Balancers, Network Load Balancers e Gateway Load Balancers. Use a tabela apresentada a seguir para descobrir qual tipo de balanceador de carga usar.

Tipo de balanceador de carga	Uso para estes casos	
Application Load Balancer	Roteamento do tráfego HTTP/HTTPS (ou da camada 7). Os application load balancers oferecem vários recursos que os tornam atrativos para uso com serviços do Amazon ECS: • Cada serviço pode atender a tráfego de vários load balancers e expor várias	

Tipo de balanceador de carga	Uso para estes casos	
	portas de balanceamento de carga especificando vários grupos de destino. • Eles são compatíveis com tarefas hospedadas em instâncias do Fargate e do EC2. • Os application load balancers permitem que os contêineres usem mapeamento de porta host dinâmico (de modo que várias tarefas do mesmo serviço sejam permitidas por instância de contêiner). • Os Application Load Balancers oferecem suporte a regras de roteamento e prioridade com base no caminho (de modo que vários serviços possam usar a mesma porta de listener em um único Application Load Balancer).	
Network Load Balancer	Roteamento do tráfego TCP ou UDP (ou da camada 4).	

Tipo de balanceador de carga	Uso para estes casos
Gateway Load Balancer	Roteamento do tráfego TCP ou UDP (ou da camada 4). Use dispositivos virtuais, como firewalls, sistemas de detecção e prevenção de intrusão e sistemas de inspeção profunda de pacotes.

Recomendamos usar Application Load Balancers nos serviços do Amazon ECS para aproveitar os recursos mais recentes, a menos que seu serviço exija um recurso que esteja disponível apenas com Network Load Balancers ou Gateway Load Balancers. Para obter mais informações sobre Elastic Load Balancing e as diferenças entre esses tipos de balanceadores de carga, consulte o Guia do usuário do Elastic Load Balancing.

Com o load balancer, você paga somente pelo que utilizar. Para obter mais informações, consulte Preço do Elastic Load Balancing.

Otimização dos parâmetros de verificação de integridade do balanceador de carga para o Amazon ECS

Os balanceadores de carga encaminham solicitações apenas para os destinos íntegros nas zonas de disponibilidade do balanceador de carga. Cada destino é registrado em um grupo de destino. O balanceador de carga verifica a integridade de cada destino usando as configurações de verificação de integridade do grupo de destino. Após você registrar o destino, ele deverá ser aprovado por uma verificação de integridade para ser considerado íntegro. O Amazon ECS realiza o monitoramento do balanceador de carga. O balanceador de carga envia periodicamente verificações de integridade para o contêiner do Amazon ECS. O agente do Amazon ECS monitora e aguarda que o balanceador de carga informe sobre a integridade do contêiner. Ele faz isso antes de considerar que o contêiner está em um estado íntegro.

Dois parâmetros de verificação de integridade do Elastic Load Balancing afetam a velocidade de implantação:

- Intervalo da verificação de integridade: determina o tempo aproximado, em segundos, entre verificações de integridade de um contêiner individual. Por padrão, o balanceador de carga verifica a cada 30 segundos.

  Esse parâmetro é denominado:

  - `HealthCheckIntervalSeconds` na API do Elastic Load Balancing
  - Intervalo no console do Amazon EC2
- Contagem de limites íntegros: determina o número de verificações de integridade consecutivas bem-sucedidas necessárias para que um contêiner não íntegro seja considerado íntegro. Por padrão, o balanceador de carga exige cinco verificações de integridade aprovadas antes de informar que o contêiner de destino está íntegro.

  Esse parâmetro é denominado:

  - `HealthyThresholdCount` na API do Elastic Load Balancing
  - Limite íntegro no console do Amazon EC2

Importante: para destinos recém-registrados, é necessária apenas uma verificação de integridade bem-sucedida para considerar o destino íntegro, independentemente da configuração da contagem de limiar de integridade. A contagem de limiar de integridade só se aplica quando um destino estiver fazendo a transição de um estado não íntegro para um estado íntegro.

Com as configurações padrão, se um destino perder a integridade e então se recuperar, o tempo total para determinar a integridade de um contêiner será de 2 minutos e 30 segundos (`30 seconds * 5 = 150 seconds`).

Você pode acelerar o processo de verificação de integridade se o serviço for inicializado e estabilizado em menos de 10 segundos. Para acelerar o processo, reduza o intervalo de verificação de integridade e a contagem de limite de integridade.

- `HealthCheckIntervalSeconds` (nome da API do Elastic Load Balancing) ou Intervalo (nome do console do Amazon EC2): 5
- `HealthyThresholdCount` (nome da API do Elastic Load Balancing) ou Limite íntegro (nome do console do Amazon EC2): 2

Com essa configuração, o processo de verificação de integridade leva 10 segundos quando comparado ao padrão de 2 minutos e 30 segundos.

Para obter mais informações sobre os parâmetros de verificação de integridade do Elastic Load Balancing, consulte Health checks for your target groups no Guia do usuário do Elastic Load Balancing.

Otimização de parâmetros de drenagem da conexão do balanceador de carga para o Amazon ECS

Para permitir a otimização, os clientes mantêm uma conexão keep alive com o serviço de contêiner. Isto permite que solicitações subsequentes desse cliente reutilizem a conexão existente. Quando quiser interromper o tráfego de um contêiner, você notifica o balanceador de carga. O balanceador de carga verifica periodicamente se o cliente fechou a conexão keep-alive. O agente do Amazon ECS monitora o balanceador de carga e aguarda que ele informe que a conexão keep alive está fechada (o destino está em um estado `UNUSED`).

A quantidade de tempo que o balanceador de carga aguarda para mover o destino para o estado `UNUSED` equivale ao atraso no cancelamento do registro. Você pode configurar o parâmetro do balanceador de carga a seguir para acelerar as implantações.

- `deregistration_delay.timeout_seconds`: 300 (padrão)

Quando houver um serviço com um tempo de resposta inferior a 1 segundo, defina o parâmetro com o seguinte valor para que o balanceador de carga espere apenas cinco segundos antes de interromper a conexão entre o cliente e o serviço de backend:

- `deregistration_delay.timeout_seconds`: 5

> **Note**
>
> Não defina o valor como cinco segundos quando tiver um serviço com solicitações de longa duração, como uploads lentos de arquivos ou conexões de fluxo.

Capacidade de resposta do SIGTERM

O Amazon ECS primeiro envia um sinal de SIGTERM à tarefa para notificar que a aplicação precisa ser concluída e desligada. Em seguida, o Amazon ECS envia uma mensagem de SIGKILL. Quando as aplicações ignoram o SIGTERM, o serviço do Amazon ECS deve esperar para enviar o sinal de SIGKILL antes de encerrar o processo.

O tempo que o Amazon ECS espera para enviar a mensagem de SIGKILL é determinado pela seguinte opção de agente do Amazon ECS:

- `ECS_CONTAINER_STOP_TIMEOUT`: 30 (padrão)

  Para obter mais informações sobre o parâmetro do agente de contêiner, consulte Amazon ECS Container Agent no GitHub.

Para acelerar o período de espera, defina o parâmetro do agente do Amazon ECS com o seguinte valor:

- `ECS_CONTAINER_STOP_TIMEOUT`: 2

  Se a aplicação levar mais de um segundo, multiplique o valor por dois e use esse número como valor.

Nesse caso, o Amazon ECS aguarda dois segundos para que o contêiner seja desligado e, em seguida, envia uma mensagem de SIGKILL quando a aplicação não é interrompida.

Você também pode modificar o código da aplicação para capturar o sinal de SIGTERM e reagir a ele. O seguinte exemplo está em JavaScript:

```
process.on('SIGTERM', function() {
  server.close();
})
```

Esse código faz com que o servidor HTTP pare de receber novas solicitações, termine de responder a todas as solicitações em andamento e, em seguida, encerra o processo Node.js porque o loop de eventos não tem nada para fazer. Diante disso, se o processo levar apenas 500 ms para concluir as solicitações em andamento, ele é encerrado mais cedo, sem precisar esperar o tempo limite de encerramento e receber um SIGKILL.

Uso de um Application Load Balancer do Amazon ECS

Um Application Load Balancer toma decisões de roteamento na camada da aplicação (HTTP/HTTPS), oferece suporte ao roteamento com base em caminho e pode encaminhar solicitações para uma ou mais portas em cada instância de contêiner no cluster. Os application load balancers oferecem suporte ao mapeamento de porta de host dinâmico. Por exemplo, se a definição de contêiner da tarefa especifica a porta 80 para uma porta de contêiner NGINX e a porta 0 para a porta

do host, a porta do host é escolhida dinamicamente com base no intervalo de portas temporário da instância de contêiner (como entre 32768 e 61000 na AMI otimizado para Amazon ECS mais recente). Quando a tarefa é iniciada, o contêiner NGINX é registrado no Application Load Balancer como uma combinação entre o ID da instância e a porta, e o tráfego é distribuído para o ID da instância e para a porta correspondente a esse contêiner. Esse mapeamento dinâmico permite várias tarefas de um único serviço na mesma instância de contêiner. Para obter mais informações, consulte o Guia do usuário para application load balancers.

Para obter informações sobre as práticas recomendadas para a definição de parâmetros a fim de acelerar as implantações, consulte:

- Otimização dos parâmetros de verificação de integridade do balanceador de carga para o Amazon ECS
- Otimização de parâmetros de drenagem da conexão do balanceador de carga para o Amazon ECS

Considere o seguinte ao usar Application Load Balancers com o Amazon ECS:

- O Amazon ECS exige a função do IAM vinculada ao serviço, que fornece as permissões necessárias para registrar e cancelar o registro de destinos com o balanceador de carga quando as tarefas são criadas e interrompidas. Para obter mais informações, consulte Uso de perfis vinculados ao serviço para o Amazon ECS.
- O grupo de destino deve ter o tipo de endereço IP definido como IPv4.

  Embora as tarefas ofereçam suporte a IPv6, a integração de serviços com o Elastic Load Balancing IPv6 não é possível.
- Para serviços com tarefas que usam o modo de rede `awsvpc`, ao criar um grupo de destino para o serviço, é necessário escolher `ip` como o tipo de destino, e não `instance`. Isso ocorre porque as tarefas que usam o modo de rede `awsvpc` estão associadas a uma interface de rede elástica, e não a uma instância do Amazon EC2.
- Se o serviço requerer acesso a diversas portas com balanceamento de carga, como a porta 80 e a porta 443 para um serviço HTTP/HTTPS, será possível configurar dois receptores. Um listener é responsável pelo HTTPS que encaminha a solicitação para o serviço e outro listener que é responsável por redirecionar solicitações HTTP para a porta HTTPS apropriada. Para obter mais informações, consulte Criar um listener no Application Load Balancer no Guia do usuário dos Application Load Balancers.

- A configuração de sub-rede do load balancer deve incluir todas as zonas de disponibilidade nas quais as instâncias de contêiner residem.
- Após você criar um serviço, a configuração do balanceador de carga não poderá ser alterada no AWS Management Console. É possível usar o AWS Copilot, o AWS CloudFormation, a AWS CLI ou o SDK para modificar a configuração do balanceador de carga somente para o controlador de implantação rolling do `ECS`, e não o AWS CodeDeploy azul/verde ou externo. Quando você adiciona, atualiza ou remove uma configuração de balanceador de carga, o Amazon ECS inicia uma nova implantação com a configuração atualizada do Elastic Load Balancing. Isso faz com que as tarefas se registrem e cancelem o registro dos balanceadores de carga. Recomendamos verificar isso em um ambiente de teste antes de atualizar a configuração do Elastic Load Balancing. Para obter informações sobre como modificar a configuração, consulte UpdateService na Referência da API do Amazon Elastic Container Service.
- Caso a tarefa de um serviço não passe nos critérios de verificação de integridade do balanceador de carga, a tarefa é interrompida e reiniciada. Esse processo continuará até que o serviço alcance o número de tarefas em execução desejadas.
- Caso você esteja enfrentando problemas com os serviços habilitados pelo balanceador de carga, consulte Solução de problemas relacionados aos balanceadores de carga de serviço no Amazon ECS.
- Ao usar o tipo de destino `instance`, as tarefas e o balanceador de carga devem estar na mesma VPC. Ao usar o tipo de destino `ip`, há suporte para conectividade entre VPCs.
- Use um grupo de destino exclusivo para cada serviço.

  Usar o mesmo grupo de destino para vários serviços pode causar problemas durante a implantação do serviço.

Para obter informações sobre como criar um Application Load Balancer, consulte Create an Application Load Balancer em Application Load Balancers

Uso de um Network Load Balancer no Amazon ECS

Um Network Load Balancer toma decisões de roteamento na camada de transporte (TCP/SSL). Ele pode processar milhões de solicitações por segundo. Após o load balancer receber uma conexão, ele seleciona um destino do grupo de destino para a regra padrão usando um algoritmo de roteamento de hash de fluxo. Ele tenta abrir uma conexão TCP para o destino selecionado na porta especificada na configuração do listener. Ele encaminha a solicitação sem modificar os cabeçalhos. Os network load balancers oferecem suporte ao mapeamento de porta de host

dinâmico. Por exemplo, se a definição de contêiner da tarefa especifica a porta 80 para uma porta de contêiner NGINX e a porta 0 para a porta do host, a porta do host é escolhida dinamicamente com base no intervalo de portas temporário da instância de contêiner (como entre 32768 e 61000 na AMI otimizado para Amazon ECS mais recente). Quando a tarefa é inicializada, o contêiner NGINX é registrado no Network Load Balancer como uma combinação de ID e porta da instância, e o tráfego é distribuído para o ID e para a porta da instância correspondentes a este contêiner. Esse mapeamento dinâmico permite várias tarefas de um único serviço na mesma instância de contêiner. Para obter mais informações, consulte o Guia do usuário de network load balancers.

Para obter informações sobre as práticas recomendadas para a definição de parâmetros a fim de acelerar as implantações, consulte:

- Otimização dos parâmetros de verificação de integridade do balanceador de carga para o Amazon ECS
- Otimização de parâmetros de drenagem da conexão do balanceador de carga para o Amazon ECS

Considere o seguinte ao usar Network Load Balancers com o Amazon ECS:

- O Amazon ECS exige a função do IAM vinculada ao serviço, que fornece as permissões necessárias para registrar e cancelar o registro de destinos com o balanceador de carga quando as tarefas são criadas e interrompidas. Para obter mais informações, consulte Uso de perfis vinculados ao serviço para o Amazon ECS.
- Não é possível anexar mais de cinco grupos de destino a um serviço.
- Para serviços com tarefas que usam o modo de rede `awsvpc`, ao criar um grupo de destino para o serviço, é necessário escolher `ip` como o tipo de destino, e não `instance`. Isso ocorre porque as tarefas que usam o modo de rede `awsvpc` estão associadas a uma interface de rede elástica, e não a uma instância do Amazon EC2.
- A configuração de sub-rede do load balancer deve incluir todas as zonas de disponibilidade nas quais as instâncias de contêiner residem.
- Após você criar um serviço, a configuração do balanceador de carga não poderá ser alterada no AWS Management Console. É possível usar o AWS Copilot, o AWS CloudFormation, a AWS CLI ou o SDK para modificar a configuração do balanceador de carga somente para o controlador de implantação rolling do `ECS`, e não o AWS CodeDeploy azul/verde ou externo. Quando você adiciona, atualiza ou remove uma configuração de balanceador de carga, o Amazon ECS inicia uma nova implantação com a configuração atualizada do Elastic Load Balancing.

Isso faz com que as tarefas se registrem e cancelem o registro dos balanceadores de carga. Recomendamos verificar isso em um ambiente de teste antes de atualizar a configuração do Elastic Load Balancing. Para obter informações sobre como modificar a configuração, consulte UpdateService na Referência da API do Amazon Elastic Container Service.

- Caso a tarefa de um serviço não passe nos critérios de verificação de integridade do balanceador de carga, a tarefa é interrompida e reiniciada. Esse processo continuará até que o serviço alcance o número de tarefas em execução desejadas.
- Ao usar um Gateway Load Balancer configurado com endereços IP como destinos e a Preservação do IP do Cliente desativada, as solicitações parecem se originar do endereço IP privado do Gateway Load Balancers. Isso significa que, ao permitir solicitações de entrada e verificações de integridade no grupo de segurança de destino, os serviços protegidos por um Gateway Load Balancer ficam efetivamente expostos.
- Para tarefas do Fargate, você deve usar a versão da plataforma `1.4.0` (Linux) ou `1.0.0` (Windows).
- Caso você esteja enfrentando problemas com os serviços habilitados pelo balanceador de carga, consulte Solução de problemas relacionados aos balanceadores de carga de serviço no Amazon ECS.
- Ao usar o tipo de destino `instance`, as tarefas e o balanceador de carga devem estar na mesma VPC. Ao usar o tipo de destino `ip`, há suporte para conectividade entre VPCs.
- A preservação do endereço IP do cliente do Network Load Balancer é compatível com os destinos do Fargate.
- Use um grupo de destino exclusivo para cada serviço.

  Usar o mesmo grupo de destino para vários serviços pode causar problemas durante a implantação do serviço.

Para obter informações sobre como criar um Network Load Balancer, consulte Create a Network Load Balancer em Network Load Balancers

> **Important**
>
> Se a definição de tarefa do serviço usar o modo de rede `awsvpc` (exigido para o tipo de inicialização do Fargate), será preciso escolher `ip` como tipo de destino, e não `instance`. Isso ocorre porque as tarefas que usam o modo de rede `awsvpc` estão associadas a uma interface de rede elástica, e não a uma instância do Amazon EC2.

Você não pode registrar instâncias por ID de instância se eles tiverem os seguintes tipos de instância: C1, CC1, CC2, CG1, CG2, CR1, G1, G2, HI1, HS1, M1, M2, M3 e T1. É possível registrar instâncias desses tipos por endereço IP.

Uso de um Gateway Load Balancer para o Amazon ECS

Um balanceador de carga de gateway opera na terceira camada do modelo Open Systems Interconnection (OSI), a camada de rede. Ele escuta todos os pacotes IP em todas as portas e encaminha o tráfego para o grupo de destino especificado na regra do listener. Ele mantém a perdurabilidade dos fluxos para um dispositivo de destino específico usando 5 tuplas (para fluxos TCP/UDP) ou 3 tuplas (para fluxos não TCP/UDP). Por exemplo, se a definição de contêiner da tarefa especifica a porta 80 para uma porta de contêiner NGINX e a porta 0 para a porta do host, a porta do host é escolhida dinamicamente com base no intervalo de portas temporário da instância de contêiner (como entre 32768 e 61000 na AMI otimizado para Amazon ECS mais recente). Quando a tarefa é iniciada, o contêiner NGINX é registrado no Gateway Load Balancer como uma combinação entre o ID da instância e a porta, e o tráfego é distribuído para o ID da instância e para a porta correspondente a esse contêiner. Esse mapeamento dinâmico permite várias tarefas de um único serviço na mesma instância de contêiner. Para obter mais informações, consulte What is a Gateway Load Balancer? em Gateway Load Balancers.

Para obter informações sobre as práticas recomendadas para a definição de parâmetros a fim de acelerar as implantações, consulte:

- Otimização dos parâmetros de verificação de integridade do balanceador de carga para o Amazon ECS
- Otimização de parâmetros de drenagem da conexão do balanceador de carga para o Amazon ECS

Considere o seguinte ao usar Gateway Load Balancers com o Amazon ECS:

- O Amazon ECS exige a função do IAM vinculada ao serviço, que fornece as permissões necessárias para registrar e cancelar o registro de destinos com o balanceador de carga quando as tarefas são criadas e interrompidas. Para obter mais informações, consulte Uso de perfis vinculados ao serviço para o Amazon ECS.
- Para serviços com tarefas que usam o modo de rede `awsvpc`, ao criar um grupo de destino para o serviço, é necessário escolher `ip` como o tipo de destino, e não `instance`. Isso ocorre porque

as tarefas que usam o modo de rede `awsvpc` estão associadas a uma interface de rede elástica, e não a uma instância do Amazon EC2.

- Balanceadores de carga de gateway não são compatíveis com serviços com tarefas que não usam `awsvpc`, mas outro modo de rede.
- A configuração de sub-rede do load balancer deve incluir todas as zonas de disponibilidade nas quais as instâncias de contêiner residem.
- Após você criar um serviço, a configuração do balanceador de carga não poderá ser alterada no AWS Management Console. É possível usar o AWS Copilot, o AWS CloudFormation, a AWS CLI ou o SDK para modificar a configuração do balanceador de carga somente para o controlador de implantação rolling do `ECS`, e não o AWS CodeDeploy azul/verde ou externo. Quando você adiciona, atualiza ou remove uma configuração de balanceador de carga, o Amazon ECS inicia uma nova implantação com a configuração atualizada do Elastic Load Balancing. Isso faz com que as tarefas se registrem e cancelem o registro dos balanceadores de carga. Recomendamos verificar isso em um ambiente de teste antes de atualizar a configuração do Elastic Load Balancing. Para obter informações sobre como modificar a configuração, consulte UpdateService na Referência da API do Amazon Elastic Container Service.
- Caso a tarefa de um serviço não passe nos critérios de verificação de integridade do balanceador de carga, a tarefa é interrompida e reiniciada. Esse processo continuará até que o serviço alcance o número de tarefas em execução desejadas.
- Ao usar um Gateway Load Balancer configurado com endereços IP como destinos, as solicitações parecem se originar do endereço IP privado do Gateway Load Balancers. Isso significa que, ao permitir solicitações de entrada e verificações de integridade no grupo de segurança de destino, os serviços protegidos por um Gateway Load Balancer ficam efetivamente expostos.
- Para tarefas do Fargate, você deve usar a versão da plataforma `1.4.0` (Linux) ou `1.0.0` (Windows).
- Caso você esteja enfrentando problemas com os serviços habilitados pelo balanceador de carga, consulte Solução de problemas relacionados aos balanceadores de carga de serviço no Amazon ECS.
- Ao usar o tipo de destino `instance`, as tarefas e o balanceador de carga devem estar na mesma VPC. Ao usar o tipo de destino `ip`, há suporte para conectividade entre VPCs.
- Use um grupo de destino exclusivo para cada serviço.

  Usar o mesmo grupo de destino para vários serviços pode causar problemas durante a implantação do serviço.

Para obter informações sobre como criar um Gateway Load Balancer, consulte Introdução a Gateway Load Balancers em Gateway Load Balancers

> **Important**
>
> Se a definição de tarefa do serviço usar o modo de rede `awsvpc` (exigido para o tipo de inicialização do Fargate), será preciso escolher `ip` como tipo de destino, e não `instance`. Isso ocorre porque as tarefas que usam o modo de rede `awsvpc` estão associadas a uma interface de rede elástica, e não a uma instância do Amazon EC2.
> Você não pode registrar instâncias por ID de instância se eles tiverem os seguintes tipos de instância: C1, CC1, CC2, CG1, CG2, CR1, G1, G2, HI1, HS1, M1, M2, M3 e T1. É possível registrar instâncias desses tipos por endereço IP.

Registrar vários grupos de destino em um serviço Amazon ECS

O serviço do Amazon ECS pode atender ao tráfego de vários balanceadores de carga e expor várias portas com balanceamento de carga quando você especificar vários grupos de destino em uma definição de serviço.

Para criar um serviço que especifique vários grupos de destino, é necessário criar o serviço usando a API do Amazon ECS, o SDK, a AWS CLI ou um modelo do AWS CloudFormation. Depois que o serviço é criado, você pode visualizar o serviço e os grupos de destino registrados nele com o AWS Management Console. Você deve usar `UpdateService` para modificar a configuração do balanceador de carga de um serviço existente.

Vários grupos de destino podem ser especificados em uma definição de serviço usando o seguinte formato. Para obter a sintaxe completa de uma definição de serviço, consulte Modelo de definição de serviço.

```
"loadBalancers":[
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"container_name",
      "containerPort":container_port
   },
   {
```

```
 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"container_name",
      "containerPort":container_port
   }
]
```

Considerações

Quando você especificar vários grupos de destino em uma definição de serviço, considere o seguinte.

- Para serviços que usam um Application Load Balancer ou um Network Load Balancer, não é possível anexar mais de cinco grupos de destino a um serviço.
- Especificar vários grupos de destino em uma definição de serviço só é compatível com as seguintes condições:
  - O serviço deve usar um Application Load Balancer ou um Network Load Balancer.
  - O serviço deve usar o tipo de controlador de implantação de atualização contínua (ECS).
- Especificar vários grupos de destino é compatível com serviços que contêm tarefas que usam os tipos de inicialização do Fargate e do EC2.
- Quando você criar um serviço que especifica vários grupos de destino, deverá ser criada a função vinculada ao serviço do Amazon ECS. A função é criada omitindo o parâmetro `role` nas solicitações de API ou a propriedade `Role` no AWS CloudFormation. Para obter mais informações, consulte Uso de perfis vinculados ao serviço para o Amazon ECS.

Exemplos de definições de serviço

Veja a seguir alguns exemplos de casos de uso para especificar vários grupos de destino em uma definição de serviço. Para obter a sintaxe completa de uma definição de serviço, consulte Modelo de definição de serviço.

Ter balanceadores de carga separados para tráfego interno e externo

No seguinte caso de uso, um serviço usa dois load balancers, um para tráfego interno e um segundo para tráfego voltado para a Internet, para o mesmo contêiner e porta.

```
"loadBalancers":[
```

```
   //Internal ELB
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"nginx",
      "containerPort":8080
   },
   //Internet-facing ELB
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"nginx",
      "containerPort":8080
   }
]
```

Expor várias portas do mesmo contêiner

No seguinte caso de uso, um serviço usa um load balancer, mas expõe várias portas do mesmo contêiner. Por exemplo, um contêiner Jenkins pode expor a porta 8080 para a interface da Web do Jenkins e a porta 50000 para a API.

```
"loadBalancers":[
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"jenkins",
      "containerPort":8080
   },
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"jenkins",
      "containerPort":50000
   }
]
```

Expor portas de vários contêineres

No seguinte caso de uso, um serviço usa um load balancer e dois grupos de destino para expor portas de contêineres separados.

```
"loadBalancers":[
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"webserver",
      "containerPort":80
   },
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"database",
      "containerPort":3306
   }
]
```

Como escalar automaticamente o serviço do Amazon ECS

Escalabilidade automática é a capacidade de aumentar ou diminuir automaticamente o número de tarefas desejadas no serviço do Amazon ECS. O Amazon ECS utiliza o serviço do Application Auto Scaling para fornecer essa funcionalidade. Para obter mais informações, consulte o Guia do usuário do Application Auto Scaling.

O Amazon ECS publica métricas do CloudWatch com o uso médio, pelo serviço, da CPU e da memória. Para obter mais informações, consulte Métricas de utilização do serviço do Amazon ECS. É possível usar essas e outras métricas do CloudWatch para expandir o serviço (adicionar mais tarefas) para lidar com alta demanda em horários de pico e para reduzir o serviço (executar menos tarefas) de modo a reduzir os custos durante períodos de baixa utilização.

O Auto Scaling do serviço do Amazon ECS oferece suporte aos seguintes tipos de escalabilidade automática:

- Use uma métrica de destino para escalar os serviços do Amazon ECS: aumenta ou diminui o número de tarefas que o serviço executa com base em um valor de destino para uma métrica

específica. Isso é semelhante à forma como o termostato mantém a temperatura da casa. Você seleciona a temperatura, e o termostato faz o resto.

- Usar incrementos predefinidos com base em alarmes do CloudWatch para ajustar a escala de serviços do Amazon ECS: aumenta ou diminui o número de tarefas que o serviço executa com base em um conjunto de ajustes de escalabilidade, conhecidos como ajustes em etapas, que variam conforme o tamanho da violação do alarme.
- Usar ações programadas para escalar os serviços do Amazon ECS: aumenta ou diminui o número de tarefas que o serviço executa com base na data e hora.
- Usar padrões históricos para escalar os serviços do Amazon ECS com escalabilidade preditiva: aumenta ou diminui o número de tarefas que o serviço executa com base em data analytics da carga histórica para detectar padrões diários ou semanais nos fluxos de tráfego.

Considerações

Ao usar políticas de escalabilidade, considere o seguinte:

- O Amazon ECS envia dados de métricas ao CloudWatch em intervalos de um minuto. As métricas não estarão disponíveis até que os clusters e os serviços enviem as métricas para o CloudWatch, e você não poderá criar alarmes do CloudWatch para métricas não existentes.
- As políticas de escalabilidade são compatíveis com um período de desaquecimento. Esse é o número de segundos para esperar que uma ação de escalabilidade anterior entre em vigor.
  - Para eventos de aumento, a intenção é aumentar de forma contínua (mas não excessiva). Depois que o Auto Scaling do serviço é expandido com êxito usando uma política de escalabilidade, ele começará a calcular o tempo de desaquecimento. A política de escalabilidade não aumentará novamente a capacidade desejada, a menos que um aumento da escala horizontalmente seja iniciado ou que o período de esfriamento termine. Enquanto o período de desaquecimento após expansão estiver em vigor, a capacidade adicionada pela ação de expansão de início será calculada como parte da capacidade desejada para a próxima ação de expansão.
  - Para eventos de redução, a intenção é reduzir de forma conservadora para proteger a disponibilidade da aplicação, de modo que as ações de redução sejam bloqueadas até que o período de desaquecimento tenha expirado. Porém, se outro alarme acionar uma ação de aumento da escala na horizontal durante o período de esfriamento da redução da escala na horizontal, o Service Auto Scaling aumentará a escala horizontalmente do destino

imediatamente. Nesse caso, o período de desaquecimento após redução é interrompido e não é concluído.

- O programador de serviço respeita a contagem desejada em todos os momentos, mas desde que você tenha políticas de escalabilidade e alarmes ativos em um serviço, o Auto Scaling do serviço pode alterar uma contagem desejada que você definiu manualmente.
- Se a contagem desejada de um serviço estiver definida abaixo do valor de capacidade mínimo e um alarme acionar uma atividade de aumento horizontal de escala, o Auto Scaling do serviço aumenta a escala da contagem desejada até o valor de capacidade mínimo e continua aumentando conforme necessário, com base na política de ajuste de escala associada ao alarme. Porém, uma atividade de redução não ajusta a contagem desejada, pois ela já está abaixo do valor mínimo de capacidade.
- Se a contagem desejada de um serviço for definida acima do valor máximo de capacidade e um alarme acionar uma atividade de redução de escala, o Auto Scaling do serviço aumenta a contagem desejada até o valor máximo de capacidade e continua a reduzir conforme necessário, com base na política de ajuste de escala associada ao alarme. Porém, uma atividade de expansão não ajusta a contagem desejada, pois ela já está acima do valor máximo de capacidade.
- Durante ações de escalabilidade, a contagem real de tarefas em execução em um serviço é o valor que o Auto Scaling do serviço usa como ponto de partida, ao contrário da contagem desejada. Esta deve ser a capacidade de processamento. Isso evita a escalabilidade excessiva (sem controle) que não pode ser atendida, por exemplo, caso não haja recursos de instância de contêiner suficientes para colocar as tarefas adicionais. Se a capacidade da instância de contêiner estiver disponível depois, a ação de escalabilidade pendente poderá continuar, e as ações de escalabilidade adicionais poderão continuar depois do período de desaquecimento.
- Se quiser que a contagem de tarefas seja dimensionada em zero quando não houver trabalho a ser feito, defina uma capacidade mínima de 0. Com políticas de dimensionamento com monitoramento do objetivo, quando a capacidade real é 0 e a métrica indica que há demanda de workload, o Auto Scaling do serviço aguarda o envio de um ponto de dados antes da expansão. Nesse caso, ele é expandido pela quantidade mínima possível como ponto de partida e, em seguida, retoma a escalabilidade com base na contagem real de tarefas em execução.
- O Application Auto Scaling desativa processos de redução da escala na horizontal enquanto as implantações do Amazon ECS estão em andamento. No entanto, processos de aumento continuam a ocorrer, a menos que sejam suspensos, durante uma implantação. Esse comportamento não se aplica aos serviços do Amazon ECS usando o controlador de implantação externo. Para obter mais informações, consulte Escalabilidade automática e implantações do serviço.

- Você tem várias opções de ajuste de escala automático de aplicação para tarefas do Amazon ECS. O rastreamento de destinos é o modo mais fácil de usar. Com isso, tudo o que você precisa fazer é definir um valor de destino para uma métrica, como a utilização média da CPU. Em seguida, o escalador automático gerencia automaticamente o número de tarefas necessárias para atingir esse valor. Com a escalabilidade por etapas, é possível reagir mais rapidamente às mudanças na demanda, pois define os limites específicos para suas métricas de escalabilidade e quantas tarefas adicionar ou remover quando os limites são ultrapassados. E, o mais importante, é possível reagir rapidamente às mudanças na demanda minimizando a quantidade de tempo em que um alarme de limite é violado.

Para obter mais informações sobre as práticas recomendadas para o ajuste de escala automático do serviço, consulte Otimização do ajuste de escala automático de serviços do Amazon ECS.

Escalabilidade automática e implantações do serviço

O Application Auto Scaling desativa processos de redução da escala na horizontal enquanto as implantações do Amazon ECS estão em andamento. No entanto, processos de aumento continuam a ocorrer, a menos que sejam suspensos, durante uma implantação. Esse comportamento não se aplica aos serviços do Amazon ECS usando o controlador de implantação externo. Se você quiser suspender processos de aumento enquanto as implantações estiverem em andamento, siga as etapas a seguir.

1. Chame o comando describe-scalable-targets, especificando o ID do recurso do serviço associado ao destino escalável no Application Auto Scaling (Exemplo: `service/default/sample-webapp`). Registre a saída. Você precisará dela quando chamar o próximo comando.
2. Chame o comando register-scalable-target, especificando o ID, o namespace e a dimensão escalável do recurso. Especifique `true` para `DynamicScalingInSuspended` e `DynamicScalingOutSuspended`.
3. Depois que a implantação estiver concluída, será possível chamar o comando register-scalable-target para retomar a escalabilidade.

Para obter mais informações, consulte Suspender e retomar a escalabilidade do Application Auto Scaling.

Use uma métrica de destino para escalar os serviços do Amazon ECS

Com as políticas de dimensionamento com monitoramento do objetivo, você seleciona uma métrica e define um valor pretendido. O Amazon ECS Service Auto Scaling cria e gerencia os alarmes do CloudWatch que controla a política de escalabilidade e calcula o ajuste da escalabilidade com base na métrica e no valor de destino. A política de escalabilidade adiciona ou remove tarefas de serviço conforme necessário para manter a métrica no valor de destino especificado ou próxima a ele. Além de manter a métrica próxima ao valor de destino, uma política de escalabilidade de rastreamento de destino também se ajusta às flutuações na métrica, devido a um padrão de carga de flutuação, e minimiza as flutuações rápidas no número de tarefas que estão sendo executadas no serviço.

As políticas de rastreamento de destinos eliminam a necessidade de definir manualmente os alarmes e os ajustes de escalabilidade do CloudWatch. O Amazon ECS lida com isso automaticamente com base no destino definido.

Considere os fatores a seguir quando usar políticas de rastreamento de destino:

- Uma política de escalabilidade de rastreamento de destino pressupõe que ela deve aumentar a escalabilidade quando a métrica especificada estiver acima do valor de destino. Você não pode usar uma política de escalabilidade de rastreamento de destino para expandir quando a métrica especificada estiver abaixo do valor de destino.
- Uma política de escalabilidade de rastreamento de destino não escala quando a métrica especificada tem dados insuficientes. Ela não aumenta a escalabilidade porque não interpreta dados insuficientes como baixa utilização.
- É possível ver lacunas entre o valor de destino e os pontos de dados de métrica reais. Isso ocorre porque o Auto Scaling do serviço sempre atua de forma conservadora, ao arredondar para cima ou para baixo ao determinar a capacidade a ser adicionada ou removida. Isso evita que ele adicione capacidade insuficiente ou remova muita capacidade.
- Para garantir a disponibilidade do aplicativo, o serviço se expande proporcionalmente à métrica o mais rápido possível, mas é reduzido gradualmente.
- O Application Auto Scaling desativa processos de redução da escala na horizontal enquanto as implantações do Amazon ECS estão em andamento. No entanto, processos de aumento continuam a ocorrer, a menos que sejam suspensos, durante uma implantação. Esse comportamento não se aplica aos serviços do Amazon ECS usando o controlador de implantação externo. Para obter mais informações, consulte Escalabilidade automática e implantações do serviço.

- É possível ter várias políticas de dimensionamento com monitoramento do objetivo para um serviço do Amazon ECS, desde que cada uma delas use uma métrica diferente. A intenção do Auto Scaling do serviço é sempre priorizar a disponibilidade. Portanto, seu comportamento será diferente, dependendo de as políticas de monitoramento do objetivo estarem prontas para aumento ou redução. Ele vai aumentar a escala do serviço horizontalmente se qualquer uma das políticas de rastreamento de destino estiver pronta para aumentar a escala horizontalmente, mas só vai reduzir a escala horizontalmente se todas as políticas de monitoramento de destino (com a parte da redução da escala horizontalmente ativada) estiverem prontas para reduzir a escala horizontalmente.
- Não edite nem exclua os alarmes do CloudWatch que o Auto Scaling do serviço gerencia para uma política de dimensionamento com monitoramento do objetivo. O Auto Scaling do serviço exclui os alarmes automaticamente quando você exclui a política de dimensionamento.
- A métrica `ALBRequestCountPerTarget` para as políticas de dimensionamento de monitoramento de destinos não é compatível com o tipo de implantação azul/verde.

Para obter mais informações sobre políticas de escalabilidade de rastreamento de destino, consulte Políticas de escalabilidade de rastreamento de destino no Guia do usuário do Application Auto Scaling.

Criar uma política de escalabilidade de rastreamento de destino para ajuste de escala automático do serviço do Amazon ECS

Crie uma política de escalabilidade de rastreamento de destino para que o Amazon ECS aumente ou diminua automaticamente a contagem de tarefas desejadas em seu serviço. O rastreamento de destino funciona a partir de um valor métrico de destino.

Console

1. Além das permissões padrão do IAM para criar e atualizar serviços, você precisa de permissões adicionais. Para obter mais informações, consulte Permissões obrigatórias do IAM para o ajuste de escala automático do serviço Amazon ECS.
2. Determine as métricas a serem usadas para a política. As seguintes métricas estão disponíveis:
   - ECSServiceAverageCPUUtilization: utilização média da CPU que o serviço deveria usar.
   - ECSServiceAverageMemoryUtilization: utilização média da memória que o serviço deveria usar.

- ALBRequestCountPerTarget: o número médio de solicitações por minuto que a tarefa deve idealmente receber.

3. Abra o console em https://console.aws.amazon.com/ecs/v2.
4. Na página Clusters, escolha o cluster.
5. Na página de detalhes do cluster, na seção Serviços, escolha o serviço.

   A página de detalhes do serviço é exibida.
6. Escolha Definir o número de tarefas.
7. Em Contagem de tarefas do serviço do Amazon ECS, escolha Usar ajuste de escala automático.

   A seção Contagem de tarefas é exibida.

   a. Em Número mínimo de tarefas, insira o limite inferior do número de tarefas a serem usadas pelo ajuste de escala automático. A contagem desejada não será inferior a essa contagem.
   b. Em Máximo, insira o limite superior do número de tarefas a serem usadas pelo ajuste de escala automático. A contagem desejada não ultrapassará essa contagem.
   c. Escolha Salvar.

      A página de políticas será exibida.
8. Escolha Criar política de escalabilidade.

   A página Criar política é exibida.
9. Em Scaling policy type (Tipo de política de escalabilidade), escolha Target tracking (Rastreamento de destino).
10. Em Policy name (Nome da política), insira o nome da política.
11. Em Tipo de métrica, escolha suas métricas na lista de opções.
12. Em Utilização pretendida, insira o valor desejado para o percentual de tarefas que o Amazon ECS deve manter. O ajuste de escala automático do serviço expandirá sua capacidade até que a utilização média seja igual à utilização pretendida ou até atingir o número máximo de tarefas especificado.
13. Em Configurações adicionais, faça o seguinte

    a. Em Período de espera após reduzir a escala horizontalmente, insira a quantidade de tempo em segundos após a conclusão de uma ação de redução antes que uma outra atividade de redução possa iniciar.

b. Em Período de espera após aumentar a escala horizontalmente, insira a quantidade de tempo em segundos para aguardar que uma atividade de ajuste de escala anterior entre em vigor.

c. Para criar somente uma política de aumento da escala horizontal, selecione Desabilitar a ação de redução.

14. Escolha Criar política de escalabilidade.

AWS CLI

1. Registre seu serviço do Amazon ECS como um destino escalável usando o comando register-scalable-target.
2. Crie uma política de escalabilidade usando o comando put-scaling-policy.

Usar incrementos predefinidos com base em alarmes do CloudWatch para ajustar a escala de serviços do Amazon ECS

Com as políticas de escalabilidade em etapas, é possível criar e gerenciar os alarmes do CloudWatch que invocam o processo de escalabilidade. Quando um alarme é violado, o Amazon ECS inicia a política de escalabilidade associada a esse alarme. A política de escalabilidade em etapas escala as tarefas usando um conjunto de ajustes conhecido como ajustes em etapas. A dimensão dos ajustes varia de acordo com a magnitude da violação do alarme.

- Se a violação exceder o primeiro limite, o Amazon ECS aplicará o primeiro ajuste em etapa.
- Se a violação exceder o segundo limite, o Amazon ECS aplicará o segundo ajuste em etapa, e assim por diante.

É altamente recomendável usar uma política de escalabilidade de rastreamento de destino para escalar métricas, como utilização média de CPU ou contagem média de solicitações por destino. Métricas que diminuem quando a capacidade aumenta e aumentam quando a capacidade diminui podem ser usadas para aumentar ou reduzir a escala do número de tarefas proporcionalmente usando o rastreamento de destino. Isso ajuda a garantir que o Amazon ECS siga estritamente a curva de demanda para seus aplicativos.

Criar uma política de escalabilidade em etapas para ajuste de escala automático do serviço do Amazon ECS

Crie uma política de escalabilidade em etapas para que o Amazon ECS aumente ou diminua automaticamente o número desejado de tarefas em seu serviço. A escalabilidade em etapas é executada com base em um conjunto de ajustes de escalabilidade, conhecidos como ajustes em etapas, que variam com base no tamanho da ruptura do alarme.

Console

1. Além das permissões padrão do IAM para criar e atualizar serviços, você precisa de permissões adicionais. Para obter mais informações, consulte Permissões obrigatórias do IAM para o ajuste de escala automático do serviço Amazon ECS.
2. Determine as métricas a serem usadas para a política. As seguintes métricas estão disponíveis:
   - ECSServiceAverageCPUUtilization: utilização média da CPU que o serviço deveria usar.
   - ECSServiceAverageMemoryUtilization: utilização média da memória que o serviço deveria usar.
   - ALBRequestCountPerTarget: o número médio de solicitações por minuto que a tarefa deve idealmente receber.
3. Crie um alarme do CloudWatch para as métricas. Para obter mais informações, consulte Criar um alarme do CloudWatch com base em um limite estático no Guia do usuário do Amazon CloudWatch.
4. Abra o console em https://console.aws.amazon.com/ecs/v2.
5. Na página Clusters, escolha o cluster.
6. Na página de detalhes do cluster, na seção Serviços, escolha o serviço.

   A página de detalhes do serviço é exibida.
7. Escolha Definir o número de tarefas.
8. Em Contagem de tarefas do serviço do Amazon ECS, escolha Usar ajuste de escala automático.

   A seção Contagem de tarefas é exibida.

   a. Em Número mínimo de tarefas, insira o limite inferior do número de tarefas a serem usadas pelo ajuste de escala automático. A contagem desejada não será inferior a essa contagem.

   b. Em Máximo, insira o limite superior do número de tarefas a serem usadas pelo ajuste de escala automático. A contagem desejada não ultrapassará essa contagem.

c. Escolha Salvar.

A página de políticas será exibida.

9. Escolha Criar política de escalabilidade.

A página Criar política é exibida.

10. Em Tipo de política de escalabilidade, escolha Escalabilidade em etapas.
11. Configure as propriedades de aumentar a escala horizontalmente. Em Etapas para adicionar tarefas, faça o seguinte:

a. Em Policy name (Nome da política), insira o nome da política.
b. Em Nome do alarme do CloudWatch, escolha o alarme do CloudWatch.
c. Em Tipo de agregação de métrica, escolha como comparar a métrica selecionada ao limite definido.
d. Em Tipos de ajuste, escolha se o ajuste é baseado em uma alteração no número de tarefas ou em uma alteração na porcentagem de tarefas.
e. Em Ações a serem tomadas, insira os valores da ação a ser tomada.

Escolha Adicionar etapa para adicionar outras ações.

12. Configure as propriedades de reduzir a escala horizontalmente. Em Etapas para remover tarefas, faça o seguinte:

a. Em Policy name (Nome da política), insira o nome da política.
b. Em Nome do alarme do CloudWatch, escolha o alarme do CloudWatch.
c. Em Tipo de agregação de métrica, escolha como comparar a métrica selecionada ao limite definido.
d. Em Tipos de ajuste, escolha se o ajuste é baseado em uma alteração no número de tarefas ou em uma alteração na porcentagem de tarefas.
e. Em Ações a serem tomadas, insira os valores da ação a ser tomada.

Escolha Adicionar etapa para adicionar outras ações.

13. Em Período de espera, insira a quantidade de tempo, em segundos, para aguardar que uma atividade de ajuste de escala anterior entre em vigor. Para uma política de adição, esse é o momento após uma atividade de aumento horizontal de escala em que a política de ajuste de escala bloqueia as atividades de redução horizontal de escala e limita quantas tarefas podem ser ampliadas por vez. Para uma política de remoção, esse é o momento após uma atividade de

redução horizontal de escala que precisa ser concluída antes que outras atividades do mesmo tipo possam iniciar.

14. Escolha Criar política de escalabilidade.

AWS CLI

1. Registre seu serviço do Amazon ECS como um destino escalável usando o comando register-scalable-target.
2. Crie uma política de escalabilidade usando o comando put-scaling-policy.

Usar ações programadas para escalar os serviços do Amazon ECS

Com a escalabilidade programada, é possível configurar a escalabilidade automática para a aplicação com base em alterações de carga previsíveis ao criar ações programadas que aumentam ou diminuem o número de tarefas em momentos específicos. Isso permite escalar a aplicação de forma proativa para corresponder às alterações de carga previsíveis.

Essas ações de escalabilidade programadas permitem otimizar os custos e a performance. Sua aplicação tem um número de tarefas suficiente para lidar com o pico de tráfego de meio de semana, mas não faz provisionamento excessivo de número de tarefas em outros momentos.

É possível usar a escalabilidade programada e as políticas de escalabilidade em conjunto para obter os benefícios de abordagens proativas e reativas para a escalabilidade. Após a execução de uma ação de escalabilidade programada, a política de escalabilidade pode continuar a tomar decisões sobre a necessidade de escalar ainda mais o número de tarefas. Isso ajuda a garantir que você tenha um número de tarefas suficiente para lidar com a carga de sua aplicação. Embora sua aplicação seja escalada para atender à demanda, a capacidade atual deve estar dentro dos números de tarefas mínimo e máximo definidos pela ação programada.

Você pode configurar a escala do cronograma usando a AWS CLI. Para obter mais informações sobre o ajuste de escala programado, consulte Scheduled Scaling no Guia do usuário do Application Auto Scaling.

Criar uma ação programada para o ajuste de escala automático do serviço do Amazon ECS

Crie uma ação programada para fazer com que o Amazon ECS aumente ou diminua o número de tarefas executadas pelo serviço com base na data e na hora.

Console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, escolha o cluster.
3. Na página de detalhes do cluster, na seção Serviços, escolha o serviço.

   A página de detalhes do serviço é exibida.
4. Escolha Ajuste de escala automático do serviço.

   A página de ajuste de escala automático do serviço é exibida.
5. Se você não configurou a escalabilidade automática do serviço, escolha Definir o número de tarefas.

   A seção Contagem de tarefas de serviços do Amazon ECS é exibida.

   Em Contagem de tarefas do serviço Amazon ECS, escolha Usar ajuste de escala automático do serviço para ajustar a contagem de tarefas desejada do seu serviço.

   A seção Contagem de tarefas é exibida.

   a. Em Número mínimo de tarefas, insira o limite inferior do número de tarefas a serem usadas pelo ajuste de escala automático. A contagem desejada não será inferior a essa contagem.
   b. Em Máximo, insira o limite superior do número de tarefas a serem usadas pelo ajuste de escala automático. A contagem desejada não ultrapassará essa contagem.
   c. Escolha Escolher Salvar.

      A página de políticas será exibida.
6. Escolha Ações programadas e, em seguida, escolha Criar.

   A página Criar ação de programação é exibida.
7. Em Nome da ação, insira um nome exclusivo.
8. Em Time zone (Fuso horário), escolha um fuso horário.

   Todos os fusos horários listados são do banco de dados de fuso horário da IANA. Para obter mais informações, consulte a Lista de fusos horários no banco de dados de FH.
9. Em Hora de início, insira a Data e a Hora em que a ação começa.

Se você escolher uma programação recorrente, o horário inicial definirá quando a primeira ação programada na série recorrente será executada.

10. Em Recurrence (Recorrência), selecione uma das opções disponíveis.

    - Para escalar em uma programação recorrente, escolha com que frequência o Amazon ECS deverá executar a ação programada.
      - Se você escolher uma opção que começa com Taxa, a expressão cron será criada para você.
      - Se você escolher Cron, insira uma expressão do cron que especifique quando executar a ação, em UTC.
    - Para escalar apenas uma vez, escolha Uma vez.

11. Em Ajustes de tarefas, faça o seguinte:

    - Em Mínimo, insira o número mínimo de tarefas que o serviço deve executar.
    - Em Máximo, insira o número máximo de tarefas que o serviço deve executar.

12. Escolha Criar ação programada.

CLI

Use o AWS CLI da seguinte maneira para configurar políticas de escalonamento agendado para seu serviço. Substitua cada *espaço reservado para entrada do usuário* por suas próprias informações.

Exemplo: para escalar apenas uma vez

Utilize o seguinte comando put-scheduled-action com o `--start-time "YYYY-MM-DDThh:mm:ssZ"` e uma ou ambas as opções `--MinCapacity` e `--MaxCapacity`.

```
aws application-autoscaling put-scheduled-action --service-namespace ecs \
  --resource-id service/my-cluster/my-service \
  --scheduled-action-name my-one-time-schedule \
  --start-time 2021-01-30T12:00:00 \
  --scalable-target-action MinCapacity=3,MaxCapacity=10
```

Para programar a escalabilidade em uma programação recorrente

Utilize o seguinte comando put-scheduled-action. Substitua os valores de *user input* pelos seus.

```
aws application-autoscaling put-scheduled-action --service-namespace ecs \
  --resource-id service/my-cluster/my-service \
  --scheduled-action-name my-recurring-action \
  --schedule "rate(5 hours)" \
  --start-time 2021-01-30T12:00:00 \
  --end-time 2021-01-31T22:00:00 \
  --scalable-target-action MinCapacity=3,MaxCapacity=10
```

O cronograma de recorrência especificado é executado de acordo com o fuso horário UTC. Para especificar um fuso horário diferente, inclua a opção `--time-zone` e especifique o nome canônico do fuso horário IANA, como no exemplo a seguir.

```
--time-zone "America/New_York"
```

Para obter mais informações, consulte a Lista de fusos horários no banco de dados de FH.

Usar padrões históricos para escalar os serviços do Amazon ECS com escalabilidade preditiva

A escalabilidade preditiva utiliza dados de carga anteriores dos fluxos de tráfego para analisar padrões diários ou semanais. Em seguida, ela usa essa análise para prever necessidades futuras e aumentar proativamente as tarefas em seu serviço, conforme necessário.

O ajuste de escala automático preditivo é mais útil nas situações a seguir.

- Tráfego cíclico: maior utilização de recursos durante o horário comercial e utilização reduzida de recursos durante a noite e nos fins de semana.
- Padrões recorrentes de workloads intermitentes: exemplos incluem processamento em lote, testes ou análise periódica de dados.
- Aplicações com tempo de inicialização longo: isso pode afetar a performance da aplicação durante eventos de aumentar a escala horizontalmente, causando latência perceptível.

Se suas aplicações demoram muito para inicializar e o tráfego aumenta em um padrão regular, considere o uso de escalabilidade preditiva. Ela ajuda você a escalar mais rapidamente aumentando proativamente o número de tarefas para cargas previstas, em vez de usar políticas de escalabilidade dinâmica, como Rastreamento de destino ou Escalabilidade em etapas, sozinhas. Ao ajudar você a evitar a possibilidade de provisionar excessivamente o número de tarefas, a escalabilidade preditiva também pode economizar dinheiro.

Por exemplo, considere uma aplicação com elevado índice de utilização durante o horário comercial e baixo uso durante a noite. No início de cada dia útil, a escalabilidade preditiva pode aumentar a escala para execução de tarefas antes do primeiro fluxo de tráfego. Isso ajuda sua aplicação a manter alta disponibilidade e performance ao passar de um período de menor utilização para um período de maior utilização. Você não precisa esperar que a escalabilidade dinâmica reaja à mudança de tráfego. Você também não precisa gastar tempo revisando os padrões de carga da aplicação e tentando alocar a quantidade certa de tarefas usando a escalabilidade programada.

O ajuste de escala preditivo é um recurso de nível de serviço que dimensiona a tarefa do seu serviço independentemente do dimensionamento da capacidade computacional subjacente (por exemplo, EC2 ou Fargate). Para o Fargate, a AWS gerencia e ajusta automaticamente a escala da capacidade subjacente com base nos requisitos da tarefa. Para a capacidade do EC2, é possível usar provedores de capacidade do grupo do Auto Scaling para escalar automaticamente as instâncias do EC2 subjacentes com base nos requisitos de ajuste de escala das suas tarefas.

Conteúdo

- Saiba como o ajuste de escala preditivo funciona com o Amazon ECS
- Criar uma política de escalabilidade preditiva para ajuste de escala automático do serviço do Amazon ECS
- Avaliar as políticas de escalabilidade preditiva para o Amazon ECS
- Usar ações programadas para substituir os valores de previsão para o Amazon ECS
- Política avançada de escalabilidade preditiva usando métricas personalizadas para o Amazon ECS

Saiba como o ajuste de escala preditivo funciona com o Amazon ECS

Aqui você pode conhecer as considerações relacionadas o uso da escalabilidade preditiva, como ela funciona e quais são os limites.

Considerações sobre o uso da escalabilidade preditiva

- Você deseja garantir que a escalabilidade preditiva seja adequada à sua workload. Você pode verificar isso configurando políticas de escalabilidade no modo somente previsão e ver a recomendação feita pelo console. Você deve avaliar a previsão e as recomendações antes de começar a usar a escalabilidade preditiva.
- Antes que a escalabilidade preditiva possa iniciar a previsão, pelo menos 24 horas de dados históricos são necessários. Quanto mais dados históricos estiverem disponíveis, mais eficaz será a previsão, sendo duas semanas o ideal. Também será necessário aguardar 24 horas antes que

a escalabilidade preditiva possa gerar novas previsões quando um serviço do Amazon ECS é excluído e um novo serviço é criado. Uma maneira de acelerar isso é usar métricas personalizadas para agregar métricas no novo serviço e no serviço antigo do Amazon ECS.

- Escolha uma métrica de carga que represente com precisão a carga total da sua aplicação e que seja o aspecto mais importante a ser escalado.
- A escalabilidade dinâmica com escalabilidade preditiva ajuda você a acompanhar de perto a demanda da sua aplicação para que você possa reduzir a escala durante períodos de calmaria e aumentá-la durante aumentos inesperados no tráfego. Quando várias políticas de escalabilidade estão ativas, cada política determina o número de tarefas desejadas de forma independente, e esse número é definido como o maior entre eles.
- É possível usar a escalabilidade preditiva junto com suas políticas de escalabilidade dinâmica, como rastreamento de destino ou escalabilidade em etapas, para que suas aplicações sejam dimensionadas com base em padrões históricos e em tempo real. Por si só, a escalabilidade preditiva não reduz a escala para suas tarefas.
- Se você usar um perfil personalizado ao chamar a API `register-scalable-target`, poderá receber um erro dizendo que a política de escalabilidade preditiva só pode funcionar com o SLR habilitado. Nesse caso, você deve chamar `register-scalable-target` novamente, mas sem role-arn. Use SLR ao registrar o destino escalável e chame a API `put-scaling-policy`.

Como a escalabilidade preditiva funciona

Use a escalabilidade preditiva criando uma política de escala preditiva que especifique a métrica do CloudWatch a ser monitorada e analisada. A escalabilidade preditiva deve ter pelo menos 24 horas de dados para começar a prever valores futuros.

Depois que você cria a política, a escala preditiva começa a analisar os dados de métricas dos últimos 14 dias para identificar padrões. Essa análise é usada para gerar as próximas 48 horas de previsões horárias de requisitos. Os dados mais recentes do CloudWatch são usados para atualizar a previsão a cada seis horas. À medida que novos dados são recebidos, a escalabilidade preditiva é capaz de melhorar continuamente a precisão das previsões futuras.

Quando você ativa a escala preditiva pela primeira vez, ela é executada no modo somente de previsão. Eae gera previsões nesse modo, mas não escala seu serviço do Amazon ECS com base nessas previsões. Isso significa que você pode avaliar a exatidão e a adequação da previsão. Você pode visualizar os dados de previsão usando a operação da API `GetPredictiveScalingForecast` ou o AWS Management Console.

Quando você decidir começar a usar a escala preditiva, alterne a política de escalabilidade para o modo prever e escalar. O cenário a seguir ocorre nesse modo.

Seu serviço do Amazon ECS é escalado no início de cada hora com base na previsão para essa hora, por padrão. Você pode optar por começar mais cedo usando a propriedade `SchedulingBufferTime` na operação da API `PutScalingPolicy`. Isso faz com que novas tarefas sejam iniciadas antes da demanda prevista e concede a elas tempo para inicializar e se preparar para lidar com o tráfego.

Limite máximo de tarefas

Ao registrar serviços do Amazon ECS para escalabilidade, você define um número máximo de tarefas que podem ser iniciadas por serviço. Por padrão, quando políticas de escalabilidade estão definidas, não é possível aumentar o número de tarefas além do limite máximo.

Como alternativa, é possível permitir que o número máximo de tarefas do serviço seja aumentado automaticamente se a previsão se aproximar ou exceder o número máximo de tarefas do serviço do Amazon ECS.

> **Warning**
>
> Tenha cuidado ao permitir que o número máximo de tarefas seja aumentado automaticamente. Isso pode fazer com que mais instâncias do que o pretendido sejam executadas se a capacidade máxima aumentada não for monitorada e gerenciada. O número máximo de tarefas aumentado passará a ser o novo número máximo normal de tarefas para o serviço do Amazon ECS até que você o atualize manualmente. O número máximo de tarefas não diminui automaticamente de volta para o máximo original.

Regiões compatíveis

- Leste dos EUA (Norte da Virgínia)
- Leste dos EUA (Ohio)
- Oeste dos EUA (Norte da Califórnia)
- Oeste dos EUA (Oregon)
- África (Cidade do Cabo)
- Ásia-Pacífico (Hong Kong)

- Ásia-Pacífico (Jacarta)
- Ásia-Pacífico (Mumbai)
- Ásia-Pacífico (Osaka)
- Ásia-Pacífico (Seul)
- Ásia-Pacífico (Singapura)
- Ásia-Pacífico (Sydney)
- Ásia-Pacífico (Tóquio)
- Canadá (Central)
- China (Pequim)
- China (Ningxia)
- Europa (Frankfurt)
- Europa (Irlanda)
- Europa (Londres)
- Europa (Milão)
- Europa (Paris)
- Europa (Estocolmo)
- Oriente Médio (Bahrein)
- South America (São Paulo)
- AWS GovCloud (Leste dos EUA)
- AWS GovCloud (Oeste dos EUA)

Criar uma política de escalabilidade preditiva para ajuste de escala automático do serviço do Amazon ECS

Crie uma política de escalabilidade preditiva para que o Amazon ECS aumente ou diminua o número de tarefas executadas pelo serviço com base em dados de histórico.

> **Note**
>
> Um novo serviço precisa fornecer pelo menos 24 horas de dados antes que uma previsão possa ser gerada.

Console

1. Além das permissões padrão do IAM para criar e atualizar serviços, você precisa de permissões adicionais. Para obter mais informações, consulte Permissões obrigatórias do IAM para o ajuste de escala automático do serviço Amazon ECS.
2. Determine as métricas a serem usadas para a política. As seguintes métricas estão disponíveis:

   - ECSServiceAverageCPUUtilization: utilização média da CPU que o serviço deveria usar.
   - ECSServiceAverageMemoryUtilization: utilização média da memória que o serviço deveria usar.
   - ALBRequestCountPerTarget: o número médio de solicitações por minuto que a tarefa deve idealmente receber.

   Como alternativa, você pode usar uma métrica personalizada. É necessário definir os seguintes valores:

   - Carga: uma métrica que representa com precisão a carga total da aplicação e é o aspecto mais importante a ser escalado.
   - Métrica de escalabilidade: o melhor indicador de quanta utilização é ideal para sua aplicação.
3. Abra o console em https://console.aws.amazon.com/ecs/v2.
4. Na página Clusters, escolha o cluster.
5. Na página de detalhes do cluster, na seção Serviços, escolha o serviço.

   A página de detalhes do serviço é exibida.
6. Escolha Ajuste de escala automático do serviço e depois Definir o número de tarefas.
7. Em Contagem de tarefas do serviço do Amazon ECS, escolha Usar ajuste de escala automático.

   A seção Contagem de tarefas é exibida.

   a. Em Número mínimo de tarefas, insira o limite inferior do número de tarefas a serem usadas pelo ajuste de escala automático. A contagem desejada não será inferior a essa contagem.
   b. Em Máximo, insira o limite superior do número de tarefas a serem usadas pelo ajuste de escala automático. A contagem desejada não ultrapassará essa contagem.
   c. Escolha Salvar.

A página de políticas será exibida.

8. Escolha Criar política de escalabilidade.

   A página Criar política é exibida.

9. Em Tipo de política de escalabilidade, escolha Escalabilidade preditiva.
10. Em Policy name (Nome da política), insira o nome da política.
11. Em Métricas, escolha suas métricas na lista de opções.

    Se tiver escolhido Application Load Balancer request count per target (Número de solicitações do Application Load Balancer por destino), escolha um grupo de destino em Target group (Grupo de destino). A opção Número de solicitações do Application Load Balancer por destino só será válida de você tiver anexado um grupo de destino do Application Load Balancer para seu serviço.

    Se você escolheu Par de métricas personalizadas, escolha métricas individuais nas listas para Métrica de carga e Métrica de escalabilidade.

12. Em Utilização pretendida, insira o valor desejado para o percentual de tarefas que o Amazon ECS deve manter. O ajuste de escala automático do serviço expandirá sua capacidade até que a utilização média seja igual à utilização pretendida ou até atingir o número máximo de tarefas especificado.
13. Escolha Criar política de escalabilidade.

AWS CLI

Use a AWS CLI conforme mostrado a seguir para configurar políticas de escalabilidade preditiva para seu serviço do Amazon ECS. Substitua cada *espaço reservado para entrada do usuário* por suas próprias informações.

Para obter mais informações sobre as métricas do CloudWatch que você pode especificar para uma política de escalabilidade preditiva, consulte PredictivEscalingMetricSpecification na Referência da API do Amazon EC2 Auto Scaling.

Exemplo 1: uma política de escalabilidade preditiva com memória predefinida.

Veja a seguir um exemplo de política com uma configuração de memória predefinida.

```
cat policy.json
```

```
{
    "MetricSpecifications": [
        {
            "TargetValue": 40,
            "PredefinedMetricPairSpecification": {
                "PredefinedMetricType": "ECSServiceMemoryUtilization"
            }
        }
    ],
    "SchedulingBufferTime": 3600,
    "MaxCapacityBreachBehavior": "HonorMaxCapacity",
    "Mode": "ForecastOnly"
}
```

O exemplo a seguir ilustra a criação da política via execução do comando put-scaling-policy com o arquivo de configuração especificado.

```
aws application-autoscaling put-scaling-policy \
--service-namespace ecs \
--region us-east-1 \
--policy-name predictive-scaling-policy-example \
--resource-id service/MyCluster/test \
--policy-type PredictiveScaling \
--scalable-dimension ecs:service:DesiredCount \
--predictive-scaling-policy-configuration file://policy.json
```

Em caso de êxito, esse comando retornará o ARN da política.

```
{
    "PolicyARN": "arn:aws:autoscaling:us-
east-1:012345678912:scalingPolicy:d1d72dfe-5fd3-464f-83cf-824f16cb88b7:resource/ecs/
service/MyCluster/test:policyName/predictive-scaling-policy-example",
    "Alarms": []
}
```

Exemplo 2: uma política de escalabilidade preditiva com CPU predefinida.

Veja a seguir um exemplo de política com uma configuração de CPU predefinida.

```
cat policy.json
{
    "MetricSpecifications": [
```

```
        {
            "TargetValue": 0.00000004,
            "PredefinedMetricPairSpecification": {
                "PredefinedMetricType": "ECSServiceCPUUtilization"
            }
        }
    ],
    "SchedulingBufferTime": 3600,
    "MaxCapacityBreachBehavior": "HonorMaxCapacity",
    "Mode": "ForecastOnly"
}
```

O exemplo a seguir ilustra a criação da política via execução do comando put-scaling-policy com o arquivo de configuração especificado.

```
aws aas put-scaling-policy \
--service-namespace ecs \
--region us-east-1 \
--policy-name predictive-scaling-policy-example \
--resource-id service/MyCluster/test \
--policy-type PredictiveScaling \
--scalable-dimension ecs:service:DesiredCount \
--predictive-scaling-policy-configuration file://policy.json
```

Em caso de êxito, esse comando retornará o ARN da política.

```
{
    "PolicyARN": "arn:aws:autoscaling:us-
east-1:012345678912:scalingPolicy:d1d72dfe-5fd3-464f-83cf-824f16cb88b7:resource/ecs/
service/MyCluster/test:policyName/predictive-scaling-policy-example",
    "Alarms": []
}
```

Avaliar as políticas de escalabilidade preditiva para o Amazon ECS

Antes de usar uma política de escalabilidade preditiva para escalar seus serviços, revise as recomendações e outros dados da política no console do Amazon ECS. Isso é importante porque você não quer que uma política de escalabilidade preditiva escale sua capacidade real até saber que suas previsões estão corretas.

Se o serviço for novo, aguarde 24 horas para criar a primeira previsão.

Ao criar uma previsão, o AWS usa dados históricos. Se seu serviço ainda não tiver muitos dados históricos recentes, a escalabilidade preditiva poderá preencher temporariamente a previsão com agregados criados com base nos agregados históricos disponíveis no momento. As previsões são preenchidas até duas semanas anteriores à data de criação da política.

Visualizar recomendações de escalabilidade preditiva

Para realizar uma análise eficaz, o ajuste de escala automático do serviço deve ter pelo menos duas políticas de escalabilidade preditiva para comparação. (Porém, ainda é possível analisar as conclusões de uma única política.) Ao criar várias políticas, é possível avaliar uma política que usa uma métrica em relação a uma política que usa outra métrica. Também é possível avaliar o impacto de diferentes combinações de valores de destino e métricas. Depois que as políticas de escalabilidade preditiva são criadas, o Amazon ECS imediatamente começa a avaliar qual política faria um trabalho melhor ao escalar seu grupo.

Para visualizar suas recomendações no console do Amazon ECS

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, escolha o cluster.
3. Na página de detalhes do cluster, na seção Serviços, escolha o serviço.

   A página de detalhes do serviço é exibida.
4. Escolha Ajuste de escala automático do serviço.
5. Escolha a política de escalabilidade preditiva e, em seguida, escolha Ações, Escalabilidade preditiva e Visualizar recomendação.

   Você pode visualizar detalhes sobre uma política junto com nossa recomendação. A recomendação indica se é melhor usar a política de escalabilidade preditiva ou não usá-la.

   Se você não tiver certeza se uma política de escalabilidade preditiva é apropriada para seu grupo, analise as colunas Impacto na disponibilidade e Impacto no custo para escolher a política certa. As informações de cada coluna indicam qual é o impacto da política.

   - Impacto na disponibilidade: descreve se a política evitaria um impacto negativo na disponibilidade ao provisionar tarefas suficientes para lidar com a workload em comparação com não usar a política.
   - Impacto no custo: descreve se a política evitaria um impacto negativo em seus custos ao não provisionar em excesso as tarefas em comparação com não usar a política. Com o

provisionamento excessivo, seus serviços tornam-se subutilizados ou ociosos, o que só aumenta o impacto nos custos.

Se você tiver várias políticas, uma etiqueta Melhor previsão será exibida ao lado do nome da política que oferece mais benefícios de disponibilidade a um custo menor. O impacto na disponibilidade tem um peso maior.

6. (Opcional) Para selecionar o período de tempo desejado para os resultados da recomendação, escolha o valor de sua preferência no menu suspenso Período de avaliação: 2 dias, 1 semana ou 2 semanas. Por padrão, o período de avaliação são as duas últimas semanas. Um período de avaliação mais longo oferece mais pontos de dados para os resultados da recomendação. Porém, adicionar mais pontos de dados pode não melhorar os resultados, se seus padrões de carga tiverem sido alterados, como após um período de demanda excepcional. Nesse caso, é possível obter uma recomendação mais focada analisando dados mais recentes.

> **Note**
>
> As recomendações são geradas somente para políticas que estão no modo Somente previsão. O recurso de recomendações funciona melhor quando uma política está no modo Somente previsão durante o período de avaliação. Se você iniciar uma política no modo de Prever e escalar e alterná-la para o modo Somente previsão posteriormente, é provável que as conclusões dessa política tenham desvios. Isso ocorre porque a política já contribuiu em favor da capacidade real.

Analisar grafos de monitoramento de escalabilidade preditiva

No console, é possível analisar a previsão dos dias, semanas ou meses anteriores para visualizar a performance da política ao longo do tempo. Você também pode usar essas informações para avaliar a precisão das previsões ao decidir se permitirá que uma política escale o número real de tarefas.

Para revisar grafos de monitoramento de escalabilidade preditiva no console do Amazon ECS

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, escolha o cluster.
3. Na página de detalhes do cluster, na seção Serviços, escolha o serviço.

   A página de detalhes do serviço é exibida.

4. Escolha Ajuste de escala automático do serviço.
5. Escolha a política de escalabilidade preditiva e, em seguida, escolha Ações, Escalabilidade preditiva e Visualizar grafo.
6. Na seção Monitorar, você pode visualizar as previsões passadas e futuras de carga e de capacidade da política em relação aos valores reais. O grafo Carga exibe a previsão de carga e os valores reais para a métrica de carga escolhida. O grafo Capacidade mostra o número de tarefas previstas pela política. Também inclui o número real de tarefas iniciadas. A linha vertical separa os valores históricos das previsões futuras. Esses grafos ficam disponíveis logo após a criação da política.
7. (Opcional) Para alterar a quantidade de dados históricos exibidos no gráfico, escolha o valor de sua preferência no menu suspenso Período de avaliação na parte superior da página. O período de avaliação não transforma os dados desta página de maneira alguma. Ele altera apenas a quantidade de dados históricos exibidos.

Compare dados no grafo Carga

Cada linha horizontal representa um conjunto diferente de pontos de dados relatados em intervalos de uma hora:

1. A Carga real observada usa a estatística SUM da métrica de carga escolhida para exibir a carga horária total no passado.
2. A Carga prevista pela política exibe a previsão de carga horária. Essa previsão é baseada nas duas semanas anteriores de observações da carga real.

Compare dados no grafo Capacidade

Cada linha horizontal representa um conjunto diferente de pontos de dados relatados em intervalos de uma hora:

1. Número real de tarefas observado mostra a capacidade real do serviço do Amazon ECS no passado, o que depende de suas outras políticas de escalabilidade e do tamanho mínimo do grupo em vigor no período selecionado.
2. Capacidade prevista pela política exibe a capacidade básica que você pode esperar ter no início de cada hora quando a política estiver no modo de Prever e escalar.
3. Número de tarefas necessário inferido mostra o número ideal de tarefas em seu serviço para manter a métrica de escalabilidade no valor de destino que você escolheu.

4. Número mínimo de tarefas mostra o número mínimo de tarefas em seu serviço.
5. Capacidade máxima mostra o número máximo de tarefas em seu serviço.

Com o objetivo de calcular a capacidade necessária inferida, começamos supondo que cada tarefa é igualmente utilizada em um valor de destino especificado. Na prática, o número de tarefas não é utilizado igualmente. Porém, ao supor que a utilização é distribuída uniformemente entre tarefas, podemos fazer uma estimativa da probabilidade de quantidade de capacidade necessária. O requisito de número de tarefas é calculado de modo a ser inversamente proporcional à métrica de escalabilidade que você usou para sua política de escalabilidade preditiva. Em outras palavras, à medida que o número de tarefas aumenta, a métrica de escalabilidade diminui na mesma proporção. Por exemplo, se o número de tarefas dobra, a métrica de escalabilidade deve diminuir pela metade.

A fórmula da capacidade necessária inferida:

```
sum of (actualServiceUnits*scalingMetricValue)/(targetUtilization)
```

Por exemplo, pegamos `actualServiceUnits` (`10`) e `scalingMetricValue` (`30`) para determinada hora. Em seguida, pegamos a `targetUtilization` especificada em sua política de escalabilidade preditiva (`60`) e calculamos a capacidade necessária inferida para a mesma hora. Isso retorna um valor de `5`. Isso significa que cinco é a quantidade inferida de capacidade necessária para manter a capacidade em proporção inversa direta ao valor de destino da métrica de escala.

> **Note**
>
> Há várias alavancas disponíveis para você ajustar e melhorar a economia de custos e a disponibilidade de sua aplicação.
>
> - Utilize escalabilidade preditiva para a capacidade de linha de base e escalabilidade dinâmica para lidar com capacidade adicional. A escalabilidade dinâmica funciona independentemente da escalabilidade preditiva, aumentando e reduzindo a escala horizontalmente com base na utilização atual. Primeiro, o Amazon ECS calcula o número recomendado de tarefas para cada política de escalabilidade não programada. Em seguida, ele escala com base na política que fornece o maior número de tarefas.
> - Para permitir que a redução da escala horizontalmente ocorra quando a carga diminui, seu serviço deve sempre ter pelo menos uma política de escalabilidade dinâmica com a parte de redução da escala horizontalmente habilitada.
> - Você pode melhorar a performance da escalabilidade verificando se a capacidade mínima e máxima não são muito restritivas. Uma política com um número recomendado de

tarefas que não esteja dentro da faixa de capacidade mínima e máxima será impedida de aumentar e reduzir a escala horizontalmente.

Monitorar métricas de escalabilidade preditiva para Amazon ECS com o CloudWatch

É possível usar o Amazon CloudWatch para monitorar seus dados para escalabilidade preditiva. Uma política de escalabilidade preditiva coleta dados que são usados para prever sua carga futura. Os dados coletados são armazenados automaticamente no CloudWatch em intervalos regulares e podem ser usados para visualizar a performance da política ao longo do tempo. Você também pode criar alarmes do CloudWatch para receber uma notificação quando os indicadores de desempenho mudarem além dos limites definidos por você.

Visualizar dados históricos de previsão

Os dados de previsão de carga para uma política de escalabilidade preditiva podem ser visualizados no CloudWatch e podem ser úteis ao visualizar previsões em relação a outras métricas do CloudWatch em um único grafo. Você também pode ver tendências ao longo do tempo visualizando um intervalo de tempo mais amplo. É possível acessar até 15 meses de métricas históricas a fim de obter uma melhor visão do desempenho da política.

Para visualizar dados históricos de previsão usando o console do CloudWatch

1. Abra o console do CloudWatch, em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Metrics (Métricas) e, em seguida, All metrics (Todas as métricas).
3. Selecione o namespace da métrica Ajuste de escala automático da aplicação.
4. Escolha Previsões de carga de escalabilidade preditiva.
5. No campo de pesquisa, insira o nome da política de escalabilidade preditiva ou o nome do grupo do serviço do Amazon ECS e pressione a tecla Enter para filtrar os resultados.
6. Para criar um gráfico de uma métrica, marque a caixa de seleção ao lado da métrica. Para alterar o nome do gráfico, escolha o ícone de lápis. Para alterar o período, selecione um dos valores predefinidos ou escolha custom (personalizado). Para obter mais informações, consulte Representar uma métrica em gráficos no Guia do usuário do Amazon CloudWatch.
7. Para alterar a estatística, escolha a guia Métricas em gráfico. Escolha o cabeçalho de coluna ou um valor individual e, em seguida, escolha uma estatística diferente. Embora seja possível escolher qualquer estatística para cada métrica, nem todas as estatísticas são úteis para

as métricas PredictiveScalingLoadForecast. Por exemplo, as estatísticas Average (Média), Minimum (Mínimo) e Maximum (Máximo) são úteis, mas a estatística Sum (Soma) não.

8. Para adicionar outra métrica ao gráfico, em Browse (Procurar), escolha All (Todas), encontre a métrica específica e marque a caixa de seleção ao lado dela. Adicione até 10 métricas.
9. (Opcional) Para adicionar o gráfico a um painel do CloudWatch, escolha Actions (Ações), Add to dashboard (Adicionar ao painel).

Criar métricas de precisão usando matemática métrica

Com matemática métrica, você pode consultar várias métricas do CloudWatch e usar expressões matemáticas para criar novas séries temporais de acordo com essas métricas. Você pode visualizar as séries temporais resultantes no console do CloudWatch e adicioná-las aos painéis. Para obter mais informações sobre matemática métrica, consulte Usar matemática métrica no Guia do usuário do Amazon CloudWatch.

Usando matemática métricas, é possível representar graficamente de diferentes maneiras os dados gerados pelo ajuste de escala automático para a escalabilidade preditiva. Isso ajuda a monitorar o desempenho da política ao longo do tempo e a entender se é possível melhorar sua combinação de métricas.

Por exemplo, você pode usar uma expressão de matemática métrica para monitorar o mean absolute percentage error (MAPE – Erro percentual absoluto médio). A métrica MAPE ajuda a monitorar a diferença entre os valores previstos e os valores efetivos observados durante uma determinada janela de previsão. Mudanças no valor de MAPE podem indicar se o desempenho da política está se degradando ao longo do tempo conforme a natureza da sua aplicação muda. Um aumento no MAPE sinaliza uma lacuna maior entre os valores previstos e os valores efetivos.

Exemplo: expressão de matemática métrica

Para começar a usar esse tipo de gráfico, você pode criar uma expressão de matemática métrica como a mostrada no exemplo a seguir.

Em vez de uma só métrica, há uma matriz de estruturas de consulta de dados métricos para `MetricDataQueries`. Cada item em `MetricDataQueries` obtém uma métrica ou executa uma expressão matemática. O primeiro item, `e1`, é a expressão matemática. A expressão designada define o parâmetro `ReturnData` como `true`, resultando na produção de uma única série temporal. Para todas as outras métricas, o valor `ReturnData` é `false`.

No exemplo, a expressão designada usa os valores efetivos e previstos como entrada, e retorna a nova métrica (MAPE). `m1` é a métrica do CloudWatch que contém os valores efetivos de carga (supondo que a utilização de CPU seja a métrica de carga originalmente especificada para a política denominada `my-predictive-scaling-policy`). `m2` é a métrica do CloudWatch que contém os valores previstos de carga. A sintaxe matemática para a métrica MAPE é a seguinte:

média de (abs ((efetivo - previsto)/(efetivo)))

Visualizar suas métricas de precisão e definir alarmes

Para visualizar os dados da métrica de precisão, selecione a guia Metrics (Métricas) no console do CloudWatch. Nele, é possível representar graficamente os dados. Para obter mais informações, consulte Adição de uma expressão matemática a um gráfico do CloudWatch no Guia do usuário do Amazon CloudWatch.

Na seção Metrics (Métricas), você também pode criar um alarme com base em uma métrica que esteja monitorando. Enquanto estiver na guia Graphed metrics (Métricas representadas em gráficos), selecione o ícone Create alarm (Criar alarme) na coluna Actions (Ações). O ícone Create alarm (Criar alarme) é representado como um pequeno sino. Para obter mais informações e opções de notificação, consulte Criação de um alarme do CloudWatch com base em uma expressão matemática de métrica e Notificação de usuários sobre alterações de alarme no Guia do usuário do Amazon CloudWatch.

Como alternativa, você pode usar GetMetricData e PutMetricAlarm para realizar cálculos usando matemática métrica e criar alarmes com base na saída.

Usar ações programadas para substituir os valores de previsão para o Amazon ECS

Às vezes, você pode ter informações adicionais sobre seus futuros requisitos de aplicações que o cálculo de previsão não pode levar em conta. Por exemplo, os cálculos de previsão podem subestimar as tarefas necessárias para um evento de marketing futuro. Você pode usar ações programadas para substituir temporariamente a previsão durante períodos futuros. As ações programadas podem ser executadas de forma recorrente ou em uma data e hora específicas quando houver flutuações de demanda únicas.

Por exemplo, você pode criar uma ação programada com um número maior de tarefas que o previsto. Em runtime, o Amazon ECS atualiza o número mínimo de tarefas em seu serviço. Como o ajuste de escala preditivo otimiza o número de tarefas, uma ação programada com um número mínimo de tarefas maior do que os valores de previsão é respeitada. Isso impede que o número de

tarefas seja inferior ao esperado. Para interromper a substituição da previsão, use uma segunda ação agendada para retornar o número mínimo de tarefas à configuração original.

O procedimento a seguir descreve as etapas necessárias para substituir a previsão durante períodos futuros.

Tópicos

- Etapa 1: (Opcional) Analisar dados de séries temporais
- Etapa 2: Criar duas ações programadas

> **Important**
>
> Este tópico pressupõe que você esteja tentando substituir a previsão para escalar para uma capacidade maior do que a prevista. Se você precisar diminuir temporariamente o número de tarefas sem a interferência de uma política de ajuste de escala preditivo, use o modo somente previsão. Enquanto estiver no modo somente de previsão, a escala preditiva continuará a gerar previsões, mas não aumentará automaticamente o número de tarefas. Você então poderá monitorar a utilização dos recursos e diminuir manualmente o número de tarefas conforme necessário.

Etapa 1: (Opcional) Analisar dados de séries temporais

Comece analisando os dados de séries temporais de previsão. Essa é uma etapa opcional, mas é útil quando você deseja entender os detalhes da previsão.

1. Recuperar a previsão

   Após a criação da previsão, é possível consultar um período específico na previsão. O objetivo da consulta é obter uma visão completa dos dados de séries temporais para um período específico.

   Sua consulta pode incluir até dois dias de dados de previsão futura. Se você usa a escalabilidade preditiva há algum tempo, também pode acessar seus dados de previsão anteriores. No entanto, a duração máxima de tempo entre as horas inicial e final é de 30 dias.

   Para obter a previsão usando o comando da AWS CLI get-predictive-scaling-forecast, forneça os seguintes parâmetros no comando:

- Inclua o nome do cluster no parâmetro `resource-id`.
- Insira o nome da política no parâmetro `--policy-name`.
- Insira a hora de início no parâmetro `--start-time` para retornar apenas os dados de previsão para depois ou no horário especificado.
- Insira a hora de término no parâmetro `--end-time` para retornar apenas os dados de previsão para antes do horário especificado.

```
aws application-autoscaling get-predictive-scaling-forecast \
    --service-namespace ecs \
    --resource-id service/MyCluster/test \
    --policy-name cpu40-predictive-scaling-policy \
    --scalable-dimension ecs:service:DesiredCount \
    --start-time "2021-05-19T17:00:00Z" \
    --end-time "2021-05-19T23:00:00Z"
```

Se bem-sucedido, o comando retornará uma resposta semelhante à seguinte.

```
{
    "LoadForecast": [
        {
            "Timestamps": [
                "2021-05-19T17:00:00+00:00",
                "2021-05-19T18:00:00+00:00",
                "2021-05-19T19:00:00+00:00",
                "2021-05-19T20:00:00+00:00",
                "2021-05-19T21:00:00+00:00",
                "2021-05-19T22:00:00+00:00",
                "2021-05-19T23:00:00+00:00"
            ],
            "Values": [
                153.0655799339254,
                128.8288551285919,
                107.1179447150675,
                197.3601844551528,
                626.4039934516954,
                596.9441277518481,
                677.9675713779869
            ],
            "MetricSpecification": {
```

```
                "TargetValue": 40.0,
                "PredefinedMetricPairSpecification": {
                    "PredefinedMetricType": "ASGCPUUtilization"
                }
            }
        }
    ],
    "CapacityForecast": {
        "Timestamps": [
            "2021-05-19T17:00:00+00:00",
            "2021-05-19T18:00:00+00:00",
            "2021-05-19T19:00:00+00:00",
            "2021-05-19T20:00:00+00:00",
            "2021-05-19T21:00:00+00:00",
            "2021-05-19T22:00:00+00:00",
            "2021-05-19T23:00:00+00:00"
        ],
        "Values": [
            2.0,
            2.0,
            2.0,
            2.0,
            4.0,
            4.0,
            4.0
        ]
    },
    "UpdateTime": "2021-05-19T01:52:50.118000+00:00"
}
```

A resposta inclui duas previsões: `LoadForecast` e `CapacityForecast`. `LoadForecast` mostra a previsão de carga horária. `CapacityForecast` mostra os valores de previsão para a capacidade que é necessária em uma base horária para lidar com a carga prevista enquanto mantém um `TargetValue` de 40,0 (40% de utilização média da CPU).

2. Identificar o período-alvo

   Identifique a hora ou horas em que a flutuação de demanda única deverá ocorrer. Lembre-se de que as datas e os horários mostrados na previsão estão em UTC.

Etapa 2: Criar duas ações programadas

Em seguida, crie duas ações programadas para um período específico em que sua aplicação terá uma carga maior do que a prevista. Por exemplo, se você tiver um evento de marketing que irá direcionar o tráfego para seu site por um período limitado, poderá programar uma ação única para atualizar a capacidade mínima quando ele começar. Em seguida, agende outra ação para retornar a capacidade mínima para a configuração original quando o evento terminar.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, escolha o cluster.
3. Na página de detalhes do cluster, na seção Serviços, escolha o serviço.

   A página de detalhes do serviço é exibida.
4. Escolha Ajuste de escala automático do serviço.

   A página de políticas será exibida.
5. Escolha Ações programadas e, em seguida, escolha Criar.

   A página Criar ação de programação é exibida.
6. Em Nome da ação, insira um nome exclusivo.
7. Em Time zone (Fuso horário), escolha um fuso horário.

   Todos os fusos horários listados são do banco de dados de fuso horário da IANA. Para obter mais informações, consulte a Lista de fusos horários no banco de dados de FH.
8. Em Hora de início, insira a Data e a Hora em que a ação começa.
9. Em Recurrence (Recorrência), escolha Once (Uma vez).
10. Em Ajustes de tarefas, para Mínimo, insira um valor inferior ou igual ao número máximo de tarefas.
11. Escolha Criar ação programada.

    A página de políticas será exibida.
12. Configure uma segunda ação programada para retornar o número mínimo de tarefas para a configuração original no final do evento. A escalabilidade preditiva pode escalar o número de tarefas somente quando o valor definido para Minínimo é menor que os valores da previsão.

Para criar duas ações programadas para eventos únicos (AWS CLI)

Para usar o AWS CLI para criar as ações programadas, use o comando put-scheduled-update-group-action.

Por exemplo, vamos definir uma programação que mantenha uma capacidade mínima de três instâncias em 19 de maio às 17h por oito horas. Os comandos a seguir mostram como implementar esse cenário.

O primeiro comando put-scheduled-update-group-action instrui o Amazon EC2 Auto Scaling a atualizar a capacidade mínima do grupo do Auto Scaling especificado às 17h UTC em 19 de maio de 2021.

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-event-start \
  --auto-scaling-group-name my-asg --start-time "2021-05-19T17:00:00Z" --minimum-capacity 3
```

O segundo comando instrui o Amazon EC2 Auto Scaling a definir a capacidade mínima do grupo como um à 1h da manhã UTC em 20 de maio de 2021.

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-event-end \
  --auto-scaling-group-name my-asg --start-time "2021-05-20T01:00:00Z" --minimum-capacity 1
```

Após você adicionar essas ações programadas ao grupo do Auto Scaling, o Amazon EC2 Auto Scaling fará o seguinte:

- Às 17h UTC em 19 de maio de 2021, a primeira ação programada é executada. Se o grupo tiver menos de três instâncias, ele será expandido para três instâncias. Durante esse período e nas próximas oito horas, o Amazon EC2 Auto Scaling poderá continuar a aumentar a escala na horizontal se a capacidade prevista for maior do que a capacidade real ou se houver uma política de escalabilidade dinâmica em vigor.
- À 1h da manhã UTC em 20 de maio de 2021, a segunda ação programada é executada. Isso retorna a capacidade mínima para sua configuração original no final do evento.

Escalabilidade com base em programações recorrentes

Para substituir a previsão para o mesmo período de tempo todas as semanas, crie duas ações programadas e forneça a lógica de hora e data usando uma expressão cron.

A expressão cron consiste em cinco campos separados por espaços: [Minute] [Hour] [Day_of_Month] [Month_of_Year] [Day_of_Week]. Os campos podem conter quaisquer valores permitidos, incluindo caracteres especiais.

Por exemplo, esta expressão cron executa a ação todas as terças-feiras às 6h30. O asterisco é usado como um curinga para corresponder a todos os valores de um campo.

```
30 6 * * 2
```

Consulte também

Para obter mais informações sobre como gerenciar ações programadas, consulte Usar ações programadas para escalar os serviços do Amazon ECS.

Política avançada de escalabilidade preditiva usando métricas personalizadas para o Amazon ECS

É possível usar métricas predefinidas ou personalizadas em uma política de escalabilidade preditiva. As métricas personalizadas são úteis quando métricas predefinidas (como CPU, memória etc.) não são suficientes para descrever suficientemente a carga da aplicação.

Ao criar uma política de escalabilidade preditiva com métricas personalizadas, você pode especificar outras métricas do CloudWatch fornecidas pela AWS. Como alternativa, você pode especificar métricas que define e publica. Você também pode usar a matemática de métricas para agregar e transformar métricas existentes em uma nova série temporal que a AWS não rastreia automaticamente. Um exemplo é combinar valores em seus dados calculando novas somas ou médias, o que é chamado de agregação. Os dados resultantes são chamados de um agregado.

A seção a seguir contém as práticas recomendados e exemplos de como sstruturar o JSON para a política.

Pré-requisitos

Para adicionar métricas personalizadas à política de escalação preditiva, você deve ter as permissões `cloudwatch:GetMetricData`.

Para especificar suas próprias métricas em vez de usar as métricas que a AWS fornecer, você deve primeiro publicá-las no CloudWatch. Para mais informações, consulte Publishing custom metrics (Publicar métricas personalizadas) no Guia do usuário do Amazon CloudWatch.

Se publicar suas próprias métricas, certifique-se de publicar os pontos de dados com uma frequência mínima de cinco minutos. Os pontos de dados são recuperados do CloudWatch com base na

duração do período necessário. Por exemplo, a especificação da métrica de carga usa métricas por hora para medir a carga em sua aplicação. O CloudWatch usa seus dados de métrica publicados para fornecer um único valor de dados para qualquer período de uma hora, agregando todos os pontos de dados com a data/hora que caem dentro de cada período de uma hora.

Práticas recomendadas

As seguintes práticas recomendadas podem ajudar no uso mais eficaz de métricas personalizadas:

- A métrica mais útil para a especificação da métrica de carga é uma métrica que representa a carga em um grupo do Auto Scaling como um todo.
- A métrica mais útil para escalar a especificação da métrica de escalabilidade é um throughput médio ou métrica de utilização por tarefa.
- A utilização visada deve corresponder ao tipo de métrica de escalabilidade. Para uma configuração de política que usa utilização da CPU, isso é um percentual desejado, por exemplo.
- Se essas recomendações não forem seguidas, provavelmente os valores futuros previstos da série temporal estarão incorretos. Para validar se os dados estão corretos, você pode visualizar os valores previstos no console. Como alternativa, depois de criar sua política de escalabilidade preditiva, inspecione os objetos `LoadForecast` retornados por uma chamada para a API GetPredictiveScalingForecast.
- Recomendamos enfaticamente configurar a escalabilidade preditiva no modo apenas previsão para que você possa avaliar a previsão antes que a escalabilidade preditiva comece a modificar ativamente a escala.

Limitações

- Você pode consultar pontos de dados de até 10 métricas em uma especificação métrica.
- Para os propósitos desse limite, uma expressão conta como uma métrica.

Solucionar problemas com uma política de escalabilidade preditiva com métricas personalizadas

Se ocorrer um problema ao usar métricas personalizadas, recomendamos fazer o seguinte:

- Se você encontrar um problema em uma implantação azul/verde enquanto usa uma expressão de pesquisa, certifique-se de criar uma expressão de pesquisa que procure uma correspondência parcial, e não uma correspondência exata. Verifique também se a consulta está encontrando apenas grupos do Auto Scaling em execução na aplicação específica. Para mais informações

sobre essa sintaxe de expressão de pesquisa, consulte CloudWatch search expression syntax (Sintaxe de expressão de pesquisa do CloudWatch) no Guia do usuário do Amazon CloudWatch.

- O comando put-scaling-policy valida uma expressão quando você cria sua política de dimensionamento. No entanto, existe a possibilidade de que esse comando não identifique a causa exata dos erros detectados. Para corrigir os problemas, solucione os erros que você recebe em uma resposta de uma solicitação para o comando get-metric-data. Você também pode solucionar problemas da expressão no console do CloudWatch.
- Você deve especificar `false` para `ReturnData` se `MetricDataQueries` especificar a função SEARCH() (BUSCAR) por conta própria sem uma função matemática como SUM() (SOMA). Isso ocorre porque as expressões de pesquisa podem retornar várias séries temporais, e uma especificação métrica baseado em uma expressão pode retornar apenas uma série temporal.
- Todas as métricas envolvidas em uma expressão de pesquisa devem ter a mesma resolução.

Construir o JSON para métricas personalizadas de escalabilidade preditiva com o Amazon ECS

A seção a seguir contém exemplos de como configurar escalação preditiva para consultar dados do CloudWatch. Há dois métodos diferentes de configurar essa opção, e o método escolhido afeta qual será o formato usado para estruturar JSON para a política de escalação preditiva. Quando você usa matemática de métricas, o formato do JSON varia ainda mais com base na matemática de métrica que está sendo aplicada.

1. Para criar uma política que obtenha dados diretamente de outras métricas do CloudWatch fornecidas pela AWS ou das métricas que você publica no CloudWatch, consulte Exemplo de política de escalabilidade preditiva com métricas personalizadas de carga e escalabilidade usando a AWS CLI.

Exemplo de política de escalabilidade preditiva com métricas personalizadas de carga e escalabilidade usando a AWS CLI

Para criar uma política de escalação preditiva com métricas personalizadas de carga e dimensionamento com a AWS CLI, armazene os argumentos para a `--predictive-scaling-configuration` em um arquivo JSON denominado `config.json`.

Você começa a adicionar métricas personalizadas substituindo os valores substituíveis no exemplo a seguir por suas métricas e sua meta de utilização.

```
{
```

```
  "MetricSpecifications": [
    {
      "TargetValue": 50,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "scaling_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "MyUtilizationMetric",
                "Namespace": "MyNameSpace",
                "Dimensions": [
                  {
                    "Name": "MyOptionalMetricDimensionName",
                    "Value": "MyOptionalMetricDimensionValue"
                  }
                ]
              },
              "Stat": "Average"
            }
          }
        ]
      },
      "CustomizedLoadMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "MyLoadMetric",
                "Namespace": "MyNameSpace",
                "Dimensions": [
                  {
                    "Name": "MyOptionalMetricDimensionName",
                    "Value": "MyOptionalMetricDimensionValue"
                  }
                ]
              },
              "Stat": "Sum"
            }
          }
        ]
      }
    }
```

```
  ]
}
```

Para obter mais informações, consulte MetricDataQuery na Amazon EC2 Auto Scaling API Reference (Referência de API do Amazon EC2 Auto Scaling).

> **Note**
>
> Veja a seguir alguns recursos adicionais que podem ajudar você a encontrar nomes de métricas, namespaces, dimensões e estatísticas para as métricas do CloudWatch:
>
> - Para mais informações sobre as métricas disponíveis para produtos da AWS, consulte Produtos da AWS que publicam métricas do CloudWatch no Guia do usuário do Amazon CloudWatch.
> - Para obter os valores exatos de nome da métrica, namespace e dimensões (se aplicável) para uma métrica do CloudWatch com a AWS CLI, consulte list-metrics.

Para criar essa política, execute o comando put-scaling-policy usando como entrada o arquivo JSON, como demonstrado no exemplo a seguir.

```
aws application-autoscaling put-scaling-policy --policy-name my-predictive-scaling-
policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
```

Se bem-sucedido, esse comando gerará o nome do recurso da Amazon (ARN) da política.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-predictive-scaling-policy",
  "Alarms": []
}
```

Usar expressões de matemática métrica

A seção a seguir fornece informações sobre o uso matemática métrica com políticas de escalabilidade preditiva em sua política.

Noções básicas de matemática métrica

Se tudo o que deseja fazer é agregar dados métricos existentes, a matemática métrica do CloudWatch poupa o esforço e o custo de publicar outra métrica no CloudWatch. Você pode usar qualquer métrica que a AWS fornece, e também pode usar métricas definidas como parte de suas aplicações.

Para obter mais informações, consulte Usar matemática métrica no Guia do usuário do Amazon CloudWatch.

Se você optar por usar uma expressão matemática métrica em sua política de escalabilidade preditiva, considere os seguintes pontos:

- As operações matemáticas métricas usam os pontos de dados da combinação exclusiva de nome da métrica, namespace e pares de métricas de chaves-valor da dimensão.
- Você pode usar qualquer operador aritmético (+ - * / ^), função estatística (como AVG ou SUM) ou outra função compatível com o CloudWatch.
- Você pode usar as métricas e os resultados de outras expressões matemáticas nas fórmulas da expressão matemática.
- Suas expressões matemáticas métricas podem ser compostas de agregações diferentes. No entanto, uma prática recomendada para o resultado final da agregação é usar `Average` para a métrica de escalabilidade e `Sum` para a métrica de carga.
- Qualquer expressão usada em uma especificação de métrica deve eventualmente retornar uma única série temporal.

Para usar matemática métrica, faça o seguinte:

- Escolha uma ou mais métricas do CloudWatch. Em seguida, crie a expressão. Para obter mais informações, consulte Usar matemática métrica no Guia do usuário do Amazon CloudWatch.
- Verifique se a expressão matemática métrica é válida usando o console do CloudWatch ou a API do CloudWatch GetMetricData.

Exemplo de política de escalação preditiva que combina métricas por meio da matemática de métricas (AWS CLI)

Às vezes, ao invés de especificar a métrica diretamente, talvez seja necessário processar seus dados de alguma forma, primeiramente. Por exemplo, você pode ter uma aplicação que extrai o

trabalho de uma fila do Amazon SQS e talvez queira usar o número de itens na fila como critério para escalabilidade preditiva. O número de mensagens na fila não define unicamente o número necessário de instâncias. Portanto, é necessário mais trabalho para criar uma métrica que possa ser usada para calcular a lista de pendências por instância.

Veja a seguir um exemplo de política de escalabilidade preditiva para esse cenário. Ele especifica métricas de escalabilidade e carga baseadas na métrica `ApproximateNumberOfMessagesVisible` do Amazon SQS, que é o número de mensagens disponíveis para recuperação da fila. Ele também usa a métrica `GroupInServiceInstances` do Amazon EC2 Auto Scaling e uma expressão matemática para calcular a lista de pendências por instância para a métrica de escalabilidade.

```
aws application-autoscaling put-scaling-policy --policy-name my-sqs-custom-metrics-
policy \
  --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
  --service-namespace ecs \
  --resource-id service/MyCluster/test \
  "MetricSpecifications": [
    {
      "TargetValue": 100,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Label": "Get the queue size (the number of messages waiting to be
 processed)",
            "Id": "queue_size",
            "MetricStat": {
              "Metric": {
                "MetricName": "ApproximateNumberOfMessagesVisible",
                "Namespace": "AWS/SQS",
                "Dimensions": [
                  {
                    "Name": "QueueName",
                    "Value": "my-queue"
                  }
                ]
              },
              "Stat": "Sum"
            },
            "ReturnData": false
          },
```

```
        {
          "Label": "Get the group size (the number of running instances)",
          "Id": "running_capacity",
          "MetricStat": {
            "Metric": {
              "MetricName": "GroupInServiceInstances",
              "Namespace": "AWS/AutoScaling",
              "Dimensions": [
                {
                  "Name": "AutoScalingGroupName",
                  "Value": "my-asg"
                }
              ]
            },
            "Stat": "Sum"
          },
          "ReturnData": false
        },
        {
          "Label": "Calculate the backlog per instance",
          "Id": "scaling_metric",
          "Expression": "queue_size / running_capacity",
          "ReturnData": true
        }
      ]
    },
    "CustomizedLoadMetricSpecification": {
      "MetricDataQueries": [
        {
          "Id": "load_metric",
          "MetricStat": {
            "Metric": {
              "MetricName": "ApproximateNumberOfMessagesVisible",
              "Namespace": "AWS/SQS",
              "Dimensions": [
                {
                  "Name": "QueueName",
                  "Value": "my-queue"
                }
              ],
            },
            "Stat": "Sum"
          },
          "ReturnData": true
```

```
          }
        ]
      }
    }
  ]
}
```

O exemplo retorna o ARN da política.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-sqs-custom-metrics-policy",
  "Alarms": []
}
```

Interconexão de serviços do Amazon ECS

As aplicações executadas em tarefas do Amazon ECS muitas vezes precisam receber conexões da Internet ou se conectar a outras aplicações executadas nos serviços do Amazon ECS. Se você precisar de conexões externas da Internet, recomendamos usar o Elastic Load Balancing. Para obter mais informações sobre balanceamento de carga integrado, consulte the section called “Uso do balanceamento de carga para distribuir o tráfego de serviço”.

Se você precisar de uma aplicação para se conectar a outras aplicações executadas nos serviços do Amazon ECS, o Amazon ECS oferecerá as maneiras a seguir de fazer isso sem um balanceador de carga:

- Amazon ECS Service Connect

  Recomendamos o Service Connect, que fornece a configuração do Amazon ECS para descoberta de serviços, conectividade e monitoramento de tráfego. Com o Service Connect, as aplicações podem usar nomes curtos e portas padrão para se conectar aos serviços do Amazon ECS no mesmo cluster e em outros clusters, inclusive entre VPCs na mesma Região da AWS.

  Ao usar o Service Connect, o Amazon ECS gerencia todas as partes da descoberta de serviços: criar os nomes que podem ser descobertos, gerenciar dinamicamente as entradas de cada tarefa à medida que as tarefas começam e são interrompidas, executar um agente em cada tarefa configurada para descobrir os nomes. Sua aplicação pode pesquisar os nomes usando a funcionalidade padrão para nomes do DNS e fazendo conexões. Caso a aplicação já faça isso, não será necessário modificá-la para usar o Service Connect.

Você fornece a configuração completa dentro de cada serviço e definição de tarefa. O Amazon ECS gerencia as alterações nessa configuração em cada implantação de serviço, para garantir que todas as tarefas em uma implantação se comportem da mesma maneira. Por exemplo, um problema comum do DNS como descoberta de serviços é controlar uma migração. Se você alterar um nome do DNS de modo a direcionar para os novos endereços IP substitutos, poderá levar o tempo máximo de TTL antes que todos os clientes comecem a usar o novo serviço. Com o Service Connect, a implantação do cliente atualiza a configuração substituindo as tarefas do cliente. Você pode configurar o disjuntor de implantação e outras configurações de implantação para afetar as alterações do Service Connect da mesma forma que qualquer outra implantação.

Para obter mais informações, consulte Uso do Service Connect para conectar serviços do Amazon ECS com nomes abreviados.

- Descoberta de serviço do Amazon ECS

  Outra abordagem para a comunicação entre serviços é a comunicação direta por meio da descoberta de serviços. Nessa abordagem, você pode usar a integração da descoberta de serviços do AWS Cloud Map com o Amazon ECS. Usando a descoberta de serviços, o Amazon ECS sincroniza a lista de tarefas executadas com o AWS Cloud Map, que mantém um nome de host do DNS que resolve os endereços IP internos de uma ou mais tarefas desse serviço específico. Outros serviços na Amazon VPC podem usar esse nome de host do DNS para enviar tráfego diretamente a outro contêiner usando o endereço IP interno.

  Essa abordagem de comunicação entre serviços fornece baixa latência. Não há componentes extras entre os contêineres. O tráfego viaja diretamente de um contêiner para o outro.

  Essa abordagem é adequada ao usar o modo de rede `awsvpc`, em que cada tarefa tem seu próprio endereço IP exclusivo. A maioria dos softwares é compatível apenas com o uso de registros `A` do DNS, que resolvem diretamente os endereços IP. Ao usar o modo de rede `awsvpc`, o endereço IP de cada tarefa é um registro `A`. No entanto, se você estiver usando o modo de rede `bridge`, vários contêineres podem estar compartilhando o mesmo endereço IP. Além disso, os mapeamentos dinâmicos de portas fazem com que os contêineres recebam números de porta aleatoriamente nesse único endereço IP. A essa altura, um registro `A` não é mais suficiente para a descoberta de serviços. Você também deve usar um registro `SRV`. Esse tipo de registro pode rastrear endereços IP e números de porta, mas exige que você configure as aplicações adequadamente. Algumas aplicações pré-criadas que você usa podem não oferecer suporte a registros `SRV`.

Outra vantagem do modo de rede `awsvpc` é ter um grupo de segurança exclusivo para cada serviço. Você pode configurar esse grupo de segurança para permitir conexões de entrada somente dos serviços upstream específicos que precisam se comunicar com esse serviço.

A principal desvantagem da comunicação direta entre serviços usando a descoberta de serviços é que você deve implementar uma lógica extra para ter repetições e lidar com falhas de conexão. Os registros DNS têm um período de vida útil (TTL) que controla por quanto tempo são armazenados em cache. Leva algum tempo para que o registro DNS seja atualizado e o cache expire para que as aplicações possam obter a versão mais recente do registro DNS. Portanto, sua aplicação pode acabar resolvendo o registro DNS de modo a direcionar para outro contêiner que não está mais lá. A aplicação precisa processar repetições e ter uma lógica para ignorar backends problemáticos.

Para obter mais informações, consulte Uso da descoberta de serviços para conectar serviços do Amazon ECS com nomes DNS.

- Amazon VPC Lattice

  O Amazon VPC Lattice é um serviço gerenciado de rede de aplicações usado pelos clientes do Amazon ECS para observar, proteger e monitorar aplicações criadas em serviços de computação, VPCs e contas da AWS sem precisar modificar seu código.

  O VPC Lattice usa grupos de destino, que são uma coleção de recursos computacionais. Esses destinos executam a aplicação ou serviço e podem ser instâncias do Amazon EC2, endereços IP, funções do Lambda e Application Load Balancers. Ao associarem seus serviços do Amazon ECS a um grupo de destino do VPC Lattice, os clientes agora podem habilitar tarefas do Amazon ECS como destinos IP no VPC Lattice. O Amazon ECS registra automaticamente as tarefas no grupo de destino do VPC Lattice quando as tarefas do serviço registrado são iniciadas.

  Para obter mais informações, consulte Use o Amazon VPC Lattice para conectar, observar e proteger seus serviços do Amazon ECS.

Tabela de compatibilidade de modo de rede

A tabela a seguir inclui a compatibilidade entre essas opções e os modos de rede de tarefas. Na tabela, “client” (cliente) refere-se à aplicação que está fazendo as conexões de dentro de uma tarefa do Amazon ECS.

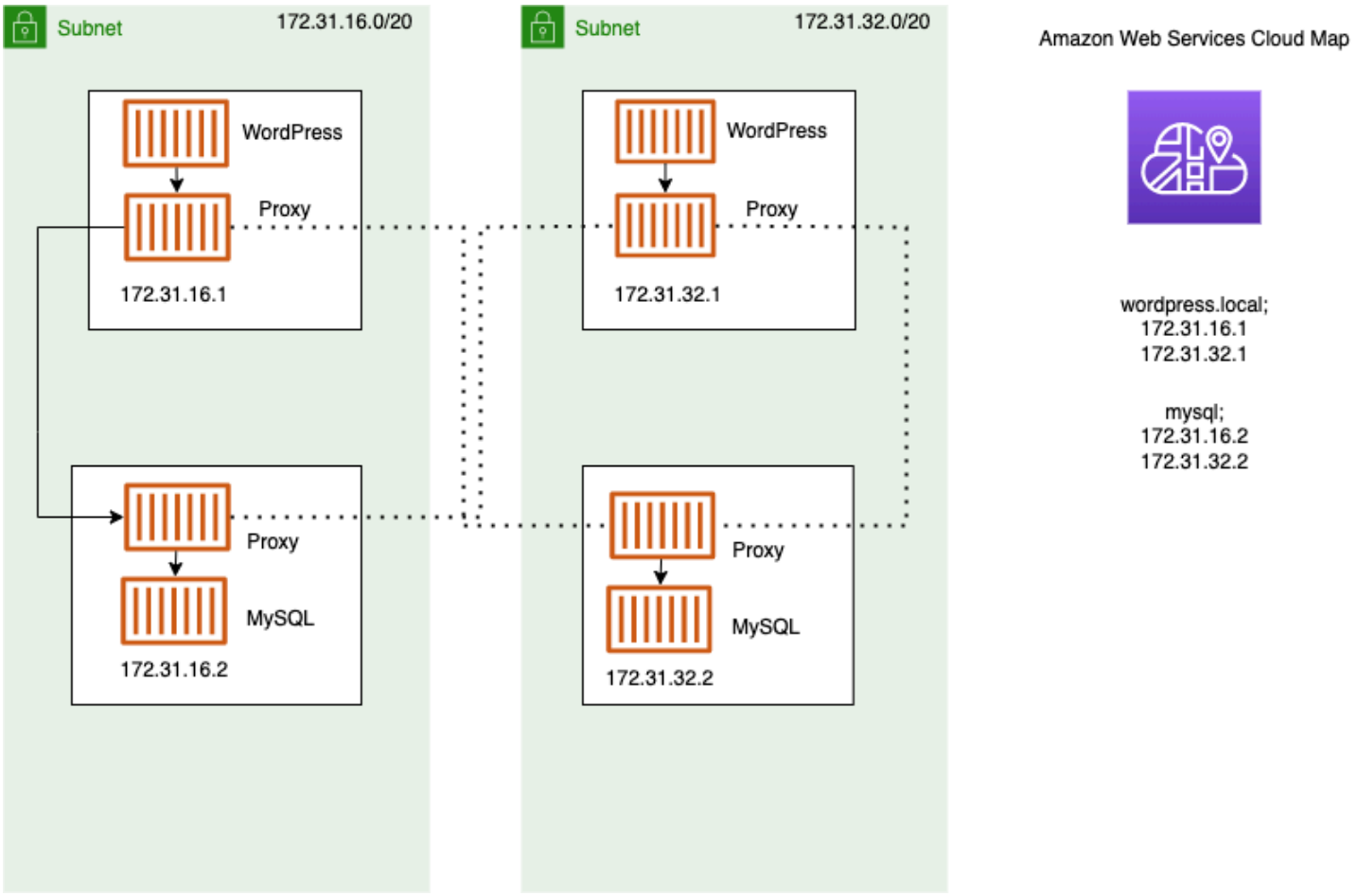
Opções de interconexão	Conectado	**awsvpc**	Host
Descoberta de serviço	sim, mas exige que os clientes conheçam os registros SRV no DNS sem `hostPort`.	sim	sim, mas exige que os clientes conheçam os registros SRV no DNS sem `hostPort`.
Service Connect	sim	sim	não
VPC Lattice	sim	sim	sim

Uso do Service Connect para conectar serviços do Amazon ECS com nomes abreviados

O Amazon ECS Service Connect fornece gerenciamento da comunicação entre serviços conforme a configuração do Amazon ECS. Ele cria uma descoberta de serviços e uma malha de serviços no Amazon ECS. Isso fornece a configuração completa dentro de cada serviço que você gerencia por implantações de serviço, uma maneira unificada de consultar os serviços nos namespaces que não dependem da configuração de DNS da VPC, além de métricas e dos logs padronizados para monitorar todas as aplicações. O Service Connect apenas realiza a interconexão de serviços.

O diagrama a seguir mostra um exemplo de rede do Service Connect com duas sub-redes na VPC e dois serviços. Um serviço do cliente que executa o WordPress com uma tarefa em cada sub-rede. Um serviço de servidor que executa o MySQL com uma tarefa em cada sub-rede. Ambos os serviços são altamente disponíveis e resilientes a problemas de tarefas e zonas de disponibilidade, pois cada serviço executa várias tarefas espalhadas por duas sub-redes. As setas sólidas mostram uma conexão do WordPress com o MySQL. Por exemplo, um comando da CLI `mysql --host=mysql` executado de dentro do contêiner do WordPress na tarefa com o endereço IP `172.31.16.1`. O comando usa o nome curto `mysql` na porta padrão do MySQL. O nome e a porta se conectam ao proxy do Service Connect na mesma tarefa. O proxy na tarefa do WordPress usa balanceamento de carga round-robin e qualquer informação de falha anterior na detecção de valores discrepantes para escolher a qual tarefa do MySQL se conectar. Conforme mostrado pelas setas sólidas no diagrama, o proxy se conecta ao segundo proxy na tarefa do MySQL com o endereço IP `172.31.16.2`. O segundo proxy se conecta ao servidor MySQL local na mesma tarefa. Ambos os proxies relatam o desempenho da conexão que é visível em gráficos nos consoles Amazon ECS e Amazon

CloudWatch para que você possa obter métricas de performance de todos os tipos de aplicações da mesma forma.



Os termos a seguir são usados com o Service Connect.

nome da porta

A configuração de definição de tarefa do Amazon ECS que atribui um nome a um mapeamento de porta específico. Essa configuração só é usada pelo Amazon ECS Service Connect.

alias do cliente

A configuração do serviço do Amazon ECS que atribui o número da porta usada no endpoint. Além disso, o alias do cliente pode atribuir o nome DNS do endpoint, substituindo o nome da descoberta. Se um nome de descoberta não for fornecido no serviço do Amazon ECS, o nome do alias do cliente substituirá o nome da porta como o nome do endpoint. Para ver exemplos de endpoints, consulte a definição de endpoint. Vários aliases de cliente podem ser atribuídos a um serviço do Amazon ECS. Essa configuração só é usada pelo Amazon ECS Service Connect.

nome da descoberta

O nome intermediário opcional que você pode criar para uma porta especificada na definição da tarefa. Esse nome é usado para criar um serviço do AWS Cloud Map. Se esse nome não for fornecido, será usado o nome da porta da definição da tarefa. Vários nomes de descoberta podem ser atribuídos a uma porta específica de um serviço do Amazon ECS. Essa configuração só é usada pelo Amazon ECS Service Connect.

Os nomes dos serviços do AWS Cloud Map devem ser exclusivos dentro de um namespace. Devido a essa limitação, você só pode ter uma configuração do Service Connect sem um nome de descoberta para uma definição de tarefa específica em cada namespace.

endpoint

O URL para se conectar a uma API ou site. O URL contém o protocolo, um nome DNS e a porta. Para obter mais informações sobre endpoints em geral, consulte endpoint no Glossário da AWS na Referência geral da Amazon Web Services.

O Service Connect cria endpoints que se conectam aos serviços do Amazon ECS e configura as tarefas nos serviços do Amazon ECS para se conectarem aos endpoints. O URL contém o protocolo, um nome DNS e a porta. Você seleciona o protocolo e o nome da porta na definição da tarefa, pois a porta deve corresponder à aplicação que está dentro da imagem do contêiner. No serviço, você seleciona cada porta pelo nome e pode atribuir o nome DNS. Se você não especificar um nome DNS na configuração do serviço do Amazon ECS, será usado por padrão o nome da porta da definição da tarefa. Por exemplo, um endpoint do Service Connect pode ser `http://blog:80`, `grpc://checkout:8080` ou `http://_db.production.internal:99`.

Serviço do Service Connect

A configuração de um único endpoint em um serviço do Amazon ECS. Isso faz parte da configuração do Service Connect, que consiste em uma única linha na Service Connect and discovery name configuration (Configuração do Service Connect e do nome da descoberta) no console ou em um objeto da lista `services` da configuração JSON de um serviço do Amazon ECS. Essa configuração só é usada pelo Amazon ECS Service Connect.

Para obter mais informações, consulte ServiceConnectService na Referência de API do Amazon Elastic Container Service.

namespace

O nome do recurso da Amazon (ARN) abreviado ou completo do namespace AWS Cloud Map para uso com o Service Connect. O namespace deve estar na mesma Região da AWS que

o serviço e o cluster do Amazon ECS. O tipo de namespace no AWS Cloud Map não afeta o Service Connect.

O Service Connect usa o namespace AWS Cloud Map como um agrupamento lógico de tarefas do Amazon ECS que se comunicam entre si. Cada serviço do Amazon ECS pode pertencer a apenas um namespace. Os serviços em um namespace podem ser distribuídos entre diferentes clusters do Amazon ECS dentro da mesma Região da AWS na mesma Conta da AWS. É possível organizar serviços de forma flexível, seguindo qualquer critério desejado.

serviço de cliente

Um serviço que executa uma aplicação cliente de rede. Esse serviço deve ter um namespace configurado. Cada tarefa do serviço pode descobrir e se conectar a todos os endpoints do namespace por meio de um contêiner do proxy do Service Connect.

Se algum dos seus contêineres na tarefa precisar se conectar a um endpoint em um serviço em um namespace, escolha um serviço de cliente. Se uma aplicação de frontend, de proxy reverso ou de balanceador de carga receber tráfego externo por meio de outros métodos, como o Elastic Load Balancing, ela poderá usar esse tipo de configuração do Service Connect.

serviço cliente-servidor

Um serviço do Amazon ECS que executa uma aplicação de rede ou de serviço Web. Esse serviço deve ter um namespace e pelo menos um endpoint configurados. Cada tarefa do serviço pode ser acessada por meio dos endpoints. O contêiner do proxy do Service Connect recebe o nome e a porta do endpoint para direcionar o tráfego para os contêineres da aplicação na tarefa.

Se algum dos contêineres expuser e receber tráfego de rede em uma porta, escolha um serviço cliente-servidor. Essas aplicações não precisam se conectar a outros serviços cliente-servidor no mesmo namespace, mas a configuração do cliente é necessária. Esse tipo de configuração do Service Connect pode ser utilizado por backend, middleware, camada de negócios ou pela maioria dos microsserviços. Se você quiser que uma aplicação de frontend, de proxy reverso ou de balanceador de carga receba tráfego de outros serviços configurados com o Service Connect no mesmo namespace, esses serviços deverão usar esse tipo de configuração do Service Connect.

O recurso Service Connect cria uma rede virtual de serviços relacionados. A mesma configuração de serviços pode ser usada em vários namespaces diferentes para executar conjuntos de aplicações independentes, mas idênticas. O Service Connect define o contêiner do proxy no serviço do Amazon ECS. Dessa forma, a mesma definição de tarefa pode ser usada para executar aplicações idênticas

em namespaces diferentes com diferentes configurações do Service Connect. Cada tarefa realizada pelo serviço executa um contêiner proxy na tarefa.

O Service Connect é adequado para conexões entre serviços do Amazon ECS dentro do mesmo namespace. Para as seguintes aplicações, você precisa usar um método de interconexão adicional para se conectar a um serviço do Amazon ECS que esteja configurado com o Service Connect:

- Tarefas que estão configuradas em outros namespaces
- Tarefas que não estão configuradas para o Service Connect
- Outras aplicações fora do Amazon ECS

Essas aplicações podem se conectar por meio do proxy do Service Connect, mas não conseguem resolver os nomes dos endpoints do Service Connect.

Para que essas aplicações resolvam os endereços IP das tarefas do Amazon ECS, é necessário usar outro método de interconexão.

Tópicos

- Preços
- Componentes do Amazon ECS Service Connect
- Visão geral da configuração do Amazon ECS Service Connect
- Criptografia do tráfego do Amazon ECS Service Connect
- Configuração do Amazon ECS Service Connect com a AWS CLI

Preços

- O preços do Amazon ECS Service Connect depende de você usar ou não a infraestrutura do AWS Fargate ou do Amazon EC2 para hospedar workloads em contêineres. Ao usar o Amazon ECS no AWS Outposts, os preços seguem o mesmo modelo de quando você está usando o Amazon EC2 diretamente. Para obter mais informações, consulte Preços do Amazon ECS.
- O uso do Amazon ECS Service Connect não gera custos adicionais.
- O uso do AWS Cloud Map é totalmente gratuito quando usado pelo Service Connect.
- Os clientes pagam pelos recursos computacionais usados pelo Amazon ECS Service Connect, incluindo vCPU e memória. Como o agente do Amazon ECS Service Connect é executado dentro de uma tarefa do cliente, sua execução não gera custos adicionais. Os recursos de tarefa são compartilhados entre a workload do cliente e o agente do Amazon ECS Service Connect.

- Quando os clientes usam a funcionalidade de criptografia de tráfego do Amazon ECS Service Connect com AWS Private CA, eles pagam pela autoridade de certificação privada criada e por cada certificado do TLS emitido. Para obter mais detalhes, consulte Preço do AWS Private Certificate Authority. Para estimar o custo mensal dos certificados do TLS, os clientes precisam saber o número dos serviços do Amazon ECS que têm o TLS habilitado, multiplicar esse número pelo custo do certificado e depois multiplicar por seis. Como o Amazon ECS Service Connect faz automaticamente o rodízio de certificados do TLS a cada cinco dias, em média, seis certificados são emitidos por serviço do Amazon ECS, por mês.

Componentes do Amazon ECS Service Connect

Ao usar o Amazon ECS Service Connect, você configura cada serviço do Amazon ECS para executar uma aplicação de servidor que recebe solicitações de rede (serviço cliente-servidor) ou para executar uma aplicação cliente que faz as solicitações (serviço cliente).

Quando você se preparar para começar a usar o Service Connect, comece com um serviço de cliente-servidor. É possível adicionar uma configuração do Service Connect a um novo serviço ou a um serviço existente. O Amazon ECS cria um endpoint do Service Connect no namespace. Além disso, o Amazon ECS cria uma nova implantação no serviço para substituir as tarefas que estão em execução no momento.

As tarefas existentes e outras aplicações podem continuar a se conectar aos endpoints existentes e a aplicações externas. Se um serviço cliente-servidor adicionar tarefas por meio da ação de aumentar a escala horizontalmente, novas conexões de clientes serão balanceadas entre todas as tarefas. Se um serviço cliente-servidor for atualizado, as novas conexões de clientes serão balanceadas entre as tarefas da nova versão.

As tarefas existentes não podem ser resolvidas e conectadas ao novo endpoint. Somente novas tarefas com uma configuração do Service Connect no mesmo namespace e que começam a ser executadas após essa implantação podem ser resolvidas e fazer conexão com esse endpoint.

Isso significa que o operador da aplicação cliente determina quando a configuração da aplicação muda, mesmo que o operador da aplicação do servidor possa alterar sua configuração a qualquer momento. A lista de endpoints no namespace poderá sofrer alterações sempre que qualquer serviço no namespace for implantado. As tarefas existentes e as tarefas de substituição mantêm o mesmo comportamento após a implantação mais recente.

Considere os seguintes exemplos:

Primeiro, suponha que você esteja criando uma aplicação que esteja disponível para a Internet pública em um único modelo do AWS CloudFormation e em uma única pilha do AWS CloudFormation. A descoberta e a acessibilidade públicas devem ser criadas por último pelo AWS CloudFormation, incluindo o serviço de cliente de frontend. Os serviços precisam ser criados nessa ordem para evitar um período em que o serviço do cliente de frontend esteja em execução e disponível ao público, mas um backend não. Isso evita que mensagens de erro sejam enviadas ao público durante esse período. No AWS CloudFormation, você deve usar `dependsOn` para indicar ao AWS CloudFormation que vários serviços do Amazon ECS não podem ser executados em paralelo ou simultaneamente. Você deve adicionar `dependsOn` ao serviço de cliente de frontend para cada serviço cliente-servidor de backend ao qual as tarefas do cliente se conectam.

Em segundo lugar, suponha que exista um serviço de frontend sem a configuração do Service Connect. As tarefas estão se conectando a um serviço de backend existente. Adicione primeiro uma configuração do Service Connect de cliente-servidor ao serviço de backend, usando o mesmo nome no DNS ou o `clientAlias` que o frontend usa. Isso cria uma nova implantação para que toda a detecção de reversão da implantação ou o AWS Management Console, a AWS CLI, AWS SDKs e outros métodos revertam o serviço de backend para a implantação e a configuração anteriores. Se você estiver satisfeito com a performance e o comportamento do serviço de backend, adicione uma configuração do Service Connect de cliente ou cliente-servidor ao serviço de frontend. Somente as tarefas da nova implantação usam o proxy do Service Connect que é adicionado a essas novas tarefas. Se você tiver problemas com essa configuração, poderá reverter para sua configuração anterior usando a detecção de reversão de implantação ou o AWS Management Console, a AWS CLI, AWS SDKs e outros métodos para reverter o serviço de backend para a implantação e a configuração anteriores. Se você usar outro sistema de descoberta de serviços baseado no DNS em vez de no Service Connect, qualquer aplicação de frontend ou cliente começará a usar novos endpoints e uma configuração alterada do endpoint após a expiração do cache de DNS local, que normalmente leva várias horas.

Redes

Por padrão, o proxy do Service Connect atua como receptor na `containerPort` usando o mapeamento da porta de definição de tarefa. As regras do seu grupo de segurança devem permitir o tráfego de entrada (ingresso) para esta porta proveniente das sub-redes em que os clientes serão executados.

Mesmo se você definir um número de porta na configuração do serviço do Service Connect, isso não alterará a porta do serviço cliente-servidor que o proxy do Service Connect recebe. Quando você define esse número de porta, o Amazon ECS altera a porta do endpoint ao qual os serviços do

cliente se conectam no proxy do Service Connect dentro dessas tarefas. O proxy no serviço cliente se conecta ao proxy no serviço cliente-servidor usando a `containerPort`.

Se você quiser alterar a porta na qual o proxy do Service Connect recebe, altere a `ingressPort0verride` na configuração do Service Connect do serviço cliente-servidor. Se você alterar esse número de porta, deverá permitir o tráfego de entrada na porta que está sendo usada pelo tráfego para esse serviço.

O tráfego que suas aplicações enviam para os serviços do Amazon ECS configurados para o Service Connect exige que a Amazon VPC e as sub-redes tenham regras de tabela de rotas e regras de ACL de rede que permitam os números de porta `containerPort` e `ingressPort0verride` que você está usando.

É possível usar o Service Connect para enviar tráfego entre VPCs. Os mesmos requisitos para as regras da tabela de rotas, para as ACLs da rede e para os grupos de segurança se aplicam a ambas as VPCs.

Por exemplo, dois clusters criam tarefas em VPCs diferentes. Um serviço em cada cluster é configurado para usar o mesmo namespace. As aplicações nesses dois serviços podem resolver todos os endpoints no namespace sem qualquer configuração DNS da VPC. No entanto, os proxies não podem se conectar, a menos que o emparelhamento de VPC, as tabelas de rotas de VPC ou de sub-rede e as ACLs da rede da VPC permitam o tráfego nos números de porta `containerPort` e `ingressPort0verride`.

Para tarefas que usam o modo de rede `bridge`, você deve criar um grupo de segurança com uma regra de entrada que permita tráfego no intervalo dinâmico superior de portas. Em seguida, atribua o grupo de segurança a todas as instâncias do EC2 no cluster do Service Connect.

Proxy do Service Connect

Se você criar ou atualizar um serviço com a configuração do Service Connect, o Amazon ECS adicionará um novo contêiner a cada nova tarefa à medida que elas forem iniciadas. Esse padrão de uso de um contêiner separado é chamado de um `sidecar`. Esse contêiner não está presente na definição da tarefa e você não pode configurá-lo. O Amazon ECS gerencia a configuração do Contêiner no serviço. Isso permite reutilizar as mesmas definições de tarefa entre diversos serviços, namespaces e tarefas sem a necessidade de usar o Service Connect.

Recursos de proxy

- Para definições de tarefas, é necessário configurar os parâmetros de CPU e de memória.

Recomendamos a adição de 256 unidades de CPU extras e pelo menos 64 MiB de memória à CPU e à memória da tarefa para o contêiner do proxy do Service Connect. No AWS Fargate, a menor quantidade de memória que você pode definir é 512 MiB. No Amazon EC2, a memória de definição de tarefa é obrigatória.

- Para o serviço, você define a configuração do log na configuração do Service Connect.
- Se você espera que as tarefas desse serviço recebam mais de 500 solicitações por segundo em sua carga máxima, recomendamos a adição de 512 unidades de CPU à sua CPU de tarefas nessa definição de tarefa para o contêiner do proxy do Service Connect.
- Se você espera criar mais de 100 serviços do Service Connect no namespace ou 2.000 tarefas no total em todos os serviços do Amazon ECS dentro do namespace, recomendamos a adição de 128 MiB de memória à sua memória de tarefas para o contêiner do proxy do Service Connect. Você deve fazer isso em todas as definições de tarefa usadas por todos os serviços do Amazon ECS no namespace.

Configuração do proxy

Suas aplicações se conectam ao proxy no contêiner de arquivo associado na mesma tarefa em que a aplicação está. O Amazon ECS configura a tarefa e os contêineres para que as aplicações se conectem ao proxy somente quando a aplicação estiver conectada aos nomes de endpoint no mesmo namespace. Todos os outros tráfegos não usam o proxy. O outro tráfego inclui endereços IP na mesma VPC, endpoints de serviço da AWS e tráfego externo.

Balanceamento de carga

O Service Connect configura o proxy para usar a estratégia round-robin para balancear a carga entre as tarefas em um endpoint do Service Connect. O proxy local que está na tarefa de onde vem a conexão escolhe uma das tarefas no serviço cliente-servidor que fornece o endpoint.

Por exemplo, considere uma tarefa que executa o WordPress em um serviço que está configurado como um serviço cliente em um namespace chamado local. Há outro serviço com duas tarefas que executam o banco de dados MySQL. Esse serviço é configurado para fornecer um endpoint chamado `mysql` por meio do Service Connect no mesmo namespace. Na tarefa do WordPress, a aplicação do WordPress se conecta ao banco de dados usando o nome do endpoint. As conexões para esse nome serão encaminhadas ao proxy que está em execução em um contêiner de arquivo associado na mesma tarefa. Em seguida, o proxy pode se conectar a qualquer uma das tarefas do MySQL usando a estratégia round-robin.

Estratégias de balanceamento de carga: round-robin

Detecção de discrepâncias

Esse recurso usa dados que o proxy tem sobre conexões falhadas anteriores para evitar o envio de novas conexões aos hosts que tiveram as conexões com falha. O Service Connect configura o recurso de detecção de valores discrepantes do proxy para fornecer verificações de integridade passivas.

usando o exemplo anterior, o proxy pode se conectar a qualquer uma das tarefas do MySQL. Se o proxy fez várias conexões com uma tarefa específica do MySQL e 5 ou mais conexões falharam nos últimos 30 segundos, o proxy evitará essa tarefa do MySQL por 30 a 300 segundos.

Repetições

O Service Connect configura o proxy para tentar novamente a conexão que passa pelo proxy e falha, e a segunda tentativa evita o uso do host da conexão anterior. Isso garante que cada conexão por meio do Service Connect não falhe por motivos pontuais.

Número de novas tentativas: 2

Timeout (Tempo limite)

O Service Connect configura o proxy para aguardar o máximo de tempo até que suas aplicações cliente-servidor respondam. O valor de tempo limite padrão é 15 segundos, mas pode ser atualizado.

Parâmetros opcionais:

idleTimeout: o tempo, em segundos, durante o qual uma conexão permanece ativa, embora ociosa. Um valor de `0` desabilita `idleTimeout`.

O padrão de `idleTimeout` para `HTTP/HTTP2/GRPC` é 5 minutos.

O `idleTimeout` padrão para `TCP` é 1 hora.

perRequestTimeout: a quantidade de tempo de espera para que o upstream responda com uma resposta completa por solicitação. Um valor `0` desativa o parâmetro `perRequestTimeout`. Isso poderá ser definido somente quando o `appProtocol` para o contêiner da aplicação for `HTTP/HTTP2/GRPC`. O padrão é 15 segundos.

> **Note**
>
> Se `idleTimeout` estiver definido para um tempo menor que `perRequestTimeout`, a conexão será fechada quando `idleTimeout` for atingido e não o `perRequestTimeout`.

Considerações

Considere o seguinte ao usar o Service Connect:

- As tarefas que são executadas no Fargate devem usar a versão da plataforma `1.4.0` ou versões posteriores do Linux para Fargate a fim de usar o Service Connect.
- A versão do agente do Amazon ECS na instância de contêiner deve ser `1.67.2` ou versões posteriores.
- As instâncias de contêiner devem executar a versão `20230428` ou posterior da AMI do Amazon Linux 2023 otimizada para o Amazon ECS, ou a versão `2.0.20221115` da AMI do Amazon Linux 2 otimizada para o Amazon ECS para usar o Service Connect. Essas versões têm o agente do Service Connect, além do agente de contêiner do Amazon ECS. Para obter mais informações sobre o agente do Service Connect, consulte Agente do Service Connect do Amazon ECS no GitHub.
- As instâncias de contêiner devem ter a permissão `ecs:Poll` para o recurso `arn:aws:ecs:`*`region`*`:`*`0123456789012`*`:task-set/`*`cluster`*`/*`. Se você estiver usando `ecsInstanceRole`, não precisará adicionar outras permissões. A política gerenciada `AmazonEC2ContainerServiceforEC2Role` tem as permissões necessárias. Para obter mais informações, consulte Função do IAM de instância de contêiner do Amazon ECS.
- As tarefas que usam o modo de rede `bridge` e o Service Connect não são compatíveis com parâmetro de definição de contêiner `hostname`.
- As definições de tarefa devem definir o limite de memória da tarefa para usar o Service Connect. Para obter mais informações, consulte Proxy do Service Connect.
- Não há suporte para as definições de tarefas que estabelecem limites de memória do contêiner.

  É possível definir limites de memória nos seus contêineres, mas deve definir o limite de memória da tarefa como um número maior que a soma dos limites de memória dos contêineres. A CPU e a memória adicionais nos limites de tarefas que não são alocadas nos limites de contêineres são usadas pelo contêiner do proxy do Service Connect e por outros contêineres que não definem limites de contêineres. Para obter mais informações, consulte Proxy do Service Connect.

- É possível configurar o Service Connect para usar qualquer namespace do AWS Cloud Map em uma mesma região na mesma Conta da AWS.
- Cada serviço pode pertencer a somente um namespace.
- Há suporte somente para as tarefas criadas pelos serviços.
- Todos os endpoints devem ser exclusivos dentro de um namespace.
- Todos os nomes de descoberta devem ser exclusivos dentro de um namespace.
- Você deve reimplantar os serviços existentes para que as aplicações possam resolver novos endpoints. Novos endpoints adicionados ao namespace após a implantacão mais recente não serão adicionados à configuração da tarefa. Para obter mais informações, consulte the section called “Componentes do Service Connect”.
- O Service Connect não exclui namespaces quando os clusters são excluídos. Você deve excluir os namespaces no AWS Cloud Map.
- O tráfego do Application Load Balancer usa como padrão o roteamento por meio do agente do Service Connect no modo de rede `awsvpc`. Se quiser que o tráfego que não seja de serviço ignore o agente do Service Connect, use o parâmetro `ingressPortOverride` na configuração de serviço do Service Connect.
- O Service Connect com TLS não é compatível com o modo de rede Bridge. Só há compatibilidade com o modo de rede `awsvpc`.

O Service Connect não é compatível com:

- Contêineres do Windows
- HTTP 1.0
- Tarefas autônomas
- Serviços que usam os tipos de implantação azul/verde baseada no CodeDeploy e externa
- Instâncias de contêiner `External` para Amazon ECS Anywhere não são compatíveis com o Service Connect.
- PPv2

Visão geral da configuração do Amazon ECS Service Connect

Ao usar o Service Connect, existem parâmetros que você precisa configurar em seus recursos.

A tabela a seguir descreve os parâmetros de configuração dos recursos do Amazon ECS.

Local dos parâmetros	Tipo de aplicação	Descrição	Obrigatório
Definição de tarefa	Cliente	Não há alterações disponíveis para o Service Connect nas definições de tarefa do cliente.	N/D
Definição de tarefa	Cliente-servidor	Os servidores devem adicionar campos `name` às portas nos `portMappings` de contêineres. Para obter mais informações, consulte `portMappings`.	Sim
Definição de tarefa	Cliente-servidor	Opcionalmente, os servidores podem fornecer um protocolo de aplicação (por exemplo, HTTP) para receber métricas específicas do protocolo para suas aplicações de servidor (por exemplo, `HTTP 5xx`).	Não
Definição de serviço	Cliente	Os serviços do cliente devem adicionar uma `serviceConnectConfiguration` para configurar o namespace a ser associado. Esse namespace deve conter todos os serviços de servidor que esse serviço precisa descobrir. Para obter mais informações, consulte `serviceConnectConfiguration`.	Sim
Definição de serviço	Cliente-servidor	Os serviços do servidor devem adicionar uma `serviceConnectConfiguration` para configurar os nomes de DNS, números de portas e namespace nos quais o serviço está disponível. Para obter mais informaçõ	Sim

Local dos parâmetros	Tipo de aplicação	Descrição	Obrigatório
		es, consulte serviceConnectConfiguration.	
Cluster	Cliente	Os clusters podem adicionar um namespace padrão do Service Connect. Novos serviços no cluster herdam o namespace quando o Service Connect é configurado em um serviço.	Não
Cluster	Cliente-servidor	Não há alterações disponíveis para o Service Connect em clusters que se aplicam aos serviços de servidor. As definições de tarefa e os serviços do servidor devem definir a respectiva configuração.	N/D

Visão geral das etapas para configurar o Service Connect

As etapas apresentadas a seguir fornecem uma visão geral de como configurar o Service Connect.

Important

- O Service Connect cria serviços do AWS Cloud Map em sua conta. Modificar esses recursos AWS Cloud Map registrando/cancelando manualmente o registro de instâncias, alterando os atributos da instância ou excluindo um serviço, pode causar um comportamento inesperado no tráfego da sua aplicação ou em implantações subsequentes.
- O Service Connect não oferece suporte a links na definição de tarefa.

1. Adicione os nomes das portas aos mapeamentos das portas nas definições das suas tarefas. Além disso, você pode identificar o protocolo de camada 7 da aplicação para obter métricas adicionais.

2. Crie um cluster com um namespace do AWS Cloud Map ou crie o namespace separadamente. Para uma organização simples, crie um cluster com o nome desejado para o namespace e especifique o nome idêntico para o namespace. Nesse caso, o Amazon ECS cria um novo namespace HTTP com a configuração necessária. O Service Connect não usa nem cria zonas hospedadas por DNS no Amazon Route 53.
3. Configure serviços para criar endpoints do Service Connect dentro do namespace.
4. Implemente serviços para criar os endpoints. O Amazon ECS adiciona um contêiner do proxy do Service Connect a cada tarefa e cria os endpoints do Service Connect no AWS Cloud Map. Esse contêiner não é configurado na definição da tarefa e a definição da tarefa pode ser reutilizada sem modificação para criar vários serviços no mesmo namespace ou em vários namespaces.
5. Implemente aplicações clientes como serviços para conexão aos endpoints. O Amazon ECS as conecta a endpoints do Service Connect por meio do proxy do Service Connect de cada tarefa.

   As aplicações só usam o proxy para se conectar aos endpoints do Service Connect. Não há configuração adicional para usar o proxy. O proxy executa balanceamento de carga round-robin, detecção de valores discrepantes e novas tentativas. Para obter mais informações sobre o proxy, consulte Proxy do Service Connect.
6. Monitore o tráfego por meio do proxy Service Connect no Amazon CloudWatch.

Configuração do cluster

É possível configurar um namespace padrão para o Service Connect ao criar ou ao atualizar o cluster. Se você especificar um nome de namespace que não existe na mesma conta e Região da AWS, será criado um novo namespace HTTP.

Se você criar um cluster e especificar um namespace padrão do Service Connect, o cluster aguardará no status `PROVISIONING` enquanto o Amazon ECS cria o namespace. É possível ver um `attachment` no status do cluster que mostra o status do namespace. Os anexos não são exibidos por padrão na AWS CLI. Você deve adicionar `--include ATTACHMENTS` para vê-los.

Configuração de serviço

O Service Connect foi projetado para exigir a configuração mínima. Você precisa definir um nome para cada mapeamento de porta que você gostaria de usar com o Service Connect na definição da tarefa. No serviço, você precisa ativar o Service Connect e selecionar um namespace para criar um serviço de cliente. Para criar um serviço de cliente-servidor, você precisa adicionar uma única configuração do serviço do Service Connect que corresponda ao nome de um dos mapeamentos

de porta. O Amazon ECS reutiliza o número da porta e o nome da porta da definição da tarefa para definir o serviço e o endpoint do Service Connect. Para substituir esses valores, você pode usar os outros parâmetros Discovery (Descoberta), DNS e Port (Porta) no console ou no `discoveryName` e no `clientAliases`, respectivamente, na API do Amazon ECS.

Criptografia do tráfego do Amazon ECS Service Connect

O Service Connect do Amazon ECS oferece suporte à criptografia automática de tráfego com certificados Transport Layer Security (TLS) para serviços do Amazon ECS. Ao direcionar os serviços do Amazon ECS para uma AWS Private Certificate Authority (AWS Private CA), o Amazon ECS provisiona automaticamente certificados TLS para criptografar o tráfego entre os serviços do Service Connect para Amazon ECS. O Amazon ECS gera, alterna e distribui certificados TLS usados na criptografia de tráfego.

A criptografia automática de tráfego com o Service Connect usa funcionalidades de criptografia líderes do setor para proteger a comunicação entre os serviços, ajudando você a cumprir os requisitos de segurança. Ela oferece suporte a certificados TLS da AWS Private Certificate Authority com criptografia `256-bit ECDSA` e `2048-bit RSA`. Você também tem controle total sobre certificados privados e chaves de assinatura para ajudar a atender aos requisitos de conformidade. Por padrão, o TLS 1.3 é compatível, mas o TLS 1.0 a 1.2 não é. O Service Connect oferece suporte ao TLS 1.3 com as seguintes cifras:

- `TLS_AES_128_GCM_SHA256`
- `TLS_AES_256_GCM_SHA384`
- `TLS_CHACHA20_POLY1305_SHA256`

> **Note**
>
> Para usar o protocolo TLS 1.3, é necessário habilitá-lo no receptor do destino.
> Somente o tráfego de entrada e de saída que é transferido pelo agente do Amazon ECS é criptografado.

Service Connect e verificações de integridade do Application Load Balancer

Você pode usar o Service Connect com as verificações de integridade do Application Load Balancer e criptografia TLS 1.3.

Configuração do Application Load Balancer

Defina o Application Load Balancer com as seguintes configurações:

- Configure um receptor TLS com uma política de segurança TLS 1.3 (como “ELBSecurityPolicy-TLS13-1-2-2021-06”). Para obter mais informações, consulte Políticas de segurança para seu Application Load Balancer.
- Crie um grupo de destino com as seguintes configurações:
  - Definir o protocolo como HTTPS
  - Anexar o grupo de destino ao receptor TLS
  - Configurar a porta de verificação de integridade para corresponder à porta de contêiner do serviço do Service Connect

Configuração do Service Connect

Defina um serviço com as seguintes configurações:

- Configure o serviço para usar o modo de rede `awsvpc`, pois o modo de rede `bridge` não é compatível.
- Habilite o Service Connect para o serviço.
- Defina o balanceador de carga com as seguintes configurações:
  - Especificar o grupo de destino que você configurou para o Application Load Balancer
  - Definir a porta do contêiner para corresponder à porta do contêiner do serviço TLS do Service Connect
- Evite configurar `ingressPortOverride` para o serviço. Para obter mais informações, consulte ServiceConnectService na Referência de API do Amazon Elastic Container Service.

Considerações

Considere o seguinte ao usar o Application Load Balancer, o TLS e o Service Connect:

- Use o modo de rede `awsvpc` em vez do modo de rede `bridge` para verificações de integridade de HTTPS ao usar o Service Connect com criptografia TLS. As verificações de integridade HTTP continuarão funcionando com o modo `bridge`.
- Configure a porta de verificação de integridade do grupo de destino para corresponder à porta do contêiner do serviço Service Connect, não à porta HTTPS padrão (443).

Certificados da AWS Private Certificate Authority e Service Connect

É necessário ter o perfil do IAM de infraestrutura. Para obter mais informações sobre esse perfil, consulte Perfil do IAM de infraestrutura do Amazon ECS.

Modos da AWS Private Certificate Authority para Service Connect

A AWS Private Certificate Authority pode ser executada em dois modos: de uso geral e de curta duração.

- Uso geral: certificados que podem ser configurados com qualquer data de expiração.
- De curta duração: certificados com validade máxima de sete dias.

Embora o Amazon ECS ofereça suporte aos dois modos, recomendamos usar certificados de curta duração. Por padrão, os certificados são alternados a cada cinco dias, e a execução em modo de curta duração oferece economias de custo significativas em relação ao de uso geral.

O Service Connect não oferece suporte à revogação de certificados. Em vez disso, utiliza certificados de curta duração com alternância frequente de certificados. Você tem autoridade para modificar a frequência de alternância, desabilitar ou excluir os segredos usando a rotação gerenciada no Secrets Manager, mas isso pode acarretar as possíveis consequências a seguir.

- Frequência de alternância mais curta: uma frequência de alternância mais curta incorre em custos mais altos porque AWS Private CA, AWS KMS, Secrets Manager e Auto Scaling enfrentam uma maior workload de alternância.
- Frequência de alternância mais longa: as comunicações das aplicações falham se a frequência de alternância exceder sete dias.
- Exclusão do segredo: excluir o segredo resulta em falha na alternância e afeta as comunicações da aplicação com o cliente.

Caso ocorra falha na alternância do segredo, um evento `RotationFailed` é publicado no AWS CloudTrail. Você também pode configurar um alarme do CloudWatch para `RotationFailed`.

> **Important**
>
> Não adicione regiões de réplica aos segredos. Isso evita que o Amazon ECS exclua o segredo, pois o Amazon ECS não tem permissão para remover regiões da replicação. Se você já adicionou a replicação, execute o comando a seguir.

```
aws secretsmanager remove-regions-from-replication \
 --secret-id SecretId \
 --remove-replica-regions region-name
```

Autoridades de certificação subordinadas

Você pode trazer qualquer AWS Private CA, raiz ou subordinado, ao protocolo TLS do Service Connect a fim de emitir certificados de entidades finais para os serviços. O emissor fornecido é tratado como signatário e raiz da confiança em todos os lugares. Você pode emitir certificados de entidade final para diferentes partes da aplicação em diferentes CAs subordinadas. Ao usar a AWS CLI, forneça o nome do recurso da Amazon (ARN) da CA para estabelecer a cadeia de confiança.

Autoridades de certificação on-premises

Para usar a CA on-premises, você cria e configura uma CA subordinada na AWS Private Certificate Authority. Isso garante que todos os certificados TLS emitidos para as workloads do Amazon ECS compartilhem a cadeia de confiança com as workloads que você executa on-premises e possam se conectar com segurança.

> **Important**
>
> Adicione a etiqueta obrigatória `AmazonECSManaged : true` em seu AWS Private CA.

Infraestrutura como código

Ao usar o TLS no Service Connect com as ferramentas de infraestrutura como código (IaC), é importante configurar as dependências corretamente para evitar problemas, como serviços presos no esgotamento. A chave AWS KMS, se fornecida, o perfil do IAM e as dependências de AWS Private CA devem ser excluídas após o serviço Amazon ECS.

Service Connect e Secrets Manager

Quando o Amazon ECS Service Connect com criptografia TLS é usado, o serviço interage com o Secrets Manager das seguintes formas:

O Service Connect utiliza o perfil de infraestrutura fornecido para criar segredos no Secrets Manager. Esses segredos são usados para armazenar as chaves privadas associadas aos seus certificados TLS para criptografar o tráfego entre os serviços do Service Connect.

> **Warning**
>
> A criação e o gerenciamento automáticos desses segredos pelo Service Connect simplificam o processo de implementação da criptografia TLS para seus serviços. No entanto, é importante estar ciente das possíveis implicações de segurança. Outros perfis do IAM que têm acesso de leitura ao Secrets Manager podem acessar esses segredos criados automaticamente. Isso poderá expor material criptográfico confidencial a partes não autorizadas se os controles de acesso não estiverem configurados adequadamente. Para minimizar esse risco, siga estas práticas recomendadas:
>
> - Gerencie e restrinja cuidadosamente o acesso ao Secrets Manager, especialmente para segredos criados pelo Service Connect.
> - Audite regularmente os perfis do IAM e suas permissões para garantir que o princípio do privilégio mínimo seja mantido.

Ao conceder acesso de leitura ao Secrets Manager, considere excluir as chaves privadas TLS criadas pelo Service Connect. É possível fazer isso usando uma condição em suas políticas do IAM para excluir segredos com ARNs que correspondem ao padrão:

```
"arn:aws:secretsmanager:::secret:ecs-sc!"
```

Um exemplo de política do IAM que nega a ação `GetSecretValue` a todos os segredos com o prefixo `ecs-sc!`:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "arn:aws:secretsmanager:*:*:secret:ecs-sc!*"
        }
```

```
    ]
}
```

> **Note**
>
> Esse é um exemplo geral e pode precisar ser ajustado com base no seu caso de uso específico e na configuração da conta da AWS. Sempre teste suas políticas do IAM minuciosamente para garantir que elas forneçam o acesso pretendido e, ao mesmo tempo, mantenham a segurança.

Ao entender como o Service Connect interage com o Secrets Manager, você pode gerenciar melhor a segurança dos seus serviços do Amazon ECS enquanto aproveita os benefícios da criptografia TLS automática.

Service Connect e AWS Key Management Service

Você pode usar o AWS Key Management Service para criptografar e descriptografar os recursos do Service Connect. O AWS KMS é um serviço gerenciado pela AWS no qual você pode criar e gerenciar chaves criptográficas que protegem seus dados.

Ao usar o AWS KMS com o Service Connect, você pode optar por usar uma chave de propriedade da AWS que a AWS gerencia para você ou escolher uma chave do AWS KMS existente. Você também pode criar uma chave do AWS KMS para usar.

Como fornecer sua própria chave de criptografia

Você pode fornecer seus próprios materiais de chave ou usar um repositório de chaves externo por meio do AWS Key Management Service Import your own key into AWS KMS e especificar o nome do recurso da Amazon (ARN) dessa chave no Service Connect do Amazon ECS.

Veja abaixo um exemplo de política AWS KMS. Substitua os valores das *entradas do usuário* pelos seus.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Sid": "id",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/role-name"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:GenerateDataKey",
        "kms:GenerateDataKeyPair"
      ],
      "Resource": "*"
    }
  ]
}
```

Para obter mais informações sobre políticas de chave, consulte Creating a key policy no Guia do desenvolvedor do AWS Key Management Service.

> **Note**
>
> O Service Connect só oferece suporte a chaves de criptografia simétricas do AWS KMS. Você não pode usar nenhum outro tipo de chave do AWS KMS para criptografar os recursos do Service Connect. Para obter ajuda para determinar se uma chave do AWS KMS é simétrica, consulte Identificar chaves do KMS assimétricas.

Para obter mais informações sobre chaves de criptografia simétricas do AWS Key Management Service, consulte Symmetric encryption AWS KMS keys no Guia do desenvolvedor do AWS Key Management Service.

Habilitação do protocolo TLS para o Amazon ECS Service Connect

Você habilita a criptografia de tráfego ao criar ou atualizar um serviço do Service Connect.

Para habilitar a criptografia de tráfego de um serviço em um namespace existente usando o AWS Management Console

1. É necessário ter o perfil do IAM de infraestrutura. Para obter mais informações sobre esse perfil, consulte Perfil do IAM de infraestrutura do Amazon ECS.

2. Abra o console em https://console.aws.amazon.com/ecs/v2.
3. No painel de navegação, escolha Namespaces.
4. Escolha o Namespace com o Serviço para o qual gostaria de habilitar a criptografia de tráfego.
5. Escolha o Serviço para o qual gostaria de habilitar a criptografia de tráfego.
6. Escolha Atualizar serviço no canto superior direito e role para baixo até a seção Service Connect.
7. Escolha Ativar criptografia de tráfego nas informações do serviço para habilitar o TLS.
8. Em Perfil do TLS do Service Connect, escolha um perfil do IAM de infraestrutura existente ou crie um novo.
9. Em Autoridade de certificação de signatário, escolha uma autoridade de certificação existente ou crie uma nova.

   Para obter mais informações, consulte Certificados da AWS Private Certificate Authority e Service Connect.
10. Em Escolher uma AWS KMS key, escolha uma chave gerenciada e de propriedade da AWS ou selecione uma chave diferente. Você também pode criar uma nova.

Para obter um exemplo de uso da AWS CLI para configurar o protocolo TLS para seu serviço, consulte Configuração do Amazon ECS Service Connect com a AWS CLI.

Verificação da habilitação do protocolo TLS para o Amazon ECS Service Connect

O Service Connect inicia o TLS no agente do Service Connect e o encerra no agente de destino. Como resultado, o código da aplicação nunca vê interações TLS. Use as etapas apresentadas a seguir para verificar se o protocolo TLS está habilitado.

1. Inclua a CLI do `openssl` na imagem da sua aplicação.
2. Habilite o ECS Exec nos serviços para se conectar às tarefas via SSM. Como alternativa, é possível iniciar uma instância do Amazon EC2 na mesma Amazon VPC do serviço.
3. Recupere o IP e a porta de uma tarefa de um serviço que deseja verificar. O endereço IP da tarefa pode ser recuperado no console do AWS Cloud Map. As informações estão na página de detalhes do serviço do namespace.
4. Faça login em qualquer uma de suas tarefas usando `execute-command` como no exemplo apresentado a seguir. Como alternativa, faça login na instância do Amazon EC2 criada na Etapa 2.

```
$ aws ecs execute-command --cluster cluster-name \
    --task task-id  \
    --container container-name \
    --interactive \
    --command "/bin/sh"
```

> **Note**
>
> Chamar o nome do DNS diretamente não revela o certificado.

5. No shell conectado, use a CLI `openssl` para verificar e visualizar o certificado anexado à tarefa.

   Exemplo:

```
openssl s_client -connect 10.0.147.43:6379 < /dev/null 2> /dev/null \
| openssl x509 -noout -text
```

   Exemplo de resposta:

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            <serial-number>
        Signature Algorithm: ecdsa-with-SHA256
        Issuer: <issuer>
        Validity
            Not Before: Jan 23 21:38:12 2024 GMT
            Not After : Jan 30 22:38:12 2024 GMT
        Subject: <subject>
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                pub:
                    <pub>
                ASN1 OID: prime256v1
                NIST CURVE: P-256
        X509v3 extensions:
            X509v3 Subject Alternative Name:
                DNS:redis.yelb-cftc
```

```
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Authority Key Identifier:
                keyid:<key-id>

            X509v3 Subject Key Identifier:
                1D:<id>
            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment
            X509v3 Extended Key Usage:
                TLS Web Server Authentication, TLS Web Client Authentication
    Signature Algorithm: ecdsa-with-SHA256
        <hash>
```

Configuração do Amazon ECS Service Connect com a AWS CLI

É possível criar um serviço do Amazon ECS para uma tarefa do Fargate que usa o Service Connect com a AWS CLI.

> **Note**
>
> É possível usar endpoints de serviço de pilha dupla para interagir com o Amazon ECS via AWS CLI, SDKs e API do Amazon ECS sobre IPv4 e IPv6. Para obter mais informações, consulte Usar endpoints de pilha dupla do Amazon ECS.

Pré-requisitos

Confira a seguir os pré-requisitos do Service Connect:

- Verifique se a versão mais recente da AWS CLI está instalada e configurada. Para obter mais informações, consulte Instalar ou atualizar para a versão mais recente da AWS CLI.
- O usuário do IAM tem as permissões necessárias especificadas no exemplo de política do IAM de AmazonECS_FullAccess.
- Você tem uma VPC, uma sub-rede, uma tabela de rotas e um grupo de segurança criados para serem usados. Para obter mais informações, consulte the section called “Criar uma nuvem privada virtual”.
- Você tem um perfil de execução de tarefa com o nome `ecsTaskExecutionRole` e a política gerenciada `AmazonECSTaskExecutionRolePolicy` está anexada ao perfil. Esse perfil permite

que o Fargate grave os logs da aplicação NGINX e os logs do proxy do Service Connect no Amazon CloudWatch Logs. Para obter mais informações, consulte Criar a função de execução de tarefa do.

Etapa 1: criar um cluster

Use as etapas a seguir para criar um cluster e um namespace do Amazon ECS.

Para criar o cluster e o namespace AWS Cloud Map do Amazon ECS

1. Crie um cluster do Amazon ECS denominado `tutorial` a ser usado. O parâmetro `--service-connect-defaults` define o namespace padrão do cluster. Na saída do exemplo, um namespace do AWS Cloud Map com o nome `service-connect` não existe nessa conta nem na Região da AWS. Portanto, o namespace é criado pelo Amazon ECS. O namespace é criado no AWS Cloud Map na conta e é visível com todos os outros namespaces. Portanto, use um nome que indique a finalidade.

```
aws ecs create-cluster --cluster-name tutorial --service-connect-defaults
 namespace=service-connect
```

Saída:

```
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/tutorial",
        "clusterName": "tutorial",
        "serviceConnectDefaults": {
            "namespace": "arn:aws:servicediscovery:us-
west-2:123456789012:namespace/ns-EXAMPLE"
        },
        "status": "PROVISIONING",
        "registeredContainerInstancesCount": 0,
        "runningTasksCount": 0,
        "pendingTasksCount": 0,
        "activeServicesCount": 0,
        "statistics": [],
        "tags": [],
        "settings": [
            {
                "name": "containerInsights",
                "value": "disabled"
```

```
            }
        ],
        "capacityProviders": [],
        "defaultCapacityProviderStrategy": [],
        "attachments": [
            {
                "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
                "type": "sc",
                "status": "ATTACHING",
                "details": []
            }
        ],
        "attachmentsStatus": "UPDATE_IN_PROGRESS"
    }
}
}
```

2. Verifique se o cluster foi criado:

```
aws ecs describe-clusters --clusters tutorial
```

Saída:

```
{
    "clusters": [
        {
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/tutorial",
            "clusterName": "tutorial",
            "serviceConnectDefaults": {
                "namespace": "arn:aws:servicediscovery:us-
west-2:123456789012:namespace/ns-EXAMPLE"
            },
            "status": "ACTIVE",
            "registeredContainerInstancesCount": 0,
            "runningTasksCount": 0,
            "pendingTasksCount": 0,
            "activeServicesCount": 0,
            "statistics": [],
            "tags": [],
            "settings": [],
            "capacityProviders": [],
            "defaultCapacityProviderStrategy": []
        }
```

```
    ],
    "failures": []
}
```

3. (opcional) verifique se o namespace foi criado no AWS Cloud Map. É possível usar o AWS Management Console ou a configuração normal da AWS CLI, pois ela é criada no AWS Cloud Map.

   Por exemplo, use a AWS CLI:

   ```
   aws servicediscovery get-namespace --id ns-EXAMPLE
   ```

   Saída:

   ```
   {
       "Namespace": {
           "Id": "ns-EXAMPLE",
           "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
   EXAMPLE",
           "Name": "service-connect",
           "Type": "HTTP",
           "Properties": {
               "DnsProperties": {
                   "SOA": {}
               },
               "HttpProperties": {
                   "HttpName": "service-connect"
               }
           },
           "CreateDate": 1661749852.422,
           "CreatorRequestId": "service-connect"
       }
   }
   ```

Etapa 2: criar o serviço para o servidor

O recurso Service Connect é destinado à interconexão de várias aplicações no Amazon ECS. Pelo menos uma dessas aplicações precisa fornecer um serviço Web ao qual se conectar. Nessa etapa, você criará:

- A definição de tarefa que usa a imagem oficial do contêiner NGINX não modificada e inclui a configuração do Service Connect.
- A definição de serviço do Amazon ECS que configura o Service Connect para fornecer a descoberta de serviços e o proxy de malha de serviços para o tráfego desse serviço. A configuração reutiliza o namespace padrão da configuração do cluster para reduzir a quantidade de configuração de serviço feita para cada serviço.
- O serviço do Amazon ECS. Ele executa uma tarefa usando a definição de tarefa e insere um contêiner adicional para o proxy do Service Connect. O proxy recebe a porta proveniente do mapeamento da porta do contêiner na definição de tarefa. Em uma aplicação cliente executada no Amazon ECS, o proxy na tarefa do cliente recebe as conexões de saída para o nome da porta de definição de tarefa, o nome da descoberta de serviços ou o nome do alias do cliente de serviço e o número da porta do alias do cliente.

Como criar o serviço Web com o Service Connect

1. Registre uma definição de tarefa compatível com o Fargate e use o modo de rede `awsvpc`. Siga estas etapas:

   a. Crie um arquivo denominado `service-connect-nginx.json` com o conteúdo da seguinte definição de tarefa.

      Essa definição de tarefa configura o Service Connect adicionando os parâmetros `name` e `appProtocol` ao mapeamento de portas. O nome da porta torna essa porta mais identificável na configuração do serviço quando várias portas são usadas. O nome da porta também é usado por padrão como o nome detectável para uso por outras aplicações no namespace.

      A definição da tarefa contém o perfil do IAM da tarefa, pois o serviço tem o ECS Exec ativado.

      > **Important**
      >
      > Essa definição de tarefa usa a `logConfiguration` para enviar a saída do nginx de `stdout` e `stderr` para o Amazon CloudWatch Logs. Esse perfil de execução de tarefas não tem as permissões extras necessárias para a criação do grupo de logs do CloudWatch Logs. Crie o grupo de logs no CloudWatch Logs usando o AWS

Management Console ou a AWS CLI. Se você não quiser enviar os logs do nginx para o CloudWatch Logs, poderá remover a `logConfiguration`.
Substitua o ID da Conta da AWS no perfil de execução de tarefas pelo ID da sua Conta da AWS.

```
{
    "family": "service-connect-nginx",
    "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
    "taskRoleArn": "arn:aws:iam::123456789012:role/ecsTaskRole",
    "networkMode": "awsvpc",
    "containerDefinitions": [
        {
        "name": "webserver",
        "image": "public.ecr.aws/docker/library/nginx:latest",
        "cpu": 100,
        "portMappings": [
            {
                "name": "nginx",
                "containerPort": 80,
                "protocol": "tcp",
                "appProtocol": "http"
            }
        ],
        "essential": true,
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-nginx",
                "awslogs-region": "region",
                "awslogs-stream-prefix": "nginx"
            }
        }
        }
    ],
    "cpu": "256",
    "memory": "512"
}
```

b. Registre a definição de tarefa usando o arquivo `service-connect-nginx.json`:

```
aws ecs register-task-definition --cli-input-json file://service-connect-
nginx.json
```

2. Crie um serviço:

   a. Crie um arquivo denominado `service-connect-nginx-service.json` com o conteúdo do serviço do Amazon ECS que você está criando. Este exemplo usa a definição de tarefa criada na etapa anterior. Uma `awsvpcConfiguration` é necessária, pois o exemplo de definição de tarefa usa o modo de rede `awsvpc`.

   Ao criar o serviço do ECS, especifique o tipo de inicialização do Fargate e a versão `LATEST` da plataforma que oferece suporte ao Service Connect. Os `securityGroups` e as `subnets` devem pertencer a uma VPC que tenha os requisitos para usar o Amazon ECS. É possível obter os IDs de grupos de segurança e sub-redes no console da Amazon VPC.

   Esse serviço configura o Service Connect adicionando o parâmetro `serviceConnectConfiguration`. O namespace não é necessário porque o cluster tem um namespace padrão configurado. As aplicações cliente em execução no ECS no namespace se conectam a esse serviço usando o `portName` e a porta no `clientAliases`. Por exemplo, esse serviço pode ser acessado usando `http://nginx:80/`, pois o nginx fornece uma página de boas-vindas no `/` do local raiz. Aplicações externas que não estão sendo executadas no Amazon ECS ou não estão no mesmo namespace podem acessar essa aplicação por meio do proxy do Service Connect usando o endereço IP da tarefa e o número da porta da definição de tarefa. Na configuração do `tls`, adicione o `arn` do certificado para `awsPcaAuthorityArn`, `kmsKey` e `roleArn` do perfil do IAM.

   Esse serviço usa uma `logConfiguration` para enviar a saída do proxy de conexão do serviço de `stdout` e `stderr` para o Amazon CloudWatch Logs. Esse perfil de execução de tarefas não tem as permissões extras necessárias para a criação do grupo de logs do CloudWatch Logs. Crie o grupo de logs no CloudWatch Logs usando o AWS Management Console ou a AWS CLI. Recomendamos que você crie esse grupo de logs e armazene os logs de proxy no CloudWatch Logs. Se você não quiser enviar os logs do proxy para o CloudWatch Logs, poderá remover a `logConfiguration`.

```
{
    "cluster": "tutorial",
```

```
    "deploymentConfiguration": {
        "maximumPercent": 200,
        "minimumHealthyPercent": 0
    },
    "deploymentController": {
        "type": "ECS"
    },
    "desiredCount": 1,
    "enableECSManagedTags": true,
    "enableExecuteCommand": true,
    "launchType": "FARGATE",
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "assignPublicIp": "ENABLED",
            "securityGroups": [
                "sg-EXAMPLE"
            ],
            "subnets": [
                "subnet-EXAMPLE",
                "subnet-EXAMPLE",
                "subnet-EXAMPLE"
            ]
           }
    },
    "platformVersion": "LATEST",
    "propagateTags": "SERVICE",
    "serviceName": "service-connect-nginx-service",
    "serviceConnectConfiguration": {
        "enabled": true,
        "services": [
            {
                "portName": "nginx",
                "clientAliases": [
                    {
                        "port": 80
                    }
                ],
                "tls": {
                   "issuerCertificateAuthority": {
                      "awsPcaAuthorityArn": "certificateArn"
                   },
                   "kmsKey": "kmsKey",
                   "roleArn": "iamRoleArn"
                }
```

```
            }
        ],
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-proxy",
                "awslogs-region": "region",
                "awslogs-stream-prefix": "service-connect-proxy"
            }
        }
    },
    "taskDefinition": "service-connect-nginx"
}
```

b. Crie um serviço usando o arquivo `service-connect-nginx-service.json`:

```
aws ecs create-service --cluster tutorial --cli-input-json file://service-
connect-nginx-service.json
```

Saída:

```
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/tutorial/
service-connect-nginx-service",
        "serviceName": "service-connect-nginx-service",
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/tutorial",
        "loadBalancers": [],
        "serviceRegistries": [],
        "status": "ACTIVE",
        "desiredCount": 1,
        "runningCount": 0,
        "pendingCount": 0,
        "launchType": "FARGATE",
        "platformVersion": "LATEST",
        "platformFamily": "Linux",
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
service-connect-nginx:1",
        "deploymentConfiguration": {
            "deploymentCircuitBreaker": {
                "enable": false,
```

```
                "rollback": false
            },
            "maximumPercent": 200,
            "minimumHealthyPercent": 0
        },
        "deployments": [
            {
                "id": "ecs-svc/3763308422771520962",
                "status": "PRIMARY",
                "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-
definition/service-connect-nginx:1",
                "desiredCount": 1,
                "pendingCount": 0,
                "runningCount": 0,
                "failedTasks": 0,
                "createdAt": 1661210032.602,
                "updatedAt": 1661210032.602,
                "launchType": "FARGATE",
                "platformVersion": "1.4.0",
                "platformFamily": "Linux",
                "networkConfiguration": {
                    "awsvpcConfiguration": {
                        "assignPublicIp": "ENABLED",
                        "securityGroups": [
                            "sg-EXAMPLE"
                        ],
                        "subnets": [
                            "subnet-EXAMPLEf",
                            "subnet-EXAMPLE",
                            "subnet-EXAMPLE"
                        ]
                    }
                },
                "rolloutState": "IN_PROGRESS",
                "rolloutStateReason": "ECS deployment ecs-
svc/3763308422771520962 in progress.",
                "failedLaunchTaskCount": 0,
                "replacedTaskCount": 0,
                "serviceConnectConfiguration": {
                    "enabled": true,
                    "namespace": "service-connect",
                    "services": [
                        {
                            "portName": "nginx",
```

```
                            "clientAliases": [
                                {
                                    "port": 80
                                }
                            ]
                        }
                    ],
                    "logConfiguration": {
                        "logDriver": "awslogs",
                        "options": {
                            "awslogs-group": "/ecs/service-connect-proxy",
                            "awslogs-region": "us-west-2",
                            "awslogs-stream-prefix": "service-connect-proxy"
                        },
                        "secretOptions": []
                    }
                },
                "serviceConnectResources": [
                    {
                        "discoveryName": "nginx",
                        "discoveryArn": "arn:aws:servicediscovery:us-
west-2:123456789012:service/srv-EXAMPLE"
                    }
                ]
            }
        ],
        "roleArn": "arn:aws:iam::123456789012:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
        "version": 0,
        "events": [],
        "createdAt": 1661210032.602,
        "placementConstraints": [],
        "placementStrategy": [],
        "networkConfiguration": {
            "awsvpcConfiguration": {
                "assignPublicIp": "ENABLED",
                "securityGroups": [
                    "sg-EXAMPLE"
                ],
                "subnets": [
                    "subnet-EXAMPLE",
                    "subnet-EXAMPLE",
                    "subnet-EXAMPLE"
                ]
```

```
            }
        },
        "schedulingStrategy": "REPLICA",
        "enableECSManagedTags": true,
        "propagateTags": "SERVICE",
        "enableExecuteCommand": true
    }
}
```

A `serviceConnectConfiguration` que você forneceu aparece na primeira implantação da saída. À medida que você faz alterações no serviço do ECS de maneiras que precisam fazer alterações nas tarefas, uma nova implantação é criada pelo Amazon ECS.

Etapa 3: verificar se você pode se conectar

Para verificar se o Service Connect está configurado e funcionando, siga estas etapas para se conectar ao serviço Web em uma aplicação externa. Em seguida, consulte as métricas adicionais que o proxy do Service Connect cria no CloudWatch.

Para se conectar ao serviço Web em uma aplicação externa

- Conecte-se ao endereço IP da tarefa e à porta do contêiner usando o endereço IP da tarefa

  Use a AWS CLI para obter o ID da tarefa, usando o `aws ecs list-tasks --cluster tutorial`.

  Se suas sub-redes e seu grupo de segurança permitirem tráfego da Internet pública na porta da definição da tarefa, será possível se conectar ao IP público no seu computador. No entanto, o IP público não está disponível em “describe-tasks”. Portanto, as etapas envolvem acessar o AWS Management Console ou a AWS CLI do Amazon EC2 para obter os detalhes da interface de rede elástica.

  Nesse exemplo, uma instância do Amazon EC2 na mesma VPC usa o IP privado da tarefa. A aplicação é nginx, mas o cabeçalho `server: envoy` mostra que o proxy do Service Connect é usado. O proxy do Service Connect recebe a porta do contêiner vindo da definição de tarefa.

  ```
  $ curl -v 10.0.19.50:80/
  *   Trying 10.0.19.50:80...
  ```

```
* Connected to 10.0.19.50 (10.0.19.50) port 80 (#0)
> GET / HTTP/1.1
> Host: 10.0.19.50
> User-Agent: curl/7.79.1
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< server: envoy
< date: Tue, 23 Aug 2022 03:53:06 GMT
< content-type: text/html
< content-length: 612
< last-modified: Tue, 16 Apr 2019 13:08:19 GMT
< etag: "5cb5d3c3-264"
< accept-ranges: bytes
< x-envoy-upstream-service-time: 0
<
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

Para visualizar as métricas do Service Connect

O proxy do Service Connect cria métricas de aplicações (conexão HTTP, HTTP2, gRPC ou TCP) nas métricas do CloudWatch. Ao usar o console do CloudWatch, consulte as dimensões de métricas adicionais de DiscoveryName, (DiscoveryName, ServiceName, ClusterName), TargetDiscoveryNam e (TargetDiscoveryName, ServiceName, ClusterName) no namespace do Amazon ECS. Para obter mais informações sobre essas métricas e dimensões, consulte Visualizar métricas disponíveis no Guia do usuário do Amazon CloudWatch Logs.

Uso da descoberta de serviços para conectar serviços do Amazon ECS com nomes DNS

O serviço do Amazon ECS pode ser opcionalmente configurado para usar a descoberta de serviço do Amazon ECS. A descoberta de serviço usa ações de API do AWS Cloud Map para gerenciar namespaces HTTP e DNS para serviços do Amazon ECS. Para obter mais informações, consulte O que é o AWS Cloud Map? no Guia do desenvolvedor do AWS Cloud Map.

A descoberta de serviço está disponível nas seguintes regiões da AWS:

Nome da região	Região
Leste dos EUA (Norte da Virgínia)	us-east-1
Leste dos EUA (Ohio)	us-east-2
Oeste dos EUA (N. da Califórnia)	us-west-1
Oeste dos EUA (Oregon)	us-west-2
África (Cidade do Cabo)	af-south-1
Ásia-Pacífico (Hong Kong)	ap-east-1
Ásia-Pacífico (Mumbai)	ap-south-1
Ásia-Pacífico (Hyderabad)	ap-south-2
Ásia-Pacífico (Tóquio)	ap-northeast-1
Ásia-Pacífico (Seul)	ap-northeast-2

Nome da região	Região
Ásia-Pacífico (Osaka)	ap-northeast-3
Ásia-Pacífico (Singapura)	ap-southeast-1
Ásia-Pacífico (Sydney)	ap-southeast-2
Ásia-Pacífico (Jacarta)	ap-southeast-3
Ásia-Pacífico (Melbourne)	ap-southeast-4
Canadá (Central)	ca-central-1
Oeste do Canadá (Calgary)	ca-west-1
China (Pequim)	cn-north-1
China (Ningxia)	cn-northwest-1
Europa (Frankfurt)	eu-central-1
Europa (Zurique)	eu-central-2
Europa (Irlanda)	eu-west-1
Europa (Londres)	eu-west-2
Europa (Paris)	eu-west-3
Europa (Milão)	eu-south-1
Europa (Estocolmo)	eu-north-1
Israel (Tel Aviv)	il-central-1
Europa (Espanha)	eu-south-2
Oriente Médio (Emirados Árabes Unidos)	me-central-1
Oriente Médio (Bahrein)	me-south-1

Nome da região	Região
América do Sul (São Paulo)	sa-east-1
AWS GovCloud (Leste dos EUA)	us-gov-east-1
AWS GovCloud (Oeste dos EUA)	us-gov-west-1

Conceitos de descoberta de serviço

A descoberta de serviço consiste nos seguintes componentes:

- Namespace de descoberta de serviço: grupo lógico de serviços de descoberta de serviço que compartilham o mesmo nome de domínio, como `example.com`, que é onde você deseja rotear o tráfico. É possível criar um namespace com uma chamada para o comando `aws servicediscovery create-private-dns-namespace` ou no console do Amazon ECS. É possível usar o comando `aws servicediscovery list-namespaces` para exibir as informações resumidas sobre os namespaces criados pela conta atual. Para obter mais informações sobre os comandos de descoberta de serviço, consulte `create-private-dns-namespace` e `list-namespaces` no Guia de referência da AWS CLI do AWS Cloud Map (descoberta de serviço).
- Serviço de descoberta de serviço: existe no namespace de descoberta de serviço e consiste no nome do serviço e na configuração do DNS para o namespace. Ele fornece os seguintes componentes principais:
  - Service registry (Registro do serviço): permite pesquisar um serviço por meio de DNS ou ações de API do AWS Cloud Map disponíveis que podem ser usados para se conectar ao serviço.
- Instância de descoberta de serviço: existe no serviço de descoberta de serviço e consiste nos atributos associados a cada serviço do Amazon ECS no diretório de serviços.
  - Atributos de instância: os metadados a seguir são adicionados como atributos personalizados para cada serviço do Amazon ECS configurado para usar a descoberta de serviço:
    - **`AWS_INSTANCE_IPV4`**: para um registro `A`, o endereço IPv4 que o Route 53 retorna em resposta a consultas de DNS e que o AWS Cloud Map retorna ao descobrir detalhes da instância, por exemplo, `192.0.2.44`.
    - **`AWS_INSTANCE_PORT`**: o valor da porta associado ao serviço de descoberta de serviço.
    - **`AVAILABILITY_ZONE`**: a zona de disponibilidade na qual a tarefa foi iniciada. Para tarefas que usam o tipo de inicialização do EC2, essa é a zona de disponibilidade na qual a instância

de contêiner existe. Para tarefas que usam o tipo de inicialização do Fargate, essa é a zona de disponibilidade na qual a interface de rede elástica existe.

- **`REGION`**: a região em que a tarefa existe.
- **`ECS_SERVICE_NAME`**: o nome do serviço do Amazon ECS ao qual a tarefa pertence.
- **`ECS_CLUSTER_NAME`**: o nome do cluster do Amazon ECS ao qual a tarefa pertence.
- **`EC2_INSTANCE_ID`**: o ID da instância de contêiner na qual a tarefa foi posicionada. Esse atributo personalizado não é adicionado se a tarefa está usando o tipo de inicialização do Fargate.
- **`ECS_TASK_DEFINITION_FAMILY`**: a família de definições de tarefa que a tarefa está usando.
- **`ECS_TASK_SET_EXTERNAL_ID`**: se um conjunto de tarefas for criado para uma implantação externa e for associado a um registro de descoberta de serviço, o atributo `ECS_TASK_SET_EXTERNAL_ID` conterá o ID externo do conjunto de tarefas.

- Verificações de integridade do Amazon ECS: o Amazon ECS executa verificações de integridade periódicas em nível de contêiner. Se não passar na verificação de integridade, o endpoint é removido do roteamento de DNS e marcado como não íntegro.

Considerações sobre descoberta de serviço

As seguintes informações devem ser considerada ao usar a descoberta de serviço:

- A descoberta de serviços é compatível com as tarefas no Fargate se você está usando a versão 1.1.0 ou posterior da plataforma. Para obter mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.
- Os serviços configurados para usar a descoberta de serviços têm um limite de 1.000 tarefas por serviço. Isso se deve a uma cota de serviço do Route 53.
- O fluxo de trabalho Create Service (Criar serviço) no console do Amazon ECS é compatível apenas com o registro de serviços em namespaces DNS privados. Quando um namespace DNS privado do AWS Cloud Map for criado, será criada automaticamente uma zona hospedada privada do Route 53.
- Os atributos DNS da VPC devem ser configurados para resoluções DNS bem-sucedidas. Para obter informações sobre como configurar os atributos, consulte Suporte a DNS na sua VPC no Guia do usuário da Amazon VPC.
- Os registros de DNS criados para um serviço de descobertra de serviço sempre são registrados com o endereço IP privado da tarefa, em vez do endereço IP público, mesmo quando são usados namespaces públicos.

- A descoberta de serviço exige que as tarefas especifiquem o modo de rede `awsvpc`, `bridge` ou `host` (`none` não é compatível).
- Se a definição de tarefa de serviço usa o modo de rede `awsvpc`, será possível criar qualquer combinação de registros `A` ou SRV para cada tarefa de serviço. Se você usar registros SRV, uma porta é necessária.
- Se a definição de tarefa de serviço usa o modo de rede `bridge` ou `host`, o único tipo de registro DNS com suporte é o registro SRV. Crie um registro SRV para cada tarefa de serviço. O registro SRV deve especificar o nome do contêiner e a combinação de portas do contêiner da definição de tarefa.
- Os registros de DNS de um serviço de descoberta de serviço podem ser consultados na VPC. Eles usam o seguinte formato: `<service-discovery-service-name>.<service-discovery-namespace>`.
- Ao fazer uma consulta de DNS no nome do serviço, os registros `A` retornam um conjunto de endereços IP correspondentes às suas tarefas. Os registros SRV retornam um conjunto de endereços IP e portas para cada tarefa.
- Se você tiver oito ou menos registros íntegros, o Route 53 responderá a todas as consultas DNS com todos os registros íntegros.
- Quando nenhum dos registros estiver íntegro, o Route 53 responderá às consultas DNS com até oito registros não íntegros.
- É possível configurar a descoberta de serviço para um serviço que esteja atrás de um balanceador de carga, mas o tráfego da descoberta de serviço sempre será roteado para a tarefa, e não para o balanceador de carga.
- A descoberta de serviço não oferece suporte ao uso de Classic Load Balancers.
- Convém usar verificações de integridade em nível de contêiner gerenciadas pelo Amazon ECS para o serviço de descoberta de serviços.
  - HealthCheckCustomConfig: o Amazon ECS gerencia as verificações de integridade em seu nome. O Amazon ECS usa informações de contêiner e das verificações de integridade, além do estado da tarefa, para atualizar a integridade com o AWS Cloud Map. Isso é especificado pelo uso do parâmetro `--health-check-custom-config` ao ser criado o serviço de descoberta de serviço. Para obter mais informações, consulte HealthCheckCustomConfig na Referência da API do AWS Cloud Map.
- Os recursos do AWS Cloud Map criados quando a descoberta de serviço é usada devem ser limpos manualmente.

- As tarefas e instâncias são registradas como UNHEALTHY até que as verificações de integridade do contêiner retornem um valor. Se as verificações de integridade forem aprovadas, o status é atualizado para HEALTHY. Se as verificações de integridade do contêiner falharem, o registro da instância da descoberta de serviços é cancelado.

Preço da descoberta de serviço

Os clientes que usam a descoberta de serviço do Amazon ECS são cobrados pelos recursos do Route 53 e pelas operações da API de descoberta do AWS Cloud Map. Isso envolve os custos de criação de zonas hospedadas do Route 53 e de consultas ao registro do serviço. Para obter mais informações, consulte Preços do AWS Cloud Map no Guia do desenvolvedor do AWS Cloud Map.

O Amazon ECS executa verificações de integridade no nível do contêiner e as expõe às operações da API de verificação de integridade personalizada do AWS Cloud Map. Atualmente, isso é disponibilizado aos clientes sem nenhum custo extra. Se configurar verificações de integridade de rede adicionais para tarefas expostas publicamente, você será cobrado por essas verificações.

Criar um novo serviço Amazon ECS que usa descoberta de serviços

Saiba como criar um serviço que contém uma tarefa do Fargate que usa descoberta de serviços com a AWS CLI.

Para obter uma lista de Regiões da AWS que oferecem suporte à descoberta de serviços, consulte Uso da descoberta de serviços para conectar serviços do Amazon ECS com nomes DNS.

Para obter informações sobre as regiões que oferecem suporte ao Fargate, consulte the section called “Regiões do AWS Fargate”.

> **Note**
>
> É possível usar endpoints de serviço de pilha dupla para interagir com o Amazon ECS via AWS CLI, SDKs e API do Amazon ECS sobre IPv4 e IPv6. Para obter mais informações, consulte Usar endpoints de pilha dupla do Amazon ECS.

Pré-requisitos

Antes de começar este tutorial, certifique-se de que os seguintes pré-requisitos foram atendidos:

- A versão mais recente da AWS CLI está instalada e configurada. Para obter mais informações, consulte Instalar ou atualizar para a versão mais recente da AWS CLI.
- As etapas descritas em Configuração para usar o Amazon ECS estão concluídas.
- O usuário do IAM tem as permissões necessárias especificadas no exemplo de política do IAM de AmazonECS_FullAccess.
- Você criou pelo menos uma VPC e um grupo de segurança. Para obter mais informações, consulte the section called “Criar uma nuvem privada virtual”.

Etapa 1: criar os recursos da descoberta de serviços no AWS Cloud Map

Siga estas etapas para criar o namespace de descoberta de serviços e o serviço de descoberta de serviços.

1. Crie um namespace privado de descoberta de serviços do Cloud Map. Este exemplo cria um namespace denominado `tutorial`. Substitua *vpc-abcd1234* pelo ID de uma das VPCs existentes.

```
aws servicediscovery create-private-dns-namespace \
      --name tutorial \
      --vpc vpc-abcd1234
```

   A saída deste comando é a seguinte.

```
{
    "OperationId": "h2qe3s6dxftvvt7riu6lfy2f6c3jlhf4-je6chs2e"
}
```

2. Usando o `OperationId` do resultado da etapa anterior, verifique se o namespace privado foi criado com êxito. Anote o ID de namespace, pois ele será usado em comandos subsequentes.

```
aws servicediscovery get-operation \
      --operation-id h2qe3s6dxftvvt7riu6lfy2f6c3jlhf4-je6chs2e
```

   A saída é a seguinte:

```
{
    "Operation": {
        "Id": "h2qe3s6dxftvvt7riu6lfy2f6c3jlhf4-je6chs2e",
```

```
        "Type": "CREATE_NAMESPACE",
        "Status": "SUCCESS",
        "CreateDate": 1519777852.502,
        "UpdateDate": 1519777856.086,
        "Targets": {
           "NAMESPACE": "ns-uejictsjen2i4eeg"
        }
    }
}
```

3. Usando o ID do NAMESPACE da saída da etapa anterior, crie um serviço de descoberta de serviços. Este exemplo cria um serviço denominado `myapplication`. Anote o ID do serviço e o ARN, pois você os usará em comandos subsequentes.

```
aws servicediscovery create-service \
      --name myapplication \
      --dns-config "NamespaceId="ns-
uejictsjen2i4eeg",DnsRecords=[{Type="A",TTL="300"}]" \
      --health-check-custom-config FailureThreshold=1
```

A saída é a seguinte:

```
{
    "Service": {
       "Id": "srv-utcrh6wavdkggqtk",
        "Arn": "arn:aws:servicediscovery:region:aws_account_id:service/srv-
utcrh6wavdkggqtk",
        "Name": "myapplication",
        "DnsConfig": {
            "NamespaceId": "ns-uejictsjen2i4eeg",
            "DnsRecords": [
                {
                    "Type": "A",
                    "TTL": 300
                }
            ]
        },
        "HealthCheckCustomConfig": {
            "FailureThreshold": 1
        },
        "CreatorRequestId": "e49a8797-b735-481b-a657-b74d1d6734eb"
    }
```

```
}
```

Etapa 2: criar os recursos do Amazon ECS

Siga estas etapas para criar seu cluster, definição de tarefa e serviço do Amazon ECS:

1. Crie um cluster do Amazon ECS. Este exemplo cria um cluster que denominado `tutorial`.

```
aws ecs create-cluster \
      --cluster-name tutorial
```

2. Registre uma definição de tarefa compatível com o Fargate e use o modo de rede `awsvpc`. Siga estas etapas:

   a. Crie um arquivo denominado `fargate-task.json` com o conteúdo da seguinte definição de tarefa.

```
{
    "family": "tutorial-task-def",
        "networkMode": "awsvpc",
        "containerDefinitions": [
            {
                "name": "sample-app",
                "image": "public.ecr.aws/docker/library/httpd:2.4",
                "portMappings": [
                    {
                        "containerPort": 80,
                        "hostPort": 80,
                        "protocol": "tcp"
                    }
                ],
                "essential": true,
                "entryPoint": [
                    "sh",
                    "-c"
                ],
                "command": [
                    "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style>
 </head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample
 App</h1> <h2>Congratulations!</h2> <p>Your application is now running on a
```

```
 container in Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/
htdocs/index.html && httpd-foreground\""
                ]
            }
        ],
        "requiresCompatibilities": [
            "FARGATE"
        ],
        "cpu": "256",
        "memory": "512"
}
```

b. Registre a definição de tarefa usando `fargate-task.json`.

```
aws ecs register-task-definition \
      --cli-input-json file://fargate-task.json
```

3. Crie um serviço do ECS seguindo estas etapas:

a. Crie um arquivo denominado `ecs-service-discovery.json` com o conteúdo do serviço do ECS que você está criando. Este exemplo usa a definição de tarefa criada na etapa anterior. Uma `awsvpcConfiguration` é necessária, pois o exemplo de definição de tarefa usa o modo de rede `awsvpc`.

Ao criar o serviço do ECS, especifique o tipo de inicialização do Fargate e a versão `LATEST` da plataforma que oferece suporte à descoberta de serviços. Quando o serviço de descoberta de serviços é criado no AWS Cloud Map, `registryArn` é o ARN retornado. `securityGroups` e `subnets` devem pertencer à VPC utilizada para criar o namespace do Cloud Map. É possível obter os IDs de grupos de segurança e sub-redes no console da Amazon VPC.

```
{
    "cluster": "tutorial",
    "serviceName": "ecs-service-discovery",
    "taskDefinition": "tutorial-task-def",
    "serviceRegistries": [
       {
          "registryArn":
 "arn:aws:servicediscovery:region:aws_account_id:service/srv-utcrh6wavdkggqtk"
       }
    ],
    "launchType": "FARGATE",
```

```
    "platformVersion": "LATEST",
    "networkConfiguration": {
       "awsvpcConfiguration": {
          "assignPublicIp": "ENABLED",
          "securityGroups": [ "sg-abcd1234" ],
          "subnets": [ "subnet-abcd1234" ]
       }
    },
    "desiredCount": 1
}
```

b. Crie seu serviço do ECS usando `ecs-service-discovery.json`.

```
aws ecs create-service \
      --cli-input-json file://ecs-service-discovery.json
```

Etapa 3: verificar a descoberta de serviços no AWS Cloud Map

Verifique se tudo foi criado corretamente consultando suas informações da descoberta de serviços. Depois que a descoberta de serviços estiver configurada, será possível usar operações de API do AWS Cloud Map ou chamar `dig` de uma instância na sua VPC. Siga estas etapas:

1. Usando o ID de serviço de descoberta de serviço, liste as instâncias da descoberta de serviço. Anote o ID da instância (marcado em negrito) para a limpeza de recursos.

```
aws servicediscovery list-instances \
      --service-id srv-utcrh6wavdkggqtk
```

A saída é a seguinte:

```
{
    "Instances": [
        {
            "Id": "16becc26-8558-4af1-9fbd-f81be062a266",
            "Attributes": {
                "AWS_INSTANCE_IPV4": "172.31.87.2"
                "AWS_INSTANCE_PORT": "80",
                "AVAILABILITY_ZONE": "us-east-1a",
                "REGION": "us-east-1",
                "ECS_SERVICE_NAME": "ecs-service-discovery",
                "ECS_CLUSTER_NAME": "tutorial",
```

```
                "ECS_TASK_DEFINITION_FAMILY": "tutorial-task-def"
            }
        }
    ]
}
```

2. Use o namespace de descoberta de serviços, o serviço e parâmetros adicionais, como o nome do cluster do ECS, para consultar detalhes sobre as instâncias de descoberta de serviços.

```
aws servicediscovery discover-instances \
      --namespace-name tutorial \
      --service-name myapplication \
      --query-parameters ECS_CLUSTER_NAME=tutorial
```

3. Os registros DNS criados na zona hospedada do Route 53 para o serviço de descoberta de serviço podem ser consultados com os comandos da AWS CLI a seguir:

   a. Usando o ID do namespace, obtenha informações sobre o namespace, o que inclui o ID da zona hospedada do Route 53.

   ```
   aws servicediscovery \
         get-namespace --id ns-uejictsjen2i4eeg
   ```

   A saída é a seguinte:

   ```
   {
       "Namespace": {
           "Id": "ns-uejictsjen2i4eeg",
           "Arn": "arn:aws:servicediscovery:region:aws_account_id:namespace/ns-
   uejictsjen2i4eeg",
           "Name": "tutorial",
           "Type": "DNS_PRIVATE",
           "Properties": {
                "DnsProperties": {
                   "HostedZoneId": "Z35JQ4ZFDRYPLV"
               }
           },
           "CreateDate": 1519777852.502,
           "CreatorRequestId": "9049a1d5-25e4-4115-8625-96dbda9a6093"
       }
   }
   ```

b. Usando o ID da zona hospedada do Route 53 da etapa anterior (veja o texto em negrito), obtenha o registro de recurso definido para a zona hospedada.

```
aws route53 list-resource-record-sets \
      --hosted-zone-id Z35JQ4ZFDRYPLV
```

4. Você também pode consultar o DNS de uma instância na sua VPC usando `dig`.

```
dig +short myapplication.tutorial
```

Etapa 4: limpar

Ao concluir este tutorial, limpe os recursos associados para evitar cobranças por recursos não utilizados. Siga estas etapas:

1. Cancele o registro das instâncias do serviço de descoberta de serviços, usando o ID do serviço e o ID da instância anotados anteriormente.

```
aws servicediscovery deregister-instance \
      --service-id srv-utcrh6wavdkggqtk \
      --instance-id 16becc26-8558-4af1-9fbd-f81be062a266
```

A saída é a seguinte:

```
{
    "OperationId": "xhu73bsertlyffhm3faqi7kumsmx274n-jh0zimzv"
}
```

2. Usando o `OperationId` da saída da etapa anterior, verifique se as instâncias do serviço de descoberta de serviços foram canceladas com êxito.

```
aws servicediscovery get-operation \
      --operation-id xhu73bsertlyffhm3faqi7kumsmx274n-jh0zimzv
```

```
{
  "Operation": {
        "Id": "xhu73bsertlyffhm3faqi7kumsmx274n-jh0zimzv",
        "Type": "DEREGISTER_INSTANCE",
        "Status": "SUCCESS",
```

```
        "CreateDate": 1525984073.707,
        "UpdateDate": 1525984076.426,
        "Targets": {
            "INSTANCE": "16becc26-8558-4af1-9fbd-f81be062a266",
            "ROUTE_53_CHANGE_ID": "C5NSRG1J4I1FH",
            "SERVICE": "srv-utcrh6wavdkggqtk"
        }
    }
}
```

3. Exclua o serviço de descoberta de serviços usando o ID do serviço.

```
aws servicediscovery delete-service \
      --id srv-utcrh6wavdkggqtk
```

4. Exclua o namespace de descoberta de serviços usando o ID do namespace.

```
aws servicediscovery delete-namespace \
      --id ns-uejictsjen2i4eeg
```

   A saída é a seguinte:

```
{
    "OperationId": "c3ncqglftesw4ibgj5baz6ktaoh6cg4t-jh0ztysj"
}
```

5. Usando o `OperationId` da saída da etapa anterior, verifique se o namespace da descoberta de serviços foi excluído com êxito.

```
aws servicediscovery get-operation \
      --operation-id c3ncqglftesw4ibgj5baz6ktaoh6cg4t-jh0ztysj
```

   A saída é a seguinte:

```
{
    "Operation": {
        "Id": "c3ncqglftesw4ibgj5baz6ktaoh6cg4t-jh0ztysj",
        "Type": "DELETE_NAMESPACE",
        "Status": "SUCCESS",
        "CreateDate": 1525984602.211,
        "UpdateDate": 1525984602.558,
```

```
        "Targets": {
            "NAMESPACE": "ns-rymlehshst7hhukh",
            "ROUTE_53_CHANGE_ID": "CJP2A2M86XW3O"
        }
    }
}
```

6. Atualize para`0` a contagem desejada para o serviço Amazon ECS. Isso deve ser feito para excluir o serviço na etapa seguinte.

```
aws ecs update-service \
      --cluster tutorial \
      --service ecs-service-discovery \
      --desired-count 0
```

7. Exclua o serviço do Amazon ECS.

```
aws ecs delete-service \
      --cluster tutorial \
      --service ecs-service-discovery
```

8. Exclua o cluster do Amazon ECS.

```
aws ecs delete-cluster \
      --cluster tutorial
```

Use o Amazon VPC Lattice para conectar, observar e proteger seus serviços do Amazon ECS

O Amazon VPC Lattice é um serviço totalmente gerenciado de rede de aplicações que possibilita que os clientes do Amazon ECS observem, protejam e monitorem as aplicações criadas em serviços de computação, VPCs e contas da AWS sem precisar de qualquer alteração no código.

O VPC Lattice usa grupos de destino, que são uma coleção de recursos computacionais. Esses destinos executam a aplicação ou serviço e podem ser instâncias do Amazon EC2, endereços IP, funções do Lambda e Application Load Balancers. Ao associarem seus serviços do Amazon ECS a um grupo de destino do VPC Lattice, os clientes agora podem habilitar tarefas do Amazon ECS como destinos IP no VPC Lattice. O Amazon ECS registra automaticamente as tarefas no grupo de destino do VPC Lattice quando as tarefas do serviço registrado são iniciadas.

> **Note**
>
> Ao usar cinco configurações de VPC Lattice, seu tempo de implantação pode ser um pouco maior do que ao usar menos configurações.

Uma regra de receptor é usada para encaminhar o tráfego para um grupo de destino especificado quando as condições são atendidas. Um receptor verifica solicitações de conexão usando o protocolo na porta que você configurou. Um serviço roteia solicitações para seus destinos registrados com base nas regras definidas por você ao configurar seu receptor.

O Amazon ECS também substitui automaticamente uma tarefa quando ela não está íntegra de acordo com as verificações de saúde do VPC Lattice. Uma vez associados ao VPC Lattice, os clientes do Amazon ECS também podem aproveitar muitos outros recursos de conectividade entre computadores, segurança e observabilidade no VPC Lattice, como conexão a serviços entre clusters, VPCs e contas com AWS Resource Access Manager, integração IAM para autorização e autenticação e recursos avançados de gerenciamento de tráfego.

Os clientes do Amazon ECS podem se beneficiar do VPC Lattice conforme mostrado a seguir.

- Maior produtividade do desenvolvedor: o VPC Lattice aumenta a produtividade do desenvolvedor, permitindo que você se concentre na criação de recursos enquanto o VPC Lattice lida com os desafios de rede, segurança e observabilidade de forma uniforme em todas as plataformas de computação.
- Melhor postura de segurança: o VPC Lattice permite que seus desenvolvedores autentiquem e protejam facilmente a comunicação entre aplicações e plataformas de computação, apliquem criptografia em trânsito e apliquem controles de acesso granulares com políticas de autenticação do VPC Lattice. Isso permite que você adote uma postura de segurança mais forte que atenda aos principais requisitos regulatórios e de conformidade do setor.
- Escalabilidade e resiliência aprimoradas de aplicações: o VPC Lattice permite criar uma rede de aplicações implantadas com recursos como roteamento, autenticação, autorização e monitoramento baseados em caminhos, cabeçalhos e métodos. Esses benefícios são fornecidos sem sobrecarga de recursos nas workloads e podem suportar implantações de vários clusters que geram milhões de solicitações por segundo sem adicionar latência significativa.
- Flexibilidade de implantação com infraestrutura heterogênea: o VPC Lattice fornece recursos consistentes em todos os serviços de computação, como Amazon ECS, Fargate, Amazon EC2,

Amazon EKS e Lambda e permite à sua organização a flexibilidade de escolher a infraestrutura adequada para cada aplicação.

Como o VPC Lattice funciona com outros serviços do Amazon ECS

Usar o VPC Lattice com o Amazon ECS pode mudar a forma como você usa outros serviços do Amazon ECS, enquanto alguns permanecem os mesmos.

Application Load Balancers

Não é mais necessário criar um Application Load Balancer específico para usar com o tipo de grupo de destino do Application Load Balancer no VPC Lattice, que então se vincula ao serviço do Amazon ECS. Em vez disso, basta configurar o serviço do Amazon ECS com um grupo de destino do VPC Lattice. Você também pode optar por usar o Application Load Balancer com o Amazon ECS ao mesmo tempo.

Implantações contínuas do Amazon ECS

Somente as implantações contínuas do Amazon ECS funcionam com o VPC Lattice, e o Amazon ECS insere tarefas com segurança e as remove dos serviços durante a implantação. Não há suporte a implantação de código nem a implantações azuis/verdes.

Para saber mais sobre o VPC Lattice, consulte o Guia do usuário do Amazon VPC Lattice.

Criar um serviço que usa o VPC Lattice

Você pode usar o AWS Management Console ou o AWS CLI para criar um serviço com o VPC Lattice.

Pré-requisitos

Antes de começar este tutorial, certifique-se de que os seguintes pré-requisitos foram atendidos:

- A versão mais recente da AWS CLI está instalada e configurada. Para obter mais informações, consulte Instalar a AWS Command Line Interface.

> **Note**
>
> É possível usar endpoints de serviço de pilha dupla para interagir com o Amazon ECS via AWS CLI, SDKs e API do Amazon ECS sobre IPv4 e IPv6. Para obter mais informações, consulte Usar endpoints de pilha dupla do Amazon ECS.

- As etapas descritas em Configuração para usar o Amazon ECS estão concluídas.
- O usuário do IAM tem as permissões necessárias especificadas no exemplo de política do IAM de AmazonECS_FullAccess.

Criar um serviço que usa o VPC Lattice com o AWS Management Console

Siga estas etapas para criar um serviço com o VPC Lattice usando o AWS Management Console.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página de navegação, escolha Clusters.
3. Na página Clusters, selecione o cluster no qual o serviço será criado.
4. Na guia Services (Serviços), escolha Create (Criar).

   Se você nunca criou um serviço antes, siga as etapas encontradas em Criar um serviço do Amazon ECS usando o console e continue com estas etapas ao chegar à seção VPC Lattice.
5. Selecione o botão para Ativar o VPC Lattice.
6. Para usar um perfil existente, em Perfil da infraestrutura do ECS para Amazon ECS, escolha um que você já criou para usar ao criar o grupo de destino da VPC Lattice. Para criar um novo perfil, selecione Criar perfil da infraestrutura do ECS.
7. Escolha a VPC.

   A VPC depende do modo de rede que você selecionou ao registrar sua definição de tarefa. Se você usar o modo `host` ou `network` com o tipo de inicialização do EC2, escolha aqui a VPC.

   Para o modo `awsvpc`, a VPC será selecionada automaticamente com base na VPC que você escolheu em Rede e não poderá ser alterada.
8. Em Grupos de destino, escolha um ou mais grupos de destino. Escolha pelo menos um grupo de destino e, no máximo, cinco. Escolha Adicionar grupo de destino para adicionar mais grupos de destino. Escolha o Nome da porta, o Protocolo e a Porta para cada grupo de destino escolhido. Para excluir um grupo de destino, escolha Remover.

   > **Note**
   >
   > - Se desejar adicionar grupos de destino existentes, será necessário usar a AWS CLI. Para obter instruções sobre como adicionar grupos de destino usando o AWS CLI, consulte register-targets na Referência do AWS Command Line Interface.

- Embora um serviço do VPC Lattice possa ter vários grupos de destino, cada grupo de destino só pode ser adicionado a um serviço.

9. Nesse ponto, navegue até o console do VPC Lattice para continuar a configuração. É aqui que você inclui seus novos grupos de destino na ação padrão do receptor ou nas regras de um serviço do VPC Lattice existente.

   Para obter mais informações, consulte Regras de receptor para o serviço do VPC Lattice.

> **Important**
>
> É necessário permitir o prefixo `vpc-lattice` da regra de entrada em seu grupo de segurança. Caso contrário, as tarefas e verificações de integridade poderão falhar.

Criar um serviço que usa o VPC Lattice com a AWS CLI

Use a AWS CLI para criar um serviço com o VPC Lattice. Substitua cada *espaço reservado para entrada do usuário* por suas próprias informações.

1. Crie um arquivo de configuração de grupo de destino. O exemplo a seguir é denominado `tg-config.json`

```
{
    "ipAddressType": "IPV4",
    "port": 443,
    "protocol": "HTTPS",
    "protocolVersion": "HTTP1",
    "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
}
```

2. Use o comando a seguir para criar um grupo de destino do VPC Lattice.

```
aws vpc-lattice create-target-group \
    --name my-lattice-target-group-ip \
    --type IP \
    --config file://tg-config.json
```

   Resultado do exemplo:

```
{
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/
tg-0eaa4b9ab4EXAMPLE",
    "config": {
        "healthCheck": {
            "enabled": true,
            "healthCheckIntervalSeconds": 30,
            "healthCheckTimeoutSeconds": 5,
            "healthyThresholdCount": 5,
            "matcher": {
                "httpCode": "200"
            },
            "path": "/",
            "protocol": "HTTPS",
            "protocolVersion": "HTTP1",
            "unhealthyThresholdCount": 2
        },
        "ipAddressType": "IPV4",
        "port": 443,
        "protocol": "HTTPS",
        "protocolVersion": "HTTP1",
        "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
    },
    "id": "tg-0eaa4b9ab4EXAMPLE",
    "name": "my-lattice-target-group-ip",
    "status": "CREATE_IN_PROGRESS",
    "type": "IP"
}
```

3. O arquivo JSON a seguir, chamado *ecs-service-vpc-lattice.json*, é um exemplo usado para anexar um serviço do Amazon ECS a um grupo de destino do VPC Lattice. O `portName` no exemplo abaixo é o mesmo que você definiu no campo `name` da propriedade `portMappings` da definição da sua tarefa.

```
{
    "serviceName": "ecs-service-vpc-lattice",
    "taskDefinition": "ecs-task-def",
        "vpcLatticeConfigurations": [
        {
            "targetGroupArn": "arn:aws:vpc-lattice:us-
west-2:123456789012:targetgroup/tg-0eaa4b9ab4EXAMPLE",
            "portName": "testvpclattice",
```

```
            "roleArn": "arn:aws:iam::123456789012:role/
ecsInfrastructureRoleVpcLattice"
        }
    ],
    "desiredCount": 5,
    "role": "ecsServiceRole"
}
```

Use o comando a seguir para criar um serviço do Amazon ECS e anexá-lo ao grupo de destino do VPC Lattice usando o exemplo de json acima.

```
aws ecs create-service \
    --cluster clusterName \
    --serviceName ecs-service-vpc-lattice \
    --cli-input-json file://ecs-service-vpc-lattice.json
```

> **Note**
>
> A VPC Lattice não é compatível com o Amazon ECS Anywhere.

Como proteger as tarefas do Amazon ECS de serem encerradas por eventos de redução horizontal da escala

Você pode usar a proteção de tarefa na redução de escala do Amazon ECS para impedir que as tarefas sejam encerras por eventos de redução de escala horizontal do ajuste de escala automático ou implantações do serviço.

Certas aplicações exigem um mecanismo para impedir que tarefas de missão crítica sejam encerradas por meio de eventos de redução da escala horizontalmente durante períodos de baixa utilização ou durante implantações de serviços. Por exemplo:

- Você tem uma aplicação assíncrona de processamento de filas, como um trabalho de transcodificação de vídeo, em que algumas tarefas precisam ser executadas por horas, mesmo quando a utilização cumulativa do serviço é baixa.
- Você tem uma aplicação de jogos que executa servidores de jogos como tarefas do Amazon ECS que precisam continuar em execução mesmo que todos os usuários tenham se desconectado para reduzir a latência de inicialização de uma reinicialização do servidor.

- Quando implantar uma nova versão de código, você precisa que as tarefas continuem em execução, pois seria caro reprocessar.

Para proteger as tarefas pertencentes ao seu serviço de encerramentos em um evento de redução da escala na horizontal, defina o atributo `ProtectionEnabled` como `true`. Ao definir `ProtectionEnabled` como verdadeiro, as tarefas ficam protegidas por 2 horas por padrão. Em seguida, é possível personalizar o período de proteção usando o atributo `ExpiresInMinutes`. É possível proteger suas tarefas por no mínimo um minuto e até no máximo 2.880 minutos (48 horas). Se estiver usando o AWS CLI, você poderá especificar a opção `--protection-enabled`.

Depois que uma tarefa terminar o trabalho necessário, será possível definir o atributo `ProtectionEnabled` como `false`, permitindo que a tarefa seja encerrada por eventos subsequentes de redução da escala na horizontal. Se estiver usando o AWS CLI, você poderá especificar a opção `--no-protection-enabled`.

Mecanismos de proteção de tarefa na redução da escala horizontalmente

É possível definir e obter proteção de tarefa na redução da escala na horizontal usando o endpoint do agente de contêiner do Amazon ECS ou a API do Amazon ECS.

- Endpoint do agente de contêiner do Amazon ECS

  Recomendamos o uso do endpoint do agente de contêiner do Amazon ECS para tarefas que possam autodeterminar a necessidade de proteção. Use essa abordagem para workloads baseadas em filas ou em processamento de tarefas.

  Quando um contêiner começa a processar o trabalho, por exemplo, consumindo uma mensagem do SQS, você pode definir o atributo `ProtectionEnabled` por meio do caminho `$ECS_AGENT_URI/task-protection/v1/state` do endpoint de proteção de tarefa na redução da escala horizontalmente de dentro do contêiner. O Amazon ECS não encerrará essa tarefa durante eventos de redução da escala horizontalmente. Depois que a tarefa concluir o trabalho, você pode limpar o atributo `ProtectionEnabled` usando o mesmo endpoint, tornando a tarefa elegível para encerramento durante eventos subsequentes de redução horizontal na escala.

  Para obter mais informações sobre o endpoint do agente de container do Amazon ECS, consulte Endpoint de Proteção de Tarefa na Redução de Escala do Amazon ECS.

- API do Amazon ECS

É possível usar a API do Amazon ECS para definir e recuperar a proteção de tarefa na redução horizontal da escala se a aplicação tiver um componente que rastreie o status das tarefas ativas. Use `UpdateTaskProtection` para marcar uma ou mais tarefas como protegidas. Use `GetTaskProtection` para recuperar o status da proteção.

Um exemplo dessa abordagem é o caso de a sua aplicação hospedar sessões de servidores de jogos como tarefas do Amazon ECS. Quando um usuário faz login em uma sessão no servidor (tarefa), você pode marcar a tarefa como protegida. Depois que o usuário se desconecta, é possível desmarcar a proteção especificamente para essa tarefa ou desmarcar periodicamente a proteção para tarefas semelhantes que não tenham mais sessões ativas, dependendo da sua necessidade de manter servidores ociosos.

Para obter mais informações, consulte UpdateTaskProtection e GetTaskProtection na Referência de APIs do Amazon Elastic Container Service.

É possível combinar as duas abordagens. Por exemplo, use o endpoint do agente do Amazon ECS para definir a proteção de tarefas de dentro de um contêiner e use a API do Amazon ECS para remover a proteção de tarefas do seu serviço de controlador externo.

Considerações

Considere os seguintes pontos antes de usar a proteção de tarefa na redução da escala horizontalmente:

- A proteção de redução horizontal da escala de tarefas é compatível apenas com tarefas implantadas de um serviço.
- Recomendamos o uso do endpoint do agente de contêiner do Amazon ECS porque o agente do Amazon ECS tem mecanismos de repetição integrados e uma interface mais simples.
- É possível redefinir o período de validade da proteção de tarefa na redução da escala na horizontal invocando `UpdateTaskProtection` para uma tarefa que já tenha a proteção ativada.
- Determine quanto tempo uma tarefa precisaria para concluir o trabalho necessário e defina a propriedade `expiresInMinutes` adequadamente. Se você definir a validade da proteção por mais tempo do que o necessário, incorrerá em custos e enfrentará atrasos na implantação de novas tarefas.
- Há suporte para a proteção contra redução de escala na horizontal no agente de contêiner do Amazon ECS `1.65.0` ou posterior.

É possível adicionar suporte a esse recurso em instâncias do Amazon EC2 usando versões mais antigas do agente de contêiner do Amazon ECS atualizando o agente para a versão mais recente. Para obter mais informações, consulte Atualizar o agente de contêiner do Amazon ECS.

- Considerações de implantação:
  - Se o serviço estiver usando uma atualização contínua, novas tarefas serão criadas, mas as tarefas que executem versões mais antigas não serão encerradas até que `protectionEnabled` seja desmarcada ou expire. É possível ajustar o parâmetro `maximumPercentage` na configuração de implantação para um valor que permita que novas tarefas sejam criadas quando tarefas antigas estiverem protegidas.
  - Se uma atualização azul/verde for aplicada, a implantação azul com tarefas protegidas não será removida se as tarefas tiverem `protectionEnabled`. O tráfego será desviado para as novas tarefas que surgirem e as tarefas mais antigas só serão removidas quando `protectionEnabled` não estiver definida ou quando expirar. Dependendo do tempo limite das atualizações do CodeDeploy ou do CloudFormation, a implantação pode expirar e as tarefas azuis antigas ainda podem estar presentes.
  - Caso use o CloudFormation, a update-stack tem um tempo limite de três horas. Portanto, se você definir a proteção de tarefas por mais de três horas, a implantação do CloudFormation poderá resultar em falha e reversão.

    Durante o tempo em que suas tarefas antigas estão protegidas, a pilha do CloudFormation exibirá `UPDATE_IN_PROGRESS`. Se a proteção de tarefa na redução da escala na horizontal for removida ou expirar dentro da janela de três horas, sua implantação será bem-sucedida e passará para o status `UPDATE_COMPLETE`. Se a implantação ficar presa em `UPDATE_IN_PROGRESS` por mais de três horas, apresentará falha, mostrará o estado `UPDATE_FAILED` e, em seguida, será revertida para o antigo conjunto de tarefas.
  - O Amazon ECS enviará eventos de serviço quando as tarefas protegidas estiverem impedindo que uma implantação (contínua ou azul/verde) atinja o estado estacionário para que você possa tomar medidas corretivas. Ao tentar atualizar o status de proteção de uma tarefa, se você receber a mensagem de erro `DEPLOYMENT_BLOCKED`, isso significa que o serviço tem mais tarefas protegidas do que a contagem desejada de tarefas para o serviço. Para corrigir esse erro, execute um dos seguintes procedimentos:
    - Aguarde até que a proteção da tarefa atual expire. Em seguida, defina a proteção da tarefa.
    - Determine quais tarefas podem ser interrompidas. Em seguida, use `UpdateTaskProtection` com a opção `protectionEnabled` definida como `false` para essas tarefas.

- Aumente a contagem de tarefas desejada do serviço para mais do que o número de tarefas protegidas.

Permissões do IAM necessárias para a proteção de tarefa na redução da escala horizontalmente

A tarefa deve ter o perfil de tarefa do Amazon ECS com as seguintes permissões:

- `ecs:GetTaskProtection`: permite que o agente de contêiner do Amazon ECS chame `GetTaskProtection`.
- `ecs:UpdateTaskProtection`: permite que o agente de contêiner do Amazon ECS chame `UpdateTaskProtection`.

Endpoint de Proteção de Tarefa na Redução de Escala do Amazon ECS

O agente de contêiner do Amazon ECS injeta automaticamente a variável de ambiente `ECS_AGENT_URI` nos contêineres de tarefas do Amazon ECS para fornecer um método de interação com o endpoint da API do agente de contêiner.

Recomendamos o uso do endpoint do agente de contêiner do Amazon ECS para tarefas que possam autodeterminar a necessidade de proteção.

Quando um contêiner começa a processar o trabalho, você pode definir o atributo `protectionEnabled` por meio do caminho `$ECS_AGENT_URI/task-protection/v1/state` do endpoint de proteção de tarefa na redução de escala de dentro do contêiner.

Use uma solicitação PUT para esse URI de dentro de um contêiner para definir a proteção de tarefa na redução da escala. Uma solicitação GET para esse URI retorna o status atual da proteção de uma tarefa.

Parâmetros de solicitação da proteção de tarefa na redução de escala

É possível definir a proteção de tarefa na redução da escala na horizontal usando o endpoint `${ECS_AGENT_URI}/task-protection/v1/state` com os parâmetros de solicitação a seguir.

`ProtectionEnabled`

Especifique `true` para que uma tarefa receba proteção. Especifique `false` para remover a proteção e tornar a tarefa elegível para encerramento.

Tipo: booliano

Obrigatório: Sim

`ExpiresInMinutes`

O número de minutos em que a tarefa é protegida. É possível especificar, no mínimo, de 1 minuto a 2.880 minutos (48 horas). Durante esse período, a tarefa não será encerrada por eventos de redução da escala na horizontal decorrentes de ajuste de escala automático do serviço ou de implantações. Após esse período, o parâmetro `protectionEnabled` será definido como `false`

Se você não especificar o período, a tarefa será automaticamente protegida por 120 minutos (2 horas).

Tipo: inteiro

Obrigatório: Não

Os exemplos a seguir mostram como definir uma proteção de tarefa na redução da escala horizontalmente com diferentes durações.

Exemplo de como proteger uma tarefa com o período padrão

Este exemplo mostra como proteger uma tarefa com o período padrão de duas horas.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true}'
```

Exemplo de como proteger uma tarefa por 60 minutos

Este exemplo mostra como proteger uma tarefa por 60 minutos usando o parâmetro `expiresInMinutes`.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true,"ExpiresInMinutes":60}'
```

Exemplo de como proteger uma tarefa por 24 horas

Este exemplo mostra como proteger uma tarefa por 24 horas usando o parâmetro `expiresInMinutes`.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true,"ExpiresInMinutes":1440}'
```

Exemplos de contêineres do Windows

Para contêineres do Windows, você pode usar o cmdlet `Invoke-RestMethod` do PowerShell em vez do curl. Os exemplos a seguir mostram os equivalentes do PowerShell dos comandos curl anteriores.

Exemplo de como proteger uma tarefa de contêiner do Windows com o período de tempo padrão

Este exemplo mostra como proteger uma tarefa com o período de tempo padrão de duas horas usando o PowerShell.

```
Invoke-RestMethod -Uri $env:ECS_AGENT_URI/task-protection/v1/state -Method Put -Body
 '{"ProtectionEnabled":true}' -ContentType 'application/json'
```

Exemplo de como proteger uma tarefa de contêiner do Windows por 60 minutos

Este exemplo mostra como proteger uma tarefa por 60 minutos usando o parâmetro `expiresInMinutes` com o PowerShell.

```
Invoke-RestMethod -Uri $env:ECS_AGENT_URI/task-protection/v1/state -Method Put -Body
 '{"ProtectionEnabled":true,"ExpiresInMinutes":60}' -ContentType 'application/json'
```

Exemplo de como proteger uma tarefa de contêiner do Windows por 24 horas

Este exemplo mostra como proteger uma tarefa por 24 horas usando o parâmetro `expiresInMinutes` com o PowerShell.

```
Invoke-RestMethod -Uri $env:ECS_AGENT_URI/task-protection/v1/state -Method Put -Body
 '{"ProtectionEnabled":true,"ExpiresInMinutes":1440}' -ContentType 'application/json'
```

A solicitação PUT retorna a resposta a seguir.

```
{
  "protection": {
    "ExpirationDate": "2023-12-20T21:57:44.837Z",
    "ProtectionEnabled": true,
```

```
    "TaskArn": "arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0"
  }
}
```

Parâmetros de resposta da proteção de tarefa na redução de escala

As informações a seguir são retornadas do endpoint de proteção contra redução da escala na horizontal da tarefa `${ECS_AGENT_URI}/task-protection/v1/state` na resposta em JSON .

`ExpirationDate`

O período em que a proteção da tarefa expirará. Se a tarefa não estiver protegida, esse valor será nulo.

`ProtectionEnabled`

O status de proteção da tarefa. Se a proteção contra redução da escala horizontalmente estiver ativada para uma tarefa, o valor será `true`. Caso contrário, ele será `false`.

`TaskArn`

O nome de recurso da Amazon (ARN) da tarefa à qual o contêiner pertence.

O exemplo a seguir mostra os detalhes retornados para uma tarefa protegida.

```
curl --request GET ${ECS_AGENT_URI}/task-protection/v1/state
```

Para contêineres do Windows, use o seguinte comando do PowerShell para obter o status da proteção:

```
Invoke-RestMethod -Uri $env:ECS_AGENT_URI/task-protection/v1/state -Method Get
```

```
{
    "protection":{
        "ExpirationDate":"2023-12-20T21:57:44Z",
        "ProtectionEnabled":true,
        "TaskArn":"arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0"
    }
}
```

As informações a seguir são retornadas quando ocorre uma falha.

`Arn`

O nome completo do recurso da Amazon (ARN) da tarefa.

`Detail`

Os detalhes relacionados à falha.

`Reason`

O motivo da falha.

O exemplo a seguir mostra os detalhes retornados para uma tarefa que não está protegida.

```
{
    "failure":{
        "Arn":"arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0",
        "Detail":null,
        "Reason":"TASK_NOT_VALID"
    }
}
```

As informações a seguir são retornadas quando ocorre uma exceção.

`requestID`

O ID da solicitação da AWS para a chamada de API do Amazon ECS que resulta em uma exceção.

`Arn`

O nome completo do recurso da Amazon (ARN) da tarefa ou serviço.

`Code`

O código do erro.

`Message`

A mensagem de erro.

> **Note**
>
> Se surgir um erro `RequestError` ou `RequestTimeout`, provavelmente será um problema de rede. Experimente usar endpoints da VPC para o Amazon ECS.

O exemplo a seguir mostra os detalhes retornados quando ocorre um erro.

```
{
    "requestID":"12345-abc-6789-0123-abc",
    "error":{
        "Arn":"arn:aws:ecs:us-west-2:555555555555:task/my-cluster-
name/1234567890abcdef0",
        "Code":"AccessDeniedException",
        "Message":"User: arn:aws:sts::444455556666:assumed-role/my-ecs-task-
role/1234567890abcdef0 is not authorized to perform: ecs:GetTaskProtection on resource:
 arn:aws:ecs:us-west-2:555555555555:task/test/1234567890abcdef0 because no identity-
based policy allows the ecs:GetTaskProtection action"
    }
}
```

O erro a seguir será exibido se o agente do Amazon ECS não conseguir obter uma resposta do endpoint do Amazon ECS devido a problemas de rede ou inoperância do ambiente de gerenciamento do Amazon ECS.

```
{
  "error": {
    "Arn": "arn:aws:ecs:us-west-2:555555555555:task/my-cluster-name/1234567890abcdef0",
    "Code": "RequestCanceled",
    "Message": "Timed out calling Amazon ECS Task Protection API"
  }
}
```

O erro a seguir é exibido quando o agente do Amazon ECS recebe uma exceção de controle de utilização do Amazon ECS.

```
{
  "requestID": "12345-abc-6789-0123-abc",
  "error": {
    "Arn": "arn:aws:ecs:us-west-2:555555555555:task/my-cluster-name/1234567890abcdef0",
    "Code": "ThrottlingException",
    "Message": "Rate exceeded"
  }
}
```

Usar a injeção de falhas com suas workloads do Amazon ECS e do Fargate

Os clientes podem usar a injeção de falhas com o Amazon ECS no Amazon EC2 e Fargate para testar como suas aplicações respondem a determinados cenários de comprometimento. Esses testes fornecem informações que você pode utilizar para otimizar a performance e a resiliência das aplicações.

Quando a injeção de falhas está habilitada, o agente de contêiner do Amazon ECS permite que as tarefas acessem novos endpoints de injeção de falhas. Você precisa fazer a opção para usar a injeção de falhas definindo o valor do parâmetro de definição de tarefa `enableFaultInjection` como `true`. O valor padrão é `false`.

```
{
    ...
   "enableFaultInjection": true
}
```

> **Note**
>
> A injeção de falhas funciona somente com tarefas que usam os modos de rede `awsvpc` ou `host`.
> O recurso de injeção de falhas não está disponível no Windows.

Para obter informações sobre como habilitar a injeção de falhas no AWS Management Console, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

Será necessário habilitar o recurso para testes no AWS Fault Injection Service. Para obter mais informações, consulte Utilizar as ações do AWS FIS aws:ecs:task.

> **Note**
>
> Se você não usar as novas AMIs otimizadas do Amazon ECS ou tiver uma AMI personalizada, instale as seguintes dependências:
>
> - `tc`
> - Módulo de kernel `sch_netem`

Endpoints para injeção de falhas do Amazon ECS

O agente de contêiner do Amazon ECS injeta automaticamente a variável de ambiente `ECS_AGENT_URI` nos contêineres de tarefas do Amazon ECS para fornecer um método de interação com o endpoint da API do agente de contêiner. Cada endpoint inclui um endpoint `/start`, `/stop` e `/status`. Esses endpoints somente aceitam solicitações de tarefas que habilitaram a injeção de falhas, e cada um tem um limite de taxa de 1 solicitação a cada 5 segundos por contêiner. Exceder o limite resulta em um erro.

> **Note**
>
> O Agente do Amazon ECS `version 1.88.0+` é necessário para utilizar endpoints de injeção de falhas.

Os três endpoints para uso com a injeção de falhas são:

- Endpoint de porta de buraco negro de rede
- Endpoint de perda de pacotes de rede
- Endpoint de latência da rede

Uma solicitação com êxito resulta em um código de resposta `200` com uma mensagem de `running` ao chamar o endpoint `/start`, `stopped` para o endpoint `/stop` e `running` ou `not-running` para o endpoint `/status`.

```
{
    "Status": <string>
}
```

Uma solicitação sem êxito retorna um destes códigos de erro:

- `400`: solicitação inválida
- `409`: a solicitação de injeção de falhas está em conflito com outra falha em execução
- `429`: a solicitação foi limitada
- `500`: o servidor teve um erro inesperado

```
{
```

```
 "Error":  <string message>
}
```

> **Note**
>
> Uma falha de latência de rede ou uma falha de perda de pacote de rede pode ser injetada por vez. A tentativa de injetar mais de uma faz com que a solicitação seja rejeitada.

Endpoint de porta de buraco negro de rede

O endpoint `{ECS_AGENT_URI}/fault/v1/network-blackhole-port` elimina o tráfego de entrada ou saída de uma porta e protocolo específicos no namespace de rede de uma tarefa e é compatível com dois modos:

- awsvpc: as alterações são aplicadas ao namespace da rede de tarefas
- host: as alterações são aplicadas à instância de contêiner do namespace de rede padrão

{ECS_AGENT_URI}/fault/v1/network-blackhole-port/start

Esse endpoint inicia as injeções de falha na porta de buraco negro de rede e inclui os seguintes parâmetros:

Porta

A porta especificada para uso na injeção de falhas na porta de buraco negro.

Tipo: inteiro

Obrigatório: Sim

Protocolo

O protocolo para uso na injeção de falhas na porta de buraco negro.

Tipo: String

Valores válidos: `tcp | udp`

Obrigatório: Sim

TrafficType

O tipo de tráfego utilizado pela injeção de falhas.

Tipo: String

Valores válidos: `ingress | egress`

Obrigatório: Sim

SourcesToFilter

Uma matriz JSON de endereços IPv4 ou blocos CIDR protegidos contra a falha.

Tipo: matriz de strings

Obrigatório: Não

O exemplo a seguir mostra como utilizar o endpoint `start` (substitua os valores em *vermelho* pelos seus próprios):

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-blackhole-port/start

Http method:POST

Request payload:
{
    "Port": 1234,
    "Protocol": "tcp|udp",
    "TrafficType": "ingress|egress"
    "SourcesToFilter": ["${IP1}", "${IP2}", ...],
}
```

{ECS_AGENT_URI}/fault/v1/network-blackhole-port/stop

Esse endpoint interrompe a falha especificada na solicitação. Esse endpoint tem os seguintes parâmetros:

Porta

A porta afetada pela falha que deve ser interrompida.

Tipo: inteiro

Obrigatório: Sim

Protocolo

O protocolo a ser usado para interromper essa falha.

Tipo: String

Valores válidos: `tcp | udp`

Obrigatório: Sim

TrafficType

O tipo de tráfego utilizado pela injeção de falhas.

Tipo: String

Valores válidos: `ingress | egress`

Obrigatório: Sim

O exemplo a seguir mostra como utilizar o endpoint `stop` (substitua os valores em *vermelho* pelos seus próprios):

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-blackhole-port/stop

Http method: POST

Request payload:
{
    "Port": 1234,
    "Protocol": "tcp|udp",
    "TrafficType": "ingress|egress",
}
```

{ECS_AGENT_URI}/fault/v1/network-blackhole-port/status

Esse endpoint é usado para verificar o status da injeção de falhas. Esse endpoint tem os seguintes parâmetros:

Porta

A porta afetada para verificar o status da falha.

Tipo: inteiro

Obrigatório: Sim

Protocolo

O protocolo a ser utilizado ao verificar o status da falha.

Tipo: String

Valores válidos: `tcp | udp`

Obrigatório: Sim

TrafficType

O tipo de tráfego utilizado pela injeção de falhas.

Tipo: String

Valores válidos: `ingress | egress`

Obrigatório: Sim

O exemplo a seguir mostra como utilizar o endpoint `status` (substitua os valores em *vermelho* pelos seus próprios):

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-blackhole-port/status

Http method: POST

Request payload:
{
   "Port": 1234,
   "Protocol": "tcp|udp",
   "TrafficType": "ingress|egress",
}
```

Endpoint de latência da rede

O endpoint `{ECS_AGENT_URI}/fault/v1/network-latency` adiciona atraso e instabilidade à interface de rede da tarefa para o tráfego para uma origem específica. O endpoint é compatível com dois modos:

- awsvpc: as alterações são aplicadas à interface de rede da tarefa
- host: as alterações são aplicadas à interface de rede padrão

{ECS_AGENT_URI}/fault/v1/network-latency/start

Esse endpoint `/start` inicia a injeção de falha de latência da rede e tem os seguintes parâmetros:

DelayMilliseconds

O número de milissegundos de atraso a serem adicionados à interface de rede para uso na injeção de falhas.

Tipo: inteiro

Obrigatório: Sim

JitterMilliseconds

O número de milissegundos de instabilidade a serem adicionados à interface de rede para uso na injeção de falhas.

Tipo: inteiro

Obrigatório: Sim

Origens

Uma matriz JSON de endereços IPv4 ou blocos CIDR que são destinos para uso com a injeção de falhas.

Tipo: matriz de strings

Obrigatório: Sim

SourcesToFilter

Uma matriz JSON de endereços IPv4 ou blocos CIDR protegidos contra a falha. `SourcesToFilter` tem prioridade sobre `Sources`.

Tipo: matriz de strings

Obrigatório: Não

O exemplo a seguir mostra como utilizar o endpoint `/start` (substitua os valores em *vermelho* pelos seus próprios):

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-latency/start

Http method: POST

Request payload:
{
    "DelayMilliseconds": 123,
    "JitterMilliseconds": 123,
    "Sources": ["${IP1}", "${IP2}", ...],
    "SourcesToFilter": ["${IP1}", "${IP2}", ...],
}
```

{ECS_AGENT_URI}/fault/v1/network-latency/stop and /status

O endpoint `{ECS_AGENT_URI}/fault/v1/network-latency/stop` interrompe a falha, e o `{ECS_AGENT_URI}/fault/v1/network-latency/status` verifica o status da falha.

Veja a seguir dois exemplos de solicitações para usar os endpoints `/stop` e `/status`. Ambos utilizam o método `POST` HTTP.

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-latency/stop
```

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-latency/status
```

Endpoint de perda de pacotes de rede

O endpoint `{ECS_AGENT_URI}/fault/v1/network-packet-loss` adiciona perda de pacotes à interface de rede fornecida. Esse endpoint é compatível com dois modos:

- awsvpc: as alterações são aplicadas à interface de rede da tarefa

- host: as alterações são aplicadas à interface de rede padrão

{ECS_AGENT_URI}/fault/v1/network-packet-loss/start

Esse endpoint `/start` inicia a injeção de falha de perda de pacotes de rede e tem os seguintes parâmetros:

LossPercent

A porcentagem de perda de pacotes

Tipo: inteiro

Obrigatório: Sim

Origens

Uma matriz JSON de endereços IPv4 ou blocos CIDR a serem usados para os testes de injeção de falhas.

Tipo: matriz de strings

Obrigatório: Sim

SourcesToFilter

Uma matriz JSON de endereços IPv4 ou blocos CIDR protegidos contra a falha. `SourcesToFilter` tem prioridade sobre `Sources`.

Tipo: matriz de strings

Obrigatório: Não

O exemplo a seguir mostra como utilizar o endpoint `start` (substitua os valores em *vermelho* pelos seus próprios):

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-packet-loss/start

Http method: POST

{
```

```
    "LossPercent": 6,
    "Sources": ["${IP1}", "${IP2}", ...],
    "SourcesToFilter": ["${IP1}", "${IP2}", ...],
}
```

{ECS_AGENT_URI}/fault/v1/network-packet-loss/stop and /status

O endpoint `{ECS_AGENT_URI}/fault/v1/network-packet-loss/stop` interrompe a falha, e o `{ECS_AGENT_URI}/fault/v1/network-packet-loss/status` verifica o status da falha. Somente um de cada tipo de falha é compatível de cada vez.

Veja a seguir dois exemplos de solicitações para usar os endpoints `/stop` e `/status`. Ambos utilizam o método `POST HTTP`.

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-packet-loss/stop
```

```
Endpoint: ${{ECS_AGENT_URI}/fault/v1/network-packet-loss/status
```

Atualizar parâmetros de serviço do Amazon ECS

Após criar um serviço, talvez você precise atualizar os parâmetros do serviço, por exemplo, o número de tarefas.

Quando o agendador de serviços inicia novas tarefas, ele determina o posicionamento das tarefas em seu cluster com a seguinte lógica.

- Determine quais instâncias de contêiner no seu cluster podem oferecer suporte à definição de tarefa do seu serviço. Por exemplo, elas têm os atributos necessários de CPU, memória, portas e de instância de contêiner.
- Por padrão, o agendador de serviços tenta equilibrar as tarefas entre as zonas de disponibilidade dessa maneira, mesmo que você possa escolher uma estratégia de posicionamento diferente.
  - Classifique instâncias de contêiner válidas pelo menor número de tarefas em execução neste serviço na mesma zona de disponibilidade da instância. Por exemplo, se a zona A tiver uma tarefa de serviço em execução e as zonas B e C tiverem nenhuma, as instâncias de contêiner válidas nas zonas B ou C serão consideradas ideais para a colocação.
  - Coloque a nova tarefa de serviço em uma instância de contêiner válida em uma zona de disponibilidade ideal (com base nas etapas anteriores), favorecendo as instâncias de contêiner com o menor número de tarefas em execução neste serviço.

Quando o programador de serviços parar de executar tarefas, ele tentará manter o equilíbrio entre as zonas de disponibilidade do cluster usando a seguinte lógica:

- Classifique as instâncias de contêiner pelo maior número de tarefas em execução para esse serviço na mesma zona de disponibilidade da instância. Por exemplo, se a zona A tiver uma tarefa de serviço em execução e as zonas B e C tiverem duas, as instâncias de contêiner nas zonas B ou C serão consideradas ideais para o encerramento.
- Interrompa a tarefa em uma instância de contêiner em zona de disponibilidade ideal (com base nas etapas anteriores), favorecendo instâncias de contêiner com o número maior de tarefas em execução neste serviço.

Use a lista para determinar se você pode alterar o parâmetro do serviço.

Rebalanceamento de zona de disponibilidade

Indica se o rebalanceamento da zona de disponibilidade deve ou não ser usado para o serviço.

Você pode alterar esse parâmetro para implantações contínuas.

Estratégia de provedor de capacidade

Os detalhes de uma estratégia de provedor de capacidade. Você pode definir um provedor de capacidade ao criar um cluster, executar uma tarefa ou atualizar um serviço.

Quando você usa o Fargate, os provedores de capacidade são `FARGATE` ou `FARGATE_SPOT`.

Quando você usa o Amazon EC2, os provedores de capacidade são grupos do Auto Scaling.

Você pode alterar provedores de capacidade para implantações contínuas e implantações azul/verde.

A lista a seguir apresenta as transições válidas:

- Atualize o tipo de inicialização do Fargate para um provedor de capacidade de grupo do Auto Scaling.
- Atualize o tipo de inicialização do Amazon EC2 para um provedor de capacidade do Fargate.
- Atualize o provedor de capacidade do Fargate para um provedor de capacidade de grupo do Auto Scaling.
- Atualize o provedor de capacidade do Amazon EC2 para um provedor de capacidade do Fargate.

- Atualize o grupo do Auto Scaling ou o provedor de capacidade do Fargate de volta para um tipo de inicialização. Ao usar a CLI ou a API, você transmite uma lista vazia no parâmetro `capacityProviderStrategy`.

Cluster

Não é possível mudar o nome de um cluster.

Configuração de implantação

A configuração de implantação inclui os alarmes do CloudWatch e o disjuntor usado para detectar falhas e a configuração necessária.

O disjuntor de implantação determina se uma implantação de serviço apresentará falha se o serviço não conseguir alcançar um estado estacionário. Se você usar o disjuntor de implantação, uma implantação de serviço transitará para um estado com falha e parará de executar novas tarefas. Se você usar a opção de reversão, quando uma implantação de serviço apresentar falha, o serviço será revertido para a última implantação concluída com êxito.

Quando você atualiza um serviço que usa o disjuntor do Amazon ECS, o Amazon ECS cria uma implantação e uma revisão do serviço. Esses recursos permitem que você visualize informações detalhadas sobre o histórico de serviços. Para obter mais informações, consulte Visualize o histórico de serviços usando implantações de serviços do Amazon ECS.

O programador de serviço usa os parâmetros de porcentagem íntegros mínimos e máximos (na configuração de implantação para o serviço) para determinar a estratégia de implantação.

Se um serviço usar o tipo de implantação de atualização contínua (`ECS`), a porcentagem de integridade mínima representará um limite menor no número de tarefas em um serviço que deverão permanecer no estado `RUNNING` durante uma implantação, como uma porcentagem do número desejado de tarefas (arredondado para cima, para o número inteiro mais próximo). O parâmetro também se aplica enquanto qualquer instância de contêiner estiver no estado `DRAINING` se o serviço contiver tarefas usando o tipo de inicialização do EC2. Use esse parâmetro para implantar sem usar capacidade adicional de cluster. Por exemplo, se o serviço tiver um número desejado de quatro tarefas e uma porcentagem de integridade mínima de 50%, o programador poderá interromper duas tarefas existentes para liberar a capacidade do cluster antes de iniciar duas novas tarefas. O serviço considera as tarefas íntegras para serviços que não usam um balanceador de carga se estiverem no estado `RUNNING`. O serviço considera as tarefas íntegras para serviços que usam um balanceador de carga se estiverem com o status `RUNNING` e forem relatadas como íntegras pelo balanceador de carga. O valor padrão da porcentagem mínima de integridade é 100%.

Se um serviço usar o tipo de implantação de atualização contínua (ECS), o parâmetro porcentagem máxima representará um limite superior no número permitido de tarefas em um serviço no estado PENDING, RUNNING ou STOPPING durante uma implantação, como uma porcentagem do número desejado de tarefas (arredondado para baixo, para o menor número inteiro mais próximo). O parâmetro também se aplica enquanto qualquer instância de contêiner estiver no estado DRAINING se o serviço contiver tarefas usando o tipo de inicialização do EC2. Use esse parâmetro para definir o tamanho do lote de implantação. Por exemplo, se o serviço tiver um número desejado de quatro tarefas e um valor máximo de porcentagem de 200%, o programador poderá iniciar quatro tarefas novas antes de interromper as quatro tarefas mais antigas. Isso é fornecido desde que os recursos de cluster necessários para isso estejam disponíveis. O valor padrão da porcentagem máxima é 200%.

Quando o programador de serviço substitui uma tarefa durante uma atualização, o serviço primeiro eliminará a tarefa do load balancer (se usado) e esperará as conexões se dissiparem. Em seguida, o equivalente de docker stop será enviado para os contêineres executados na tarefa. Isso resulta em um sinal SIGTERM e um tempo limite de 30 segundos, após o qual SIGKILL é enviado, e os contêineres são interrompidos à força. Se o contêiner administra o sinal SIGTERM com tranquilidade e sai dentro de 30 segundos após recebê-lo, nenhum sinal SIGKILL é enviado. O programador de serviço inicia e interrompe tarefas conforme definidas por suas configurações de porcentagem íntegras mínimas e máximas.

O programador de serviços também substitui tarefas consideradas não íntegras após uma falha na verificação de integridade do contêiner ou na verificação de integridade do grupo-alvo do balanceador de carga. Essa substituição depende dos parâmetros de definição do serviço maximumPercent e desiredCount. Se uma tarefa for marcada como não íntegra, o programador de serviços iniciará primeiro uma tarefa de substituição. Então, acontece o seguinte.

- Se o status de integridade da tarefa substituta for HEALTHY, o agendador do serviço interromperá a tarefa que não está íntegra
- Se a tarefa de substituição tiver um status de integridade de UNHEALTHY, o programador interromperá a tarefa de substituição não íntegra ou a tarefa não íntegra existente para que a contagem total de tarefas seja igual a desiredCount.

Se o parâmetro maximumPercent limitar o programador a iniciar uma tarefa de substituição primeiro, o programador interromperá aleatoriamente uma tarefa não íntegra, uma de cada vez, para liberar capacidade e, em seguida, iniciará uma tarefa de substituição. O processo de iniciar e parar continua até que todas as tarefas não íntegras sejam substituídas por tarefas íntegras. Depois que todas as tarefas não íntegras forem substituídas e somente as tarefas íntegras

estiverem em execução, se a contagem total de tarefas exceder a `desiredCount`, as tarefas íntegras serão interrompidas aleatoriamente até que a contagem total de tarefas seja igual a `desiredCount`. Para obter mais informações sobre `maximumPercent` e `desiredCount`, consulte Parâmetros de definição de serviço.

Controlador de implantação

O controlador de implantação a ser usado para o serviço. Existem três tipos de controlador de implantação disponíveis:

- `ECS`
- `EXTERNAL`
- `CODE_DEPLOY`

Ao atualizar um serviço, você pode atualizar o controlador de implantação que ele usa. A lista a seguir apresenta as transições válidas:

- Atualização das implantações azul/verde do CodeDeploy (`CODE_DEPLOY`) para implantações azul/verde ou contínuas do ECS (`ECS`).
- Atualização das implantações azul/verde do CodeDeploy (`CODE_DEPLOY`) para implantações externas (`EXTERNAL`).
- Atualização das implantações azul/verde ou contínuas do ECS (`ECS`) para implantações externas (`EXTERNAL`).
- Atualização das implantações externas (`EXTERNAL`) para implantações azul/verde ou contínuas do ECS (`ECS`).

Considere o seguinte ao atualizar o controlador de implantação de um serviço:

- Você não pode atualizar o controlador de implantação de um serviço do controlador de implantação do `ECS` para nenhum dos outros controladores se ele usar o VPC Lattice ou o Service Connect do Amazon ECS.
- Você não pode atualizar o controlador de implantação de um serviço durante uma implantação de serviço em andamento.
- Você não pode atualizar o controlador de implantação de um serviço para `CODE_DEPLOY` se não houver balanceadores de carga no serviço.
- Você não pode atualizar o controlador de implantação de um serviço do `ECS` para nenhum dos outros controladores se a `deploymentConfiguration` incluir alarmes, um disjuntor de implantação ou uma estratégia de implantação `BLUE_GREEN`. Para obter mais informações, consulte Controladores e estratégias de implantação de serviços do Amazon ECS.

- O valor especificado para `versionConsistency` na definição do contêiner não será usado pelo Amazon ECS se você atualizar o controlador de implantação do serviço do ECS para qualquer um dos outros controladores.
- Se você atualizar o controlador de implantação de um serviço do ECS para qualquer um dos outros controladores, as respostas das APIs `UpdateService` e `DescribeService` ainda retornarão `deployments` em vez de `taskSets`. Para obter mais informações sobre `UpdateService` e `CreateService`, consulte UpdateService e CreateService na Referência de APIs do Amazon ECS.
- Se um serviço usar uma estratégia de implantação de atualização contínua, atualizar o controlador de implantação do ECS para qualquer um dos outros controladores mudará a forma como o valor `maximumPercent` na `deploymentConfiguration` é usado. Em vez de ser usado apenas como um limite no total de tarefas em uma implantação de atualização contínua, `maximumPercent` será usado para substituir tarefas não íntegras. Para obter mais informações sobre como o agendador substitui tarefas não íntegras, consulte Serviços do Amazon ECS.
- Se você atualizar o controlador de implantação de um serviço do ECS para qualquer um dos outros controladores de implantação, qualquer `advancedConfiguration` que você especificar com a configuração do balanceador de carga será ignorado. Para obter mais informações, consulte LoadBalancer e AdvancedConfiguration na Referência de APIs do Amazon ECS.

Ao atualizar o controlador de implantação de um serviço usando o CloudFormation, considere as situações a seguir, dependendo do tipo de migração que você está realizando.

- Se você tiver um modelo do CloudFormation que contenha as informações do controlador de implantação `EXTERNAL` e os recursos `TaskSet` e `PrimaryTaskSet`, e remover os recursos do conjunto de tarefas do modelo ao atualizar de `EXTERNAL` para `ECS`, as chamadas de API `DescribeTaskSet` e `DeleteTaskSet` retornarão um erro 400 após a atualização do controlador de implantação para `ECS`. Isso resultará em uma falha de exclusão do CloudFormation nos recursos do conjunto de tarefas, mesmo que a pilha do CloudFormation passe para o status `UPDATE_COMPLETE`. Para obter mais informações, consulte Recurso removido da pilha, mas não excluído no Guia do usuário do AWS CloudFormation. Para corrigir esse problema, exclua os conjuntos de tarefas diretamente usando a API `DeleteTaskSet` do Amazon ECS. Para obter mais informações sobre como excluir um conjunto de tarefas, consulte DeleteTaskSet na Referência de APIs do Amazon Elastic Container Service.

- Se você estiver migrando de `CODE_DEPLOY` para `ECS` com uma nova definição de tarefa e o CloudFormation realizar uma operação de reversão, a solicitação `UpdateService` do Amazon ECS falhará com o seguinte erro:

```
Resource handler returned message: "Invalid request provided: Unable to update
task definition on services with a CODE_DEPLOY deployment controller. Use AWS
CodeDeploy to trigger a new deployment. (Service: Ecs, Status Code: 400,
Request ID: 0abda1e2-f7b3-4e96-b6e9-c8bc585181ac) (SDK Attempt Count: 1)"
(RequestToken: ba8767eb-c99e-efed-6ec8-25011d9473f0, HandlerErrorCode:
 InvalidRequest)
```

- Depois de uma migração bem-sucedida do controlador de implantação de `ECS` para `EXTERNAL`, você precisará remover manualmente o conjunto de tarefas `ACTIVE`, pois o Amazon ECS não gerenciará mais a implantação. Para obter informações sobre como excluir um conjunto de tarefas, consulte DeleteTaskSet na Referência de APIs do Amazon Elastic Container Service.

Contagem de tarefas desejadas

O número de instanciações da tarefa a posicionar e manter em execução no seu serviço.

Caso queira interromper temporariamente o serviço, defina esse valor como 0. Em seguida, quando estiver tudo pronto para iniciar o serviço, atualize-o com o valor original.

Você pode alterar esse parâmetro para implantações contínuas e implantações azul/verde.

Habilitar tags gerenciadas

Determina se as tags gerenciadas do Amazon ECS devem ou não ser ativadas para as tarefas no serviço.

Somente as tarefas iniciadas após a atualização refletirão a atualização. Para atualizar as tags em todas as tarefas, use a opção de implantação forçada.

Você pode alterar esse parâmetro para implantações contínuas e implantações azul/verde.

Habilitar o ECS Exec

Determina se o Amazon ECS Exec será usado.

Se você não quiser substituir o valor que foi definido quando o serviço foi criado, você poderá defini-lo como null ao executar essa ação.

Você pode alterar esse parâmetro para implantações contínuas.

Período de carência da verificação de integridade

O período, em segundos, durante o qual o programador de serviço do Amazon ECS ignora verificações de integridade não íntegras de Elastic Load Balancing, VPC Lattice e contêiner depois que uma tarefa é iniciada pela primeira vez. Se você não especificar um valor de período de tolerância de verificação de integridade, o valor padrão `0` será usado. Se você não usar nenhuma das verificações de integridade, então `healthCheckGracePeriodSeconds` não será usado.

Se as tarefas do serviço demorarem um pouco para iniciar e responder às verificações de integridade, você poderá especificar um período de carência de até 2.147.483.647 segundos (cerca de 69 anos) para a verificação de integridade. Durante esse tempo, o programador do Amazon ECS Service ignorará o status de verificações de integridade. Esse período de carência pode evitar que o programador do serviço marque tarefas como não íntegras e as interrompa antes de terem tempo de surgir.

Você pode alterar esse parâmetro para implantações contínuas e implantações blue/green.

balanceador de cargas

Você precisa usar um perfil vinculado ao serviço ao atualizar um balanceador de carga.

Uma lista de objetos de balanceador de carga do Elastic Load Balancing. Ela contém o nome do balanceador de carga, o nome do contêiner e a porta do contêiner para acesso do balanceador de carga. O nome do contêiner como aparece em uma definição de contêiner.

O Amazon ECS não atualiza automaticamente os grupos de segurança associados aos balanceadores de carga do Elastic Load Balancing ou às instâncias de contêiner do Amazon ECS.

Quando você adiciona, atualiza ou remove uma configuração de balanceador de carga, o Amazon ECS inicia novas tarefas com a configuração atualizada do Elastic Load Balancing e então interromperá as tarefas antigas quando as novas tarefas estiverem em execução.

Para serviços que usam atualizações contínuas, você pode adicionar, atualizar ou remover grupos de destino do Elastic Load Balancing. Você pode atualizar de um único grupo de destino para vários grupos de destino e de vários grupos de destino para um único grupo de destino.

Para serviços que usam implantações azul/verde, você pode atualizar os grupos de destino do Elastic Load Balancing usando `CreateDeployment` por meio do CodeDeploy. Observe que

não há suporte para vários grupos de destino para implantações azul/verde. Para obter mais informações, consulte Registrar vários grupos de destino em um serviço.

Para serviços que usam o controlador de implantação externo, você pode adicionar, atualizar ou remover balanceadores de carga usando CreateTaskSet. Observe que não há suporte para vários grupos de destino para implantações externas. Para obter mais informações, consulte Registrar vários grupos de destino em um serviço.

Transmita uma lista vazia para remover os balanceadores de carga.

Você pode alterar esse parâmetro para implantações contínuas.

Configuração de rede

A configuração de rede para o serviço.

Você pode alterar esse parâmetro para implantações contínuas.

Restrições de posicionamento

Uma matriz de objetos de restrição de posicionamento de tarefa para atualizar o serviço a ser usado. Se nenhum valor for especificado, as restrições de posicionamento existentes para o serviço permanecerão inalteradas. Se esse valor for especificado, ele substituirá todas as restrições de posicionamento existentes definidas para o serviço. Para remover todas as restrições de posicionamento existentes, especifique uma matriz vazia.

É possível especificar no máximo 10 restrições por tarefa. Esse limite inclui restrições na definição de tarefa e aquelas especificadas em runtime.

Você pode alterar esse parâmetro para implantações contínuas e implantações azul/verde.

Estratégia de posicionamento

Os objetos da estratégia de posicionamento de tarefa para atualizar o serviço a ser usado. Se nenhum valor for especificado, a estratégia de posicionamento existente para o serviço permanecerá inalterada. Se esse valor for especificado, ele substituirá todas a estratégia de posicionamento existente definida para o serviço. Para remover uma estratégia de posicionamento existente, especifique um objeto vazio.

Você pode alterar esse parâmetro para implantações contínuas e implantações azul/verde.

Versão da plataforma

A versão da plataforma do Fargate na qual seu serviço funciona.

Não é possível atualizar um serviço que usa uma versão da plataforma Linux para usar uma versão da plataforma Windows e vice-versa.

Você pode alterar esse parâmetro para implantações contínuas.

Propagar tags

Determina se as tags da definição de tarefa ou do serviço devem ser propagadas para a tarefa. Se nenhum valor for especificado, as tags não serão propagadas.

Somente as tarefas iniciadas após a atualização refletirão a atualização. Para atualizar as tags em todas as tarefas, defina `forceNewDeployment` como `true`, para que o Amazon ECS inicie novas tarefas com as tags atualizadas.

Você pode alterar esse parâmetro para implantações contínuas e implantações azul/verde.

Configuração do Service Connect

A configuração do Amazon ECS Service Connect. Esse parâmetro determina como o serviço se conecta a outros serviços dentro da sua aplicação.

Você pode alterar esse parâmetro para implantações contínuas.

Registros de serviço

Você precisa usar um perfil vinculado ao serviço ao atualizar os registros de serviço.

Os detalhes dos registros de descoberta de serviços a serem atribuídos a este serviço. Para obter mais informações, consulte Descoberta de serviços.

Quando você adiciona, atualiza ou remove a configuração dos registros de serviço, o Amazon ECS inicia novas tarefas com a configuração atualizada dos registros de serviço e então interromperá as tarefas antigas quando as novas tarefas estiverem em execução.

Transmita uma lista vazia para remover os registros de serviço.

Você pode alterar esse parâmetro para implantações contínuas.

Definição de tarefa

A definição e a revisão da tarefa a serem usadas para o serviço.

Se você alterar as portas usadas por contêineres em uma definição de tarefa, talvez seja necessário atualizar os grupos de segurança da instância de contêiner para que funcionem com as portas atualizadas.

Se você atualizar a definição de tarefa para o serviço, o nome do contêiner e a porta do contêiner especificados na configuração do balanceador de carga deverão permanecer na definição da tarefa.

O comportamento de extração de imagem de contêiner é diferente dependendo do tipo de inicialização. Para obter mais informações, consulte um dos seguintes:

- Tipo de inicialização do Fargate para o Amazon ECS
- Tipo de inicialização do EC2 para o Amazon ECS
- Tipo de inicialização externa (Amazon ECS Anywhere) para o Amazon ECS

Você pode alterar esse parâmetro para implantações contínuas.

Configurações de volume

Os detalhes do volume que foi `configuredAtLaunch`. Quando `configuredAtLaunch` é definido como `true` na definição da tarefa, esse parâmetro de serviço configura um volume do Amazon EBS para cada tarefa no serviço a ser criada e anexada durante a implantação. Você pode configurar os parâmetros size, volumeType, IOPS, throughput, snapshot e encryption em ServiceManagedEBSVolumeConfiguration. O `name` do volume precisa corresponder ao `name` da definição da tarefa. Se definido como null, nenhuma nova implantação será acionada. Caso contrário, se essa configuração for diferente da existente, ela acionará uma nova implantação.

Você pode alterar esse parâmetro para implantações contínuas.

Configuração do VPC Lattice

A configuração do VPC Lattice para seu serviço. Isso define como o serviço se integra ao VPC Lattice para comunicação entre serviços.

Você pode alterar esse parâmetro para implantações contínuas.

Considerações sobre o AWS CDK

O AWS CDK não rastreia os estados dos recursos. Ele não sabe se você está criando ou atualizando um serviço. Os clientes devem usar a escotilha de fuga para acessar diretamente o constructo `Service` L1.

Para obter informações sobre escotilhas de fuga, consulte Personalizar constructos da AWS Construct Library no Guia do desenvolvedor AWS Cloud Development Kit (AWS CDK) v2.

Para migrar o serviço existente para o constructo `ecs.Service`, faça o seguinte:

1. Use a escotilha de fuga para acessar o constructo `Service` L1.
2. Defina manualmente as seguintes propriedades no constructo `Service` L1.

   Se o seu serviço usar a capacidade do Amazon EC2:

   - `daemon?`
   - `placementConstraints?`
   - `placementStrategies?`
   - Se você usar o modo de rede `awsvpc`, precisará definir os constructos `vpcSubnets?` e `securityGroups?`.

   Se o seu serviço usar o Fargate:

   - `FargatePlatformVersion`
   - Os constructos `vpcSubnets?` e `securityGroups?`.
3. Configure o `launchType` da seguinte maneira:

```
const cfnEcsService = service.node.findChild('Service') as ecs.CfnService;
cfnEcsService.launchType = "FARGATE";
```

Para migrar de um tipo de execução para um provedor de capacidade, faça o seguinte:

1. Use a escotilha de fuga para acessar o constructo `Service` L1.
2. Adicione o constructo `capacityProviderStrategies?`.
3. Implante o serviço.

Atualizar um serviço do Amazon ECS

Após criar um serviço, talvez você precise atualizar os parâmetros do serviço, por exemplo, o número de tarefas.

Quando você atualiza um serviço que usa o disjuntor do Amazon ECS, o Amazon ECS cria uma implantação e uma revisão do serviço. Esses recursos permitem que você visualize informações detalhadas sobre o histórico de serviços. Para obter mais informações, consulte Visualize o histórico de serviços usando implantações de serviços do Amazon ECS.

Pré-requisitos

Antes de atualizar um serviço, verifique quais parâmetros de serviço podem ser alterados para seu tipo de implantação. Para obter uma lista completa de parâmetros modificáveis, consulte Atualizar parâmetros de serviço do Amazon ECS.

Procedimento

Console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, escolha o cluster.
3. Na página de detalhes do cluster, na seção Serviços, marque a caixa de seleção ao lado do serviço e escolha Atualizar.
4. Para que seu serviço inicie uma nova implantação, selecione Force new deployment (Forçar nova implantação).
5. Em Definição de tarefas, escolha a família de definição de tarefa e a revisão.

   > **Important**
   >
   > O console valida se a família de definição de tarefa e a revisão selecionadas são compatíveis com a configuração de computação definida. Se você receber um aviso, verifique a compatibilidade da definição de tarefa e a configuração de computação selecionadas.

6. Caso escolha Replica (Réplica), em Desired tasks (Tarefas desejadas), especifique o número de tarefas que serão iniciadas e mantidas no serviço.
7. Se você escolheu Replica, para que o Amazon ECS monitore a distribuição de tarefas entre as zonas de disponibilidade e as redistribua quando houver um desequilíbrio, em Rebalanceamento do serviço em zonas de disponibilidade, selecione Rebalanceamento do serviço em zonas de disponibilidade.
8. Em Min running tasks (Mínimo de tarefas em execução), insira o limite inferior do número de tarefas do serviço que devem permanecer no estado `RUNNING` durante uma implantação, como uma porcentagem do número desejado de tarefas (arredondado para o número inteiro superior mais próximo). Para obter mais informações, consulte Configuração da implantação.
9. Em Max running tasks (Máximo de tarefas em execução), insira o limite superior do número de tarefas do serviço que devem permanecer no estado `RUNNING` ou `PENDING` durante uma

implantação, como uma porcentagem do número desejado de tarefas (arredondado para o número inteiro inferior mais próximo).

10. Para configurar como as tarefas são implantadas para seu serviço, expanda Opções de implantação e configure suas opções.

    a. Em Tipo de controlador de implantação, especifique o controlador de implantação do serviço. O console do Amazon ECS é compatível com os seguintes tipos de controladores: ECS.

    b. Em Estratégia de implantação, escolha a estratégia usada pelo Amazon ECS para implantar novas versões do serviço.

    c. Dependendo da estratégia de implantação escolhida, faça o seguinte:

Estratégia de implantação	Etapas
Atualização contínua	a. Em % mínimo de tarefas em execução, especifique um valor percentual mínimo de tarefas que devem ser executadas durante a implantação de um serviço. Para obter mais informações, consulte Implantação de serviços do Amazon ECS por meio da substituição de tarefas. b. Em % máximo de tarefas em execução, especifique um valor percentual máximo de tarefas que devem ser executadas durante a implantação de um serviço. Para obter mais informações,

Estratégia de implantação	Etapas
	consulte Implantação de serviços do Amazon ECS por meio da substituição de tarefas.
Azul/verde	Em Tempo de incorporação, especifique uma duração, em minutos, que as revisões de serviço azul e verde devem ser executadas simultaneamente. Para obter mais informações, consulte Implantações azuis/verdes do Amazon ECS.

d. Para executar funções do Lambda em um estágio do ciclo de vida, em Ganchos do ciclo de vida de implantação, faça o seguinte para cada função exclusiva do Lambda:

   i. Escolha Adicionar.

      Repita o procedimento para cada função exclusiva que você deseja executar.

   ii. Em Função do Lambda, insira o nome da função.

   iii. Em Perfil, escolha o perfil que você criou nos pré-requisitos com as permissões azul/verde.

      Para obter mais informações, consulte Permissões necessárias para funções do Lambda em implantações azul/verde do Amazon ECS.

   iv. Em Estágios do ciclo de vida, selecione os estágios que a função do Lambda executa.

   v. (Opcional) Em Detalhes do gancho, insira um par de chave/valor que forneça informações sobre o gancho.

11. Para configurar como o Amazon ECS detectará e lidará com falhas de implantação, expanda Deployment failure detection(Detecção de falhas de implantação) e escolha suas opções.

a. Para interromper uma implantação quando as tarefas não podem ser iniciadas, selecione Use the Amazon ECS deployment circuit breaker) (Usar o disjuntor de implantação do Amazon ECS.

   Para que o software reverta automaticamente a implantação para o último estado de implantação concluído quando o disjuntor de implantação definir a implantação para um estado de falha, selecione Reverter em caso de falha.

b. Para interromper uma implantação com base nas métricas da aplicação, selecione Usar alarmes do CloudWatch. Em seguida, em Nomes de alarmes do CloudWatch, escolha os alarmes. Para criar um alarme, acesse o console do CloudWatch.

   Para que o software reverta automaticamente a implantação para o último estado de implantação concluído quando um alarme do CloudWatch definir a implantação para um estado de falha, selecione Reverter em caso de falha.

12. Para alterar as opções de computação, expanda Configuração de computação e faça o seguinte:

    a. Para serviços no AWS Fargate, em Platform version (Versão da plataforma), escolha a nova versão.

    b. Para serviços que usam uma estratégia de provedor de capacidade, em Estratégia de provedor de capacidade, faça o seguinte:

       - Para adicionar um provedor de capacidade adicional, escolha Adicionar mais. Em seguida, em Provedor de capacidade, escolha o provedor de capacidade.
       - Para remover um provedor de capacidade, à direita do provedor de capacidade, escolha Remover.

       Um serviço que usa um provedor de capacidade do grupo do Auto Scaling não pode ser atualizado para usar um provedor de capacidade do Fargate. Um serviço que usa um provedor de capacidade do Fargate não pode ser atualizado para usar um provedor de capacidade do grupo do Auto Scaling.

13. (Opcional) Para configurar o ajuste de escala automático do serviço, expanda Ajuste de escala automático do serviço e especifique os parâmetros a seguir. Para usar o ajuste de escala automático preditivo, que utiliza os dados de carga anteriores dos fluxos de tráfego, configure-o após criar o serviço. Para obter mais informações, consulte Usar padrões históricos para escalar os serviços do Amazon ECS com escalabilidade preditiva.

a. Para usar a escalabilidade automática do serviço, selecione Service auto scaling (Escalabilidade automática do serviço).

b. Em Número mínimo de tarefas, insira o limite inferior do número de tarefas a serem usadas pelo ajuste de escala automático. A contagem desejada não será inferior a essa contagem.

c. Em Número máximo de tarefas, insira o limite superior do número de tarefas a serem usadas pelo ajuste de escala automático. A contagem desejada não ultrapassará essa contagem.

d. Escolha o tipo de política. Em Tipo de política de ajuste de escala, escolha uma das opções a seguir.

Para usar este tipo de política	Faça o seguinte	
Monitoramento do objetivo	a. Em Scaling policy type (Tipo de política de escalabilidade), escolha Target tracking (Rastreamento de destino). b. Em Policy name (Nome da política), insira o nome da política. c. Em Métrica de serviço do ECS, selecione uma das métricas a seguir. • ECSServiceAverageCPUUtilization: utilização média da CPU pelo serviço. • ECSServiceAverageMemoryUtilization: utilização média da memória pelo serviço. • ALBRequestCountPerTarget: número de solicitações concluídas por destino em um grupo de destino do Application Load Balancer. d. Em Target value (Valor de destino), insira o	

Para usar este tipo de política	Faça o seguinte	
	valor que o serviço manterá para a métrica selecionada. e. Em Aumentar a escala horizontalmente no período de espera, insira a quantidade de tempo, em segundos, após uma atividade de aumento na escala (adicionar tarefas) que deve ser concluída antes que outra atividade de aumento na escala possa ser iniciada. f. Em Reduzir a escala horizontalmente no período de espera, insira a quantidade de tempo, em segundos, após uma atividade de redução na escala (remover tarefas) que deve ser concluída antes que outra atividade de redução na escala possa ser iniciada. g. Para impedir que a política realize uma atividade de redução da escala na horizonta	

Para usar este tipo de política	Faça o seguinte	
	l, selecione Turn off scale-in (Desativar a redução da escala na horizontal). h. • (Opcional) Selecione Desativar redução horizontal de escala se quiser que a política de ajuste de escala aumente a escala horizontalmente para ampliar o tráfego, mas não precisar que ela seja reduzida quando o tráfego diminuir.	

Para usar este tipo de política	Faça o seguinte	
Escalabilidade em etapas	a. Em Scaling policy type (Tipo de política de escalabilidade), escolha Step scaling (Escalabilidade em etapas). b. Em Nome da política, insira o nome da política. c. Em Alarm name (Nome do alarme), insira um nome exclusivo para o alarme. d. Em Métrica de serviço do Amazon ECS, escolha a métrica a ser usada para o alarme. e. Em Estatística, escolha a estatística de alarme. f. Em Período, escolha o período do alarme. g. Em Condição de alarme, escolha como comparar a métrica selecionada com o limite definido. h. Em Limite para comparar métricas e Período de avaliação para iniciar o alarme, insira o limite usado para o alarme e por	

Para usar este tipo de política	Faça o seguinte	
	quanto tempo avaliar o limite. i. Em Ações de escalabilidade, faça o seguinte: • Em Ação, selecione se deseja adicionar, remover ou definir uma contagem desejada específica para o serviço. • Se você decidir adicionar ou remover tarefas, em Valor, insira o número de tarefas (ou o percentual de tarefas existentes) para adicionar ou remover quando a ação de ajuste de escala for iniciada. Se você optar por definir a contagem desejada, insira o número de tarefas. Em Tipo, selecione se o Valor é um valor inteiro ou percentual da contagem desejada existente. • Em Limite inferior e Limite superior, insira o limite inferior	

Para usar este tipo de política	Faça o seguinte	
	e o limite superior do seu ajuste de escalabilidade de etapas. Por padrão, o limite inferior de uma política de adição é o limite do alarme e o limite superior é mais (+) infinito. Por padrão, o limite superior de uma política de remoção é o limite do alarme e o limite inferior é menos (-) infinito. • (Opcional) Adicione outras opções de escalabilidade. Escolha Adicionar nova ação de escalabilidade e repita as etapas de Ações de escalabilidade. • Em Período de espera, insira a quantidade de tempo, em segundos, para aguardar que uma atividade de ajuste de escala anterior entre em vigor. Para uma política	

Para usar este tipo de política	Faça o seguinte	
	de adição, esse é o momento após uma atividade de aumento horizontal de escala em que a política de ajuste de escala bloqueia as atividades de redução horizontal de escala e limita quantas tarefas podem ser ampliadas por vez. Para uma política de remoção, esse é o momento após uma atividade de redução horizontal de escala que precisa ser concluída antes que outras atividades do mesmo tipo possam iniciar.	

14. (Opcional) Para usar o Service Connect, selecione Turn on Service Connect (Ativar o Service Connect) e especifique o seguinte:

    a. Em Service Connect configuration (Configuração do Service Connect), especifique o modo cliente.

    - Se seu serviço executa uma aplicação cliente de rede que só precisa se conectar a outros serviços no namespace, escolha Somente no lado do cliente.
    - Se seu serviço executa uma aplicação de rede ou de serviço Web e precisa fornecer endpoints para esse serviço e se conectar a outros serviços no namespace, escolha Client and server (Cliente e servidor).

b. Para usar um namespace que não seja o namespace padrão do cluster, em Namespace, escolha o namespace do serviço.

15. Caso a tarefa use um volume de dados compatível com a configuração na implantação, você pode configurar o volume expandindo Volume.

    O nome e o tipo do volume são configurados ao criar uma revisão de definição de tarefa e não podem ser alterados ao atualizar um serviço. Para atualizar o nome e o tipo do volume, você deve criar uma revisão de definição de tarefa e atualizar o serviço usando a nova revisão.

Para configurar esse tipo de volume	Faça o seguinte	
Amazon EBS	a. Em Tipo de volume do EBS, escolha o tipo de volume do EBS que você deseja anexar à tarefa. b. Em Tamanho (GiB), insira um valor válido para o tamanho do volume em gibibytes (GiB). Você pode especificar um tamanho de volume mínimo de 1 GiB e máximo de 16.384 GiB. Esse valor é obrigatório, a menos que você forneça um ID de snapshot. c. Em IOPS, insira o número máximo de operações de entrada e saída (IOPS) que o volume deve oferecer. Esse valor é configurá	

Para configurar esse tipo de volume	Faça o seguinte	
	vel somente em tipos de volumes `io1`, `io2` e `gp3`. d. Em Throughput (MiB/s), insira o throughput que o volume deve fornecer, em mebibytes por segundo (MiBps ou MiB/s). Esse valor é configurável somente no tipo de volume `gp3`. e. Em ID do snapshot, escolha um snapshot de volume existente do Amazon EBS ou insira o ARN de um snapshot se quiser criar um volume usando um snapshot. Você também pode criar um volume vazio sem escolher ou inserir um ID de snapshot. f. Se você especificar um ID de snapshot, poderá especificar uma Taxa de inicialização do volume (MiB/s). Insira um valor entre 100 e 300, em MiB/s, que determinará a rapidez com que os dados são carregados do snapshot especificado usando o ID do	

Para configurar esse tipo de volume	Faça o seguinte	
	snapshot para a criação do volume. g. Em Tipo de sistema de arquivos, escolha o tipo de sistema de arquivos que será usado para armazenamento e recuperação de dados no volume. Você pode escolher o sistema operacional padrão ou um tipo específico de sistema de arquivos. O padrão no Linux é XFS. Em volumes criados de um snapshot, você deve especificar o mesmo tipo de sistema de arquivos que o volume estava usando quando o snapshot foi criado. Se houver uma incompatibilidade no tipo do sistema de arquivos, a tarefa não será iniciada. h. Em Perfil de infraestrutura, escolha um perfil do IAM com as permissões necessárias para permitir que o Amazon ECS gerencie volumes do Amazon EBS em tarefas. Você	

Para configurar esse tipo de volume	Faça o seguinte	
	pode anexar a política gerenciada `AmazonECS Infrastru ctureRole PolicyForVolumes` ao perfil ou usar a política como guia para criar e anexar sua própria política com permissões que atendam às suas necessidades específicas. Para obter mais informações sobre as permissões necessári as, consulte Perfil do IAM de infraestrutura do Amazon ECS. i. Em Criptografia, escolha Padrão se deseja usar a criptografia do Amazon EBS por meio das configurações padrão. Se a conta tiver a Criptografia por padrão configurada, o volume será criptografado com a chave do AWS Key Management Service (AWS KMS) especific ada na configuração. Caso escolha Padrão e a criptografia padrão do Amazon EBS não estiver	

Para configurar esse tipo de volume	Faça o seguinte	
	ativada, o volume não será criptografado. Caso escolha Personalizada, você pode especificar uma AWS KMS key de sua escolha para criptografia de volume. Caso escolha Nenhum, o volume não será criptografado, a menos que você tenha a criptografia por padrão configurada ou crie um volume usando um snapshot criptografado. j. Caso tenha escolhido Personalizada em Criptografia, você deve especificar a AWS KMS key que deseja usar. Em Chave do KMS, escolha uma AWS KMS key ou insira um ARN de chave. Se você optar por criptografar o volume usando uma chave simétrica gerenciada pelo cliente, verifique se tem as permissões corretas definidas na política do AWS KMS	

Para configurar esse tipo de volume	Faça o seguinte	
	key. Para obter mais informações, consulte Data encryption for Amazon EBS volumes. k. (Opcional) Em Tags, você pode adicionar tags ao volume do Amazon EBS propagando tags da definição de tarefa ou do serviço ou fornecendo suas próprias tags. Se quiser propagar tags na definição de tarefa, escolha Definição de tarefa em Propagar tags em. Se quiser propagar tags do serviço, escolha Serviço em Propagar tags de. Caso escolha Não propagar ou se não selecionar um valor, as tags não são propagadas. Se quiser fornecer suas próprias tags, escolha Adicionar tag e forneça a chave e o valor de cada tag adicionada. Para obter mais informações sobre marcação de volumes do Amazon EBS, consulte	

Para configurar esse tipo de volume	Faça o seguinte	
	Tagging Amazon EBS volumes.	

16. (Opcional) Para ajudar a identificar o serviço, expanda a seção Tags (Etiquetas) e configure suas etiquetas.
    - [Adicionar uma tag] Selecione Adicionar tag e faça o seguinte:
      - Em Chave, insira o nome da chave.
      - Em Valor insira o valor da chave.
    - [Remover uma tag] Ao lado da tag, escolha Remove tag (Remover tag).
17. Selecione Atualizar.

AWS CLI

- Executar `update-service`. Para obter informações sobre a execução do comando, consulte update-service na Referência da AWS Command Line Interface.

  O exemplo de `update-service` a seguir atualiza a contagem de tarefas desejada do serviço `my-http-service` para 2.

  Substitua os valores das *`user-input`* pelos seus.

```
aws ecs update-service \
    --cluster MyCluster \
    --service my-http-service \
    --desired-count 2
```

Próximas etapas

Acompanhe sua implantação e visualize seu histórico de serviços para os disjuntores do Amazon ECS. Para obter mais informações, consulte Visualize o histórico de serviços usando implantações de serviços do Amazon ECS.

Exclusão de um serviço do Amazon ECS usando o console

Veja a seguir alguns dos motivos pelos quais você excluiria um serviço:

- A aplicação não é mais necessária
- Você está migrando o serviço para um novo ambiente
- A aplicação não está sendo ativamente usada
- A aplicação está usando mais recursos do que o necessário e você está tentando otimizar custos

O serviço automaticamente tem a escala reduzida verticalmente a zero antes de ser excluído. Recursos de balanceador de carga ou recursos de descoberta serviço associados ao serviço não serão afetados pela exclusão do serviço. Para excluir os recursos do Elastic Load Balancing, consulte um dos seguintes tópicos, dependendo do tipo de balanceador de carga: Excluir um Application Load Balancer ou Excluir um Network Load Balancer.

Quando você exclui um serviço, o Amazon ECS exclui todas as implantações e revisões do serviço.

Ao excluir um serviço, se ainda houver tarefas em execução que exijam limpeza, o status do serviço muda de `ACTIVE` para `DRAINING`, e não será mais possível visualizar o serviço no console ou na operação da API `ListServices`. Depois que todas as tarefas tiverem passado para o status `STOPPING` ou `STOPPED`, o status do serviço mudará de `DRAINING` para `INACTIVE`. Serviços no status `DRAINING` ou `INACTIVE` ainda podem ser visualizados com a operação de API `DescribeServices`.

> **Important**
>
> Caso tente criar um serviço com o mesmo nome de um serviço existente no status `ACTIVE` ou `DRAINING`, você receberá uma mensagem de erro.

Procedimento

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, selecione o cluster para o serviço.
3. Na página Clusters, escolha o cluster.
4. Na página Cluster: ***name***, escolha a guia Services (Serviços).
5. Selecione os serviços e, em seguida, escolha Delete (Excluir).

6. Para excluir um serviço mesmo sem reduzir a escala verticalmente para zero tarefa, selecione Force delete service (Forçar exclusão do serviço).
7. No prompt de confirmação, insira delete e escolha Excluir.

Migrar um ARN curto de um serviço do Amazon ECS para um ARN longo

O Amazon ECS atribui um nome do recurso da Amazon (ARN) exclusivo a cada serviço. Os serviços que foram criados antes de 2021 têm um formato curto de ARN:

`arn:aws:ecs:`*`region`*`:`*`aws_account_id`*`:service/`*`service-name`*

O Amazon ECS alterou o formato de ARN para incluir o nome do cluster. Este é o formato longo de ARN:

`arn:aws:ecs:`*`region`*`:`*`aws_account_id`*`:service/`*`cluster-name`*`/`*`service-name`*

Para que você possa marcar um serviço, ele deve ter o formato longo de ARN.

Você pode migrar um serviço com um formato curto de ARN para um formato longo de ARN sem precisar recriar o serviço. Você pode usar a API, a CLI ou o console. Não é possível desfazer a operação de migração.

O processo de migração é contínuo e garante zero tempo de inatividade para seu serviço. Durante a migração:

- Disponibilidade do serviço: seu serviço continua funcionando normalmente sem interrupção do tráfego ou da funcionalidade.
- Tarefas em execução: as tarefas existentes continuam sendo executadas sem interrupções. As novas tarefas inicializadas após a migração usarão o formato longo do ARN se a configuração `taskLongArnFormat` da conta estiver habilitada.
- Instâncias de contêineres: as instâncias de contêineres não são afetadas pela migração do ARN do serviço e continuam operando normalmente.
- Configuração do serviço: todas as configurações do serviço, incluindo a definição de tarefas, rede e configurações do balanceador de carga, permanecem inalteradas.

Se quiser usar o AWS CloudFormation para marcar um serviço com um formato curto de ARN, deverá migrar o serviço usando a API, a CLI ou o console. Depois que a migração for concluída, você poderá usar o AWS CloudFormation para marcar o serviço.

Se quiser usar o Terraform para marcar um serviço com formato curto de ARN, você deverá migrar o serviço usando a API, a CLI ou o console. Depois de concluída a migração, você poderá usar o Terraform para marcar o serviço.

Depois de concluída a migração, o serviço apresentará as seguintes alterações:

- O formato longo de ARN

  `arn:aws:ecs:`*`region`*`:`*`aws_account_id`*`:service/`*`cluster-name`*`/`*`service-name`*

- Quando você migra usando o console, o Amazon ECS adiciona uma tag ao serviço com a chave definida como "ecs:serviceArnMigratedAt" e o valor definido como o timestamp da migração (formato UTC).

  Essa tag conta para a cota de tags.

- Quando o `PhysicalResourceId` de uma pilha do AWS CloudFormation representa o ARN de um serviço, o valor não é alterado e continua sendo o ARN curto do serviço.

Pré-requisitos

Execute as seguintes operações antes de migrar o ARN do serviço.

1. Para ver se você tem um ARN de serviço curto, veja os detalhes do serviço no console do Amazon ECS (você verá um aviso se o serviço tiver um formato curto de ARN) ou o retorno do parâmetro `serviceARN` de `describe-services`. Quando o ARN não inclui o nome do cluster, você tem um ARN curto. Este é o formato curto de ARN:

   `arn:aws:ecs:`*`region`*`:`*`aws_account_id`*`:service/`*`service-name`*

2. Observe a data de criação.
3. Se você tiver políticas do IAM que usem o formato curto de ARN, atualize-as para o formato longo de ARN.

   Substitua cada *`espaço reservado para entrada do usuário`* por suas próprias informações.

   `arn:aws:ecs:`*`region`*`:`*`aws_account_id`*`:service/`*`cluster-name`*`/`*`service-name`*

   Para obter mais informações, consulte Editar políticas do IAM no Guia do usuário do AWS Identity and Access Management.

4. Se você tiver ferramentas que usem o formato curto de ARN, atualize-as para o formato longo de ARN.

   Substitua cada *espaço reservado para entrada do usuário* por suas próprias informações.

   `arn:aws:ecs:`*region*`:`*aws_account_id*`:service/`*cluster-name*`/`*service-name*

5. Habilitar o formato longo de ARN do serviço. Execute `put-account-setting` com a opção `serviceLongArnFormat` definida como `enabled`. Para obter mais informações, consulte put-account-setting na Referência da API do Amazon Elastic Container Service.

   Execute o comando como usuário-raiz quando seu serviço tiver uma data `createdAt` desconhecida.

   ```
   aws ecs put-account-setting --name serviceLongArnFormat --value enabled
   ```

   Exemplo de saída

   ```
   {
       "setting": {
           "name": "serviceLongArnFormat",
           "value": "enabled",
           "principalArn": "arn:aws:iam::123456789012:role/your-role",
           "type": user
       }
   }
   ```

6. Habilite o formato longo de ARN da tarefa. Essa configuração de conta controla o formato do ARN para novas tarefas que são inicializadas após a conclusão da migração do serviço. Execute `put-account-setting` com a opção `taskLongArnFormat` definida como `enabled`. Para obter mais informações, consulte put-account-setting na Referência da API do Amazon Elastic Container Service.

   Execute o comando como usuário-raiz quando seu serviço tiver uma data `createdAt` desconhecida.

   ```
   aws ecs put-account-setting --name taskLongArnFormat --value enabled
   ```

   Exemplo de saída

```
{
    "setting": {
        "name": "taskLongArnFormat",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:role/your-role",
        "type": user
    }
}
```

> **Note**
>
> A configuração `taskLongArnFormat` não migra diretamente as tarefas existentes. Isso afeta apenas o formato do ARN das novas tarefas criadas depois que a configuração é habilitada. As tarefas em execução existentes mantêm seu formato do ARN atual até serem substituídas por meio de operações normais de serviço (como implantações ou atividades de escalabilidade).

Procedimento

Use o procedimento a seguir para migrar o ARN do serviço.

Console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, escolha o cluster.
3. Na seção Serviços, escolha um serviço que tenha um aviso na coluna ARN.

   A página de detalhes do serviço é exibida.
4. Escolha Migrar para ARN longo.

   A caixa de diálogo Migrar serviço é exibida.
5. Escolha Migrate (Migrar).

CLI

Depois que atender aos pré-requisitos, você poderá marcar o serviço. Execute o seguinte comando:

O Amazon ECS considera que passar o formato longo de ARN em uma solicitação de API `tag-resource` para um serviço com um ARN curto é um sinal para migrar o serviço para usar o formato longo de ARN.

```
aws ecs tag-resource \
    --resource-arn arn:aws:ecs:region:aws_account_id:service/cluster-name/service-name
    --tags key=key1,value=value1
```

O exemplo a seguir marca MyService com uma tag que tem uma chave definida como "TestService" e um valor definido como "WebServers":

```
aws ecs tag-resource \
    --resource-arn arn:aws:ecs:us-east-1:123456789012:service/MyCluster/MyService
    --tags key=TestService1,value=WebServers
```

Terraform

Depois que atender aos pré-requisitos, você poderá marcar o serviço. Crie um recurso `aws_ecs_service` e defina a referência `tags`. Para obter mais informações, consulte Resource: aws_ecs_service na documentação do Terraform.

```
resource "aws_ecs_service" "MyService" {
  name    = "example"
  cluster = aws_ecs_cluster.MyService.id

 tags = {
 "Name"  =  "MyService"
 "Environment"  =  "Production"
 "Department"  =  "QualityAssurance"
  }
}
```

Próximas etapas

Você pode adicionar tags ao serviço. Para obter mais informações, consulte Adição de etiquetas aos recursos do Amazon ECS.

Se você quiser que o Amazon ECS propague as tags da definição de tarefa ou do serviço para a tarefa, execute `update-service` com o parâmetro `propagateTags`. Para obter mais informações, consulte update-service na AWS Command Line Interface Reference.

Solução de problemas

Alguns usuários podem encontrar o erro ao seguir ao migrar do formato de ARN curto para o formato de ARN longo.

`There was an error while migrating the ARN of service service-name. The specified account does not have serviceLongArnFormat or taskLongArnFormat account settings enabled. Add account settings in order to enable tagging.`

Se você já ativou a configuração de conta `serviceLongArnFormat`, mas ainda enfrenta esse erro, talvez as configurações da conta para o formato de ARN longo não tenham sido habilitadas para a entidade principal do IAM específica que originalmente criou o serviço.

1. Identifique a entidade principal que criou o serviço.

    1. No console, as informações estão disponíveis no campo Criado por na guia Configuração e rede na página de detalhes do serviço no console do Amazon ECS.
    2. Na AWS CLI, execute o seguinte comando:

        Substitua os valores das *user-input* pelos seus.

        ```
        aws ecs describe-services --cluster cluster-name --services service-name --query
         'services[0].{createdBy: createdBy}'
        ```

2. Habilite as configurações de conta necessárias para essa entidade principal específica. Você pode fazer isso por meio de uma das seguintes maneiras:

    a. Assuma o usuário ou perfil do IAM para essa entidade principal. Em seguida, execute `put-account-setting`.

    b. Use o usuário raiz para executar o comando enquanto especifica a entidade principal de criação com o `principal-arn`.

    Exemplo.

    Substitua *principal-arn* pelo valor na Etapa 1.

    ```
    aws ecs put-account-setting --name serviceLongArnFormat --value enabled --
    principal-arn arn:aws:iam::123456789012:role/jdoe
    ```

Ambos os métodos habilitam a configuração de conta `serviceLongArnFormat` necessária na entidade principal que criou o serviço, o que permite que a migração do ARN prossiga.

Lógica de controle de utilização do serviço do Amazon ECS

O agendador de serviços do Amazon ECS inclui uma lógica protetiva que controla a utilização das inicializações de tarefas quando elas apresentam falhas repetidas na inicialização. Isso ajuda a evitar o consumo desnecessário de recursos e reduz os custos.

Quando as tarefas em um serviço não conseguem passar do estado `PENDING` para o estado `RUNNING` e, em vez disso, passam diretamente para `STOPPED`, o agendador:

- Aumenta incrementalmente o tempo entre as tentativas de reinicialização
- Continua aumentando os atrasos até um máximo de 27 minutos entre as tentativas
- Gera uma mensagem de evento de serviço para notificar você sobre o problema

> **Note**
>
> O período máximo de atraso de 27 minutos poderá mudar em futuras atualizações.

Quando controle de utilização é habilitado, você recebe esta mensagem de evento de serviço:

```
(service service-name) is unable to consistently start tasks successfully.
```

Características importantes da lógica de controle de utilização:

- Os serviços continuam tentando novamente indefinidamente
- A única modificação é o aumento do tempo entre as reinicializações
- Não há parâmetros configuráveis pelo usuário

Resolução de problemas de controle de utilização

Para resolver o controle de utilização, você pode:

- Atualizar o serviço para usar uma nova definição de tarefa, o que retornará o serviço imediatamente à operação normal e sem controle de utilização. Para obter mais informações, consulte Atualizar um serviço do Amazon ECS.

- Resolva a causa subjacente das falhas da tarefa.

As causas comuns de falhas de tarefas que acionam o controle de utilização incluem:

- Recursos de cluster insuficientes (portas, memória ou CPU)
  - Indicado por uma mensagem de evento de serviço de recursos insuficientes
- Falhas na extração de imagens do contêiner
  - Pode ser causado por nomes de imagens inválidos, tags ou permissões insuficientes
  - Resulta em `CannotPullContainerError` em Visualizar erros de tarefa interrompida do Amazon ECS
- Espaço em disco insuficiente
  - Resulta em `CannotCreateContainerError` em erros de tarefas interrompidas
  - Para conferir as etapas de resolução, consulte Solução de problemas relacionados a `API error (500): devmapper` do Docker no Amazon ECS

> **Important**
>
> Os seguintes cenários NÃO acionam a lógica de controle de utilização:
>
> - Tarefas que são interrompidas após atingirem o estado `RUNNING`
> - Tarefas interrompidas devido a falhas nas verificações de integridade do Elastic Load Balancing
> - Tarefas em que o comando do contêiner sai com um código diferente de zero após atingir o estado `RUNNING`

Parâmetros de definição de serviço do Amazon ECS

Uma definição de serviço define como executar o serviço do .Amazon ECS. É possível especificar os parâmetros a seguir em uma definição de serviço.

Tipo de inicialização

`launchType`

Tipo: String

Valores válidos: `EC2` | `FARGATE` | `EXTERNAL`

Obrigatório: Não

O tipo de inicialização na qual executar seu serviço. Se não for especificado um tipo de inicialização, o padrão `capacityProviderStrategy` será usado. Para obter mais informações, consulte Tipos de inicialização do Amazon ECS.

Se um `launchType` for especificado, o parâmetro `capacityProviderStrategy` deverá ser omitido.

Estratégia de provedor de capacidade

`capacityProviderStrategy`

Tipo: matriz de objetos

Obrigatório: Não

A estratégia de provedor de capacidade a ser usada para o serviço.

Uma estratégia de provedor de capacidade consiste em um ou mais provedores de capacidade junto com o `base` e o `weight` a serem atribuídos a eles. Um provedor de capacidade deve ser associado ao cluster a ser usado em uma estratégia de provedor de capacidade. A API PutClusterCapacityProviders é usada para associar um provedor de capacidade a um cluster. Somente provedores de capacidade com um status `UPDATING` ou `ACTIVE` podem ser usados.

Se um `capacityProviderStrategy` for especificado, o parâmetro `launchType` deverá ser omitido. Se nenhum `capacityProviderStrategy` ou `launchType` for especificado, o `defaultCapacityProviderStrategy` do cluster será usado.

Se quiser especificar um provedor de capacidade que use um grupo do Auto Scaling, o provedor de capacidade já deverá estar criado. Novos provedores de capacidade podem ser criados com a operação de API CreateCapacityProvider.

Para usar um provedor de capacidade do AWS Fargate, especifique os provedores de capacidade `FARGATE` ou `FARGATE_SPOT`. Os provedores de capacidade do AWS Fargate estão disponíveis para todas as contas e só precisam estar associados a um cluster para serem utilizados.

A operação de API PutClusterCapacityProviders é usada para atualizar a lista de provedores de capacidade disponíveis para um cluster após a criação do cluster.

`capacityProvider`

Tipo: string

Obrigatório: Sim

O nome abreviado ou o Nome de recurso da Amazon (ARN) completo do provedor de capacidade.

`weight`

Tipo: inteiro

Intervalo válido: inteiros entre 0 e 1.000.

Obrigatório: Não

O valor do peso designa a porcentagem relativa do número total de tarefas executadas que usam o provedor de capacidade especificado.

Por exemplo, imagine que você tenha uma estratégia contendo dois provedores de capacidade e ambos têm um peso de um. Quando a base é atendida, as tarefas se dividem igualmente entre os dois provedores de capacidade. Com base nessa mesma lógica, imagine que você especifique um peso de 1 para capacityProviderA e um peso de 4 para capacityProviderB. Em seguida, para cada tarefa executada utilizando capacityProviderA, quatro tarefas utilizam capacityProviderB.

`base`

Tipo: inteiro

Intervalo válido: inteiros entre 0 e 100.000.

Obrigatório: Não

O valor da base designa o número mínimo de tarefas que serão executadas no provedor de capacidade especificado. Somente um provedor de capacidade em uma estratégia de provedor de capacidade pode ter uma base definida.

Definição de tarefa

`taskDefinition`

Tipo: String

Obrigatório: Não

O `family` e `revision` (`family:revision`) ou o nome do recurso da Amazon (ARN) completo da definição de tarefa a ser executada no serviço. Se um `revision` não for especificado, a última revisão `ACTIVE` da família especificada será usada.

É necessário especificar uma definição de tarefa ao usar o controlador de implantação (`ECS`) de atualização contínua.

Sistema operacional da plataforma

`platformFamily`

Tipo: string

Obrigatório: Condicional

Padrão: Linux

Esse parâmetro é necessário para serviços do Amazon ECS hospedados no Fargate.

Esse parâmetro é ignorado para serviços do Amazon ECS hospedados no Amazon EC2.

O sistema operacional nos contêineres que executa o serviço. Os valores válidos são `LINUX`, `WINDOWS_SERVER_2019_FULL`, `WINDOWS_SERVER_2019_CORE`, `WINDOWS_SERVER_2022_FULL` e `WINDOWS_SERVER_2022_CORE`.

O valor `platformFamily` para cada tarefa especificada para o serviço deve corresponder ao valor `platformFamily` do serviço. Por exemplo, se você definir a `platformFamily` como `WINDOWS_SERVER_2019_FULL`, o valor de `platformFamily` para todas as tarefas deve ser `WINDOWS_SERVER_2019_FULL`.

Versão da plataforma

`platformVersion`

Tipo: String

Obrigatório: Não

A versão da plataforma na qual suas tarefas no serviço estão em execução. Uma versão da plataforma é especificada apenas para tarefas que usam o tipo de inicialização do Fargate. Se não for especificada, a versão mais recente (`LATEST`) será usada como padrão.

As versões da plataforma do AWS Fargate são usadas para fazer referência a um ambiente de runtime específico para a infraestrutura de tarefas do Fargate. Ao especificar a versão da plataforma `LATEST` quando estiver executando uma tarefa ou criando um serviço, você obtém a versão de plataforma mais atual disponível para suas tarefas. Ao escalar seu serviço, essas tarefas recebem a versão de plataforma especificada na implantação atual do serviço. Para obter mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.

> **Note**
>
> Versões de plataforma não são especificadas para tarefas que usam o tipo de inicialização do EC2.

Cluster

`cluster`

Tipo: String

Obrigatório: Não

O nome abreviado ou o Nome de recurso da Amazon (ARN) completo do cluster no qual executar o serviço. Se você não especificar um cluster, consideraremos o cluster `default`.

Nome do serviço

`serviceName`

Tipo: string

Obrigatório: Sim

O nome do serviço. São permitidos até 255 letras (caixa alta e baixa), números, hífens e sublinhados. Os nomes dos serviços devem ser exclusivos em um cluster, mas é possível ter serviços com nomes semelhantes em vários clusters em uma ou várias regiões.

Estratégia de programação

`schedulingStrategy`

Tipo: String

Valores válidos: `REPLICA` | `DAEMON`

Obrigatório: Não

A estratégia de programação para usar. Se nenhuma estratégia de agendamento for especificada, será usada a estratégia `REPLICA`. Para obter mais informações, consulte Serviços do Amazon ECS.

Há duas estratégias de programador de serviços disponíveis:

- `REPLICA`: a estratégia de programação de réplica coloca e mantém o número desejado de tarefas no seu cluster. Por padrão, o programador de serviço distribui tarefas por zonas de disponibilidade. É possível usar estratégias e limitações de posicionamento de tarefas para personalizar decisões de posicionamento de tarefa. Para obter mais informações, consulte Estratégia de agendamento de réplicas.
- `DAEMON`: a estratégia de programação do daemon implanta exatamente uma tarefa em cada instância de contêiner ativa que atenda a todas as restrições de posicionamento de tarefas que você especifica no seu cluster. Ao usar essa estratégia, não há necessidade de especificar um número desejado de tarefas, uma estratégia de posicionamento de tarefas ou usar políticas de Auto Scaling do serviço. Para obter mais informações, consulte Estratégia de agendamento de daemon.

> **Note**
>
> As tarefas do Fargate não são compatíveis com a estratégia de programação do `DAEMON`.

Contagem desejada

`desiredCount`

Tipo: inteiro

Obrigatório: Não

O número de instanciações da definição de tarefa específica a posicionar e manter em execução no seu serviço.

Esse parâmetro será necessário se a estratégia de agendamento `REPLICA` for usada. Se o serviço usar a estratégia de agendamento `DAEMON`, esse parâmetro será opcional.

Quando você usa o ajuste de escala automático do serviço, ao atualizar um serviço que está em execução e que tem um `desiredCount` menor do que o número de tarefas em execução no momento, o serviço é reduzido para o `desiredCount` especificado.

Configuração de implantação

`deploymentConfiguration`

Tipo: Objeto

Obrigatório: Não

Parâmetros opcionais de implantação que controlam quantas tarefas são executadas durante a implantação e o pedido de encerramento e iniciação de tarefas.

`maximumPercent`

Tipo: inteiro

Obrigatório: Não

Se um serviço estiver usando o tipo de implantação de atualização contínua (`ECS`), o parâmetro `maximumPercent` representará um limite superior no número de tarefas do serviço que são permitidas no estado `RUNNING`, `STOPPING` ou `PENDING` durante a implantação. Ele é expresso como uma porcentagem do `desiredCount` arredondado para o número inteiro mais próximo. É possível usar esse parâmetro para definir o tamanho do lote de implantação.

Por exemplo, se o serviço estiver usando o agendador de serviços `REPLICA` e tiver uma `desiredCount` de quatro tarefas e um valor `maximumPercent` de 200%, o agendador iniciará quatro novas tarefas antes de interromper as quatro tarefas mais antigas. Isso é fornecido desde que os recursos de cluster necessários para isso estejam disponíveis. O valor padrão `maximumPercent` para um serviço que usa o programador de serviço `REPLICA` é 200%.

O agendador do Amazon ECS usa esse parâmetro para substituir tarefas não íntegras iniciando primeiro as tarefas de substituição e depois interrompendo as tarefas não íntegras, desde que os recursos do cluster para iniciar as tarefas de substituição estejam disponíveis. Para obter mais informações sobre como o agendador substitui tarefas não íntegras, consulte Serviços do Amazon ECS.

Se o seu serviço estiver usando o tipo de programador de serviços `DAEMON`, `maximumPercent` deverá permanecer em 100%. Este é o valor padrão.

O número máximo de tarefas durante uma implantação é `desiredCount` multiplicado por `maximumPercent`/100, arredondado para o valor inteiro mais próximo.

Se um serviço estiver usando o tipo de implantação azul/verde (`CODE_DEPLOY`) ou tipos de implantação `EXTERNAL` e as tarefas que usam o tipo de inicialização do EC2, os valores de porcentagem máxima serão definidos para os valores padrão. Eles são usados para definir os limites superiores do número de tarefas no serviço que permanecem no estado `RUNNING` enquanto as instâncias de contêiner estão no estado `DRAINING`.

> **Note**
>
> Você não pode especificar um valor personalizado `maximumPercent` para um serviço que usa os tipos azul/verde (`CODE_DEPLOY`) ou de implantação `EXTERNAL` e tem tarefas que usam o tipo de inicialização do EC2.

Se o serviço utilizar os tipos de implantação azul/verde (`CODE_DEPLOY`) ou `EXTERNAL`, e as tarefas no serviço utilizarem o tipo de inicialização do Fargate, o valor de porcentagem máxima não será utilizado. O valor ainda é retornado ao descrever seu serviço.

`minimumHealthyPercent`

Tipo: inteiro

Obrigatório: Não

Se um serviço estiver usando o tipo de implantação de atualização contínua (`ECS`), `minimumHealthyPercent` representará um limite inferior no número de tarefas do serviço que devem permanecer no `RUNNING` ou durante a implantação. Isso é expresso como uma porcentagem de `desiredCount`, arredondada para cima até o número inteiro mais próximo. É possível usar esse parâmetro para implantar sem usar capacidade adicional de cluster.

Por exemplo, se o serviço tiver um `desiredCount` de quatro tarefas, um `minimumHealthyPercent` de 50% e um `maximumPercent` de 100%, o agendador de serviços interromperá duas tarefas existentes para liberar a capacidade do cluster antes de iniciar duas novas tarefas.

Se alguma tarefa não estiver íntegra e `maximumPercent` não permitir que o agendador do Amazon ECS inicie tarefas de substituição, o agendador interromperá as tarefas não íntegras uma a uma usando `minimumHealthyPercent` como uma restrição para liberar capacidade para iniciar tarefas de substituição. Para obter mais informações sobre como o agendador substitui tarefas não íntegras, consulte Serviços do Amazon ECS.

Para serviços que não usam um balanceador de carga, considere o seguinte:

- Um serviço será considerado íntegro se todos os contêineres essenciais dentro das tarefas no serviço forem aprovados em suas verificações de integridade.
- Se uma tarefa não tiver contêineres essenciais com uma verificação de integridade definida, o programador de serviço aguardará 40 segundos depois que uma tarefa atingir um estado `RUNNING` antes de a tarefa ser contabilizada no percentual mínimo íntegro total.
- Se uma tarefa tiver um ou mais contêineres essenciais com uma verificação de integridade definida, o programador de serviço aguardará que a tarefa atinja um status íntegro antes de a contabilizar no percentual mínimo íntegro total. Uma tarefa é considerada íntegra quando todos os contêineres essenciais dentro da tarefa são aprovados em suas verificações de integridade. O período de tempo que o programador de serviço pode aguardar é determinado pelas configurações de verificação de integridade do contêiner. Para obter mais informações, consulte Verificação de integridade.

Para serviços que usam um balanceador de carga, considere o seguinte:

- Se uma tarefa não tiver contêineres essenciais com uma verificação de integridade definida, o programador de serviço aguardará a verificação de integridade do grupo de destino do balanceador de carga retornar um status íntegro antes de contabilizar a tarefa no percentual mínimo íntegro total.

- Se uma tarefa tiver um contêiner essencial com uma verificação de integridade definida, o programador de serviço aguardará que a tarefa atinja um status íntegro e a verificação de integridade do grupo de destino do balanceador de carga retorne um status íntegro antes de contabilizar a tarefa no percentual mínimo íntegro total.

O valor padrão para um serviço réplica para `minimumHealthyPercent` é 100%. O valor padrão `minimumHealthyPercent` para um serviço que usa o programador de serviço `DAEMON` é 0% para a AWS CLI, os AWS SDKs e as APIs, e 50% para o AWS Management Console.

O número mínimo de tarefas íntegras durante uma implantação é `desiredCount` multiplicado por `minimumHealthyPercent`/100, arredondado para o valor inteiro mais próximo acima.

Se um serviço estiver usando o tipo de implantação azul/verde (`CODE_DEPLOY`) ou tipos de implantação `EXTERNAL` e executando tarefas que usam o tipo de inicialização do EC2, os valores de porcentagem de integridade mínima serão definidos para os valores padrão. Eles são usados para definir os limites inferiores do número de tarefas no serviço que permanecem no estado `RUNNING` enquanto as instâncias de contêiner estão no estado `DRAINING`.

> **Note**
>
> Você não pode especificar um valor personalizado `maximumPercent` para um serviço que usa os tipos azul/verde (`CODE_DEPLOY`) ou de implantação `EXTERNAL` e tem tarefas que usam o tipo de inicialização do EC2.

Se um serviço estiver usando os tipos de implantação azul/verde (`CODE_DEPLOY`) ou `EXTERNAL` e estiver executando tarefas que usam o tipo de inicialização do Fargate, o valor percentual mínimo de integridade não será usado, embora ele seja retornado ao descrever o serviço.

Controlador de implantação

`deploymentController`

Tipo: Objeto

Obrigatório: Não

O controlador de implantação a ser usado para o serviço. Se nenhum controlador de implantação for especificado, será usado o controlador ECS. Para obter mais informações, consulte Serviços do Amazon ECS.

`type`

Tipo: String

Valores válidos: ECS | CODE_DEPLOY | EXTERNAL

Obrigatório: sim

O tipo de controlador de implantação a ser usado. Existem três tipos de controlador de implantação disponíveis:

`ECS`

O tipo de implantação de atualização contínua (ECS) envolve substituir a versão atual em execução do contêiner pela versão mais recente. O número de contêineres que o Amazon ECS adiciona ou remove do serviço durante uma atualização contínua é controlado ajustando-se os números mínimo e máximo de tarefas íntegras permitidas durante uma implantação de serviço, conforme especificado na deploymentConfiguration.

`CODE_DEPLOY`

O tipo de implantação azul/verde (CODE_DEPLOY) usa o modelo de implantação azul/verde desenvolvido pelo CodeDeploy, que permite que você verifique a nova implantação de um serviço antes de enviar tráfego de produção para ele.

`EXTERNAL`

Use o tipo de implantação externa quando desejar usar qualquer controlador de implantação de terceiros para o controle total do processo de implantação para um serviço do Amazon ECS.

Posicionamento de tarefas

`placementConstraints`

Tipo: matriz de objetos

Obrigatório: Não

Um array de objetos de restrição de posicionamento para usar em tarefas no serviço. É possível especificar no máximo 10 restrições por tarefa. Esse limite inclui restrições na definição de tarefa e aquelas especificadas no runtime. Se você utilizar o tipo de inicialização do Fargate, não haverá suporte para restrições de posicionamento de tarefas.

`type`

Tipo: String

Obrigatório: Não

O tipo de restrição. Use `distinctInstance` para garantir que cada tarefa em um determinado grupo esteja em execução em uma instância de contêiner diferente. Use `memberOf` para restringir a seleção a um grupo de candidatos válidos. O valor `distinctInstance` não é compatível em definições de tarefa.

`expression`

Tipo: String

Obrigatório: Não

Uma expressão de idioma de consulta de cluster a ser aplicada à restrição. Você não pode especificar uma expressão, caso o tipo de restrição seja `distinctInstance`. Para obter mais informações, consulte Criação de expressões para definir instâncias de contêiner em tarefas do Amazon ECS.

`placementStrategy`

Tipo: matriz de objetos

Obrigatório: Não

A estratégia de posicionamento de objetos para usar em tarefas no serviço. É possível especificar um máximo de quatro regras de estratégia por serviço.

`type`

Tipo: String

Valores válidos: `random` | `spread` | `binpack`

Obrigatório: Não

O tipo de estratégia de posicionamento. A estratégia de posicionamento `random` posiciona as tarefas de candidatos disponíveis aleatoriamente. A estratégia de posicionamento `spread` distribui o posicionamento entre os candidatos disponíveis uniformemente com base no parâmetro do `field`. A estratégia `binpack` posiciona as tarefas em candidatos disponíveis que tenham a menor quantia disponível do recurso que está especificado no parâmetro `field`. Por exemplo, se você efetuar binpack na memória, uma tarefa será posicionada na instância com a menor quantidade de memória remanescente (mas ainda o suficiente para executar a tarefa).

`field`

Tipo: String

Obrigatório: Não

O campo em que a estratégia de posicionamento será aplicada. Para a estratégia de posicionamento `spread`, os valores válidos são `instanceId` (ou `host`, que tem o mesmo efeito), ou qualquer atributo de plataforma ou personalizado que seja aplicado em uma instância de contêiner, como `attribute:ecs.availability-zone`. Para a estratégia de posicionamento `binpack`, os valores válidos são `cpu` e `memory`. Para a estratégia de posicionamento `random`, esse campo não é usado.

Tags

`tags`

Tipo: matriz de objetos

Obrigatório: Não

Os metadados que você aplica ao serviço para ajudá-lo a categorizá-los e organizá-los. Cada tag consiste de uma chave e um valor opcional, que podem ser definidos. Quando um serviço é excluído, as tags também são excluídas. Um máximo de 50 tags podem ser aplicadas ao serviço. Para obter mais informações, consulte Marcação de recursos do Amazon ECS.

`key`

Tipo: String

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Obrigatório: Não

Uma parte de um par de chave/valor que compõe uma tag. Uma chave é um rótulo geral que age como uma categoria para valores de tag mais específicos.

`value`

Tipo: String

Restrições de tamanho: o tamanho mínimo é 0. O tamanho máximo é 256.

Obrigatório: Não

A parte opcional de um par de chave/valor que compõe uma tag. Um valor atua como um descritor dentro de uma categoria de tag (chave).

`enableECSManagedTags`

Tipo: booliano

Valores válidos: `true` | `false`

Obrigatório: Não

Especifica se devem ser usadas as tags gerenciadas do Amazon ECS para as tarefas no serviço. Se nenhum valor for especificado, o padrão será `false`. Para obter mais informações, consulte Usar etiquetas para faturamento.

`propagateTags`

Tipo: Sequência

Valores válidos: `TASK_DEFINITION` | `SERVICE`

Obrigatório: Não

Especifica se deve copiar as tags da definição de tarefa ou do serviço para as tarefas no serviço. Se nenhum valor for especificado, as tags não serão copiadas. As tags só podem ser copiadas para tarefas no serviço durante a criação do serviço. Para adicionar etiquetas a uma tarefa após a criação do serviço ou da tarefa, use a ação de API `TagResource`.

Configuração de rede

`networkConfiguration`

Tipo: Objeto

Obrigatório: Não

A configuração de rede para o serviço. Esse parâmetro é necessário para definições de tarefa que usam o modo de rede `awsvpc` para receber sua própria Interface de rede elástica e não tem suporte para outros modos de rede. Se você estiver usando o tipo de inicialização do Fargate, será necessário o modo de rede `awsvpc`. Para obter mais informações sobre redes para o tipo de execução do Amazon EC2, consulte Opções de redes de tarefas do Amazon ECS para o tipo de inicialização do EC2. Para obter mais informações sobre redes para o tipo de execução do Fargate, consulte Opções de rede de tarefas do Amazon ECS para o tipo de execução do Fargate.

`awsvpcConfiguration`

Tipo: Objeto

Obrigatório: Não

Um objeto que representa as sub-redes e os security groups de uma tarefa ou serviço.

`subnets`

Tipo: matriz de strings

Obrigatório: Sim

As sub-redes associadas à tarefa ou ao serviço. Existe um limite de 16 sub-redes que podem ser especificadas de acordo com `awsvpcConfiguration`.

`securityGroups`

Tipo: matriz de strings

Obrigatório: Não

Os security groups associados à tarefa ou ao serviço. Se você não especificar um grupo de segurança, o grupo de segurança padrão da VPC será usado. Existe um limite de cinco grupos de segurança que podem ser especificados com base em `awsvpcConfiguration`.

`assignPublicIP`

Tipo: String

Valores válidos: `ENABLED` | `DISABLED`

Obrigatório: Não

Indica se a interface de rede elástica da tarefa recebe um endereço IP público. Se nenhum valor for especificado, será usado o valor padrão de `DISABLED`.

`healthCheckGracePeriodSeconds`

Tipo: inteiro

Obrigatório: Não

O período, em segundos, durante o qual o programador de serviço do Amazon ECS ignora verificações de integridade não íntegras de Elastic Load Balancing, VPC Lattice e contêiner depois que uma tarefa é iniciada pela primeira vez. Se você não especificar um valor de período de tolerância de verificação de integridade, o valor padrão 0 será usado. Se você não usar nenhuma das verificações de integridade, então `healthCheckGracePeriodSeconds` não será utilizado.

Se as tarefas do serviço demorarem para iniciar e responder, você poderá especificar um período de carência para a verificação de integridade de até 2.147.483.647 segundos (cerca de 69 anos). Durante esse tempo, o programador de serviço do Amazon ECS Service ignorará o status das verificações de integridade. Esse período de carência pode evitar que o programador do serviço marque tarefas como não íntegras e as interrompa antes de terem tempo de surgir.

`loadBalancers`

Tipo: matriz de objetos

Obrigatório: Não

Um objeto load balancer que representa os load balancers para uso com o serviço. Para serviços que usam um Application Load Balancer ou um Network Load Balancer, existe um limite de cinco grupos de destino que você pode anexar a um serviço.

Após você criar um serviço, a configuração do balanceador de carga não poderá ser alterada no AWS Management Console. É possível usar o AWS Copilot, o AWS CloudFormation, a AWS CLI ou o SDK para modificar a configuração do balanceador de carga somente para o controlador de implantação rolling do `ECS`, e não o AWS CodeDeploy azul/verde ou externo. Quando você adiciona, atualiza ou remove uma configuração de balanceador de carga, o Amazon

ECS inicia uma nova implantação com a configuração atualizada do Elastic Load Balancing. Isso faz com que as tarefas se registrem e cancelem o registro dos balanceadores de carga. Recomendamos verificar isso em um ambiente de teste antes de atualizar a configuração do Elastic Load Balancing. Para obter informações sobre como modificar a configuração, consulte UpdateService na Referência da API do Amazon Elastic Container Service.

Para Application Load Balancers e Network Load Balancers, esse objeto deve conter o ARN do grupo de destino do balanceador de carga, o nome do contêiner (conforme aparece em uma definição de contêiner) e a porta do contêiner para acesso a partir do balanceador de carga. Quando uma tarefa desse serviço é colocada em uma instância do contêiner, combinação da instância do contêiner e a porta é registrada como um destino no grupo de destino especificado.

`targetGroupArn`

Tipo: String

Obrigatório: Não

O nome do recurso da Amazon (ARN) completo do grupo de destino do Elastic Load Balancing associado a um serviço.

O ARN de um grupo de destino só é especificado ao usar um Application Load Balancer ou um Network Load Balancer.

`loadBalancerName`

Tipo: String

Obrigatório: Não

O nome do load balancer que deve ser associado ao serviço.

Se estiver usando um Application Load Balancer ou um Network Load Balancer, omita o parâmetro de nome do balanceador de carga.

`containerName`

Tipo: String

Obrigatório: Não

O nome do contêiner (conforme aparece na definição de contêiner) para associar ao balanceador de carga.

`containerPort`

Tipo: inteiro

Obrigatório: Não

A porta no contêiner para associar ao balanceador de carga. Essa porta deve corresponder a um `containerPort` na definição de tarefa usada pelas tarefas do serviço. Para as tarefas que usam o tipo de inicialização do EC2, a instância de contêiner deve permitir tráfego de entrada no `hostPort` do mapeamento da porta.

`role`

Tipo: String

Obrigatório: Não

O nome curto ou o ARN completo da função do IAM que permite que o Amazon ECS faça chamadas para o balanceador de carga em seu nome. Esse parâmetro só será permitido se você estiver usando um balanceador de carga com um único grupo de destino para o serviço e a definição de tarefa não usar o modo de rede `awsvpc`. Se você especificar o parâmetro `role`, você também deve especificar um objeto do load balancer com o parâmetro `loadBalancers`.

Se a função especificada tiver um caminho diferente de `/`, você deverá especificar a função completa de Nome de região da Amazon (ARN), isso é recomendado, ou prefixar o nome da função com o caminho. Por exemplo, se uma função com o nome `bar` tiver um caminho `/foo/`, você especificaria `/foo/bar` como o nome da função. Para obter mais informações, consulte Nome e caminhos amigáveis no Guia do usuário do IAM.

> **Important**
>
> Se sua conta já tiver criado a função vinculada ao serviço do Amazon ECS, essa função será usada por padrão para o serviço, a menos que você especifique uma função aqui. A função vinculada ao serviço será necessária se a definição de sua tarefa usar o modo de rede awsvpc e, nesse caso, você não deve especificar uma função aqui. Para obter mais informações, consulte Uso de perfis vinculados ao serviço para o Amazon ECS.

`serviceConnectConfiguration`

Tipo: Objeto

Obrigatório: Não

A configuração desse serviço para detectar e se conectar a serviços e para ser detectado por outros serviços e ser conectado a eles em um namespace.

Para obter mais informações, consulte Uso do Service Connect para conectar serviços do Amazon ECS com nomes abreviados.

`enabled`

Tipo: booliano

Obrigatório: Sim

Especifica se o Service Connect deverá ser usado com esse serviço.

`namespace`

Tipo: String

Obrigatório: Não

O nome do recurso da Amazon (ARN) abreviado ou completo do namespace AWS Cloud Map para uso com o Service Connect. O namespace deve estar na mesma Região da AWS que o serviço e o cluster do Amazon ECS. O tipo de namespace não afeta o Service Connect. Para obter mais informações sobre o AWS Cloud Map, consulte Working with Services (Trabalhar com serviços) no Guia do desenvolvedor do AWS Cloud Map.

`services`

Tipo: matriz de objetos

Obrigatório: Não

Uma matriz de objetos de serviço do Service Connect. Estes são os nomes e aliases (também conhecidos como endpoints) que são usados por outros serviços do Amazon ECS para se conectar a esse serviço.

Este campo não é obrigatório para um serviço “cliente” do Amazon ECS que seja membro de um namespace somente para se conectar a outros serviços dentro do namespace. Um exemplo é a aplicação de frontend que aceita solicitações recebidas de um balanceador de carga conectado ao serviço ou por outros meios.

Um objeto seleciona uma porta na definição da tarefa, atribui um nome para o serviço do AWS Cloud Map e uma matriz de aliases (também conhecidos como endpoints) e portas para que as aplicações clientes se refiram a esse serviço.

`portName`

Tipo: string

Obrigatório: Sim

O `portName` deve corresponder ao `name` de um dos `portMappings` de todos os contêineres na definição de tarefa desse serviço do Amazon ECS.

`discoveryName`

Tipo: String

Obrigatório: Não

O `discoveryName` é o nome do novo serviço do AWS Cloud Map que o Amazon ECS cria para esse serviço do Amazon ECS. Esse nome deve ser exclusivo no namespace do AWS Cloud Map.

Se esse campo não for especificado, será usado `portName`.

`clientAliases`

Tipo: matriz de objetos

Obrigatório: Não

A lista de aliases de cliente para esse serviço de conexão de serviços. Você os usa para atribuir nomes que podem ser usados por aplicações clientes. O número máximo de aliases de cliente que você pode ter nesta lista é 1.

Cada alias (“endpoint”) é um nome DNS e um número de porta que outros serviços (“clientes”) do Amazon ECS podem usar para se conectar a esse serviço.

Cada nome e cada combinação de porta deve ser exclusivo dentro do namespace.

Esses nomes são configurados em cada tarefa do serviço de cliente, não no AWS Cloud Map. As solicitações DNS para resolver esses nomes não saem da tarefa e não contam para a cota de solicitações DNS por segundo por interface de rede elástica.

`port`

Tipo: inteiro

Obrigatório: Sim

O número da porta receptora para o proxy de conexão de serviços. Essa porta está disponível em todas as tarefas dentro do mesmo namespace.

Para evitar a alteração das suas aplicações nos serviços do cliente Amazon ECS, defina isso como a mesma porta que a aplicação cliente usa por padrão.

`dnsName`

Tipo: String

Obrigatório: Não

O `dnsName` é o nome que você usa nas aplicações das tarefas do cliente para se conectar a esse serviço. O nome deve ser um rótulo DNS válido.

O valor padrão será o `discoveryName.namespace` se esse campo não for especificado. Se a `discoveryName` não for especificada, será usada a `portName` da definição de tarefa.

Para evitar a alteração das suas aplicações nos serviços do cliente Amazon ECS, defina isso como o mesmo nome que a aplicação cliente usa por padrão. Por exemplo, alguns nomes comuns são `database`, db ou o nome em minúsculas de um banco de dados, como `mysql` ou `redis`.

`ingressPortOverride`

Tipo: inteiro

Obrigatório: Não

(Opcional) O número da porta para o proxy do Service Connect receber.

Use o valor desse campo para ignorar o proxy para tráfego no número da porta especificado na `portMapping` nomeada na definição de tarefa dessa aplicação e, em seguida, use-o em seus grupos de segurança da Amazon VPC para permitir tráfego para o proxy para esse serviço do Amazon ECS.

No modo `awsvpc`, o valor padrão é o número da porta do contêiner especificado no `portMapping` nomeado na definição de tarefa dessa aplicação. No modo `bridge`, o valor padrão é a porta temporária do proxy do Service Connect.

`logConfiguration`

Tipo: objeto LogConfiguration

Obrigatório: Não

Isso define o local em que os logs do proxy do Service Connect são publicados. Use os logs para depuração durante eventos inesperados. Essa configuração define o parâmetro `logConfiguration` no contêiner do proxy do Service Connect em cada tarefa nesse serviço do Amazon ECS. O contêiner do proxy não é especificado na definição de tarefa.

Recomendamos que você use a mesma configuração de log dos contêineres de aplicações da definição de tarefa para esse serviço do Amazon ECS. Para o FireLens, esta é a configuração de log do contêiner da aplicação. Não é o contêiner de roteador de log do FireLens que usa a imagem de contêiner `fluent-bit` ou `fluentd` .

`serviceRegistries`

Tipo: matriz de objetos

Obrigatório: Não

Os detalhes da configuração de descoberta de serviço para o seu serviço. Para obter mais informações, consulte Uso da descoberta de serviços para conectar serviços do Amazon ECS com nomes DNS.

`registryArn`

Tipo: Sequência

Obrigatório: Não

O nome do recurso da Amazon (ARN) do registro de serviço. O registro de serviço atualmente compatível é AWS Cloud Map. Para obter mais informações, consulte Trabalhar com serviços no Guia do desenvolvedor do AWS Cloud Map.

`port`

Tipo: inteiro

Obrigatório: Não

O valor da porta usado se o serviço de descoberta de serviços especificou um registro de SRV. Esse campo é necessário se o modo de rede `awsvpc` e registros de SRV são usados.

`containerName`

Tipo: String

Obrigatório: Não

O valor do nome do contêiner a ser usado para o serviço de descoberta de serviço. Esse valor é especificado na definição da tarefa. Se a definição de tarefa que sua tarefa de serviço especifica usa o modo de rede `bridge` ou `host` você deve especificar uma combinação de `containerName` e `containerPort` da definição de tarefa. Se a definição de tarefa que sua tarefa de serviço especifica usa o modo de rede `awsvpc` e um registro do tipo SRV DNS é usado, você deve especificar uma combinação de `containerName` e `containerPort` ou um valor `port`, mas não ambos.

`containerPort`

Tipo: inteiro

Obrigatório: Não

O valor do porta a ser usado para o serviço de descoberta de serviço. Esse valor é especificado na definição da tarefa. Se a definição de tarefa que sua tarefa de serviço especifica usa o modo de rede `bridge` ou `host` você deve especificar uma combinação de `containerName` e `containerPort` da definição de tarefa. Se a definição de tarefa que sua tarefa de serviço especifica usa o modo de rede `awsvpc` e um registro do tipo SRV DNS é usado, você deve especificar uma combinação de `containerName` e `containerPort` ou um valor `port`, mas não ambos.

Token de cliente

`clientToken`

Tipo: String

Obrigatório: Não

O identificador exclusivo e que diferencia maiúsculas e minúsculas que você fornece para garantir a idempotência da solicitação. Pode ter até 32 caracteres ASCII.

Rebalanceamento de zonas de disponibilidade

`availabilityZoneRebalancing`

Tipo: String

Obrigatório: Não

Indica se o serviço utiliza o rebalanceamento de zonas de disponibilidade. Os valores válidos são `ENABLED` e `DISABLED`. Para ter mais informações sobre o rebalanceamento de zonas de disponibilidade, consulte Balancear um serviço do Amazon ECS entre zonas de disponibilidade.

Configurações de volume

`volumeConfigurations`

Tipo: Objeto

Obrigatório: Não

A configuração que será usada para criar volumes de tarefas gerenciadas pelo serviço. Somente volumes marcados como `configuredAtLaunch` na definição de tarefa podem ser configurados usando esse objeto.

Esse objeto é necessário para associar volumes do Amazon EBS a tarefas gerenciadas por um serviço. Para obter mais informações, consulte Uso de volumes do Amazon EBS com o Amazon ECS.

`name`

Tipo: string

Obrigatório: Sim

O nome de um volume configurado ao criar ou atualizar um serviço. São permitidos até 255 letras (caixa alta e baixa), números, hifens (_) e sublinhados (-). Esse valor deve corresponder ao nome do volume especificado na definição de tarefa.

`managedEBSVolume`

Tipo: Objeto

Obrigatório: Não

A configuração de volume usada para criar volumes do Amazon EBS que são associados a tarefas gerenciadas por um serviço quando o serviço é criado ou atualizado. Um volume é associado por tarefa.

`encrypted`

Tipo: booliano

Obrigatório: Não

Valores válidos: `true|false`

Especifica se cada volume do Amazon EBS criado deve ser criptografado. Se você ativou a criptografia do Amazon EBS por padrão para uma Região da AWS específica para sua Conta da AWS, mas definiu esse parâmetro como `false`, esse parâmetro será substituído e os volumes serão criptografados com a chave do KMS especificada para criptografia por padrão. Para obter mais informações sobre criptografia do Amazon EBS por padrão, consulte Habilitar criptografia do Amazon EBS por padrão no Guia do usuário do Amazon EBS. Para obter mais informações sobre a criptografia de volumes do Amazon EBS associados a tarefas do Amazon ECS, consulte Criptografar dados armazenados em volumes do Amazon EBS para tarefas do Amazon ECS.

`kmsKeyId`

Tipo: String

Obrigatório: Não

O identificador da chave do AWS Key Management Service (AWS KMS) a ser usada para criptografia no Amazon EBS. Se `kmsKeyId` for especificado, o estado de criptografado deverá ser `true`.

A chave especificada usando esse parâmetro substitui o padrão do Amazon EBS ou qualquer chave do KMS em nível de cluster para a criptografia de armazenamento gerenciado do Amazon ECS que você possa ter especificado. Para obter mais informações, consulte Criptografar dados armazenados em volumes do Amazon EBS para tarefas do Amazon ECS.

Você pode especificar a chave do KMS usando qualquer um dos seguintes itens:

- ID da chave: por exemplo, `1234abcd-12ab-34cd-56ef-1234567890ab`.
- Alias de chave: por exemplo, `alias/ExampleAlias`.
- ARN de chave: por exemplo, `arn:aws:kms:us-east-1:012345678910:key/1234abcd-12ab-34cd-56ef-1234567890ab`.
- Alias de ARN: por exemplo, `arn:aws:kms:us-east-1:012345678910:alias/ExampleAlias`.

> **Important**
>
> A AWS autentica a chave do KMS de forma assíncrona. Portanto, se você especificar um ID, um alias ou um ARN que não seja válido, a ação poderá parecer estar concluída, mas falhará. Para obter mais informações, consulte Troubleshooting Amazon EBS volume attachment issues.

`volumeType`

Tipo: String

Obrigatório: Não

Valores válidos: `gp2|gp3|io1|io2|sc1|st1|standard`

O tipo de volume do Amazon EBS. Para obter mais informações sobre tipos de volumes, consulte Amazon EBS volume types no Guia do usuário do Amazon EBS. O tipo de volume padrão é `gp3`.

> **Note**
>
> O tipo de volume `standard` não é compatível com tarefas do Fargate.

`sizeInGiB`

Tipo: inteiro

Obrigatório: Não

Intervalo válido: números inteiros entre 1 e 16.384

O tamanho do volume do EBS em gibibytes (GiB). Se você não fornecer um ID de snapshot para configurar um volume para anexação, deverá fornecer um valor de tamanho. Se você configurar um volume para anexação usando um snapshot, o valor padrão será o tamanho do snapshot. Em seguida, você pode especificar um tamanho maior ou igual ao tamanho do snapshot.

Para tipos de volume `gp2` e `gp3`, o intervalo válido é de 1 a 16.384.

Para tipos de volume `io1` e `io2`, o intervalo válido é de 4 a 16.384.

Para tipos de volume `st1` e `sc1`, o intervalo válido é de 125 a 16.384.

Para o tipo de volume `standard`, o intervalo válido é de 1 a 1.024.

`snapshotId`

Tipo: String

Obrigatório: Não

O ID do snapshot de um volume do Amazon EBS existente usado pelo Amazon ECS para criar novos volumes para associação. Especifique um `snapshotId` ou um `sizeInGiB`.

`volumeInitializationRate`

Tipo: inteiro

Obrigatório: Não

A taxa, em MiB/s, na qual os dados são recuperados em um snapshot de um volume do Amazon EBS existente para criar novos volumes para associação. Essa propriedade só poderá ser especificada se você especificar uma `snapshotId`. Para obter mais informações sobre essa taxa de inicialização de volume, incluindo a faixa de taxas compatíveis para inicialização, consulte Inicializar volumes do Amazon EBS no Guia do usuário do Amazon EBS.

`iops`

Tipo: inteiro

Obrigatório: Não

O número de operações de E/S por segundo (IOPS). Para volumes de `gp3`, `io1` e `io2`, isso representa o número de IOPS provisionadas para o volume. Em volumes do `gp2`, esse valor representa o desempenho de linha de base do volume e a taxa na a qual o volume acumula créditos de E/S para intermitência. Esse parâmetro é necessário para volumes `io1` e `io2`. Esse parâmetro não é compatível com volumes `gp2`, `st1`, `sc1` ou `standard`.

Para volumes `gp3`, o intervalo válido de valores é de 3.000 a 16.000.

Para volumes `io1`, o intervalo válido de valores é de 100 a 64.000.

Para volumes `io2`, o intervalo válido de valores é de 100 a 64.000.

`throughput`

Tipo: inteiro

Obrigatório: Não

O throughput para provisionamento dos volumes associados a tarefas gerenciadas por um serviço.

> **Important**
>
> Esse parâmetro é compatível apenas com volumes do `gp3`.

`roleArn`

Tipo: string

Obrigatório: Sim

O recurso da Amazon (ARN) do perfil do AWS Identity and Access Management (IAM) da infraestrutura que fornece permissões do Amazon ECS para gerenciar recursos do Amazon EBS nas tarefas. Para obter mais informações, consulte Perfil do IAM de infraestrutura do Amazon ECS.

`tagSpecifications`

Tipo: Objeto

Obrigatório: Não

A especificação das tags a serem aplicadas a cada volume do Amazon EBS.

`resourceType`

Tipo: string

Obrigatório: Sim

Valores válidos: `volume`

O tipo de recurso a ser marcado na criação.

`tags`

Tipo: matriz de objetos

Obrigatório: Não

Os metadados que você aplica aos volumes para ajudar a categorizá-los e organizá-los. Cada tag consiste em uma chave e um valor opcional, ambos definidos por você. `AmazonECSCreated` e `AmazonECSManaged` são tags reservadas adicionadas pelo Amazon ECS em seu nome para que você possa especificar no máximo 48 tags de sua preferência. Quando um volume é excluído, as tags também são excluídas. Para obter mais informações, consulte Marcação de recursos do Amazon ECS.

`key`

Tipo: String

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Obrigatório: Não

Uma parte de um par de chave/valor que compõe uma tag. Uma chave é um rótulo geral que age como uma categoria para valores de tag mais específicos.

`value`

Tipo: String

Restrições de tamanho: o tamanho mínimo é 0. O tamanho máximo é 256.

Obrigatório: Não

A parte opcional de um par de chave/valor que compõe uma tag. Um valor atua como um descritor dentro de uma categoria de tag (chave).

`propagateTags`

Tipo: String

Valores válidos: `TASK_DEFINITION` | `SERVICE` | `NONE`

Obrigatório: Não

Especifica se as tags da definição de tarefa ou do serviço devem ser copiadas para um volume. Se `NONE` for especificado ou nenhum valor for especificado, as tags não serão copiadas.

`fileSystemType`

Tipo: String

Obrigatório: Não

Valores válidos: `xfs`|`ext3`|`ext4`|`NTFS`

O tipo de sistema de arquivos em um volume. O tipo de sistema de arquivos do volume determina como os dados são armazenados e recuperados no volume. Em volumes criados de um snapshot, você deve especificar o mesmo tipo de sistema de arquivos que o volume estava usando quando o snapshot foi criado. Se houver uma incompatibilidade no tipo do sistema de arquivos, a tarefa não será iniciada.

Os valores válidos para o Linux são `xfs`, ext3, `and` `ext4`. O padrão para volumes anexados às tarefas do Linux é `XFS`.

Os valores válidos para o Windows são `NTFS`. O padrão para volumes anexados a tarefas do Windows é `NTFS`.

Modelo de definição de serviço

Veja a seguir a representação JSON de uma definição de serviço do Amazon ECS.

Tipo de execução do Amazon EC2

```
{
    "cluster": "",
    "serviceName": "",
    "taskDefinition": "",
    "loadBalancers": [
        {
            "targetGroupArn": "",
            "loadBalancerName": "",
            "containerName": "",
            "containerPort": 0
        }
    ],
    "serviceRegistries": [
        {
            "registryArn": "",
            "port": 0,
            "containerName": "",
            "containerPort": 0
        }
    ],
    "desiredCount": 0,
    "clientToken": "",
    "launchType": "EC2",
    "capacityProviderStrategy": [
        {
            "capacityProvider": "",
            "weight": 0,
            "base": 0
        }
    ],
    "platformVersion": "",
    "role": "",
    "deploymentConfiguration": {
        "deploymentCircuitBreaker": {
            "enable": true,
            "rollback": true
        },
        "maximumPercent": 0,
        "minimumHealthyPercent": 0,
        "alarms": {
            "alarmNames": [
                ""
```

```
            ],
            "enable": true,
            "rollback": true
        }
    },
    "placementConstraints": [
        {
            "type": "distinctInstance",
            "expression": ""
        }
    ],
    "placementStrategy": [
        {
            "type": "binpack",
            "field": ""
        }
    ],
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": [
                ""
            ],
            "securityGroups": [
                ""
            ],
            "assignPublicIp": "DISABLED"
        }
    },
    "healthCheckGracePeriodSeconds": 0,
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "EXTERNAL"
    },
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "enableECSManagedTags": true,
    "propagateTags": "TASK_DEFINITION",
    "enableExecuteCommand": true,
    "availabilityZoneRebalancing": "ENABLED",
    "serviceConnectConfiguration": {
```

```
        "enabled": true,
        "namespace": "",
        "services": [
            {
                "portName": "",
                "discoveryName": "",
                "clientAliases": [
                    {
                        "port": 0,
                        "dnsName": ""
                    }
                ],
                "ingressPortOverride": 0
            }
        ],
        "logConfiguration": {
            "logDriver": "journald",
            "options": {
                "KeyName": ""
            },
            "secretOptions": [
                {
                    "name": "",
                    "valueFrom": ""
                }
            ]
        }
    },
    "volumeConfigurations": [
        {
            "name": "",
            "managedEBSVolume": {
                "encrypted": true,
                "kmsKeyId": "",
                "volumeType": "",
                "sizeInGiB": 0,
                "snapshotId": "",
                "volumeInitializationRate": 0,
                "iops": 0,
                "throughput": 0,
                "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
```

```
                            {
                                "key": "",
                                "value": ""
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn": "",
                "filesystemType": ""
            }
        }
    ]
}
```

Tipo de inicialização do Fargate

```
{
    "cluster": "",
    "serviceName": "",
    "taskDefinition": "",
    "loadBalancers": [
        {
            "targetGroupArn": "",
            "loadBalancerName": "",
            "containerName": "",
            "containerPort": 0
        }
    ],
    "serviceRegistries": [
        {
            "registryArn": "",
            "port": 0,
            "containerName": "",
            "containerPort": 0
        }
    ],
    "desiredCount": 0,
    "clientToken": "",
    "launchType": "FARGATE",
    "capacityProviderStrategy": [
        {
            "capacityProvider": "",
```

```
            "weight": 0,
            "base": 0
        }
    ],
    "platformVersion": "",
    "platformFamily": "",
    "role": "",
    "deploymentConfiguration": {
        "deploymentCircuitBreaker": {
            "enable": true,
            "rollback": true
        },
        "maximumPercent": 0,
        "minimumHealthyPercent": 0,
        "alarms": {
            "alarmNames": [
                ""
            ],
            "enable": true,
            "rollback": true
        }
    },
    "placementStrategy": [
        {
            "type": "binpack",
            "field": ""
        }
    ],
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": [
                ""
            ],
            "securityGroups": [
                ""
            ],
            "assignPublicIp": "DISABLED"
        }
    },
    "healthCheckGracePeriodSeconds": 0,
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "EXTERNAL"
    },
```

```
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "enableECSManagedTags": true,
    "propagateTags": "TASK_DEFINITION",
    "enableExecuteCommand": true,
    "availabilityZoneRebalancing": "ENABLED",
    "serviceConnectConfiguration": {
        "enabled": true,
        "namespace": "",
        "services": [
            {
                "portName": "",
                "discoveryName": "",
                "clientAliases": [
                    {
                        "port": 0,
                        "dnsName": ""
                    }
                ],
                "ingressPortOverride": 0
            }
        ],
        "logConfiguration": {
            "logDriver": "journald",
            "options": {
                "KeyName": ""
            },
            "secretOptions": [
                {
                    "name": "",
                    "valueFrom": ""
                }
            ]
        }
    },
    "volumeConfigurations": [
        {
            "name": "",
            "managedEBSVolume": {
                "encrypted": true,
```

```
                "kmsKeyId": "",
                "volumeType": "",
                "sizeInGiB": 0,
                "snapshotId": "",
                "volumeInitializationRate": 0,
                "iops": 0,
                "throughput": 0,
                "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
                            {
                                "key": "",
                                "value": ""
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn": "",
                "filesystemType": ""
            }
        }
    ]
}
```

É possível criar esse modelo de definição de serviço usando o comando da AWS CLI a seguir.

```
aws ecs create-service --generate-cli-skeleton
```

Marcação de recursos do Amazon ECS

Para ajudar a gerenciar os recursos do Amazon ECS, você pode, opcionalmente, atribuir seus próprios metadados a cada recurso usando etiquetas. Cada tag consiste em uma chave e um valor opcional.

É possível usar tags para categorizar seus recursos do Amazon ECS de diferentes formas, p. ex., por finalidade, proprietário ou ambiente. Isso é útil quando você tem muitos recursos do mesmo tipo. É possível identificar rapidamente um recurso específico com base nas tags atribuídas a ele. Por exemplo, você pode definir um conjunto de tags para as instâncias de contêiner do Amazon ECS da sua conta. Isso ajuda você a rastrear o proprietário e o nível de pilha de cada instância.

É possível usar tags para seus relatórios de custo e uso. É possível usar esses relatórios para analisar o custo e o uso de seus recursos do Amazon ECS. Para obter mais informações, consulte the section called “Relatórios de uso”.

> **Warning**
>
> Há muitas APIs que retornam chaves de tag e seus valores. Negar acesso ao `DescribeTags` não nega automaticamente acesso às tags apresentadas por outras APIs. Como uma prática recomendada, sugerimos que você não inclua dados confidenciais nas suas tags.

Recomendamos que você desenvolva um conjunto de chave de tags que atenda suas necessidades para cada tipo de recurso. É possível usar um conjunto consistente de chaves de tags para um gerenciamento mais fácil de seus recursos. É possível pesquisar e filtrar os recursos de acordo com as tags que adicionar.

As etiquetas não têm significado semântico no Amazon ECS e são interpretadas estritamente como uma string de caracteres. É possível editar chaves de tags e valores, e é possível remover as tags de um recurso a qualquer momento. É possível definir o valor de uma tag a uma string vazia, mas não pode configurar o valor de um tag como nula. Se você adicionar uma etiqueta que tenha a mesma chave de uma etiqueta existente nesse recurso, o novo valor substituirá o antigo. Ao excluir um recurso, todas as respectivas tags também são excluídas.

Se você estiver usando o AWS Identity and Access Management (IAM), poderá controlar quais usuários na sua conta da AWS têm permissão para gerenciar etiquetas.

Como os recursos são marcados

Há várias maneiras para marcar as tarefas, serviços, definições de tarefa e clusters do Amazon ECS:

- Um usuário marca manualmente um recurso usando o AWS Management Console, a API do Amazon ECS, a AWS CLI ou um AWS SDK.
- Um usuário cria um serviço ou executa uma tarefa autônoma e seleciona a opção de tags gerenciadas pelo Amazon ECS.

  O Amazon ECS marca automaticamente todas as tarefas recém-iniciadas. Para obter mais informações, consulte the section called “Tags gerenciadas pelo Amazon ECS”.
- Um usuário cria um recurso usando o console. O console marca automaticamente os recursos.

  Essas tags são retornadas nas respostas da AWS CLI e do AWS SDK e são exibidas no console. Você não pode modificar nem excluir essas tags.

  Para obter informações sobre as tags adicionadas, consulte a coluna Tags adicionadas automaticamente pelo console na tabela Suporte à marcação para recursos do Amazon ECS.

Se você especificar tags ao criar um recurso e não for possível aplicá-las, o Amazon ECS reverterá o processo de criação. Isso garante que os recursos sejam criados com tags ou, então, não criados, e que nenhum recurso seja deixado sem tags. Ao marcar os recursos no momento da criação, você elimina a necessidade de executar scripts personalizados de marcação após a criação do recurso.

A tabela a seguir descreve os recursos do Amazon ECS compatíveis com as marcações.

Recurso	Compatível com tags	Compatível com a propagação de tags	Tags adicionadas automaticamente pelo console
Tarefas do Amazon ECS	Sim	Sim, a partir da definição de tarefa.	Chave: `aws:ecs:clusterName` Valor: `cluster-name`
	Sim		Chave: `ecs:service:stackId`

Recurso	Compatível com tags	Compatível com a propagação de tags	Tags adicionadas automaticamente pelo console
Serviços do Amazon ECS		Sim, da definição de tarefa ou do serviço para as tarefas no serviço.	Valor: `arn:aws:cloudformation: arn`
Conjuntos de tarefas do Amazon ECS	Sim	Não	N/D
Definições de tarefa do Amazon ECS	Sim	Não	Chave: `ecs:taskDefinition:createdFrom` Valor: `ecs-console-v2`

Recurso	Compatível com tags	Compatível com a propagação de tags	Tags adicionadas automaticamente pelo console
Clusters do Amazon ECS	Sim	Não	Chave: `aws:cloudformation:logical-id` Valor: `ECSCluster` Chave: `aws:cloudformation:stack-id` Valor: `arn:aws:cloudformation: arn` Chave: `aws:cloudformation:stack-name` Valor: `ECS-Console-V2-Cluster- EXAMPLE`
Instâncias de contêiner do Amazon ECS	Sim	Sim, a partir da instância do Amazon EC2. Para obter mais informações, consulte Adição de etiquetas a uma instância de contêiner do Amazon ECS.	N/D
Instâncias externas do Amazon ECS	Sim	Não	N/D

Recurso	Compatível com tags	Compatível com a propagação de tags	Tags adicionadas automaticamente pelo console
Provedor de capacidade do Amazon ECS	Sim. Não é possível marcar provedores de capacidade `FARGATE` e `FARGATE_SPOT` predefinidos.	Não	N/D

Atribuição de tags a recursos durante a criação

Os seguintes recursos oferecem suporte à marcação na criação usando a API do Amazon ECS, a AWS CLI ou o AWS SDK:

- Tarefas do Amazon ECS
- Serviços do Amazon ECS
- Definição de tarefa do Amazon ECS
- Conjuntos de tarefas do Amazon ECS
- Clusters do Amazon ECS
- Instâncias de contêiner do Amazon ECS
- Provedores de capacidade do Amazon ECS

O Amazon ECS tem a opção de usar a autorização para atribuição de tags para a criação de recursos. Quando a Conta da AWS é configurada para autorização de marcação, os usuários devem ter permissões para ações que criam o recurso, como `ecsCreateCluster`. Se as tags forem especificadas na ação resource-creating, a AWS executará uma autorização adicional para verificar se os usuários ou perfis têm permissões para criar tags. Portanto, é preciso conceder permissões explícitas para usar a ação `ecs:TagResource`. Para obter mais informações, consulte the section called “Recursos de tags durante a criação”. Para obter informações sobre como configurar a opção, consulte the section called “Autorização para atribuição de tags”.

Restrições

As restrições a seguir se aplicam às tags:

- Podem ser associadas no máximo 50 tags a um recurso.
- As chaves de tag não podem ser repetidas para um recurso. As chaves de tag devem ser exclusivas, e cada chave só pode ter um valor.
- As chaves podem ter até 128 caracteres em UTF-8.
- Os valores podem ter até 256 caracteres em UTF-8.
- Se vários Serviços da AWS e recursos usam seu esquema de tags, limite os tipos de caracteres que você usa. Alguns serviços podem ter restrições quanto aos caracteres permitidos. Em geral, os caracteres permitidos são letras, números, espaços e os seguintes caracteres: **+ - = . _ : / @**.
- As chaves e valores das tags diferenciam maiúsculas de minúsculas.
- Não é possível usar `aws:`, `AWS:` ou qualquer combinação de letras maiúsculas e minúsculas como prefixo para chaves ou valores. Esses são reservados para uso pela AWS. Você não pode editar nem excluir chaves nem valores de tags com esse prefixo. As tags com esse prefixo não contam para seu limite de tags por recurso.

Tags gerenciadas pelo Amazon ECS

Quando você usa tags gerenciadas pelo Amazon ECS, ele marca automaticamente todas as tarefas recém-executadas e quaisquer volumes do Amazon EBS anexados às tarefas com as informações do cluster e as tags de definição de tarefa adicionadas pelo usuário ou as tags de serviço. Veja a seguir uma descrição das tags adicionadas:

- Tarefas autônomas: uma tag com uma Chave como `aws:ecs:clusterName` e um Valor definido como o nome do cluster. Todas as tags de definição de tarefa que foram adicionadas pelos usuários. Um volume do Amazon EBS anexado a uma tarefa autônoma recebe a tag com uma Chave, como `aws:ecs:clusterName`, e um Valor definido para o nome do cluster. Para obter mais informações sobre marcação de volumes do Amazon EBS, consulte Tagging Amazon EBS volumes.
- Tarefas que fazem parte de um serviço: uma tag com uma Chave como `aws:ecs:clusterName` e um Valor definido como o nome do cluster. Uma tag com uma Chave como `aws:ecs:serviceName` e um Valor definido como o nome do serviço. Tags de um dos seguintes recursos:

- Definições de tarefa: todas as tags de definição de tarefa que foram adicionadas pelos usuários.
- Serviços: todas as tags de serviço que foram adicionadas pelos usuários.

  Um volume do Amazon EBS anexado a uma tarefa que faz parte de um serviço recebe uma tag com uma Chave, como `aws:ecs:clusterName`, e um Valor definido para o nome do cluster, bem como uma tag com uma Chave, como `aws:ecs:serviceName`, e um Valor definido para o nome do serviço. Para obter mais informações sobre marcação de volumes do Amazon EBS, consulte Tagging Amazon EBS volumes.

As seguintes opções são necessárias para este recurso:

- É necessário que aceitar os novos formatos do nome do recurso da Amazon (ARN) e do identificador do recurso (ID). Para obter mais informações, consulte Nomes de recursos da Amazon (ARNs) e IDs.
- Ao usar as APIs para criar um serviço ou executar uma tarefa, é necessário definir `enableECSManagedTags` como `true` para `run-task` e `create-service`. Para obter mais informações, consulte create-service e run-task na Referência da API do AWS Command Line Interface.
- O Amazon ECS usa tags gerenciadas para determinar quando alguns recursos estão habilitados, por exemplo, o ajuste de escala automático de cluster. Recomendamos que você não modifique manualmente as tags para que o Amazon ECS possa gerenciar os recursos com eficiência.

Usar etiquetas para faturamento

A AWS fornece uma ferramenta de geração de relatório chamada Cost Explorer. É possível usá-la para analisar o custo e o uso de seus recursos do Amazon ECS.

É possível usar o Cost Explorer para visualizar gráficos de uso e de custo. É possível visualizar dados dos últimos 13 meses e prever o provável valor que você gastará nos próximos 3 meses. É possível usar o Cost Explorer para ver os padrões de quanto você gasta nos recursos da AWS ao longo do tempo. Por exemplo, você pode usar o Cost Explorer para identificar áreas que precisam de uma investigação mais profunda e observar tendências que podem ser usadas para o entendimento dos custos. Também é possível especificar os períodos dos dados e visualizar os dados de tempo por dia ou mês.

É possível usar tags gerenciadas pelo Amazon ECS ou tags adicionadas pelo usuário para seu relatório de custo e uso. Para obter mais informações, consulte Relatórios de uso do Amazon ECS.

Para ver o custo dos recursos combinados, é possível organizar as informações de faturamento com base nos recursos com os mesmos valores da chave da tag. Por exemplo, é possível etiquetar vários recursos com um nome de aplicação específico, e depois organizar suas informações de faturamento para ver o custo total daquela aplicação em vários serviços. Para obter mais informações sobre como configurar um relatório de alocação de custos com etiquetas, consulte Relatório mensal de alocação de custos no Guia do usuário do AWS Billing.

Além disso, é possível ativar Dados de alocação de custos divididos para obter dados de uso de CPU e memória em nível de tarefa em seus relatórios de custo e uso. Para obter mais informações, consulte Relatórios de custo e uso em nível de tarefa.

> **Note**
>
> Se a criação de relatórios tiver sido ativada, pode levar até 24 horas até que os dados do mês atual estejam disponíveis para visualização.

Adição de etiquetas aos recursos do Amazon ECS

É possível realizar a marcação de tarefas, serviços, definições de tarefas ou clusters novos ou existentes. Para obter informações sobre a marcação de instâncias de contêiner, consulte Adição de etiquetas a uma instância de contêiner do Amazon ECS.

> **Warning**
>
> Não adicione informações de identificação pessoal (PII) nem outras informações confidenciais ou sigilosas em tags. As tags são acessíveis a muitos serviços da AWS, incluindo faturamento. As tags não devem ser usadas para dados privados ou confidenciais.

É possível usar os recursos a seguir para especificar tags ao criar o recurso.

Tarefa	Console	AWS CLI	Ação API
Execute uma ou mais tarefas.	Execução de uma aplicação como uma tarefa do Amazon ECS	run-task	RunTask

Tarefa	Console	AWS CLI	Ação API
Crie um serviço.	Criação de uma implantação de atualização contínua do Amazon ECS	create-service	CreateService
Crie um conjunto de tarefas.	Implantação de serviços do Amazon ECS usando um controlador de terceiros	create-task-set	CreateTaskSet
Registre uma definição de tarefa.	the section called “Criação de uma definição de tarefa usando o console”	register-task-definition	RegisterTaskDefinition
Crie um cluster.	Criação de um cluster do Amazon ECS para o tipo de inicialização do Fargate	create-cluster	CreateCluster
Execute uma ou mais instâncias de contêiner.	Iniciar uma instância de contêiner do Linux do Amazon ECS	run-instances	RunInstances

Adicionar etiquetas a recursos existentes (console do Amazon ECS)

Você pode adicionar ou excluir tags associadas a clusters, serviços, tarefas e definições de tarefas diretamente da página do recurso.

Para modificar uma tag de um recurso individual

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na barra de navegação, selecione Região da AWS a ser usada.
3. No painel de navegação, selecione um tipo de recurso (por exemplo, Clusters).
4. Selecione o recurso na lista de recursos, escolha a guia Tags e, em seguida, escolha Manage tags (Gerenciar tags).
5. Configure suas tags.

   [Adicionar uma etiqueta] Escolha Add tag (Adicionar etiqueta) e faça o seguinte:

   - Em Chave, insira o nome da chave.
   - Em Valor insira o valor da chave.
6. Escolha Salvar.

Adicionar etiquetas a recursos existentes (AWS CLI)

É possível adicionar ou substituir uma ou mais etiquetas ao usar a AWS CLI ou uma API.

> **Note**
>
> É possível usar endpoints de serviço de pilha dupla para interagir com o Amazon ECS via AWS CLI, SDKs e API do Amazon ECS sobre IPv4 e IPv6. Para obter mais informações, consulte Usar endpoints de pilha dupla do Amazon ECS.

- AWS CLI: tag-resource
- Ação de API: TagResource

Adição de etiquetas a uma instância de contêiner do Amazon ECS

É possível associar tags a suas instâncias de contêiner usando um dos métodos a seguir:

- Método 1: ao criar a instância de contêiner usando a API, a CLI ou o console do Amazon EC2, especifique as etiquetas transmitindo os dados do usuário para a instância usando o parâmetro de configuração do agente de contêiner `ECS_CONTAINER_INSTANCE_TAGS`. Isso cria etiquetas que são associadas à instância de contêiner somente no Amazon ECS. Elas não podem ser listadas usando a API do Amazon EC2. Para obter mais informações, consulte Inicialização de instâncias de contêiner do Linux no Amazon ECS para transmitir dados.

  > **Important**
  >
  > Se você iniciar as instâncias de contêiner usando um grupo do Amazon EC2 Auto Scaling, deverá usar o parâmetro de configuração do agente ECS_CONTAINER_INSTANCE_TAGS para adicionar etiquetas. Isso é decorrente da maneira como as etiquetas são adicionadas às instâncias do Amazon EC2 que são iniciadas por meio de grupos do Auto Scaling.

  Veja a seguir um exemplo de um script de dados do usuário que associa tags à sua instância de contêiner:

  ```
  #!/bin/bash
  cat <<'EOF' >> /etc/ecs/ecs.config
  ECS_CLUSTER=MyCluster
  ECS_CONTAINER_INSTANCE_TAGS={"tag_key": "tag_value"}
  EOF
  ```

- Método 2: ao criar sua instância de contêiner usando a API, a CLI ou o console do Amazon EC2, especifique primeiramente as tags usando o parâmetro `TagSpecification.N`. Em seguida, transmita os dados do usuário para a instância usando o parâmetro `ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM` de configuração do agente de contêiner. Essa ação propaga os dados do Amazon EC2 para o Amazon ECS.

  Veja a seguir um exemplo de um script de dados do usuário que propaga as tags associadas a uma instância do Amazon EC2 e registra a instância em um cluster denominado `MyCluster`.

  ```
  #!/bin/bash
  cat <<'EOF' >> /etc/ecs/ecs.config
  ECS_CLUSTER=MyCluster
  ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM=ec2_instance
  EOF
  ```

Para fornecer acesso para permitir que as etiquetas de instância de contêiner se propaguem do Amazon EC2 para o Amazon ECS, adicione manualmente as seguintes permissões, como uma política alinhada, à função do IAM da instância de contêiner do Amazon ECS. Para obter mais informações, consulte Adicionar e remover políticas do IAM.

- `ec2:DescribeTags`

Veja a seguir um exemplo de política usada para adicionar essas permissões.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
             "ec2:DescribeTags"
            ],
            "Resource": "*"
        }
    ]
}
```

Instâncias de contêiner externas

É possível associar tags às instâncias de contêiner externas usando um dos métodos a seguir.

- Método 1: antes de executar o script de instalação para registrar a instância externa no cluster, crie ou edite o arquivo de configuração do agente de contêiner do Amazon ECS em `/etc/ecs/ecs.config` e adicione o parâmetro `ECS_CONTAINER_INSTANCE_TAGS` de configuração do agente de contêiner. Isso cria etiquetas associadas à instância externa.

  Veja a seguir um exemplo de sintaxe.

  ```
  ECS_CONTAINER_INSTANCE_TAGS={"tag_key": "tag_value"}
  ```

- Método 2: depois que a instância externa for registrada no cluster, será possível usar o AWS Management Console para adicionar tags. Para obter mais informações, consulte Adicionar etiquetas a recursos existentes (console do Amazon ECS).

Relatórios de uso do Amazon ECS

A AWS fornece uma ferramenta de geração de relatório chamada Cost Explorer. É possível usá-la para analisar o custo e o uso de seus recursos do Amazon ECS.

É possível usar o Cost Explorer para visualizar gráficos de uso e de custo. É possível visualizar dados dos últimos 13 meses e prever o provável valor que você gastará nos próximos 3 meses. É possível usar o Cost Explorer para ver os padrões de quanto você gasta nos recursos da AWS ao longo do tempo. Por exemplo, você pode usar o Cost Explorer para identificar áreas que precisam de uma investigação mais profunda e observar tendências que podem ser usadas para o entendimento dos custos. Também é possível especificar os períodos dos dados e visualizar os dados de tempo por dia ou mês.

Os dados de medição do seu relatório de uso e de custo mostram o uso em todas as tarefas do Amazon ECS. Os dados de medição incluem o uso da CPU como `vCPU-Hours` e o uso da memória como `GB-Hours` para cada tarefa executada. A forma como os dados são apresentados depende do tipo de inicialização da tarefa.

Para tarefas que usam o tipo de execução do Fargate, a coluna `lineItem/Operation` mostra `FargateTask` e você verá o custo associado a cada tarefa.

Para tarefas que usam o tipo de inicialização do EC2, a coluna `lineItem/Operation` mostra `ECSTask-EC2` e as tarefas não têm um custo direto associado a elas. Os dados de medição exibidos no relatório, como uso de memória, representam o total de recursos que a tarefa reservou durante o período de faturamento que você especificou. É possível usar esses dados para determinar o custo do cluster subjacente das instâncias do Amazon EC2. Os dados de custo e de uso das instâncias do Amazon EC2 serão listados separadamente no serviço do Amazon EC2.

Também é possível usar as tags gerenciadas do Amazon ECS para identificar o serviço ou o cluster ao qual cada tarefa pertence. Para obter mais informações, consulte Usar etiquetas para faturamento.

> **Important**
>
> Os dados de medição só podem ser visualizados para tarefas iniciadas a partir de 16 de novembro de 2018. Tarefas iniciadas antes dessa data não mostram dados de medição.

Veja a seguir um exemplo de alguns campos que você pode usar para classificar dados de alocação de custos no Cost Explorer.

- Nome do cluster
- Nome do serviço
- Tags de recursos
- Tipo de inicialização
- Região da AWS
- Tipo de uso

Para obter mais informações sobre como criar um Relatório de uso e de custos da AWS, consulte Relatório de uso e de custos da AWS no Guia do usuário do AWS Billing.

Relatórios de custo e uso em nível de tarefa

O AWS Cost Management pode fornecer dados de uso de CPU e memória no AWS Cost and Usage Report para cada tarefa no Amazon ECS, incluindo tarefas no Fargate e tarefas no EC2. Esses dados são chamados de Dados de alocação de custos divididos. É possível usar esses dados para analisar os custos e o uso das aplicações. Além disso, é possível dividir e alocar os custos para unidades de negócios e equipes individuais com tags de alocação de custos e categorias de custo. Para obter mais informações sobre Dados de alocação de custos divididos, consulte Understanding split cost allocation data no Guia do usuário do AWS Cost and Usage Report.

É possível optar por Dados de alocação de custos divididos em nível de tarefa para a conta no AWS Cost Management Console. Se você tiver uma conta de gerenciamento (pagador), poderá optar pela conta do pagador para aplicar essa configuração a todas as contas vinculadas.

Depois de configurar os Dados de alocação de custos divididos, haverá colunas adicionais no cabeçalho splitLineItem no relatório. Para obter mais informações, consulte Split line item details no Guia do usuário do AWS Cost and Usage Report

Para tarefas no EC2, esses dados dividem o custo da instância do EC2 com base no uso ou nas reservas dos recursos e nos recursos restantes na instância.

Confira a seguir os pré-requisitos:

- Defina o parâmetro de configuração do agente `ECS_DISABLE_METRICS` do Amazon ECS como `false`.

Quando essa configuração for `false`, o agente do Amazon ECS enviará métricas para o Amazon CloudWatch. No Linux, essa configuração é `false` por padrão, e as métricas são enviadas para o CloudWatch. No Windows, essa configuração é `true` por padrão, então você deve alterá-la para `false` para enviar as métricas ao CloudWatch para uso do AWS Cost Management. Para obter mais informações sobre a configuração do agente do ECS, consulte Configuração do agente de contêiner do Amazon ECS.

- A versão mínima do Docker para métricas confiáveis é o Docker versão v20.10.13 e posteriores, que está incluída na AMI otimizada para o Amazon ECS 20220607 e posteriores.

Para usar Dados de alocação de custos divididos, você deve criar um relatório e selecionar Dados de alocação de custos divididos. Para obter mais informações, consulte Creating Cost and Usage Reports no Guia do usuário do AWS Cost and Usage Report.

O AWS Cost Management calcula os Dados de alocação de custos divididos com o uso da CPU e da memória da tarefa. O AWS Cost Management pode usar a reserva de CPU e memória da tarefa em vez do uso, se o uso não estiver disponível. Se você perceber que o CUR está usando as reservas, verifique se suas instâncias de contêiner atendem aos pré-requisitos e se as métricas de uso dos recursos da tarefa aparecem no CloudWatch.

Monitorar o Amazon ECS

O monitoramento é uma parte importante para manter a confiabilidade, a disponibilidade e a performance do Amazon ECS e das soluções da AWS. Você deve coletar dados de monitoramento de todas as partes de sua solução da AWS, para facilitar a depuração de uma falha multipontos, caso ocorra. Antes de começar a monitorar o Amazon ECS, crie um plano de monitoramento que inclua as respostas para as seguintes perguntas:

- Quais são seus objetivos de monitoramento?
- Quais recursos você vai monitorar?
- Com que frequência você vai monitorar esses recursos?
- Quais ferramentas de monitoramento você usará?
- Quem realizará o monitoramento das tarefas?
- Quem deve ser notificado quando algo der errado?

As métricas disponibilizadas dependem do tipo de inicialização das tarefas e dos serviços em seus clusters. Se você estiver usando o tipo de inicialização do Fargate para seus serviços, serão fornecidas métricas de utilização de CPU e memória para auxiliar no monitoramento dos seus serviços. Para o tipo de inicialização do Amazon EC2, você possui e precisa monitorar as instâncias do EC2 que formam a infraestrutura subjacente. Métricas adicionais de utilização e reserva de memória e CPU são disponibilizadas no cluster, no serviço e na tarefa.

A próxima etapa é estabelecer uma linha de base de performance normal do Amazon ECS no seu ambiente, ao medir a performance em vários momentos e em diferentes condições de carga. À medida que você monitora o Amazon ECS, armazene dados de monitoramento históricos para compará-los com os dados de performance atuais, identificar padrões de performance normais e anomalias de performance e elaborar métodos para resolver problemas.

Para estabelecer uma linha de base, é preciso, no mínimo, monitorar os seguintes itens:

- As métricas de utilização e reserva de memória e CPU para os clusters do Amazon ECS
- As métricas de utilização de memória e CPU para os serviços do Amazon ECS

  Para obter mais informações, consulte Visualizar métricas do Amazon ECS.

Práticas recomendadas para monitorar o Amazon ECS

Use as práticas recomendadas a seguir para monitorar o Amazon ECS.

- Faça do monitoramento uma prioridade para evitar pequenos problemas antes que se tornem grandes
- Crie um plano de monitoramento que inclua respostas para a pergunta a seguir
  - Quais são seus objetivos de monitoramento?
  - Quais recursos você vai monitorar?
  - Com que frequência você vai monitorar esses recursos?
  - Quais ferramentas de monitoramento você usará?
  - Quem realizará o monitoramento das tarefas?
  - Quem deve ser notificado quando algo der errado?
- Automatize o monitoramento o máximo possível.
- Verifique os arquivos de log do Amazon ECS. Para obter mais informações, consulte Visualização de logs do agente de contêiner do Amazon ECS.
- Utilize o Runtime Monitoring para ajudar a proteger suas contas, contêineres, workloads e dados dentro do seu ambiente AWS. Para obter mais informações, consulte Identificação de comportamentos não autorizados usando o monitoramento de runtime.

Ferramentas de monitoramento do Amazon ECS

A AWS fornece várias ferramentas que podem ser usadas para monitorar o Amazon ECS. É possível configurar algumas dessas ferramentas para fazer o monitoramento em seu lugar, e, ao mesmo tempo, algumas das ferramentas exigem intervenção manual. Recomendamos que as tarefas de monitoramento sejam automatizadas ao máximo possível.

Ferramentas de monitoramento automatizadas

É possível usar as seguintes ferramentas automatizadas de monitoramento para observar o Amazon ECS e gerar relatórios quando algo estiver errado:

- Alarmes do Amazon CloudWatch: observe uma única métrica ao longo de um período que você especificar e execute uma ou mais ações com base no valor da métrica em relação a um determinado limite ao longo de vários períodos. A ação é uma notificação enviada para um tópico

do Amazon Simple Notification Service (Amazon SNS) ou uma política do Amazon EC2 Auto Scaling. Os alarmes do CloudWatch não invocam ações simplesmente por estarem em um estado específico. O estado deve ter sido alterado e mantido por um número específico de períodos. Para obter mais informações, consulte Monitoramento do Amazon ECS usando o CloudWatch.

Para serviços com tarefas que usam o tipo de inicialização do Fargate, você pode usar os alarmes do CloudWatch para expandir e reduzir as tarefas no serviço, com base em métricas do CloudWatch, como a utilização da CPU e da memória. Para obter mais informações, consulte Como escalar automaticamente o serviço do Amazon ECS.

Para clusters com tarefas ou serviços que usam o tipo de inicialização do EC2, você pode usar os alarmes do CloudWatch para expandir e reduzir as instâncias de contêiner com base em métricas do CloudWatch, como a reserva de memória do cluster.

Para instâncias de contêiner iniciadas com a AMI do Amazon Linux otimizada para Amazon ECS, é possível usar o CloudWatch Logs para exibir logs diferentes das suas instâncias de contêiner em um local conveniente. Você deve instalar o agente do CloudWatch nas suas instâncias de contêiner. Para obter mais informações, consulte Baixar e configurar o agente do CloudWatch usando a linha de comando no Guia do usuário do Amazon CloudWatch. Você também deve anexar a política `ECS-CloudWatchLogs` ao perfil `ecsInstanceRole`. Para obter mais informações, consulte Monitorar permissões de instâncias de contêiner.

- Amazon CloudWatch Logs: monitore, armazene e acesse os arquivos de log nos contêineres nas tarefas do Amazon ECS especificando o driver de log `awslogs` nas definições de tarefa. Para obter mais informações, consulte Envio de logs do Amazon ECS para o CloudWatch .

  Você também pode monitorar, armazenar e acessar os arquivos de log do sistema operacional e do agente de contêiner do Amazon ECS nas instâncias de contêiner do Amazon ECS. Este método para acessar logs pode ser usado para contêineres que usem o tipo de inicialização do EC2.

- Amazon CloudWatch Events: faça correspondência de eventos e direcione-os a uma ou mais funções ou streams de destino para fazer alterações, capturar informações de estado e realizar ações corretivas. Para obter mais informações, consulte Automatização de respostas a erros do Amazon ECS usando o EventBridge neste guia e O EventBridge é a evolução do Amazon CloudWatch Events no Guia do usuário Amazon EventBridge.
- Container Insights: colete, agregue e resuma métricas e logs de aplicações e microsserviços em contêineres. O Container Insights coleta dados como eventos de log de performance usando formato de métrica incorporado. Esses eventos de log de desempenho são entradas que usam

um esquema JSON estruturado que permite que dados de alta cardinalidade sejam ingeridos e armazenados em grande escala. Com base nesses dados, o CloudWatch cria métricas agregadas no nível de cluster, tarefa e serviço como métricas do CloudWatch. As métricas que o Container Insights coleta estão disponíveis nos painéis automáticos do CloudWatch e também podem ser visualizadas na seção Métricas do console do CloudWatch.

- Monitoramento de log do AWS CloudTrail: compartilhe arquivos de log entre contas, monitore arquivos de log do CloudTrail em tempo real enviando-os para o CloudWatch Logs, grave aplicações de processamento de logs em Java e confirme se os arquivos de log não foram alterados após a entrega pelo CloudTrail. Para obter mais informações, consulte Registro em log das chamadas de API do Amazon ECS usando o AWS CloudTrail neste guia e Trabalhar com arquivos de log do CloudTrail noGuia do usuário do AWS CloudTrail.
- Monitoramento de runtime: detecta ameaças em clusters e contêineres no ambiente da AWS. O monitoramento de runtime usa um agente de segurança do GuardDuty que adiciona visibilidade de runtime às workloads individuais do Amazon ECS, por exemplo, acesso a arquivos, execução de processos e conexões de rede.

Ferramentas de monitoramento manual

Outra parte importante do monitoramento do Amazon ECS envolve o monitoramento manual dos itens que os alarmes do CloudWatch não abrangem. O CloudWatch, o Trusted Advisor e outros painéis de console da AWS apresentam uma visão rápida do estado do seu ambiente na AWS. Recomendamos que você também verifique os arquivos de log nas instâncias de contêiner e os contêineres nas tarefas.

- Console do Amazon ECS:
  - Métricas de cluster para o tipo de execução do EC2
  - Métricas de serviço
  - Estado de integridade do serviço
  - Evenetos de implantações de serviços
- Página inicial do CloudWatch:
  - Alertas e status atual
  - Gráficos de alertas e recursos
  - Estado de integridade do serviço

  Além disso, é possível usar o CloudWatch para fazer o seguinte:

  - Criar painéis personalizados para monitorar os serviços com os quais você se preocupa.
  - Colocar em gráfico dados de métrica para solucionar problemas e descobrir tendências.
  - Pesquise e procure todas as métricas de recursos da AWS.
  - Criar e editar alertas para ser notificado sobre problemas.
- Verificação de integridade do contêiner: esses comandos são executados localmente em um contêiner e validam a integridade e a disponibilidade da aplicação. Você os configura por contêiner na definição de tarefa.
- O AWS Trusted Advisor pode ajudar a monitorar os recursos da AWS para melhorar a performance, a confiabilidade, a segurança e a economia. Quatro verificações do Trusted Advisor estão disponíveis a todos os usuários; mais de 50 verificações estão disponíveis para usuários com um plano de suporte Business ou Enterprise. Para obter mais informações, consulte AWS Trusted Advisor.

  O Trusted Advisor tem estas verificações que se referem ao Amazon ECS:

  - Uma tolerância a falhas que indica a existência de um serviço em execução em uma única zona de disponibilidade.
  - Uma tolerância a falhas indicando que você não usou a estratégia de posicionamento de distribuição para várias zonas de disponibilidade.
- O AWS Compute Optimizer é um serviço que analisa as métricas de configuração e utilização dos seus recursos da AWS. O serviço informa se seus recursos estão em condições ideais e gera recomendações de otimização para reduzir o custo e melhorar a performance de suas workloads.

  Para obter mais informações, consulte Recomendações do AWS Compute Optimizer para o Amazon ECS.

Monitoramento do Amazon ECS usando o CloudWatch

É possível monitorar os recursos do Amazon ECS usando o Amazon CloudWatch, que coleta e processa os dados brutos do Amazon ECS em métricas legíveis, quase em tempo real. Esses dados estatísticos são registrados por um período de duas semanas, para que você possa acessar informações históricas e obter uma perspectiva melhor sobre a performance dos clusters ou dos serviços. Os dados das métricas do Amazon ECS são enviados automaticamente para o CloudWatch em períodos de um minuto. Para obter mais informações sobre o CloudWatch, consulte o Guia do usuário do Amazon CloudWatch.

O Amazon ECS fornece métricas gratuitas para clusters e serviços. Por um custo adicional, você pode ativar o Amazon ECS CloudWatch Container Insights no cluster para métricas por tarefa, incluindo utilização de CPU, memória e sistema de arquivos do EBS. Para obter mais informações sobre o Container Insights, consulte Monitorar contêineres do Amazon ECS utilizando o Container Insights com observabilidade aprimorada.

Considerações

Considere os pontos a seguir ao usar as métricas do CloudWatch no Amazon ECS.

- Qualquer serviço do Amazon ECS hospedado no Fargate tem métricas de utilização de CPU e memória do CloudWatch automaticamente para que você não precise realizar nenhuma etapa manual.

- Para qualquer tarefa ou serviço do Amazon ECS hospedado em instâncias do Amazon EC2, a instância do Amazon EC2 exige uma versão `1.4.0` ou posterior (Linux) ou `1.0.0` ou posterior (Windows) do agente de contêiner para que as métricas do CloudWatch sejam geradas. Recomendamos usar a versão mais recente do atendente de contêiner. Para obter informações sobre como verificar a versão do agente e atualizar para a versão mais recente, consulte Atualizar o agente de contêiner do Amazon ECS.
- A versão mínima do Docker para métricas confiáveis do CloudWatch é a versão `20.10.13` e mais recente do Docker.
- As instâncias do Amazon EC2 também exigem a permissão `ecs:StartTelemetrySession` no perfil do IAM com o qual você executa as instâncias do Amazon EC2. Caso tenha criado o perfil do IAM da instância de contêiner do Amazon ECS antes que as métricas do CloudWatch estivessem disponíveis para o Amazon ECS, talvez seja necessário adicionar essa permissão. Para obter informações sobre o perfil do IAM da instância de contêiner e como anexar a política do IAM gerenciada para instâncias de contêiner, consulte Função do IAM de instância de contêiner do Amazon ECS.
- É possível desabilitar a coleta de métricas do CloudWatch nas instâncias do Amazon EC2 definindo `ECS_DISABLE_METRICS=true` na configuração do agente de contêiner do Amazon ECS. Para obter mais informações, consulte Configuração do agente de contêiner do Amazon ECS.

Métricas recomendadas

O Amazon ECS fornece métricas gratuitas do CloudWatch que você pode usar para monitorar os recursos. A reserva de CPU e memória e a utilização de CPU, memória e sistema de arquivos do EBS no cluster como um todo, bem como a utilização de CPU, memória e sistema de arquivos do EBS nos serviços dos clusters, podem ser medidas usando essas métricas. Para as cargas de trabalho de GPU, você pode medir a reserva de GPU no cluster.

A infraestrutura em que as tarefas do Amazon ECS estão hospedadas nos clusters determina quais métricas estão disponíveis. Para as tarefas hospedadas na infraestrutura do Fargate, o Amazon ECS fornece métricas de utilização de CPU, memória e sistema de arquivos do EBS para ajudar no monitoramento dos serviços. Para tarefas hospedadas em instâncias do EC2, o Amazon ECS fornece métricas de reserva de CPU, memória e GPU e métricas de utilização de CPU e memória no nível de cluster e de serviço. Você precisa monitorar as instâncias do Amazon EC2 que compõem sua infraestrutura subjacente separadamente. Para obter mais informações sobre monitoramento de instâncias do Amazon EC2, consulte Monitorar o Amazon EC2 no Manual do usuário do Amazon EC2.

Para obter informações sobre os alarmes recomendados para uso com o Amazon ECS, consulte um dos itens a seguir no Guia do usuário do Amazon CloudWatch Logs:

- Amazon ECS
- Amazon ECS with Container Insights

Visualizar métricas do Amazon ECS

Depois que os recursos estiverem em execução no cluster, você poderá visualizar as métricas nos consoles do Amazon ECS e do CloudWatch. O console do Amazon ECS fornece uma visualização máxima, mínima e média de 24 horas das métricas do cluster e do serviço. O console do CloudWatch fornece uma exibição detalhada e personalizável dos recursos, bem como o número de tarefas em execução em um serviço.

Console do Amazon ECS

As métricas de utilização de CPU e memória do serviço do Amazon ECS estão disponíveis no console do Amazon ECS. A visualização fornecida para as métricas de serviço mostra os valores médios, mínimos e máximos referentes ao período de 24 horas anterior, com pontos de dados

disponíveis em intervalos de 5 minutos. Para obter mais informações, consulte Métricas de utilização do serviço do Amazon ECS.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Selecione o cluster para o qual você deseja visualizar métricas.
3. Determine as métricas a serem visualizadas.

Para visualizar métricas do	Etapas
Clusters	Na página de detalhes do cluster, escolha a guia Métricas. Também há um link fornecido para o console do CloudWatch para visualizar as métricas do CloudWatch Container Insights, se estiverem ativadas.
Serviços	Na página de detalhes do cluster, na guia Serviços, selecione o serviço. As métricas estão disponíveis na guia Integridade e métricas.

Console do CloudWatch

Para o tipo de execução do Fargate, as métricas de serviço do Amazon ECS também podem ser visualizadas no console do CloudWatch. O console fornece a visualização mais detalhada das métricas do Amazon ECS e você pode personalizar as visualizações para atender às necessidades. É possível visualizar a utilização do serviço e a contagem de tarefas EM EXECUÇÃO do serviço.

Para o tipo de execução do EC2, as métricas de cluster e serviço do Amazon ECS também podem ser visualizadas no console do CloudWatch. O console fornece a visualização mais detalhada das métricas do Amazon ECS e você pode personalizar as visualizações para atender às necessidades.

Para obter informações sobre como visualizar as métricas, consulte View available metrics (Visualizar métricas disponíveis) no Guia do usuário do Amazon CloudWatch.

Métricas do Amazon ECS CloudWatch

É possível usar métricas de uso do CloudWatch para fornecer visibilidade sobre o uso dos recursos da sua conta. Use essas métricas para visualizar o uso do serviço atual nos gráficos e painéis do CloudWatch.

O Amazon ECS envia dados de métricas para o CloudWatch em intervalos de 1 minuto. Essas métricas são coletadas para recursos que tenham tarefas no estado `RUNNING`. Se um cluster, serviço ou outro recurso não tiver tarefas em execução, nenhuma métrica será relatada para esse recurso durante esse período. Por exemplo, se você tiver um cluster contendo um serviço, mas esse serviço não tiver tarefas em um estado `RUNNING`, não haverá o envio de métricas para o CloudWatch. De maneira semelhante, se você tiver dois serviços e um deles tiver tarefas em execução e o outro não, somente as métricas do serviço com tarefas em execução serão enviadas.

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
CPUReservation	A porcentagem de unidades de CPU reservadas no cluster ou serviço. A reserva de CPU (filtrada por `ClusterName`) é medida como o total de unidades de CPU reservadas pelas tarefas do Amazon ECS no cluster, dividido pelo total de unidades de CPU de todas	`ClusterName` .	Média, Mínimo, Máximo	Percentual

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
	as instâncias do Amazon EC2 registradas no cluster. Somente as instâncias do Amazon EC2 no status `ACTIVE` ou `DRAINING` afetarão as métricas de reserva da CPU. Só há suporte para essa métrica em tarefas hospedadas em uma instância do Amazon EC2.			

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
`CPUUtilization`	A porcentagem de unidades de CPU usadas pelo cluster ou serviço. A utilização da CPU no nível do cluster (filtrada por `ClusterName`) é medida como o total de unidades de CPU em uso pelas tarefas do Amazon ECS no cluster, dividido pelo total de unidades de CPU de todas as instâncias do Amazon EC2 registradas no cluster. Somente as instâncias do Amazon EC2 no status `ACTIVE` ou `DRAINING` afetarão as métricas de reserva da CPU. A métrica no nível do cluster só é compatível com tarefas	`ClusterName`, `ServiceName`	Média, Mínimo, Máximo	Percentual

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
	hospedadas em uma instância do Amazon EC2. A utilização da CPU no nível de serviço (filtrada por `ClusterName` e `ServiceName`) é medida como o total de unidades de CPU em uso pelas tarefas que pertencem ao serviço, dividido pelo número total de unidades de CPU reservadas para as tarefas que pertencem ao serviço. A métrica no nível de serviço é compatível com tarefas hospedadas em instâncias do Amazon EC2 e no Fargate.			

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
`MemoryReservation`	O percentual de memória reservada ao executar tarefas no cluster. A reserva de memória do cluster é medida como a memória total reservada pelas tarefas do Amazon ECS no cluster, dividida pela quantidade total de memória para todas as instâncias do Amazon EC2 registradas no cluster. Essa métrica só pode ser filtrada por `ClusterName` . Somente instâncias do Amazon EC2 no status `ACTIVE` ou `DRAINING` afetam as métricas de reserva da memória. A métrica para reserva de	`ClusterName` .	Média, Mínimo, Máximo	Percentual

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
	memória no nível do cluster só é compatíve l com tarefas hospedadas em uma instância do Amazon EC2. **Note** Ao calcular a utilizaçã o da memória, se `MemoryRes ervation` for especific ada, ela será usada no cálculo em vez da memória total.			

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
`MemoryUtilization`	A porcentagem de memória em uso pelo cluster ou serviço. A utilização da memória no nível do cluster (filtrada por `ClusterName`) é medida como a memória total em uso pelas tarefas do Amazon ECS no cluster, dividida pela memória total de todas as instâncias do Amazon EC2 registradas no cluster. Somente instâncias do Amazon EC2 no status `ACTIVE` ou `DRAINING` afetam as métricas de utilização de memória. A métrica no nível do cluster só é compatível com tarefas hospedadas em	`ClusterName`, `ServiceName`	Média, Mínimo, Máximo	Percentual

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
	uma instância do Amazon EC2. A utilização de memória no nível de serviço (filtrada por `ClusterName` e `ServiceName`) é medida como a memória total em uso pelas tarefas que pertencem ao serviço, dividida pela memória total reservada para as tarefas que pertencem ao serviço. A métrica no nível de serviço é compatível com tarefas hospedadas em instâncias do Amazon EC2 e no Fargate.			

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
`EBSFilesystemUtilization`	A porcentagem do sistema de arquivos do Amazon EBS que é usada por tarefas em um serviço. A métrica de utilização do sistema de arquivos do EBS no nível de serviço (filtrada por `ClusterName` e `ServiceName`) é medida como a quantidade total do sistema de arquivos do EBS em uso pelas tarefas que pertencem ao serviço, dividida pela quantidade total de armazenamento do sistema de arquivos do EBS alocada para todas as tarefas que pertencem	`ClusterName`, `ServiceName`	Média, Mínimo, Máximo	Percentual

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
	ao serviço. A métrica de utilização do sistema de arquivos do EBS no nível de serviço só está disponível para tarefas hospedadas em instâncias do Amazon EC2 (usando a versão `1.79.0` do agente de contêiner) e no Fargate (usando a versão `1.4.0` da plataforma) com um volume do EBS anexado. **Note** Para as tarefas hospedadas no Fargate, há espaço no disco que é usado			

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
	somente pelo Fargate. Não há custo associado ao espaço que Fargate usa, mas você verá esse armazenamento adicional usando ferramentas como `df`.			

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
`GPUReservation`	O percentual total de GPUs disponíveis reservadas ao executar tarefas no cluster. A métrica de reserva de GPU no nível do cluster é medida como o número de GPUs reservadas pelas tarefas do Amazon ECS no cluster, dividido pelo número total de GPUs disponíveis em todas as instâncias do Amazon EC2 com GPUs registradas no cluster. Somente instâncias do Amazon EC2 no status `ACTIVE` ou `DRAINING` afetarão as métricas de reserva de GPU.	`ClusterName`	Média, Mínimo, Máximo	Percentual

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
`ActiveConnectionCount`	O número total de conexões simultâneas ativas de clientes aos proxies do Amazon ECS Service Connect que são executadas em tarefas que compartilham o `DiscoveryName` selecionado. Essa métrica só estará disponível se você tiver configurado o Amazon ECS Service Connect. Dimensões válidas: `DiscoveryName` e `DiscoveryName`, `ServiceName`, `ClusterName`.	`DiscoveryName` e `DiscoveryName`, `ServiceName`, `ClusterName`	Média, Mínimo, Máximo, Soma	Contagem

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
`NewConnectionCount`	O número total de novas conexões estabelecidas por clientes com os proxies do Amazon ECS Service Connect que são executados em tarefas que compartilham o `DiscoveryName` selecionado. Essa métrica só estará disponível se você tiver configurado o Amazon ECS Service Connect.	`DiscoveryName` e `DiscoveryName`, `ServiceName`, `ClusterName`	Média, Mínimo, Máximo, Soma	Contagem

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
`ProcessedBytes`	O número total de bytes de tráfego de entrada processados pelos proxies do Service Connect. Essa métrica só estará disponível se você tiver configurado o Amazon ECS Service Connect.	`DiscoveryName e DiscoveryName, ServiceName, ClusterName`	Média, Mínimo, Máximo, Soma	Bytes

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
`RequestCount`	O número de solicitações de tráfego de entrada processadas pelos proxies do Service Connect. Essa métrica só estará disponível se você tiver configurado o Amazon ECS Service Connect. Além disso, você precisa configurar `appProtocol` no mapeamento de portas em sua definição de tarefa.	`DiscoveryName e DiscoveryName, ServiceName, ClusterName`	Média, Mínimo, Máximo, Soma	Contagem

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
`GrpcRequestCount`	O número de solicitações de tráfego gRPC de entrada processadas pelos proxies do Service Connect. Essa métrica só estará disponível se você tiver configurado o Amazon ECS Service Connect e se o `appProtocol` for `GRPC` no mapeamento de portas na definição da tarefa.	`DiscoveryName` e `DiscoveryName`, `ServiceName`, `ClusterName`	Média, Mínimo, Máximo, Soma	Contagem

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
`HTTPCode_Target_2XX_Count`	O número de códigos de resposta HTTP com números 200 a 299 gerados pelas aplicações nessas tarefas. Essas tarefas são os destinos. Essa métrica conta apenas as respostas enviadas aos proxies do Service Connect pelas aplicações dessas tarefas, não as respostas enviadas diretamente. Essa métrica só estará disponível se você tiver configurado o Amazon ECS Service Connect e se o `appProtocol` for `HTTP` ou `HTTP2` no mapeamento de portas na	`TargetDiscoveryName` e `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Média, Mínimo, Máximo, Soma	Contagem

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
	definição da tarefa. Dimensões válidas:			

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
`HTTPCode_Target_3XX_Count`	O número de códigos de resposta HTTP com números 300 a 399 gerados pelas aplicações nessas tarefas. Essas tarefas são os destinos. Essa métrica conta apenas as respostas enviadas aos proxies do Service Connect pelas aplicações dessas tarefas, não as respostas enviadas diretamente. Essa métrica só estará disponível se você tiver configurado o Amazon ECS Service Connect e se o `appProtocol` for `HTTP` ou `HTTP2` no mapeamento de portas na	`TargetDiscoveryName` e `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Média, Mínimo, Máximo, Soma	Contagem

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
	definição da tarefa.			

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
`HTTPCode_Target_4XX_Count`	O número de códigos de resposta HTTP com números 400 a 499 gerados pelas aplicações nessas tarefas. Essas tarefas são os destinos. Essa métrica conta apenas as respostas enviadas aos proxies do Service Connect pelas aplicações dessas tarefas, não as respostas enviadas diretamente. Essa métrica só estará disponível se você tiver configurado o Amazon ECS Service Connect e se o `appProtocol` for `HTTP` ou `HTTP2` no mapeamento de portas na	`TargetDiscoveryName` e `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Média, Mínimo, Máximo, Soma	Contagem

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
	definição da tarefa.			

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
`HTTPCode_Target_5XX_Count`	O número de códigos de resposta HTTP com números 500 a 599 gerados pelas aplicações nessas tarefas. Essas tarefas são os destinos. Essa métrica conta apenas as respostas enviadas aos proxies do Service Connect pelas aplicações dessas tarefas, não as respostas enviadas diretamente. Essa métrica só estará disponível se você tiver configurado o Amazon ECS Service Connect e se o `appProtocol` for `HTTP` ou `HTTP2` no mapeamento de portas na	`TargetDiscoveryName` e `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Média, Mínimo, Máximo, Soma	Contagem

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
	definição da tarefa.			
`RequestCountPerTarget`	O número médio de solicitações recebidas por cada destino que compartilha o `DiscoveryName` selecionado. Essa métrica só estará disponível se você tiver configurado o Amazon ECS Service Connect.	`TargetDiscoveryName` e `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Média	Contagem
`TargetProcessedBytes`	O número total de bytes processados pelos proxies do Service Connect. Essa métrica só estará disponível se você tiver configurado o Amazon ECS Service Connect.	`TargetDiscoveryName` e `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Média, Mínimo, Máximo, Soma	Bytes

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
`TargetResponseTime`	A latência do processamento da solicitação da aplicação. O tempo decorrido, em milissegundos, após a solicitação chegar ao proxy do Service Connect na tarefa de destino até que uma resposta da aplicação de destino seja recebida de volta para o proxy. Essa métrica só estará disponível se você tiver configurado o Amazon ECS Service Connect.	`TargetDiscoveryName` e `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Média, Mínimo, Máximo	Milissegundos

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
`ClientTLSNegotiationErrorCount`	O número total de vezes em que houve falha na conexão TLS. Essa métrica só é usada quando o TLS está habilitado. Essa métrica só estará disponível se você tiver configurado o Amazon ECS Service Connect.	`DiscoveryName` e `DiscoveryName`, `ServiceName`, `ClusterName`	Média, Mínimo, Máximo, Soma	Contagem

Métrica	Descrição	Dimensão válida	Estatísticas úteis	Unidade
`TargetTLSNegotiationErrorCount`	O número total de vezes em que houve falha na conexão TLS devido à falta de certificados do cliente, falhas nas verificações de AWS Private CA ou falhas nas verificações de SAN. Essa métrica só é usada quando o TLS está habilitado. Essa métrica só estará disponível se você tiver configurado o Amazon ECS Service Connect.	`ServiceName` , `ClusterName` , `TargetDiscoveryName` e `TargetDiscoveryName`	Média, Mínimo, Máximo, Soma	Contagem

Dimensões para métricas do Amazon ECS

As métricas do Amazon ECS usam o namespace `AWS/ECS` e fornecem métricas para as dimensões a seguir. O Amazon ECS só envia métricas para recursos que tenham tarefas no estado `RUNNING`. Por exemplo, se você tiver um cluster contendo um serviço, mas esse serviço não tiver tarefas no estado `RUNNING`, não haverá métricas enviadas para o CloudWatch. Se você tiver dois serviços e um deles tiver tarefas em execução e o outro não, somente as métricas do serviço com tarefas em execução serão enviadas.

Dimensão	Definição
`ClusterName`	Essa dimensão filtra os dados que você solicitar para todos os recursos em um cluster especificado. Todas as métricas do Amazon ECS são filtradas por `ClusterName` .
`ServiceName`	Essa dimensão filtra os dados que você solicitar para todos os recursos em um serviço especificado em um determinado cluster.
`DiscoveryName`	Essa dimensão filtra os dados solicitados para as métricas do tráfego para um nome de descoberta de serviço especificado do Service Connect em todos os clusters do Amazon ECS. Observe que uma porta específica em um contêiner em execução pode ter vários nomes de descoberta.
`DiscoveryName, ServiceName, ClusterName`	Essa dimensão filtra os dados que você solicita para métricas de tráfego para um nome de descoberta especificado do Service Connect em tarefas que têm esse nome de descobert a e que são criadas por esse serviço nesse cluster. Use essa dimensão para ver as métricas de tráfego de entrada de um serviço específico caso você tenha reutilizado o mesmo nome de descoberta em vários serviços em namespaces diferentes. Observe que uma porta específica em um contêiner em execução pode ter vários nomes de descoberta.

Dimensão	Definição
`TargetDiscoveryName`	Essa dimensão filtra os dados solicitados para as métricas do tráfego para um nome de descoberta de serviço especificado do Service Connect em todos os clusters do Amazon ECS. Diferentemente de `DiscoveryName` , essas métricas de tráfego medem apenas o tráfego de entrada para esse `DiscoveryName` proveniente de outras tarefas do Amazon ECS que têm uma configuração do Service Connect nesse namespace. Isso inclui tarefas feitas por serviços com uma configuração do Service Connect somente para cliente ou para cliente-servidor. Observe que uma porta específica em um contêiner em execução pode ter vários nomes de descoberta.

Dimensão	Definição
`TargetDiscoveryName, ServiceName, ClusterName`	Essa dimensão filtra os dados que você solicita para métricas de tráfego para um nome de descoberta especificado do Service Connect, mas somente conta tráfego de tarefas criadas por esse serviço nesse cluster. Use essa dimensão para ver as métricas de tráfego de entrada provenientes de um cliente específico em outro serviço. Diferentemente de `DiscoveryName, ServiceName, ClusterName` , essas métricas de tráfego medem apenas o tráfego de entrada para esse `DiscoveryName` proveniente de outras tarefas do Amazon ECS que têm uma configuração do Service Connect nesse namespace. Isso inclui tarefas feitas por serviços com uma configuração do Service Connect somente para cliente ou para cliente-servidor. Observe que uma porta específica em um contêiner em execução pode ter vários nomes de descoberta.

Métricas de uso do AWS Fargate

É possível usar métricas de uso do CloudWatch para fornecer visibilidade sobre o uso dos recursos da sua conta. Use essas métricas para visualizar o uso do serviço atual nos gráficos e painéis do CloudWatch.

As métricas de uso do AWS Fargate correspondem às cotas de serviço da AWS. Também é possível configurar alarmes que alertem você quando o uso se aproximar de uma cota de serviço. Para obter mais informações sobre cotas de serviço do Fargate, consulte Endpoints e cotas do Amazon ECS no Referência geral da Amazon Web Services.

O AWS Fargate publica as seguintes métricas no namespace `AWS/Usage`.

Métrica	Descrição
`ResourceCount`	O número total dos recursos especificados em execução na sua conta. Os recursos são definidos pelas dimensões associadas à métrica.

As dimensões a seguir são usadas para refinar as métricas de uso publicadas pelo AWS Fargate.

Dimensão	Descrição
`Service`	O nome do serviço da AWS que contém o recurso. Para as métricas de uso do AWS Fargate, o valor dessa dimensão é `Fargate`.
`Type`	O tipo de entidade que está sendo relatado. Atualmente, o único valor válido para métricas de uso do AWS Fargate é `Resource`.
`Resource`	O tipo de recurso que está em execução. O tipo de recurso que está em execução. Atualmente, o único valor válido para métricas de uso do AWS Fargate é `vCPU`, que retorna informaçõ es sobre as instâncias em execução.
`Class`	A classe do recurso sob acompanhamento. A classe do recurso sob acompanhamento. Para as métricas de uso do AWS Fargate com vCPU como o valor da dimensão Resource (Recurso), os valores válidos são `Standard/OnDemand` e `Standard/Spot` .

Você pode usar o console do Service Quotas para visualizar o uso em um gráfico e configurar alarmes que alertam quando o uso do AWS Fargate se aproximar de uma cota de serviço. Para obter informações sobre como criar um alarme do CloudWatch para enviar notificações quando estiver próximo de um limite de valor de cota, consulte Service Quotas and Amazon CloudWatch alarms no Guia do usuário do Service Quotas

.

Métricas de reserva de cluster do Amazon ECS

As métricas de reserva de cluster são medidas como a porcentagem de CPU, memória e GPUs reservada por todas as tarefas do Amazon ECS em um cluster em comparação com a CPU, memória e GPUs agregadas que foram registradas para cada instância de contêiner ativa no cluster. Somente instâncias de contêiner no status `ACTIVE` ou `DRAINING` afetarão as métricas de reserva de cluster. Essa métrica é usada somente em clusters com tarefas ou serviços hospedados em instâncias do EC2. Não há suporte em clusters com tarefas hospedadas no AWS Fargate.

```
                              (Total CPU units reserved by tasks in cluster) x 100
Cluster CPU reservation =
 ---------------------------------------------------------------
                           (Total CPU units registered by container instances in
 cluster)
```

```
                                 (Total MiB of memory reserved by tasks in cluster x
 100)
Cluster memory reservation =
 ------------------------------------------------------------------
                              (Total MiB of memory registered by container instances in
 cluster)
```

```
                                (Total GPUs reserved by tasks in cluster x 100)
Cluster GPU reservation =
 ------------------------------------------------------------------
                              (Total GPUs registered by container instances in cluster)
```

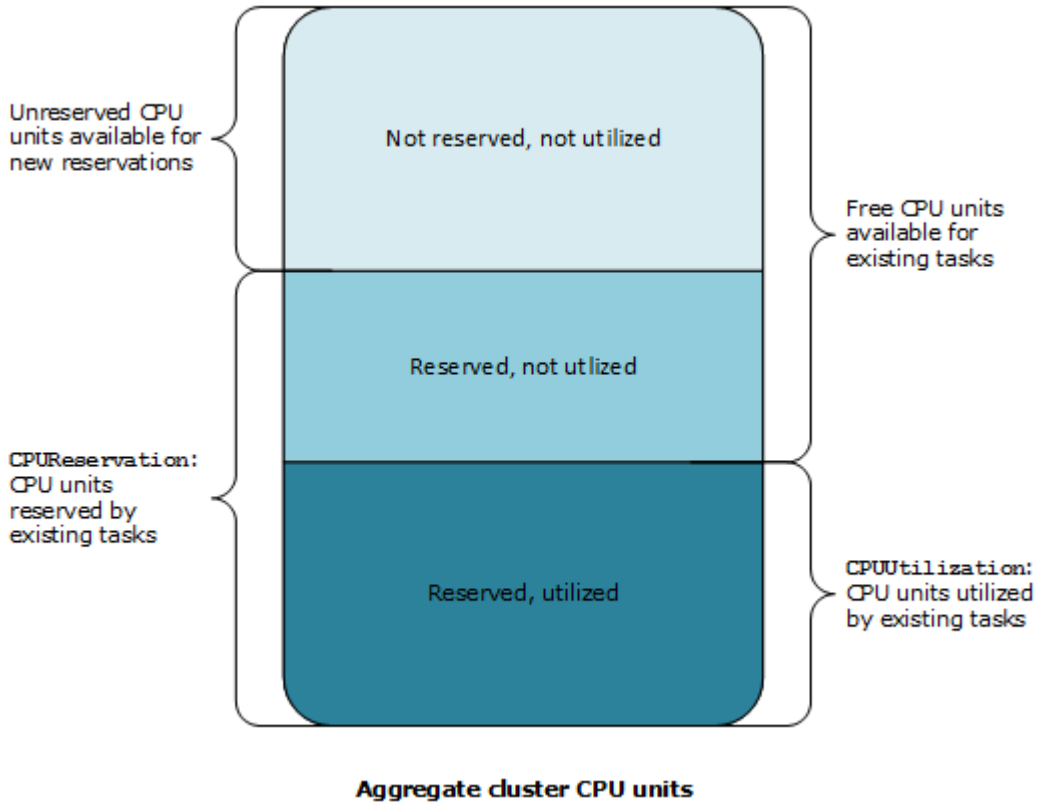
Quando você executa uma tarefa em um cluster, o Amazon ECS analisa a definição de tarefa e reserva as unidades de CPU, os MiB de memória e GPUs agregados especificados nas definições de contêiner. A cada minuto, o Amazon ECS calcula o número de unidades de CPU, MiB de memória e GPUs que estão reservados, no momento, para cada tarefa em execução no cluster. A quantidade total de CPU, memória e GPUs reservada para todas as tarefas em execução no cluster é calculada, e esses números são informados ao CloudWatch como uma porcentagem do total de recursos registrados para o cluster. Se você especificar um limite flexível (`memoryReservation`) na definição da tarefa, ele será usado para calcular a quantidade de memória reservada. Caso contrário, o limite rígido (`memory`) será usado. O total de MiB de memória reservado pelas tarefas em um cluster também inclui o tamanho do volume do sistema de arquivos temporário (`tmpfs`) e `sharedMemorySize`, se definido na definição da tarefa. Para obter mais informações sobre

limites rígidos e flexíveis, tamanho de memória compartilhada e tamanho de volume tmpfs, consulte Parâmetros de definição de tarefa.

Por exemplo, um cluster tem duas instâncias de contêiner ativas registradas: uma instância `c4.4xlarge` e uma instância `c4.large`. A instância `c4.4xlarge` registra-se no cluster com 16.384 unidades de CPU e 30.158 MiB de memória. A instância `c4.large` registra-se com 2.048 unidades de CPU e 3.768 MiB de memória. Os recursos agregados desse cluster são de 18.432 unidades de CPU e 33.926 MiB de memória.

Se uma definição de tarefa reservar 1.024 unidades de CPU e 2.048 MiB de memória, e dez tarefas forem iniciadas com essa definição de tarefa nesse cluster (e nenhuma outra tarefa estiver sendo executada), serão reservados um total de 10.240 unidades de CPU e 20.480 MiB de memória. Isso é relatado no CloudWatch como 55% de reserva de CPU e 60% de reserva de memória para o cluster.

A ilustração a seguir mostra o total de unidades de CPU registradas em um cluster e os meio da reserva e utilização para tarefas existentes e o posicionamento da nova tarefa. Os blocos inferior (Reservado, usado) e central (Reservado, não usado) representam o total de unidades de CPU reservadas para as tarefas existentes em execução no cluster, ou a métrica `CPUReservation` do CloudWatch. O bloco inferior representa as unidades reservadas de CPU que as tarefas em execução estão usando realmente no cluster, ou a métrica `CPUUtilization` do CloudWatch. O bloco superior representa as unidades de CPU que não estão reservadas por tarefas existentes; essas unidades de CPU estão disponíveis para o posicionamento da nova tarefa. As tarefas existentes podem usar essas unidades de CPU reservadas também, se a necessidade de recursos de CPU aumentar. Para mais informações, consulte a `cpu` documentação dos parâmetros de definição de tarefa.



Métricas de utilização de cluster do Amazon ECS

As métricas de utilização de cluster estão disponíveis para utilização de CPU, memória e, quando há um volume do EBS anexado às tarefas, do sistema de arquivos do EBS. Essas métricas só estão disponíveis em clusters com tarefas ou serviços hospedados em instâncias do Amazon EC2. Não há suporte para elas em clusters com tarefas hospedadas no AWS Fargate.

Métricas de utilização de memória e CPU no nível de cluster do Amazon ECS

A utilização de CPU e memória é medida como a porcentagem de CPU e memória usada por todas as tarefas em um cluster quando comparada à CPU e memória agregadas registradas para cada instância ativa do Amazon EC2 registrada no cluster. Somente instâncias do Amazon EC2 no status `ACTIVE` ou `DRAINING` afetam as métricas de utilização do cluster.

```
                                (Total CPU units used by tasks in cluster) x 100
Cluster CPU utilization =
 ------------------------------------------------------------
                           (Total CPU units registered by container instances in
 cluster)
```

```
                                    (Total MiB of memory used by tasks in cluster x
 100)
Cluster memory utilization =
 ------------------------------------------------------------------
                              (Total MiB of memory registered by container instances in
 cluster)
```

A cada minuto, o agente de contêiner do Amazon ECS em cada instância do Amazon EC2 calcula o número de unidades de CPU e MiB de memória que estão sendo usados no momento para cada tarefa em execução nessa instância do Amazon EC2, e essas informações são relatadas ao Amazon ECS. A quantidade total de CPU e de memória utilizada para todas as tarefas em execução no cluster é calculada e esses números são informados ao CloudWatch como uma porcentagem do total de recursos registrados para o cluster.

Por exemplo, um cluster tem duas instâncias ativas do Amazon EC2 registradas, uma instância `c4.4xlarge` e uma instância `c4.large`. A instância `c4.4xlarge` registra-se no cluster com `16,384` unidades de CPU e `30,158` MiB de memória. A instância `c4.large` registra-se com `2,048` unidades de CPU e `3,768` MiB de memória. Os recursos agregados desse cluster são `18,432` unidades de CPU e `33,926` MiB de memória.

Se dez tarefas estiverem sendo executadas nesse cluster e cada uma consumir `1,024` unidades de CPU e `2,048` MiB de memória, um total de `10,240` unidades de CPU e `20,480` MiB de memória serão usados no cluster. Isso é relatado no CloudWatch como 55% de utilização da CPU e 60% de utilização da memória para o cluster.

Utilização do sistema de arquivos do Amazon EBS no nível de cluster do Amazon ECS

A métrica de utilização do sistema de arquivos do EBS no nível de cluster é medida como o total do sistema de arquivos do EBS em uso por tarefas em execução no cluster, dividido pelo total de armazenamento no sistema de arquivos do EBS alocado para todas as tarefas no cluster.

```
                                    (Total GB of EBS filesystem used by tasks in
 cluster x 100)
Cluster EBS filesystem utilization =
 ------------------------------------------------------------------
                                    (Total GB of EBS filesystem allocated to tasks
 in cluster)
```

Métricas de utilização do serviço do Amazon ECS

As métricas de utilização do serviço estão disponíveis para utilização de CPU, memória e, quando há um volume do EBS anexado às tarefas, do sistema de arquivos do EBS. As métricas no nível de serviço são compatíveis com tarefas hospedadas em instâncias do Amazon EC2 e no Fargate.

Utilização de CPU e memória no nível de serviço

A utilização de CPU e memória é medida como a porcentagem de CPU e memória usada pelas tarefas do Amazon ECS que pertencem a um serviço em um cluster quando comparada à CPU e à memória especificadas na definição de tarefa do serviço.

Ao visualizar essas métricas no CloudWatch, você pode escolher estatísticas diferentes:

- Média: a utilização média em todas as tarefas do serviço. Isso é calculado usando a fórmula abaixo.
- Mínimo: a utilização da tarefa com o menor uso de recursos no serviço. Isso representa a porcentagem de CPU ou memória usada pela tarefa que consome menos recursos em comparação com o que foi especificado na definição da tarefa.
- Máximo: a utilização da tarefa com o maior uso de recursos no serviço. Isso representa a porcentagem de CPU ou memória usada pela tarefa que consome mais recursos em comparação com o que foi especificado na definição da tarefa.

As fórmulas a seguir mostram como a estatística Média é calculada:

```
                                (Total CPU units used by tasks in service) x 100
Service CPU utilization =
 ------------------------------------------------------------------
                           (Total CPU units specified in the task definition) x (Number
 of tasks in the service)
```

```
                                   (Total MiB of memory used by tasks in service x
 100)
Service memory utilization =
 ------------------------------------------------------------------
                              (Total MiB of memory specified in the task definition)  x
 (Number of tasks in the service)
```

> **Note**
>
> As fórmulas acima se aplicam somente à estatística Média. As estatísticas Mínimo e Máximo representam a tarefa individual com a menor e a maior utilização de recursos, respectivamente, em vez de um cálculo agregado em todas as tarefas.

O Amazon ECS coleta métricas a cada 20 segundos. A cada minuto, o agente de contêiner do Amazon ECS calcula o número de unidades de CPU e MiB de memória que estão sendo usados no momento para cada tarefa pertencente ao serviço. Essas informações enviadas de volta para o Amazon ECS. A quantidade total de CPU e de memória utilizada para todas as tarefas pertencentes ao serviço em execução no cluster é calculada e esses números são informados ao CloudWatch como uma porcentagem do total de recursos especificados para o serviço na definição de tarefa do serviço. Os valores mínimo e máximo são os menores e maiores das métricas de 20 segundos. A média é o agregado dos 3 valores.

Se você especificar um limite flexível (`memoryReservation`), ele será usado para calcular a quantidade de memória reservada. Caso contrário, o limite rígido (`memory`) será usado. Para obter mais informações sobre limites rígidos e flexíveis, consulte Tamanho da tarefa.

Por exemplo, a definição de tarefa para um serviço especifica um total de 512 unidades de CPU e 1.024 MiB de memória (com o parâmetro `memory` de limite fixo) para todos os contêineres. O serviço tem uma contagem desejada de 1 tarefa em execução, o serviço fica em execução em um cluster com 1 instância de contêiner `c4.large` (com 2.048 unidades de CPU e 3.768 MiB de memória total), e não há nenhuma outra tarefa em execução no cluster. Embora a tarefa especifique 512 unidades de CPU, porque é a única tarefa em execução em uma instância de contêiner com 2.048 unidades de CPU, ela pode usar até quatro vezes o valor especificado (2.048/512). No entanto, a memória especificada de 1.024 MiB é um limite rígido e não pode ser excedido. Nesse caso, a utilização de memória do serviço não pode exceder 100%.

Se o exemplo anterior tivesse usado o limite flexível `memoryReservation`, em vez do parâmetro `memory` de limite fixo, as tarefas do serviço poderiam usar mais que os 1.024 MiB de memória especificados, conforme necessário. Nesse caso, a utilização de memória do serviço poderia exceder 100%.

Se a aplicação tiver um aumento repentino na utilização da memória por um curto período de tempo, você não verá a utilização da memória de serviço aumentando porque o Amazon ECS coleta

vários pontos de dados a cada minuto e, em seguida, os agrega a um ponto de dados enviado ao CloudWatch.

Se essa tarefa estiver executando trabalho que use muito a CPU em um período e estiver usando todas as 2.048 unidades de CPU e 512 MiB de memória disponíveis, o serviço informará a utilização de 400% da CPU e a utilização de 50% da memória. Se a tarefa estiver inativa e usando 128 unidades de CPU e 128 MiB de memória, o serviço informará a utilização de 25% da CPU e utilização de 12,5% da memória.

> **Note**
>
> Neste exemplo, a utilização da CPU só ultrapassará 100% quando as unidades de CPU forem definidas no nível do contêiner. Se você definir unidades de CPU no nível da tarefa, a utilização não ultrapassará o limite definido no nível da tarefa.

Utilização do sistema de arquivos do EBS no nível de serviço

A utilização do sistema de arquivos do EBS no nível de serviço é medido como o total do sistema de arquivos do EBS em uso pelas tarefas que pertencem ao serviço, dividido pelo total de armazenamento do sistema de arquivos do EBS alocado para todas as tarefas que pertencem ao serviço.

Contagem de tarefas **`RUNNING`** de serviço

É possível usar as métricas do CloudWatch para ver o número de tarefas nos serviços que estão no estado `RUNNING`. Por exemplo, você pode definir um alarme do CloudWatch para essa métrica emitir um alerta se o número de tarefas em execução no serviço ficar abaixo de um valor especificado.

Contagem de tarefas **`RUNNING`** do serviço no Amazon ECS CloudWatch Container Insights

Uma métrica de “Número de tarefas em execução” (`RunningTaskCount`) está disponível por cluster e por serviço quando você utiliza o Amazon ECS CloudWatch Container Insights. É possível usar o Container Insights para todos os novos clusters criados ao escolher a configuração da conta `containerInsights`, em clusters individuais, ativando as configurações de cluster durante a criação do cluster ou em clusters existentes usando a API UpdateClusterSettings. As métricas coletadas pelo CloudWatch Container Insights são cobradas como métricas personalizadas. Para obter mais informações sobre os preços do CloudWatch, consulte Preço do CloudWatch.

Para obter mais informações, consulte Métricas do Amazon ECS Container Insights, no Guia do Usuário do Amazon CloudWatch.

Casos de uso de métricas de utilização do serviço do Amazon ECS

A lista a seguir fornece informações sobre quando usar as métricas do Amazon ECS.

- Monitoramento da utilização de recursos: use estatísticas médias para monitorar padrões de consumo de CPU e memória, estabelecer linhas de base de performance e detectar tendências graduais de degradação da performance.
- Otimização de custos: use estatísticas médias para monitorar o uso de recursos, dimensionar os contêineres corretamente, ajustar as reservas com base nos padrões de uso e implementar o escalabilidade agendada.
- Análise comparativa de performance: use estatísticas médias para comparar métricas entre revisões de serviços, estabelecer KPIs de performance e validar melhorias de otimização.
- Detecção do piso de recursos: use estatísticas médias para identificar as necessidades mínimas de recursos durante os períodos de ociosidade, definir reservas apropriadas e detectar quedas anormais.
- Detecção de anomalias: use estatísticas mínimas para detectar quedas incomuns na utilização de recursos que indicam possíveis problemas, como falhas de inicialização ou períodos de ociosidade inesperados.
- Refinamento da política de escalabilidade: use estatísticas mínimas para estabelecer limites ideais de redução horizontal da escala com base na utilização mínima viável para evitar escalabilidade agressiva.
- Planejamento de capacidade: use estatísticas máximas para definir tamanhos de tarefas apropriados e planejar a infraestrutura com espaço suficiente para picos de tráfego.
- Identificação de gargalos de performance: use estatísticas máximas para detectar pontos de saturação de recursos, identificar gargalos e determinar quando aumentar o tamanho da tarefa.
- Configuração da política de escalabilidade: use estatísticas máximas para definir limites ideais de aumento horizontal da escala com base nos padrões de pico e configurar a capacidade de expansão.
- Monitoramento da conformidade com SLAs: use estatísticas máximas para rastrear os tempos de resposta de pico e as taxas de erro para garantir que a performance do serviço atenda aos SLAs definidos.

Informações de métricas relacionadas

Para obter mais informações sobre o Container Insights, consulte os Casos de uso do Container Insights no Guia do usuário do CloudWatch.

Automatização de respostas a erros do Amazon ECS usando o EventBridge

Com o Amazon EventBridge, é possível automatizar os serviços da AWS e responder automaticamente a eventos do sistema, como problemas de disponibilidade de aplicações ou alterações em recursos. Os eventos dos serviços da AWS são entregues ao EventBridge quase em tempo real. É possível criar regras simples para indicar quais eventos são de seu interesse, e quais ações automatizadas devem ser tomadas quando um evento corresponder a uma regra. As ações que podem ser configuradas automaticamente incluem:

- Adicionar eventos a grupos de logs no CloudWatch Logs
- Invocar uma função do AWS Lambda
- Invocar o comando de execução do Amazon EC2
- Transmitir o evento Amazon Kinesis Data Streams
- Ativar a máquina de estado do AWS Step Functions
- Notificar um tópico do Amazon SNS ou uma fila do Amazon Simple Queue Service (Amazon SQS)

Para obter mais informações, consulte Começar a usar o Amazon EventBridge no Manual do usuário do Amazon EventBridge.

É possível usar os eventos do Amazon ECS para o Eventbridge receber notificações quase em tempo real quanto ao estado atual dos clusters do Amazon ECS. Se suas tarefas estiverem usando o tipo de inicialização do EC2, será possível ver o estado das instâncias de contêiner e o estado atual de todas as tarefas em execução nestas instâncias de contêiner. Se as tarefas usarem o tipo de execução do Fargate, você poderá ver o estado das instâncias de contêiner.

Usando o Eventbridge, você pode criar programadores personalizados para o Amazon ECS responsáveis por orquestrar tarefas em clusters e por monitorar o estado dos clusters praticamente em tempo real. É possível eliminar o código de programação e monitoramento que consulta continuamente o serviço do Amazon ECS em busca de alterações no status e, em vez disso,

lidar com alterações no estado do Amazon ECS de forma assíncrona usando qualquer destino do Eventbridge. Os destinos podem incluir AWS Lambda, Amazon Simple Queue Service, Amazon Simple Notification Service ou Amazon Kinesis Data Streams.

Uma sequência de eventos do Amazon ECS garante que cada evento seja entregue pelo menos uma vez. Se eventos duplicados forem enviados, o evento fornecerá informações suficientes para identificar as duplicatas. Para obter mais informações, consulte Processo de eventos do Amazon ECS.

Os eventos são ordenados de maneira relativa, de maneira que você possa identificar facilmente quando um evento ocorreu em relação a outros eventos.

Tópicos

- Eventos do Amazon ECS
- Processo de eventos do Amazon ECS

Eventos do Amazon ECS

O Amazon ECS rastreia o estado de cada tarefa e serviço. Se o estado de uma tarefa ou serviço for alterado, um evento será gerado e enviado ao Amazon EventBridge. Esses eventos são classificados como eventos de alteração de estado de tarefas e eventos de ação de serviços. Esses eventos e as causas possíveis serão descritos com mais detalhes nas seções a seguir.

O Amazon ECS gera e envia os seguintes tipos de eventos ao EventBridge:

- Alteração do estado da instância de contêiner
- Alteração do estado da tarefa
- Alteração no estado de implantação
- Ação do atendimento

> **Note**
>
> O Amazon ECS pode adicionar outros tipos de evento, origens e detalhes no futuro. Se você estiver desserializando dados JSON de eventos no código, certifique-se de a aplicação esteja preparada para processar propriedades desconhecidas para evitar problemas se, e quando, essas propriedades adicionais forem adicionadas.

Em alguns casos, vários eventos são gerados para a mesma atividade. Por exemplo, quando uma tarefa é iniciada em uma instância de contêiner, um evento de alteração do estado da tarefa é gerado para a nova tarefa. Um evento de alteração de estado da instância de contêiner é gerado para compensar a alteração nos recursos disponíveis, como CPU, memória, portas disponíveis, na instância de contêiner. Da mesma forma, caso uma instância de contêiner seja terminada, os eventos são gerados para a instância de contêiner, o status da conexão do agente de contêiner e cada tarefa que estava em execução na instância de contêiner.

Os eventos de alteração de estado de contêiner e os eventos de alteração de estado de tarefas contêm dois campos `version`; um no corpo principal do evento e um no objeto `detail` do evento. Veja a seguir uma descrição das diferenças entre esses dois campos.

- O campo `version` no corpo principal do evento é definido como `0` em todos os eventos. Para obter mais informações sobre parâmetros do EventBridge, consulte metadados de eventos de serviços da AWS no Guia do usuário do Amazon EventBridge.
- O campo `version` no objeto `detail` do evento descreve a versão do recurso associado. Sempre que um recurso muda de estado, essa versão é incrementada. Como os eventos podem ser enviados várias vezes, esse campo permite identificar eventos duplicados. Eventos duplicados têm a mesma versão no objeto `detail`. Caso esteja replicando o estado da tarefa e da instância de contêiner do Amazon ECS com o Eventbridge, você poderá comparar a versão de um recurso relatado pelas APIs do Amazon ECS com a versão relatada no Eventbridge para o recurso (dentro do objeto `detail`) para verificar se a versão no fluxo de eventos é atual.

Os eventos de ação de serviços contêm apenas o campo `version` no corpo principal.

Eventos de ação de serviços especificam o serviço em dois campos diferentes:

- Em caso de eventos gerados por `create-service`, o serviço está no campo `serviceName`.
- Em caso de eventos gerados por `update-service`, o serviço está no campo `service`.

Se você utiliza ferramentas automatizadas para eventos de serviços, é necessário codificar para ambos os campos.

Para obter informações sobre como criar uma regra para os eventos de ação de um serviço, consulte Eventos de ação do serviço do Amazon ECS.

Para obter outras informações sobre como integrar o Amazon ECS e o EventBridge, consulte Integrating Amazon EventBridge and Amazon ECS (Como integrar o Amazon EventBridge e o Amazon ECS).

Eventos de alteração no estado da instância de contêiner do Amazon ECS

Os seguintes cenários causam eventos de alteração de estado da instância de contêiner:

As operações de API `StartTask`, `RunTask` ou `StopTask` são chamadas diretamente ou com o AWS Management Console ou os SDKs.

A realização ou a parada de tarefas em uma instância de contêiner modifica os recursos disponíveis na instância de contêiner, como CPU, memória e portas disponíveis.

O programador de serviços do Amazon ECS inicia ou interrompe uma tarefa.

A realização ou a parada de tarefas em uma instância de contêiner modifica os recursos disponíveis na instância de contêiner, como CPU, memória e portas disponíveis.

O agente de contêiner do Amazon ECS chama a operação da API `SubmitTaskStateChange` com um status `STOPPED` para uma tarefa com um status desejado de `RUNNING`.

O agente de contêiner do Amazon ECS monitora o estado de tarefas nas instâncias de contêiner e relata todas as alterações ocorridas no estado. Caso uma tarefa que deveria ser `RUNNING` seja transicionada para `STOPPED`, o agente libera os recursos que foram alocados para a tarefa parada, como CPU, memória e portas disponíveis.

O registro da instância de contêiner é cancelado com a operação da API `DeregisterContainerInstance` diretamente, com o AWS Management Console ou os SDKs.

O cancelamento do registro de uma instância de contêiner altera o status da instância de contêiner e o status de conexão do agente de contêiner do Amazon ECS.

Uma tarefa foi parada quando a instância do EC2 tiver sido interrompida.

Quando você para uma instância de contêiner, as tarefas em execução nela são transicionadas para o status `STOPPED`.

O agente de contêiner do Amazon ECS registra uma instância de contêiner pela primeira vez.

A primeira vez em que o agente de contêiner do Amazon ECS registra uma instância de contêiner (na inicialização ou quando executado pela primeira vez manualmente) faz com que seja criado um evento de alteração do estado para a instância.

O agente de contêiner do Amazon ECS se conecta ou se desconecta do Amazon ECS.

Quando o agente de contêiner do Amazon ECS se conecta ou se desconecta do backend do Amazon ECS, ele altera o status `agentConnected` da instância de contêiner.

> **Note**
>
> O agente de contêiner do Amazon ECS se desconecta e se reconecta várias vezes por hora como parte da sua operação normal. Portanto, devem ser esperados eventos de conexão do agente. Esses eventos não são uma indicação de que há um problema com o agente de contêiner ou sua instância de contêiner.

Você faz o upgrade do agente de contêiner do Amazon ECS em uma instância.

Os detalhes da instância de contêiner contêm um objeto para a versão do agente de contêiner. Caso você atualize o agente, essas informações da versão mudam e geram um evento.

Example Evento de alteração no estado da instância de contêiner

Os eventos de alteração do estado da instância de contêiner são entregues no formato a seguir. A seção `detail` a seguir se assemelha ao objeto ContainerInstance retornado de uma operação de API DescribeContainerInstances na Referência da API do Amazon Elastic Container Service. Para obter mais informações sobre parâmetros do EventBridge, consulte metadados de eventos de serviços da AWS no Guia do usuário do Amazon EventBridge.

```
{
  "version": "0",
  "id": "8952ba83-7be2-4ab5-9c32-6687532d15a2",
  "detail-type": "ECS Container Instance State Change",
  "source": "aws.ecs",
  "account": "111122223333",
  "time": "2016-12-06T16:41:06Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ecs:us-east-1:111122223333:container-instance/
b54a2a04-046f-4331-9d74-3f6d7f6ca315"
  ],
  "detail": {
    "agentConnected": true,
    "attributes": [
```

```
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.syslog"
      },
      {
        "name": "com.amazonaws.ecs.capability.task-iam-role-network-host"
      },
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.awslogs"
      },
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.json-file"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.17"
      },
      {
        "name": "com.amazonaws.ecs.capability.privileged-container"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.18"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.19"
      },
      {
        "name": "com.amazonaws.ecs.capability.ecr-auth"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.20"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.21"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.22"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.23"
      },
      {
        "name": "com.amazonaws.ecs.capability.task-iam-role"
      }
    ],
    "clusterArn": "arn:aws:ecs:us-east-1:111122223333:cluster/default",
```

```
    "containerInstanceArn": "arn:aws:ecs:us-east-1:111122223333:container-instance/
b54a2a04-046f-4331-9d74-3f6d7f6ca315",
    "ec2InstanceId": "i-f3a8506b",
    "registeredResources": [
      {
        "name": "CPU",
        "type": "INTEGER",
        "integerValue": 2048
      },
      {
        "name": "MEMORY",
        "type": "INTEGER",
        "integerValue": 3767
      },
      {
        "name": "PORTS",
        "type": "STRINGSET",
        "stringSetValue": [
          "22",
          "2376",
          "2375",
          "51678",
          "51679"
        ]
      },
      {
        "name": "PORTS_UDP",
        "type": "STRINGSET",
        "stringSetValue": []
      }
    ],
    "remainingResources": [
      {
        "name": "CPU",
        "type": "INTEGER",
        "integerValue": 1988
      },
      {
        "name": "MEMORY",
        "type": "INTEGER",
        "integerValue": 767
      },
      {
        "name": "PORTS",
```

```
        "type": "STRINGSET",
        "stringSetValue": [
          "22",
          "2376",
          "2375",
          "51678",
          "51679"
        ]
      },
      {
        "name": "PORTS_UDP",
        "type": "STRINGSET",
        "stringSetValue": []
      }
    ],
    "status": "ACTIVE",
    "version": 14801,
    "versionInfo": {
      "agentHash": "aebcbca",
      "agentVersion": "1.13.0",
      "dockerVersion": "DockerVersion: 1.11.2"
    },
    "updatedAt": "2016-12-06T16:41:06.991Z"
  }
}
```

Eventos de alteração de estado de tarefa do Amazon ECS

Os seguintes cenários causam eventos de alteração do estado da tarefa:

As operações de API `StartTask`, `RunTask` ou `StopTask` são chamadas diretamente ou com o AWS Management Console, AWS CLI ou os SDKs.

A inicialização ou a parada de tarefas cria novos recursos de tarefa ou modifica o estado de recursos da tarefa existente.

O programador de serviços do Amazon ECS inicia ou interrompe uma tarefa.

A inicialização ou a parada de tarefas cria novos recursos de tarefa ou modifica o estado de recursos da tarefa existente.

O agente de contêiner do Amazon ECS chama a operação da API `SubmitTaskStateChange`.

Para o tipo de execução do Amazon EC2, o agente de contêiner do Amazon ECS monitora o estado das tarefas nas instâncias de contêiner. O agente de contêiner do Amazon ECS relata todas as alterações de estado. Alterações de estado podem incluir mudanças de `PENDING` para `RUNNING` ou de `RUNNING` para `STOPPED`.

O cancelamento do registro de instância de contêiner subjacente é cancelado com a operação de API `DeregisterContainerInstance` e o sinalizador `force`, seja diretamente ou com o AWS Management Console ou os SDKs.

O cancelamento do registro de uma instância de contêiner altera o status da instância de contêiner e o status de conexão do agente de contêiner do Amazon ECS. Caso as tarefas estejam em execução na instância de contêiner, o sinalizador `force` deve ser definido para permitir o cancelamento do registro. Isso para todas as tarefas na instância.

A instância de contêiner subjacente é parada ou encerrada.

Quando você para ou encerra uma instância de contêiner, as tarefas em execução nela são transicionadas para o status `STOPPED`.

Um contêiner na tarefa muda de estado.

O agente de contêiner do Amazon ECS monitora o estado dos contêineres dentro das tarefas. Por exemplo, caso um contêiner em execução dentro de uma tarefa seja interrompido, essa alteração no estado do contêiner gera um evento.

Uma tarefa que utiliza o fornecedor de capacidade do Fargate Spot recebe um aviso de término.

Quando uma tarefa está usando o provedor de capacidade do `FARGATE_SPOT` e é interrompida devido a uma interrupção do Spot, um evento de alteração de estado da tarefa é gerado.

Example Evento de alteração no estado da tarefa

Os eventos de alteração do estado da tarefa são entregues no formato a seguir. Observe o seguinte sobre os campos:

- O status de integridade do evento não está disponível no evento de alteração do estado da tarefa. Se você precisar do status de integridade da tarefa, poderá executar describe-tasks.
- Quando os contêineres usam uma imagem hospedada no Amazon ECR, o campo `imageDigest` é devolvido.

- Os valores dos campos `createdAt`, `connectivityAt`, `pullStartedAt`, `startedAt`, `pullStoppedAt` e `updatedAt` são carimbos de data/hora de string ISO.
- O valor `detail-type` é "Alteração no estado da tarefa do ECS".
- Quando o evento é gerado para uma tarefa interrompida, os campos `stoppedReason` e `stoppedCode` fornecem informações adicionais sobre por que a tarefa foi interrompida (por exemplo, "Iniciada pelo usuário").

Para obter mais informações sobre parâmetros do EventBridge, consulte Metadados de eventos de serviços da AWS na Referência de eventos do Amazon EventBridge.

Para obter informações sobre como configurar uma regra de evento do Amazon EventBridge que captura apenas eventos de tarefa em que a execução da tarefa foi interrompida porque um de seus contêineres essenciais foi encerrado, consulte Enviar alertas do Amazon Simple Notification Service de eventos de tarefa interrompida do Amazon ECS


```
{
    "version": "0",
    "id": "105f6bb1-4da6-c630-4965-35383018cbca",
    "detail-type": "ECS Task State Change",
    "source": "aws.ecs",
    "account": "123456789012",
    "time": "2025-05-06T11:02:34Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/a1173316d40a45dea9"
    ],
    "detail": {
        "attachments": [
            {
                "id": "fe3a9a46-6a47-40ee-afd9-7952ae90a75a",
                "type": "eni",
                "status": "ATTACHED",
                "details": [
                    {
                        "name": "subnetId",
                        "value": "subnet-0d0eab1bb38d5ca64"
                    },
                    {
                        "name": "networkInterfaceId",
                        "value": "eni-0103a2f01bad57d71"
                    },
```

```
                    {
                        "name": "macAddress",
                        "value": "0e:50:d1:c1:77:81"
                    },
                    {
                        "name": "privateDnsName",
                        "value": "ip-10-0-1-163.ec2.internal"
                    },
                    {
                        "name": "privateIPv4Address",
                        "value": "10.0.1.163"
                    }
                ]
            }
        ],
        "attributes": [
            {
                "name": "ecs.cpu-architecture",
                "value": "x86_64"
            }
        ],
        "availabilityZone": "us-east-1b",
        "capacityProviderName": "FARGATE",
        "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/example-cluster",
        "connectivity": "CONNECTED",
        "connectivityAt": "2025-05-06T11:02:17.19Z",
        "containers": [
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example-
cluster/a1173316d40a45dea9/a0a99b87-baa8-4bf6-b9f1-a9a95917a635",
                "lastStatus": "RUNNING",
                "name": "web",
                "image": "nginx",
                "imageDigest":
 "sha256:c15da6c91de8d2f436196f3a768483ad32c258ed4e1beb3d367a27ed67253e66",
                "runtimeId": "a1173316d40a45dea9-0265927825",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/
a1173316d40a45dea9",
                "networkInterfaces": [
                    {
                        "attachmentId": "fe3a9a46-6a47-40ee-afd9-7952ae90a75a",
                        "privateIpv4Address": "10.0.1.163"
                    }
                ],
```

```
                "cpu": "99",
                "memory": "100"
            },
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example-
cluster/a1173316d40a45dea9/a2010e2d-ba7c-4135-8b79-e0290ff3cd8c",
                "lastStatus": "RUNNING",
                "name": "aws-guardduty-agent-nm40lC",
                "imageDigest":
 "sha256:bf9197abdf853607e5fa392b4f97ccdd6ca56dd179be3ce8849e552d96582ac8",
                "runtimeId": "a1173316d40a45dea9-2098416933",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/
a1173316d40a45dea9",
                "networkInterfaces": [
                    {
                        "attachmentId": "fe3a9a46-6a47-40ee-afd9-7952ae90a75a",
                        "privateIpv4Address": "10.0.1.163"
                    }
                ],
                "cpu": "null"
            },
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example-
cluster/a1173316d40a45dea9/dccf0ca2-d929-471f-a5c3-98006fd4379e",
                "lastStatus": "RUNNING",
                "name": "aws-otel-collector",
                "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.32.0",
                "imageDigest":
 "sha256:7a1b3560655071bcacd66902c20ebe9a69470d5691fe3bd36baace7c2f3c4640",
                "runtimeId": "a1173316d40a45dea9-4027662657",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/
a1173316d40a45dea9",
                "networkInterfaces": [
                    {
                        "attachmentId": "fe3a9a46-6a47-40ee-afd9-7952ae90a75a",
                        "privateIpv4Address": "10.0.1.163"
                    }
                ],
                "cpu": "0"
            }
        ],
        "cpu": "256",
        "createdAt": "2025-05-06T11:02:13.877Z",
        "desiredStatus": "RUNNING",
```

```
        "enableExecuteCommand": false,
        "ephemeralStorage": {
            "sizeInGiB": 20
        },
        "group": "family:webserver",
        "launchType": "FARGATE",
        "lastStatus": "RUNNING",
        "memory": "512",
        "overrides": {
            "containerOverrides": [
                {
                    "name": "web"
                },
                {
                    "environment": [
                        {
                            "name": "CLUSTER_NAME",
                            "value": "example-cluster"
                        },
                        {
                            "name": "REGION",
                            "value": "us-east-1"
                        },
                        {
                            "name": "HOST_PROC",
                            "value": "/host_proc"
                        },
                        {
                            "name": "AGENT_RUNTIME_ENVIRONMENT",
                            "value": "ecsfargate"
                        },
                        {
                            "name": "STAGE",
                            "value": "prod"
                        }
                    ],
                    "memory": 128,
                    "name": "aws-guardduty-agent-nm40lC"
                },
                {
                    "name": "aws-otel-collector"
                }
            ]
        },
```

```
        "platformVersion": "1.4.0",
        "pullStartedAt": "2025-05-06T11:02:24.162Z",
        "pullStoppedAt": "2025-05-06T11:02:33.493Z",
        "startedAt": "2025-05-06T11:02:34.325Z",
        "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/
a1173316d40a45dea9",
        "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/
webserver:5",
        "updatedAt": "2025-05-06T11:02:34.325Z",
        "version": 3
    }
}
```

Example

A seguir está um exemplo de um evento de alteração no estado uma tarefa para o tipo de execução do EC2.

```
{
    "version": "0",
    "id": "a65cf262-f104-0dd5-ceda-4b09ba71a441",
    "detail-type": "ECS Task State Change",
    "source": "aws.ecs",
    "account": "123456789012",
    "time": "2025-05-12T13:12:06Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ecs:us-east-1:123456789012:task/example/c1ffa94f19a540ed8d9f7e1d2a5d"
    ],
    "detail": {
        "attachments": [
            {
                "id": "52333e3b-b812-41a8-b057-9ed184bbe5e1",
                "type": "eni",
                "status": "ATTACHED",
                "details": [
                    {
                        "name": "subnetId",
                        "value": "subnet-0d0eab1bb38d5ca64"
                    },
                    {
                        "name": "networkInterfaceId",
                        "value": "eni-0ea90f746500773a4"
```

```
                    },
                    {
                        "name": "macAddress",
                        "value": "0e:d5:9b:ce:49:fb"
                    },
                    {
                        "name": "privateDnsName",
                        "value": "ip-10-0-1-37.ec2.internal"
                    },
                    {
                        "name": "privateIPv4Address",
                        "value": "10.0.1.37"
                    }
                ]
            }
        ],
        "attributes": [
            {
                "name": "ecs.cpu-architecture",
                "value": "x86_64"
            }
        ],
        "availabilityZone": "us-east-1b",
        "capacityProviderName": "Infra-ECS-Cluster-example-fa84e0cc-
AsgCapacityProvider-OseQJU9pizmp",
        "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/example",
        "connectivity": "CONNECTED",
        "connectivityAt": "2025-05-12T13:11:44.98Z",
        "containerInstanceArn": "arn:aws:ecs:us-east-1:123456789012:container-instance/
example/d1d84798400f49f3b21cb61610c1e",
        "containers": [
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example/
c1ffa94f19a540ed8d9f7e1d2a5d3626/197d0994-5367-4a6d-9f9a-f075e4a6",
                "lastStatus": "RUNNING",
                "name": "aws-otel-collector",
                "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.32.0",
                "imageDigest":
 "sha256:7a1b3560655071bcacd66902c20ebe9a69470d5691fe3bd36baace7c2f3c4640",
                "runtimeId":
 "8e926f0ccd8fe2b459926f49584ba6d33a3d9f61398dbabe944ee6a13a8ff3a1",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example/
c1ffa94f19a540ed8d9f7e1d2a5d",
                "networkInterfaces": [
```

```
                    {
                        "attachmentId": "52333e3b-b812-41a8-b057-9ed184bbe5e1",
                        "privateIpv4Address": "10.0.1.37"
                    }
                ],
                "cpu": "0"
            },
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example/
c1ffa94f19a540ed8d9f7e1d2a5d3626/cab39ef0-9c50-459d-844b-b9d51d73d",
                "lastStatus": "RUNNING",
                "name": "web",
                "image": "nginx",
                "imageDigest":
 "sha256:c15da6c91de8d2f436196f3a768483ad32c258ed4e1beb3d367a27ed67253e66",
                "runtimeId":
 "9f1c73f0094f051541d9e5c2ab1e172d83c4eb5171bcc857c4504b02770ff3b8",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example/
c1ffa94f19a540ed8d9f7e1d2a5d",
                "networkInterfaces": [
                    {
                        "attachmentId": "52333e3b-b812-41a8-b057-9ed184bbe5e1",
                        "privateIpv4Address": "10.0.1.37"
                    }
                ],
                "cpu": "99",
                "memory": "100"
            }
        ],
        "cpu": "256",
        "createdAt": "2025-05-12T13:11:44.98Z",
        "desiredStatus": "RUNNING",
        "enableExecuteCommand": false,
        "group": "family:webserver",
        "launchType": "EC2",
        "lastStatus": "RUNNING",
        "memory": "512",
        "overrides": {
            "containerOverrides": [
                {
                    "name": "aws-otel-collector"
                },
                {
                    "name": "web"
```

```
                }
            ]
        },
        "pullStartedAt": "2025-05-12T13:11:59.491Z",
        "pullStoppedAt": "2025-05-12T13:12:05.896Z",
        "startedAt": "2025-05-12T13:12:06.053Z",
        "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example/
c1ffa94f19a540ed8d9f7e1d2a5d",
        "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/
webserver",
        "updatedAt": "2025-05-12T13:12:06.053Z",
        "version": 4
    }
}
```

Eventos de ação do serviço do Amazon ECS

O Amazon ECS envia eventos de ação de serviço com o tipo de detalhe ECS Service Action (Eventos de serviço do ECS). Ao contrário da instância de contêiner e dos eventos de alteração de estado da tarefa, os eventos de ação de serviços não incluem um número de versão no campo de resposta `details`. Veja a seguir um padrão de evento que é usado para criar uma regra do Eventbridge para eventos de ação de serviços do Amazon ECS. Para obter mais informações, consulte Introdução ao EventBridge no Guia do usuário do Amazon EventBridge.

```
{
    "source": [
        "aws.ecs"
    ],
    "detail-type": [
        "ECS Service Action"
    ]
}
```

O Amazon ECS envia eventos com os tipos de evento `INFO`, `WARN` e `ERROR`. A seguir estão os eventos de ação de serviço.

Eventos de ação de serviço com tipo de evento **`INFO`**

`SERVICE_STEADY_STATE`

O serviço é saudável e no número desejado de tarefas, atingindo assim um estado estável. O programador de serviços relata o status periodicamente, portanto, você pode receber essa mensagem várias vezes.

`TASKSET_STEADY_STATE`

O conjunto de tarefas é saudável e no número desejado de tarefas, atingindo assim um estado estável.

`CAPACITY_PROVIDER_STEADY_STATE`

Um provedor de capacidade associado a um serviço atinge um estado estável.

`SERVICE_DESIRED_COUNT_UPDATED`

Quando o agendador de serviço atualiza a contagem desejada calculada para um serviço ou conjunto de tarefas. Esse evento não é enviado quando a contagem desejada é atualizada manualmente por um usuário.

`TASKS_STOPPED`

O serviço interrompeu a tarefa em execução.

`SERVICE_DEPLOYMENT_IN_PROGRESS`

Uma implantação de serviço está em andamento. A implantação do serviço pode ser uma reversão ou uma nova revisão do serviço.

`SERVICE_DEPLOYMENT_COMPLETED`

A implantação de um serviço está em estado estável e concluída. A implantação do serviço pode ser uma reversão ou a implantação de uma revisão atualizada do serviço.

Eventos de ação de serviço com tipo de evento **`WARN`**

`SERVICE_TASK_START_IMPAIRED`

O serviço não consegue iniciar tarefas com êxito de maneira consistente.

`SERVICE_DISCOVERY_INSTANCE_UNHEALTHY`

Um serviço que usa a descoberta de serviço contém uma tarefa não íntegra. O agendador de serviço detecta que uma tarefa dentro de um registro de serviço não está íntegra.

`VPC_LATTICE_TARGET_UNHEALTHY`

O serviço que usa o VPC Lattice detectou que um dos destinos do VPC Lattice não está íntegro.

Eventos de ação de serviço com tipo de evento **`ERROR`**

`SERVICE_DAEMON_PLACEMENT_CONSTRAINT_VIOLATED`

Uma tarefa em um serviço que usa a estratégia do agendador de serviço `DAEMON` não atende mais à estratégia de restrição de posicionamento para o serviço.

`ECS_OPERATION_THROTTLED`

O programador de serviço teve sua utilização controlada devido aos limites de controle de utilização da API do Amazon ECS.

`SERVICE_DISCOVERY_OPERATION_THROTTLED`

O agendador de serviço foi limitado devido aos limites de aceleração da API do AWS Cloud Map. Isto pode ocorrer em serviços configurados para usar a descoberta de serviço.

`SERVICE_TASK_PLACEMENT_FAILURE`

O agendador de serviço não consegue colocar uma tarefa. A causa será descrita no campo `reason`.

Uma causa comum para esse evento de serviço que está sendo gerado é a falta de recursos no cluster para posicionar a tarefa. Por exemplo, não há capacidade suficiente de CPU ou memória nas instâncias de contêiner disponíveis ou não há nenhuma instância de contêiner disponível. Outra causa comum é quando o agente de contêiner do Amazon ECS é desconectado na instância de contêiner, fazendo com que o programador não consiga colocar a tarefa.

`SERVICE_TASK_CONFIGURATION_FAILURE`

O agendador de serviço não consegue colocar uma tarefa devido a um erro de configuração. A causa será descrita no campo `reason`.

Uma causa comum da geração desse evento de serviço são etiquetas que estavam sendo aplicadas ao serviço, mas o usuário ou o perfil não aceitou o novo formato de nome do recurso da Amazon (ARN) na região. Para obter mais informações, consulte Nomes de recursos da Amazon (ARNs) e IDs. Outra causa comum é que o Amazon ECS não conseguiu assumir a função do IAM da tarefa fornecida.

`SERVICE_HEALTH_UNKNOWN`

O serviço não conseguiu descrever os dados de integridade das tarefas.

`SERVICE_DEPLOYMENT_FAILED`

A implantação de um serviço não atingiu o estado estável. Isso acontece quando um CloudWatch é acionado ou o disjuntor detecta uma falha na implantação do serviço.

Example Evento de estado estacionário do serviço

Os eventos de estado constante do serviço são entregues no formato seguinte. Para obter mais informações sobre os parâmetros do EventBridge, consulte Eventos no EventBridge no Guia do usuário do Amazon EventBridge.

```
{
    "version": "0",
    "id": "af3c496d-f4a8-65d1-70f4-a69d52e9b584",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:27:22Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "INFO",
        "eventName": "SERVICE_STEADY_STATE",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "createdAt": "2019-11-19T19:27:22.695Z"
    }
}
```

Example Evento de estado estacionário do provedor de capacidade

Os eventos de estado constante do provedor de capacidade são entregues no formato a seguir.

```
{
    "version": "0",
    "id": "b9baa007-2f33-0eb1-5760-0d02a572d81f",
    "detail-type": "ECS Service Action",
```

```
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:37:00Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "INFO",
        "eventName": "CAPACITY_PROVIDER_STEADY_STATE",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "capacityProviderArns": [
            "arn:aws:ecs:us-west-2:111122223333:capacity-provider/ASG-tutorial-
capacity-provider"
        ],
        "createdAt": "2019-11-19T19:37:00.807Z"
    }
}
```

Example Evento de início da tarefa de serviço com deficiência

Os eventos de início da tarefa de serviço com deficiência são entregues no formato a seguir.

```
{
    "version": "0",
    "id": "57c9506e-9d21-294c-d2fe-e8738da7e67d",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:55:38Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "WARN",
        "eventName": "SERVICE_TASK_START_IMPAIRED",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "createdAt": "2019-11-19T19:55:38.725Z"
    }
}
```

Example Evento de falha no posicionamento da tarefas de serviço

Os eventos de falha de posicionamento de tarefas de serviço são entregues no formato a seguir. Para obter mais informações, consulte Eventos no EventBridge no Guia do usuário do Amazon EventBridge.

No exemplo a seguir, a tarefa estava tentando usar o provedor de capacidade `FARGATE_SPOT`, mas o programador de serviços não conseguiu adquirir nenhuma capacidade do Fargate Spot.

```
{
    "version": "0",
    "id": "ddca6449-b258-46c0-8653-e0e3a6d0468b",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:55:38Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "ERROR",
        "eventName": "SERVICE_TASK_PLACEMENT_FAILURE",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "capacityProviderArns": [
            "arn:aws:ecs:us-west-2:111122223333:capacity-provider/FARGATE_SPOT"
        ],
        "reason": "RESOURCE:FARGATE",
        "createdAt": "2019-11-06T19:09:33.087Z"
    }
}
```

No exemplo a seguir para o tipo de execução do EC2, a tarefa tentou iniciar na instância de contêiner `2dd1b186f39845a584488d2ef155c131`, mas o programador de serviços não conseguiu colocar a tarefa devido a CPU insuficiente.

```
{
  "version": "0",
  "id": "ddca6449-b258-46c0-8653-e0e3a6d0468b",
  "detail-type": "ECS Service Action",
  "source": "aws.ecs",
  "account": "111122223333",
```

```
  "time": "2019-11-19T19:55:38Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
  ],
  "detail": {
    "eventType": "ERROR",
    "eventName": "SERVICE_TASK_PLACEMENT_FAILURE",
    "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
    "containerInstanceArns": [
    "arn:aws:ecs:us-west-2:111122223333:container-instance/
default/2dd1b186f39845a584488d2ef155c131"
    ],
    "reason": "RESOURCE:CPU",
    "createdAt": "2019-11-06T19:09:33.087Z"
  }
}
```

Eventos de alteração no estado da implantação do serviço do Amazon ECS

O Amazon ECS envia eventos de estado de alteração de implantação de serviço com o tipo de detalhe ECS Deployment State Change (Alteração do estado de implantação do ECS). Veja a seguir um padrão de evento que é usado para criar uma regra do Eventbridge para eventos de alteração de estado de implantação de serviços do Amazon ECS. Para obter mais informações sobre a criação de uma regra do EventBridge, consulte Conceitos básicos do Amazon EventBridge no Guia do usuário do Amazon EventBridge.

```
{
    "source": [
        "aws.ecs"
    ],
    "detail-type": [
        "ECS Deployment State Change"
    ]
}
```

O Amazon ECS envia eventos com os tipos de evento `INFO` e `ERROR`. Para obter mais informações, consulte Eventos de ação do serviço do Amazon ECS.

Veja a seguir os eventos de alteração de estado de implantação de serviços.

`SERVICE_DEPLOYMENT_IN_PROGRESS`

A implantação do serviço está em andamento. Esse evento é enviado para implantações iniciais e implantações de reversão.

`SERVICE_DEPLOYMENT_COMPLETED`

A implantação do serviço foi concluída. Esse evento é enviado quando um serviço atinge um estado estacionário após uma implantação.

`SERVICE_DEPLOYMENT_FAILED`

Houve falha na implantação do serviço. Esse evento é enviado para serviços com a lógica de disjuntor de implantação ativada.

Example evento de implantação de serviço em andamento

Os eventos de implantação de serviço em andamento são fornecidos quando uma implantação inicial e uma de reversão são iniciadas. A diferença entre as duas está no campo `reason`. Para obter mais informações sobre parâmetros do EventBridge, consulte metadados de eventos de serviços da AWS no Guia do usuário do Amazon EventBridge.

Veja a seguir um exemplo de saída para o início de uma implantação inicial.

```
{
   "version": "0",
   "id": "ddca6449-b258-46c0-8653-e0e3a6EXAMPLE",
   "detail-type": "ECS Deployment State Change",
   "source": "aws.ecs",
   "account": "111122223333",
   "time": "2020-05-23T12:31:14Z",
   "region": "us-west-2",
   "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
   ],
   "detail": {
        "eventType": "INFO",
        "eventName": "SERVICE_DEPLOYMENT_IN_PROGRESS",
        "deploymentId": "ecs-svc/123",
        "updatedAt": "2020-05-23T11:11:11Z",
        "reason": "ECS deployment deploymentId in progress."
   }
```

```
}
```

Veja a seguir um exemplo de saída para o início de uma implantação de reversão. O campo `reason` fornece o ID da implantação para a qual o serviço está revertendo.

```
{
   "version": "0",
   "id": "ddca6449-b258-46c0-8653-e0e3aEXAMPLE",
   "detail-type": "ECS Deployment State Change",
   "source": "aws.ecs",
   "account": "111122223333",
   "time": "2020-05-23T12:31:14Z",
   "region": "us-west-2",
   "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
   ],
   "detail": {
        "eventType": "INFO",
        "eventName": "SERVICE_DEPLOYMENT_IN_PROGRESS",
        "deploymentId": "ecs-svc/123",
        "updatedAt": "2020-05-23T11:11:11Z",
        "reason": "ECS deployment circuit breaker: rolling back to
 deploymentId deploymentID."
   }
}
```

Example evento de implantação de serviço concluído

Os eventos de implantação de serviço no estado concluído são entregues no formato a seguir. Para obter mais informações, consulte Implantação de serviços do Amazon ECS por meio da substituição de tarefas.

```
{
   "version": "0",
   "id": "ddca6449-b258-46c0-8653-e0e3aEXAMPLE",
   "detail-type": "ECS Deployment State Change",
   "source": "aws.ecs",
   "account": "111122223333",
   "time": "2020-05-23T12:31:14Z",
   "region": "us-west-2",
   "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
```

```
    ],
    "detail": {
         "eventType": "INFO",
         "eventName": "SERVICE_DEPLOYMENT_COMPLETED",
         "deploymentId": "ecs-svc/123",
         "updatedAt": "2020-05-23T11:11:11Z",
         "reason": "ECS deployment deploymentID completed."
    }
}
```

Example evento de implantação de serviço com falha

Os eventos de implantação de serviço no estado com falha são entregues no formato a seguir. Um evento de implantação de serviço em estado com falha só será enviado para serviços com a lógica de disjuntor de implantação ativada. Para obter mais informações, consulte Implantação de serviços do Amazon ECS por meio da substituição de tarefas.

```
{
    "version": "0",
    "id": "ddca6449-b258-46c0-8653-e0e3aEXAMPLE",
    "detail-type": "ECS Deployment State Change",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2020-05-23T12:31:14Z",
    "region": "us-west-2",
    "resources": [
         "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
         "eventType": "ERROR",
         "eventName": "SERVICE_DEPLOYMENT_FAILED",
         "deploymentId": "ecs-svc/123",
         "updatedAt": "2020-05-23T11:11:11Z",
         "reason": "ECS deployment circuit breaker: task failed to start."
    }
}
```

Processo de eventos do Amazon ECS

O Amazon ECS envia eventos pelo menos uma vez. Isso significa que você pode receber várias cópias de um determinado evento. Além disso, os eventos não podem ser distribuídos para os ouvintes de evento na ordem em que os eventos ocorreram.

Para que os eventos sejam ordenados adequadamente, a seção `detail` de cada evento contém uma propriedade `version`. Sempre que um recurso muda de estado, essa `version` é incrementada. Eventos duplicados têm a mesma `version` no objeto `detail`. Caso esteja replicando o estado da tarefa e da instância de contêiner do Amazon ECS com o Eventbridge, você poderá comparar a versão de um recurso relatado pelas APIs do Amazon ECS com a `version` relatada no Eventbridge para o recurso para verificar se a versão no fluxo de eventos é atual. Os eventos com um número de propriedade da versão mais alto devem ser tratados como ocorridos depois de eventos com números da versão mais baixos.

Exemplo: processar eventos em uma função do AWS Lambda

O exemplo a seguir mostra uma função do Lambda escrita em Python 3.9 que captura os eventos de alteração na tarefa e no estado da instância de contêiner e os salva em uma das duas tabelas do Amazon DynamoDB:

- ECSCtrInstanceState: armazena o estado mais recente de uma instância de contêiner. A ID da tabela é o valor `containerInstanceArn` da instância de contêiner.
- ECSTaskState: armazena o estado mais recente de uma tarefa. A ID da tabela é o valor `taskArn` da tarefa.

```
import json
import boto3

def lambda_handler(event, context):
    id_name = ""
    new_record = {}

    # For debugging so you can see raw event format.
    print('Here is the event:')
    print((json.dumps(event)))

    if event["source"] != "aws.ecs":
       raise ValueError("Function only supports input from events with a source type
 of: aws.ecs")

    # Switch on task/container events.
    table_name = ""
    if event["detail-type"] == "ECS Task State Change":
        table_name = "ECSTaskState"
        id_name = "taskArn"
```

```
        event_id = event["detail"]["taskArn"]
    elif event["detail-type"] == "ECS Container Instance State Change":
        table_name = "ECSCtrInstanceState"
        id_name =  "containerInstanceArn"
        event_id = event["detail"]["containerInstanceArn"]
    else:
        raise ValueError("detail-type for event is not a supported type. Exiting
without saving event.")

    new_record["cw_version"] = event["version"]
    new_record.update(event["detail"])

    # "status" is a reserved word in DDB, but it appears in containerPort
    # state change messages.
    if "status" in event:
        new_record["current_status"] = event["status"]
        new_record.pop("status")


    # Look first to see if you have received a newer version of an event ID.
    # If the version is OLDER than what you have on file, do not process it.
    # Otherwise, update the associated record with this latest information.
    print("Looking for recent event with same ID...")
    dynamodb = boto3.resource("dynamodb", region_name="us-east-1")
    table = dynamodb.Table(table_name)
    saved_event = table.get_item(
        Key={
            id_name : event_id
        }
    )
    if "Item" in saved_event:
        # Compare events and reconcile.
        print(("EXISTING EVENT DETECTED: Id " + event_id + " - reconciling"))
        if saved_event["Item"]["version"] < event["detail"]["version"]:
            print("Received event is a more recent version than the stored event -
updating")
            table.put_item(
                Item=new_record
            )
        else:
            print("Received event is an older version than the stored event -
ignoring")
    else:
        print(("Saving new event - ID " + event_id))
```

```
        table.put_item(
            Item=new_record
        )
```

O exemplo a seguir do Fargate mostra uma função do Lambda gravada em Python 3.9 que captura os eventos de alteração de estado da tarefa e os salva na seguinte tabela do Amazon DynamoDB:

```
import json
import boto3

def lambda_handler(event, context):
    id_name = ""
    new_record = {}

    # For debugging so you can see raw event format.
    print('Here is the event:')
    print((json.dumps(event)))

    if event["source"] != "aws.ecs":
       raise ValueError("Function only supports input from events with a source type
 of: aws.ecs")

    # Switch on task/container events.
    table_name = ""
    if event["detail-type"] == "ECS Task State Change":
        table_name = "ECSTaskState"
        id_name = "taskArn"
        event_id = event["detail"]["taskArn"]
    else:
        raise ValueError("detail-type for event is not a supported type. Exiting
 without saving event.")

    new_record["cw_version"] = event["version"]
    new_record.update(event["detail"])

    # "status" is a reserved word in DDB, but it appears in containerPort
    # state change messages.
    if "status" in event:
        new_record["current_status"] = event["status"]
        new_record.pop("status")
```

```
    # Look first to see if you have received a newer version of an event ID.
    # If the version is OLDER than what you have on file, do not process it.
    # Otherwise, update the associated record with this latest information.
    print("Looking for recent event with same ID...")
    dynamodb = boto3.resource("dynamodb", region_name="us-east-1")
    table = dynamodb.Table(table_name)
    saved_event = table.get_item(
        Key={
            id_name : event_id
        }
    )
    if "Item" in saved_event:
        # Compare events and reconcile.
        print(("EXISTING EVENT DETECTED: Id " + event_id + " - reconciling"))
        if saved_event["Item"]["version"] < event["detail"]["version"]:
            print("Received event is a more recent version than the stored event -
updating")
            table.put_item(
                Item=new_record
            )
        else:
            print("Received event is an older version than the stored event -
ignoring")
    else:
        print(("Saving new event - ID " + event_id))

        table.put_item(
            Item=new_record
        )
```

Monitorar contêineres do Amazon ECS utilizando o Container Insights com observabilidade aprimorada

O Container Insights coleta, agrega e resume métricas e logs de suas aplicações conteinerizadas e microsserviços. O Container Insights com observabilidade aprimorada fornece todas as métricas do Container Insights, além de métricas adicionais de tarefas e contêineres.

O Container Insights detecta todos os contêineres de um cluster e coletará dados de performance em todas as camadas da pilha de performance. Os dados operacionais são coletados como eventos de log de desempenho. Essas são entradas que usam um esquema JSON estruturado que permite que dados de alta cardinalidade sejam ingeridos e armazenados em escala. Com base nesses

dados, o CloudWatch cria métricas agregadas de alto nível no nível do cluster, do serviço e da tarefa como métricas do CloudWatch. As métricas incluem a utilização de recursos, como CPU, memória, disco e rede. As métricas estão disponíveis em painéis automáticos do CloudWatch.

As métricas refletem apenas os recursos com tarefas em execução durante o intervalo de tempo especificado. Por exemplo, se você tiver um cluster contendo um serviço, mas esse serviço não tiver tarefas no estado `RUNNING`, não haverá métricas enviadas para o CloudWatch. Se você tiver dois serviços e um deles tiver tarefas em execução e o outro não, somente as métricas do serviço com tarefas em execução serão enviadas.

Em 2 de dezembro de 2024, a AWS lançou o Container Insights com observabilidade aprimorada para o Amazon ECS. Essa versão é compatível com a observabilidade aprimorada dos clusters do Amazon ECS usando os tipos de inicialização do Amazon EC2 e do Fargate. Depois de configurar o Container Insights com observabilidade aprimorada no Amazon ECS, o Container Insights coleta automaticamente a telemetria detalhada da infraestrutura do nível do cluster até o nível do contêiner em seu ambiente e exibe esses dados cruciais de performance em painéis selecionados, eliminando o trabalho pesado na configuração da observabilidade. Para obter informações sobre como configurar o Container Insights com observabilidade aprimorada, consulte Container Insights com observabilidade aprimorada.

Recomendamos que você use o Container Insights com observabilidade aprimorada em vez do Container Insights, pois ele fornece visibilidade detalhada no ambiente de contêiner, reduzindo o tempo médio de resolução. Para obter mais informações, consulte Métricas do Container Insights do Amazon ECS com observabilidade aprimorada, no Guia do usuário do Amazon CloudWatch.

Os eventos que você pode visualizar são aqueles que o Amazon ECS envia ao Amazon EventBridge. Para obter mais informações, consulte Eventos do Amazon ECS.

> **Important**
>
> As métricas coletadas pelo CloudWatch Container Insights são cobradas como métricas personalizadas. Para obter mais informações sobre os preços do CloudWatch, consulte Preço do CloudWatch.

Como determinar a integridade das tarefas do Amazon ECS usando verificações de integridade de contêineres

Ao criar uma definição de tarefa, você pode configurar uma verificação de integridade para os contêineres. As verificações de integridade são comandos executados localmente em um contêiner e validam a integridade e a disponibilidade da aplicação.

O agente de contêiner do Amazon ECS apenas monitora e comunica as verificações de integridade especificadas na definição da tarefa. O Amazon ECS não monitora as verificações de integridade do Docker incorporadas em uma imagem do contêiner, mas não especificadas na definição do contêiner. Os parâmetros de verificação de integridade especificados em uma definição de contêiner substituem as verificações de integridade do Docker existentes na imagem do contêiner.

Quando uma verificação de integridade é definida em uma definição de tarefa, o contêiner executa o processo de verificação de integridade dentro do contêiner e avalia o código de saída para determinar a integridade da aplicação.

A verificação de integridade consiste nos seguintes parâmetros:

- Comando: o comando executado pelo contêiner para determinar a integridade. A matriz de strings pode começar com `CMD` para executar os argumentos de comando diretamente ou `CMD-SHELL` para executar o comando com o shell padrão do contêiner. Use `CMD-SHELL` quando precisar de recursos de shell, como pipes, redirecionamentos, encadeamento de comandos ou expansão de variáveis de ambiente. Por exemplo, `CMD-SHELL` permite que você use comandos como `curl -f http://localhost/ || exit 1`, no qual o shell interpreta o operador `||`. Use `CMD` para comandos simples que não exigem interpretação de shell.
- Intervalo: o período (em segundos) entre cada verificação de integridade.
- Tempo limite: o período (em segundos) de espera para que uma verificação de integridade seja bem-sucedida antes de ser considerada uma falha.
- Repetições: o número de novas tentativas de uma verificação de integridade com falha até o contêiner ser considerado não íntegro.
- Período de início: o período de carência opcional para fornecer tempo aos contêineres para inicializar antes que as verificações de integridade com falha sejam contabilizadas no número máximo de novas tentativas.

Se uma verificação de integridade for bem-sucedida dentro do `startPeriod`, então o contêiner é considerado saudável e quaisquer falhas subsequentes contam para o número máximo de novas tentativas.

Para obter informações sobre como especificar uma verificação de integridade em uma definição de tarefa, consulte Verificação de integridade.

Os seguintes itens descrevem os possíveis valores do status de integridade de um contêiner:

- `HEALTHY`: a verificação de integridade do contêiner foi aprovada.
- `UNHEALTHY`: a verificação de integridade do contêiner falhou.
- `UNKNOWN`: a verificação de integridade do contêiner está sendo avaliada, não há verificação de integridade do contêiner definida ou o Amazon ECS não tem o status de integridade do contêiner definido.

Os comandos de verificação de integridade são executados no contêiner. Portanto, você deve incluir os comandos na imagem do contêiner.

A verificação de integridade se conecta à aplicação por meio da interface de loopback do contêiner em `localhost` ou `127.0.0.1`. Um código de saída `0` indica êxito, e um código de saída diferente de zero indica falha.

Considere o seguinte ao usar verificações de integridade do contêiner:

- As verificações de integridade do contêiner exigem a versão 1.17.0 ou posterior do agente de contêiner do Amazon ECS.
- As verificações de integridade do contêiner serão compatíveis com tarefas do Fargate se estiver usando a versão `1.1.0` da plataforma Linux ou posterior ou a versão `1.1.0` da plataforma Windows ou posterior.

Como o Amazon ECS determina a integridade da tarefa

Os contêineres essenciais e que têm o comando de verificação de integridade na definição da tarefa são os únicos considerados para determinar a integridade da tarefa.

As seguintes regras são avaliadas em ordem:

1. Se o status de um contêiner essencial for UNHEALTHY, o status da tarefa será UNHEALTHY.
2. Se o status de um contêiner essencial for UNKNOWN, o status da tarefa será UNKNOWN.
3. Se o status de todos os contêineres essenciais for HEALTHY, o status da tarefa será HEALTHY.

Considere o exemplo de integridade da tarefa a seguir com dois contêineres essenciais.

Integridade do contêiner 1	Integridade do contêiner 2	Integridade da tarefa
UNHEALTHY	UNKNOWN	UNHEALTHY
UNHEALTHY	HEALTHY	UNHEALTHY
HEALTHY	UNKNOWN	UNKNOWN
HEALTHY	HEALTHY	HEALTHY

Considere o exemplo de integridade da tarefa a seguir com três contêineres.

Integridade do contêiner 1	Integridade do contêiner 2	Integridade do contêiner 3	Integridade da tarefa
UNHEALTHY	UNKNOWN	UNKNOWN	UNHEALTHY
UNHEALTHY	UNKNOWN	HEALTHY	UNHEALTHY
UNHEALTHY	HEALTHY	HEALTHY	UNHEALTHY
HEALTHY	UNKNOWN	HEALTHY	UNKNOWN
HEALTHY	UNKNOWN	UNKNOWN	UNKNOWN
HEALTHY	HEALTHY	HEALTHY	HEALTHY

Como as verificações de integridade são afetadas pelas desconexões de agentes

Se o agente de contêiner do Amazon ECS for desconectado do serviço do Amazon ECS, isso não fará com que o contêiner mude para um status `UNHEALTHY`. Isso ocorre intencionalmente para garantir que os contêineres permaneçam em execução durante reinicializações do agente ou indisponibilidades temporárias. O status da verificação de integridade é a resposta para a "ultima comunicação recebida" do agente do Amazon ECS, portanto, se o contêiner foi considerado `HEALTHY` antes da desconexão, esse status permanecerá até que o agente se reconecte e ocorra outra verificação de integridade. Não há suposições sobre o status das verificações de integridade do contêiner.

Visualização da integridade do contêiner do Amazon ECS

É possível visualizar a integridade do contêiner no console e usar a API na resposta `DescribeTasks`. Para obter mais informações, consulte DescribeTasks na Referência de APIs do Amazon Elastic Container Service.

Se você usar o registro em log para o contêiner, por exemplo, o Amazon CloudWatch Logs, poderá configurar o comando de verificação de integridade para encaminhar a saída de integridade do contêiner para seus logs. Certifique-se de usar `2&1` para capturar as informações `stdout` e `stderr`. O exemplo a seguir usa `CMD-SHELL` porque precisa de recursos de shell, como os operadores pipe (`>>`) e a lógica OR (`||`):

```
"command": [
     "CMD-SHELL",
     "curl -f http://localhost/ >> /proc/1/fd/1 2>&1  || exit 1"
   ],
```

Monitoramento da integridade da instância de contêiner do Amazon ECS

O Amazon ECS fornece monitoramento de integridade de instâncias de contêiner. É possível determinar rapidamente se o Amazon ECS detectou problemas que possam impedir que suas instâncias de contêiner executem contêineres. O Amazon ECS realiza verificações automatizadas em todas as instâncias de contêiner em execução com a versão do agente `1.57.0` ou posterior para

identificar problemas. Para obter mais informações sobre como verificar a versão do agente de uma instância de contêiner, consulte Atualizar o agente de contêiner do Amazon ECS.

Você deve estar usando a AWS CLI versão `1.22.3` ou posterior ou a AWS CLI versão `2.3.6` ou posterior. Para obter informações sobre como atualizar a AWS CLI, consulte Instalação ou atualização da versão mais recente da AWS CLI no Guia do usuário da AWS Command Line Interface versão 2.

Para visualizar a integridade da instância do contêiner, execute `describe-container-instances` com a opção `CONTAINER_INSTANCE_HEALTH`.

Estes são os valores válidos para `overallStatus`:

- `OK`
- `IMPAIRED`
- `INSUFFICIENT_DATA`
- `INITIALIZING`

Veja a seguir um exemplo de como executar `describe-container-instances`.

```
aws ecs describe-container-instances \
     --cluster cluster_name \
     --container-instances 47279cd2cadb41cbaef2dcEXAMPLE \
     --include CONTAINER_INSTANCE_HEALTH
```

Veja a seguir um exemplo do objeto de status de integridade na saída.

```
"healthStatus": {
 "overallStatus": "OK",
 "details": [{
  "type": "CONTAINER_RUNTIME",
  "status": "OK",
  "lastUpdated": "2021-11-10T03:30:26+00:00",
  "lastStatusChange": "2021-11-10T03:26:41+00:00"
 }]
}
```

Problemas de integridade de instâncias de contêiner

Quando o `overallStatus` tiver qualquer status diferente de `OK`, tente o seguinte:

- Aguarde e execute `describe-container-instances`
- Visualize a integridade da sua instância de contêiner no console do EC2 ou usando a CLI.
- Analise as métricas do CloudWatch. Para obter mais informações, consulte Monitoramento do Amazon ECS usando o CloudWatch.
- Verifique em AWS Health Dashboard se há algum problema com o serviço.

Identifique oportunidades de otimização do Amazon ECS usando dados de rastreamento de aplicações

O Amazon ECS integra-se com a distribuição da AWS do OpenTelemetry para coletar dados de rastreamento da sua aplicação. O Amazon ECS usa um contêiner de arquivo associado da distribuição da AWS do OpenTelemetry para coletar e encaminhar dados de rastreamento para AWS X-Ray. Para obter mais informações, consulte Configuração da distribuição da AWS do OpenTelemetry Collector no Amazon ECS. Em seguida, você pode usar AWS X-Ray para identificar erros e exceções, analisar gargalos de desempenho e tempos de resposta.

Para que a distribuição da AWS do OpenTelemetry Collectorenvie dados de rastreamento para AWS X-Ray, sua aplicação deve ser configurada para criar os dados de rastreamento. Para obter mais informações, consulte Instrumentando sua aplicação para AWS X-Ray no Guia do desenvolvedor do AWS X-Ray.

Permissões obrigatórias do IAM para integração da distribuição da AWS do OpenTelemetry com o AWS X-Ray

A integração do Amazon ECS com o AWS Distro para OpenTelemetry requer que você crie um perfil de tarefa e especifique o perfil em sua definição de tarefa. Recomendamos configurar o arquivo associado do AWS Distro para OpenTelemetry com a finalidade de encaminhar logs de contêiner para o CloudWatch Logs.

> **Important**
>
> Se você também coletar métricas de aplicações usando a integração do AWS Distro para OpenTelemetry, certifique-se de que o perfil do IAM da tarefa também contenha as permissões necessárias para essa integração. Para obter mais informações, consulte Correlação do desempenho da aplicação do Amazon ECS usando métricas de aplicações.

Crie a política apresentada a seguir e anexe-a ao perfil de execução de tarefas.

Para usar o editor de políticas JSON para criar uma política

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Políticas.

   Se essa for a primeira vez que você escolhe Políticas, a página Bem-vindo às políticas gerenciadas será exibida. Escolha Começar.
3. Na parte superior da página, escolha Criar política.
4. Na seção Editor de políticas, escolha a opção JSON.
5. Insira o seguinte documento de política JSON:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents",
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:DescribeLogGroups",
                "logs:PutRetentionPolicy",
                "xray:PutTraceSegments",
                "xray:PutTelemetryRecords",
                "xray:GetSamplingRules",
                "xray:GetSamplingTargets",
                "xray:GetSamplingStatisticSummaries",
                "ssm:GetParameters"
            ],
            "Resource": "*"
        }
    ]
}
```

6. Escolha Próximo.

> **Note**
>
> É possível alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Próximo no editor Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte Restruturação de política no Guia do usuário do IAM.

7. Na página Revisar e criar, insira um Nome de política e uma Descrição (opcional) para a política que você está criando. Revise Permissões definidas nessa política para ver as permissões que são concedidas pela política.
8. Escolha Criar política para salvar sua nova política.

Especificação do arquivo associado da distribuição da AWS do OpenTelemetry para integração do AWS X-Ray na sua definição de tarefa

O console do Amazon ECS simplifica a criação do contêiner auxiliar AWS Distro para OpenTelemetry usando a opção Usar coleta de rastreamento. Para obter mais informações, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

Se você não estiver usando o console do Amazon ECS, poderá adicionar o contêiner de arquivo associado da distribuição da AWS do OpenTelemetry à sua definição de tarefa. O trecho de definição de tarefa a seguir mostra a definição do contêiner para adicionar o arquivo associado da distribuição da AWS do OpenTelemetry para integração do AWS X-Ray.

```
{
 "family": "otel-using-xray",
 "taskRoleArn": "arn:aws:iam::111122223333:role/AmazonECS_OpenTelemetryXrayRole",
 "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
 "containerDefinitions": [{
   "name": "aws-otel-emitter",
   "image": "application-image",
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "true",
     "awslogs-group": "/ecs/aws-otel-emitter",
     "awslogs-region": "us-east-1",
     "awslogs-stream-prefix": "ecs"
```

```
    }
   },
   "dependsOn": [{
    "containerName": "aws-otel-collector",
    "condition": "START"
   }]
  },
  {
   "name": "aws-otel-collector",
   "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.30.0",
   "essential": true,
   "command": [
    "--config=/etc/ecs/otel-instance-metrics-config.yaml"
   ],
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "True",
     "awslogs-group": "/ecs/ecs-aws-otel-sidecar-collector",
     "awslogs-region": "us-east-1",
     "awslogs-stream-prefix": "ecs"
    }
   }
  }
 ],
 "networkMode": "awsvpc",
 "requiresCompatibilities": [
  "FARGATE"
 ],
 "cpu": "1024",
 "memory": "3072"
}
```

Correlação do desempenho da aplicação do Amazon ECS usando métricas de aplicações

O Amazon ECS no Fargate oferece suporte à coleta de métricas de suas aplicações em execução no Fargate e a sua exportação para o Amazon CloudWatch ou Amazon Managed Service for Prometheus.

Você pode usar os metadados coletados para correlacionar os dados de desempenho da aplicação com os dados da infraestrutura subjacente, reduzindo o tempo médio de resolução do problema.

O Amazon ECS usa um contêiner de arquivo associado da distribuição da AWS do OpenTelemetry para coletar e encaminhar as métricas da sua aplicação até o destino. A experiência do console do Amazon ECS simplifica o processo de adicionar essa integração ao criar as definições de tarefa.

Tópicos

- Exportação de métricas de aplicações para o Amazon CloudWatch
- Exportação de métricas de aplicações para o Amazon Managed Service for Prometheus

Exportação de métricas de aplicações para o Amazon CloudWatch

O Amazon ECS no Fargate oferece suporte à exportação de métricas de aplicações personalizadas para o Amazon CloudWatch como métricas personalizadas. Isso é feito adicionando o contêiner de arquivo associado da distribuição da AWS do OpenTelemetry à sua definição de tarefa. O console do Amazon ECS simplifica esse processo adicionando a opção Usar coleta de métricas ao criar uma definição de tarefa. Para obter mais informações, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

As métricas da aplicação são exportadas para o CloudWatch Logs com o nome do grupo de logs `/aws/ecs/application/metrics` e as métricas podem ser exibidas no namespace `ECS/AWSOTel/Application`. Sua aplicação deve ser instrumentada com o SDK OpenTelemetry. Para obter mais informações, consulte Introdução à distribuição da AWS do OpenTelemetry, na documentação da distribuição da AWS do OpenTelemetry.

Considerações

As informações a seguir devem ser consideradas quando for usada a integração do Amazon ECS na Fargate com a distribuição da AWS do OpenTelemetry para o envio de métricas de aplicações para o Amazon CloudWatch.

- Essa integração só envia suas métricas personalizadas de aplicação para o CloudWatch. Se você quiser métricas por tarefa, é possível ativar o Container Insights na configuração de cluster do Amazon ECS. Para obter mais informações, consulte Monitorar contêineres do Amazon ECS utilizando o Container Insights com observabilidade aprimorada.

- A integração do AWS Distro para OpenTelemetry é compatível com workloads do Amazon ECS hospedadas no Fargate e workloads do Amazon ECS em instâncias do Amazon EC2. No momento, não há compatibilidade com instâncias externas.
- O CloudWatch é compatível com, no máximo, 30 dimensões por métrica. Por padrão, o Amazon ECS assume como padrão a inclusão das dimensões `TaskARN`, `ClusterARN`, `LaunchType`, `TaskDefinitionFamily` e `TaskDefinitionRevision` para as métricas. As demais 25 dimensões podem ser definidas pela aplicação. Se mais de 30 dimensões forem configuradas, o CloudWatch não poderá exibi-las. Quando isso ocorrer, as métricas da aplicação serão exibidas no namespace de métricas do CloudWatch `ECS/AWSOTel/Application`, mas sem nenhuma dimensão. É possível instrumentar sua aplicação para incluir dimensões adicionais. Para obter mais informações, consulte Uso de métricas do CloudWatch com a distribuição da AWS do OpenTelemetry, na documentação da distribuição da AWS do OpenTelemetry.

Permissões obrigatórias do IAM para integração da distribuição da AWS do OpenTelemetry com o Amazon CloudWatch

A integração do Amazon ECS com a distribuição da AWS do OpenTelemetry requer que você crie uma função do IAM de tarefa e especifique a função na definição de tarefa. Recomendamos que o arquivo associado da distribuição da AWS do OpenTelemetry também possa ser configurado para encaminhar logs de contêiner para o CloudWatch Logs, o que exige que uma função do IAM de execução de tarefa seja criada e especificada também na definição da tarefa. O console do Amazon ECS cuida do perfil do IAM de execução de tarefas em seu nome, mas o perfil do IAM da tarefa deve ser criado manualmente e adicionado à definição de tarefa. Para obter mais informações sobre a função do IAM de execução de tarefas, consulte Função do IAM de execução de tarefas do Amazon ECS.

> **Important**
>
> Se você também estiver coletando dados de rastreamento de aplicações usando a distribuição da AWS para integração do OpenTelemetry, certifique-se de que sua função do IAM de tarefa também contenha as permissões necessárias para essa integração. Para obter mais informações, consulte Identifique oportunidades de otimização do Amazon ECS usando dados de rastreamento de aplicações.
>
> Se a aplicação exigir permissões adicionais, você deverá adicioná-las a essa política. Cada definição de tarefa só pode especificar uma função do IAM de tarefa. Por exemplo, se você

estiver usando um arquivo de configuração personalizado armazenado no Systems Manager, você deve adicionar a permissão `ssm:GetParameters` a essa política do IAM.

Para criar um perfil de serviço do Elastic Container Service (console do IAM)

1. Faça login no AWS Management Console e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Perfis e, em seguida, Criar perfil.
3. Em Tipo de Entidade Confiável, escolha AWS service (Serviço da AWS).
4. Em Serviço ou caso de uso, escolha Elastic Container Service e selecione o caso de uso da Tarefa do Elastic Container Service.
5. Escolha Próximo.
6. Na seção Adicionar permissões, procure AWSDistroOpenTelemetryPolicyForXray e selecione a política.
7. (Opcional) Defina um limite de permissões. Esse é um atributo avançado que está disponível para perfis de serviço, mas não para perfis vinculados ao serviço.

   a. Abra a seção Definir limite de permissões e escolha Usar um limite de permissões para controlar o número máximo de permissões do perfil.

      O IAM inclui uma lista das políticas gerenciadas pela AWS e pelo cliente em sua conta.

   b. Selecione a política a ser usada para o limite de permissões.
8. Escolha Próximo.
9. Insira um nome de perfil ou um sufixo de nome de perfil para ajudar a identificar a finalidade do perfil.

   Important

   Quando nomear um perfil, observe o seguinte:

   - Os nomes do perfil devem ser exclusivos em sua Conta da AWS e não podem ser diferenciados caso a caso.

     Por exemplo, não crie dois perfis denominados **`PRODROLE`** e **`prodrole`**. Quando usado em uma política ou como parte de um ARN, o nome de perfil diferencia maiúsculas de minúsculas. No entanto, quando exibido para os clientes no console,

como durante o processo de login, o nome de perfil diferencia maiúsculas de minúsculas.

- Não é possível editar o nome do perfil depois de criá-lo porque outras entidades podem referenciar o perfil.

10. (Opcional) Em Descrição, insira uma descrição para o perfil.
11. (Opcional) Para editar os casos de uso e as permissões do perfil, escolha Editar nas seções Etapa 1: selecionar entidades confiáveis ou Etapa 2: adicionar permissões.
12. (Opcional) Para ajudar a identificar, organizar ou pesquisar o perfil, adicione tags como pares de chave-valor. Para obter mais informações sobre o uso de tags no IAM, consulte Tags para recursos do AWS Identity and Access Management no Guia do usuário do IAM.
13. Reveja a função e escolha Criar função.

Especificação do arquivo associado da distribuição da AWS do OpenTelemetry na sua definição de tarefa

O console do Amazon ECS simplifica a experiência de criação do contêiner auxiliar do AWS Distro para OpenTelemetry utilizando a opção Usar coleta de métricas. Para obter mais informações, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

Se você não estiver usando o console do Amazon ECS, poderá adicionar o contêiner de arquivo associado da distribuição da AWS do OpenTelemetry à sua definição de tarefa manualmente. O exemplo de definição de tarefa a seguir mostra a definição do contêiner para adicionar o arquivo associado da distribuição da AWS do OpenTelemetry para integração do Amazon CloudWatch.

```
{
 "family": "otel-using-cloudwatch",
 "taskRoleArn": "arn:aws:iam::111122223333:role/AmazonECS_OpenTelemetryCloudWatchRole",
 "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
 "containerDefinitions": [
    {
   "name": "aws-otel-emitter",
   "image": "application-image",
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "true",
     "awslogs-group": "/ecs/aws-otel-emitter",
```

```
      "awslogs-region": "us-east-1",
      "awslogs-stream-prefix": "ecs"
     }
    },
    "dependsOn": [{
     "containerName": "aws-otel-collector",
     "condition": "START"
    }]
   },
   {
    "name": "aws-otel-collector",
    "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.30.0",
    "essential": true,
    "command": [
     "--config=/etc/ecs/ecs-cloudwatch.yaml"
    ],
    "logConfiguration": {
     "logDriver": "awslogs",
     "options": {
      "awslogs-create-group": "True",
      "awslogs-group": "/ecs/ecs-aws-otel-sidecar-collector",
      "awslogs-region": "us-east-1",
      "awslogs-stream-prefix": "ecs"
     }
    }
   }
  ],
  "networkMode": "awsvpc",
  "requiresCompatibilities": [
   "FARGATE"
  ],
  "cpu": "1024",
  "memory": "3072"
}
```

Exportação de métricas de aplicações para o Amazon Managed Service for Prometheus

O Amazon ECS oferece suporte à exportação de métricas de CPU, memória, rede e armazenamento no nível da tarefa e suas métricas de aplicação personalizadas para o Amazon Managed Service for Prometheus. Isso é feito adicionando o contêiner de arquivo associado da distribuição da AWS do OpenTelemetry à sua definição de tarefa. O console do Amazon ECS simplifica esse processo

adicionando a opção Usar coleta de métricas ao criar uma definição de tarefa. Para obter mais informações, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

As métricas são exportadas para o Amazon Managed Service para Prometheus e podem ser exibidas usando o painel do Amazon Managed Grafana. Sua aplicação deve ser instrumentada com bibliotecas do Prometheus ou com o SDK OpenTelemetry. Para obter mais informações sobre a instrumentação da sua aplicação com o SDK OpenTelemetry, consulte Introdução à distribuição da AWS do OpenTelemetry, na documentação da distribuição da AWS do OpenTelemetry.

Ao usar as bibliotecas do Prometheus, sua aplicação deve expor um endpoint `/metrics` que é usado para extrair os dados das métricas. Para obter mais informações sobre como instrumentar a aplicação com bibliotecas do Prometheus, consulte Bibliotecas de clientes do Prometheus na documentação do Prometheus.

Considerações

Recomendamos levar as seguintes informações em consideração ao usar a integração do Amazon ECS no Fargate com o AWS Distro para OpenTelemetry para o envio de métricas de aplicações para o Amazon Managed Service para Prometheus.

- A integração do AWS Distro para OpenTelemetry é compatível com workloads do Amazon ECS hospedadas no Fargate e workloads do Amazon ECS em instâncias do Amazon EC2. No momento, não há compatibilidade com instâncias externas.
- Por padrão, a distribuição da AWS do OpenTelemetry inclui todas as dimensões de nível de tarefa disponíveis para as métricas de aplicações ao exportar para o Amazon Managed Service for Prometheus. Você também pode instrumentar sua aplicação para incluir dimensões adicionais. Para obter mais informações, consulte Primeiros passos com o Prometheus Remote Write Exporter para Amazon Managed Service for Prometheus na documentação da distribuição da AWS do OpenTelemetry.

Permissões obrigatórias do IAM para integração da distribuição da AWS do OpenTelemetry com o Amazon Managed Service for Prometheus

A integração do Amazon ECS com o Amazon Managed Service for Prometheus usando o arquivo associado da distribuição da AWS do OpenTelemetry requer que você crie uma função do IAM de tarefa e especifique a função na definição de tarefa. Essa função do IAM de tarefa deve ser criada manualmente usando as etapas abaixo antes de registrar sua definição de tarefa.

Recomendamos que o arquivo associado da distribuição da AWS do OpenTelemetry também possa ser configurado para encaminhar logs de contêiner para o CloudWatch Logs, o que exige que uma função do IAM de execução de tarefa seja criada e especificada também na definição da tarefa. O console do Amazon ECS cuida do perfil do IAM de execução da tarefa em seu nome, mas o perfil do IAM da tarefa deve ser criado manualmente. Para obter mais informações sobre a criação de uma função do IAM de execução de tarefa, consulte Função do IAM de execução de tarefas do Amazon ECS.

> **Important**
>
> Se você também estiver coletando dados de rastreamento de aplicações usando a distribuição da AWS para integração do OpenTelemetry, certifique-se de que sua função do IAM de tarefa também contenha as permissões necessárias para essa integração. Para obter mais informações, consulte Identifique oportunidades de otimização do Amazon ECS usando dados de rastreamento de aplicações.

Para criar um perfil de serviço do Elastic Container Service (console do IAM)

1. Faça login no AWS Management Console e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Perfis e, em seguida, Criar perfil.
3. Em Tipo de Entidade Confiável, escolha AWS service (Serviço da AWS).
4. Em Serviço ou caso de uso, escolha Elastic Container Service e selecione o caso de uso da Tarefa do Elastic Container Service.
5. Escolha Próximo.
6. Na seção Adicionar permissões, procure AmazonPrometheusRemoteWriteAccess e selecione a política.
7. (Opcional) Defina um limite de permissões. Esse é um atributo avançado que está disponível para perfis de serviço, mas não para perfis vinculados ao serviço.

    a. Abra a seção Definir limite de permissões e escolha Usar um limite de permissões para controlar o número máximo de permissões do perfil.

    O IAM inclui uma lista das políticas gerenciadas pela AWS e pelo cliente em sua conta.

    b. Selecione a política a ser usada para o limite de permissões.

8. Escolha Próximo.
9. Insira um nome de perfil ou um sufixo de nome de perfil para ajudar a identificar a finalidade do perfil.

> **Important**
>
> Quando nomear um perfil, observe o seguinte:
>
> - Os nomes do perfil devem ser exclusivos em sua Conta da AWS e não podem ser diferenciados caso a caso.
>
> Por exemplo, não crie dois perfis denominados **`PRODROLE`** e **`prodrole`**. Quando usado em uma política ou como parte de um ARN, o nome de perfil diferencia maiúsculas de minúsculas. No entanto, quando exibido para os clientes no console, como durante o processo de login, o nome de perfil diferencia maiúsculas de minúsculas.
> - Não é possível editar o nome do perfil depois de criá-lo porque outras entidades podem referenciar o perfil.

10. (Opcional) Em Descrição, insira uma descrição para o perfil.
11. (Opcional) Para editar os casos de uso e as permissões do perfil, escolha Editar nas seções Etapa 1: selecionar entidades confiáveis ou Etapa 2: adicionar permissões.
12. (Opcional) Para ajudar a identificar, organizar ou pesquisar o perfil, adicione tags como pares de chave-valor. Para obter mais informações sobre o uso de tags no IAM, consulte Tags para recursos do AWS Identity and Access Management no Guia do usuário do IAM.
13. Reveja a função e escolha Criar função.

Especificação do arquivo associado da distribuição da AWS do OpenTelemetry na sua definição de tarefa

O console do Amazon ECS simplifica a experiência de criação do contêiner auxiliar do AWS Distro para OpenTelemetry utilizando a opção Usar coleta de métricas. Para obter mais informações, consulte Criar uma definição de tarefa do Amazon ECS usando o console.

Se você não estiver usando o console do Amazon ECS, poderá adicionar o contêiner de arquivo associado da distribuição da AWS do OpenTelemetry à sua definição de tarefa manualmente. O exemplo de definição de tarefa a seguir mostra a definição de contêiner para adicionar o arquivo

associado da distribuição da AWS do OpenTelemetry para integração com o Amazon Managed Service for Prometheus.

```
{
 "family": "otel-using-cloudwatch",
 "taskRoleArn": "arn:aws:iam::111122223333:role/AmazonECS_OpenTelemetryCloudWatchRole",
 "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
 "containerDefinitions": [{
   "name": "aws-otel-emitter",
   "image": "application-image",
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "true",
     "awslogs-group": "/ecs/aws-otel-emitter",
     "awslogs-region": "aws-region",
     "awslogs-stream-prefix": "ecs"
    }
   },
   "dependsOn": [{
    "containerName": "aws-otel-collector",
    "condition": "START"
   }]
  },
  {
   "name": "aws-otel-collector",
   "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.30.0",
   "essential": true,
   "command": [
    "--config=/etc/ecs/ecs-amp.yaml"
   ],
   "environment": [{
    "name": "AWS_PROMETHEUS_ENDPOINT",
    "value": "https://aps-workspaces.aws-region.amazonaws.com/workspaces/
ws-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/api/v1/remote_write"
   }],
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "True",
     "awslogs-group": "/ecs/ecs-aws-otel-sidecar-collector",
     "awslogs-region": "aws-region",
     "awslogs-stream-prefix": "ecs"
```

```
    }
   }
  }
 ],
 "networkMode": "awsvpc",
 "requiresCompatibilities": [
  "FARGATE"
 ],
 "cpu": "1024",
 "memory": "3072"
}
```

Registro em log das chamadas de API do Amazon ECS usando o AWS CloudTrail

O Amazon Elastic Container Service é integrado ao AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário, um perfil ou um AWS service (Serviço da AWS). O CloudTrail captura todas as chamadas de API para o Amazon ECS como eventos. As chamadas capturadas incluem chamadas do console do Amazon ECS e chamadas de código para as operações da API do Amazon ECS. Ao fazer uso das informações coletadas pelo CloudTrail, é possível determinar a solicitação feita ao Amazon ECS, o endereço IP que a realizou, quando ela foi feita e detalhes adicionais.

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar o seguinte:

- Se a solicitação foi feita com credenciais de usuário raiz ou credenciais de usuário.
- Se a solicitação foi feita em nome de um usuário do Centro de Identidade do IAM.
- Se a solicitação foi feita com credenciais de segurança temporárias de um perfil ou de um usuário federado.
- Se a solicitação foi feita por outro AWS service (Serviço da AWS).

Quando a Conta da AWS é criada, o CloudTrail torna-se ativo nela e você tem acesso automático ao Histórico de eventos do CloudTrail. O Histórico de eventos do CloudTrail fornece um registro visualizável, pesquisável, baixável e imutável dos últimos 90 dias de eventos de gerenciamento gravados em uma Região da AWS. Para obter mais informações, consulte Trabalhar com histórico

de eventos do CloudTrail no Guia do usuário do AWS CloudTrail. Não há cobranças do CloudTrail pela visualização do Histórico de eventos.

Para obter um registro contínuo de eventos em sua Conta da AWS nos últimos 90 dias, crie uma trilha ou um armazenamento de dados de eventos do CloudTrail Lake.

Trilhas do CloudTrail

Uma trilha permite que o CloudTrail entregue arquivos de log a um bucket do Amazon S3. As trilhas criadas usando o AWS Management Console são de várias regiões. Só é possível criar uma trilha de região única ou de várias regiões usando a AWS CLI. Criar uma trilha de várias regiões é uma prática recomendada, pois você captura atividades em todas as Regiões da AWS da conta. Ao criar uma trilha de região única, é possível visualizar somente os eventos registrados na Região da AWS da trilha. Para obter mais informações sobre trilhas, consulte Criar uma trilha para a Conta da AWS e Criar uma trilha para uma organização no Guia do usuário do AWS CloudTrail.

Uma cópia dos eventos de gerenciamento em andamento pode ser entregue no bucket do Amazon S3 sem nenhum custo via CloudTrail com a criação de uma trilha; no entanto, há cobranças de armazenamento do Amazon S3. Para obter mais informações sobre os preços do CloudTrail, consulte Preços do AWS CloudTrail. Para receber informações sobre a definição de preços do Amazon S3, consulte Definição de preços do Amazon S3.

Armazenamentos de dados de eventos do CloudTrail Lake

O CloudTrail Lake permite executar consultas baseadas em SQL nos eventos. O CloudTrail Lake converte eventos existentes em formato JSON baseado em linhas para o formato Apache ORC. O ORC é um formato colunar de armazenamento otimizado para recuperação rápida de dados. Os eventos são agregados em armazenamentos de dados de eventos, que são coleções imutáveis de eventos baseados nos critérios selecionados com a aplicação de seletores de eventos avançados. Os seletores que aplicados a um armazenamento de dados de eventos controlam quais eventos persistem e estão disponíveis para consulta. Para obter mais informações sobre o CloudTrail Lake, consulte Trabalhar com o AWS CloudTrail Lake, no Guia do usuário do AWS CloudTrail.

Os armazenamentos de dados de eventos e consultas do CloudTrail Lake incorrem em custos. Ao criar um armazenamento de dados de eventos, você escolhe a opção de preço que deseja usar para ele. A opção de preço determina o custo para a ingestão e para o armazenamento de eventos, e o período de retenção padrão e máximo para o armazenamento de dados de eventos. Para obter mais informações sobre os preços do CloudTrail, consulte Preços do AWS CloudTrail.

Eventos de gerenciamento do Amazon ECS no CloudTrail

Os eventos de gerenciamento fornecem informações sobre operações de gerenciamento executadas em recursos na Conta da AWS. Elas também são conhecidas como operações de ambiente de gerenciamento. Por padrão, o CloudTrail registra eventos de gerenciamento em logs.

O Amazon Elastic Container Service registra em log todas as operações do ambiente de gerenciamento do Amazon ECS como eventos de gerenciamento. Por exemplo, chamadas para as seções `CreateService`, `RunTask` e `DeleteCluster` geram entradas nos arquivos de log CloudTrail. Para obter uma lista das operações do ambiente de gerenciamento do Amazon Elastic Container Service que o Amazon ECS registra no CloudTrail, consulte a Amazon Elastic Container Service API Reference.

Exemplos de eventos do Amazon ECS

Um evento representa uma única solicitação de qualquer origem e inclui informações sobre a operação solicitada, a data e a hora da operação da API, os parâmetros de solicitação etc. Os arquivos de log do CloudTrail não são um rastreamento de pilha ordenada de chamadas de API públicas, portanto não são exibidos em uma ordem específica.

O exemplo a seguir mostra um evento do CloudTrail que demonstra a ação `CreateService`.

> **Note**
>
> Este exemplo foi formatado para uma melhor legibilidade. Em um arquivo de log do CloudTrail, todas as entradas e eventos são concatenados em uma única linha. Além disso, este exemplo foi limitado a uma única entrada do Amazon ECS. Em um arquivo de log real do CloudTrail, você vê entradas e eventos de vários serviços da AWS.

```
{
    "eventVersion": "1.04",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:account_name",
        "arn": "arn:aws:sts::123456789012:user/Mary_Major",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
```

```
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-06-20T18:32:25Z"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::123456789012:role/Admin",
                "accountId": "123456789012",
                "userName": "Mary_Major"
            }
        }
    },
    "eventTime": "2018-06-20T19:04:36Z",
    "eventSource": "ecs.amazonaws.com",
    "eventName": "CreateCluster",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.12",
    "userAgent": "console.amazonaws.com",
    "requestParameters": {
        "clusterName": "default"
    },
    "responseElements": {
        "cluster": {
            "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/default",
            "pendingTasksCount": 0,
            "registeredContainerInstancesCount": 0,
            "status": "ACTIVE",
            "runningTasksCount": 0,
            "statistics": [],
            "clusterName": "default",
            "activeServicesCount": 0
        }
    },
    "requestID": "cb8c167e-EXAMPLE",
    "eventID": "e3c6f4ce-EXAMPLE",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```


Para obter informações sobre o conteúdo dos registros do CloudTrail, consulte Conteúdo dos registros do CloudTrail no Guia do usuário do AWS CloudTrail.

Visualização do CloudWatch Logs Live Tail para serviços e tarefas do Amazon ECS

O Live Tail do CloudWatch Logs ajuda você a solucionar incidentes rapidamente, exibindo uma lista de streaming de novos eventos de log à medida que eles são consumidos. Você pode visualizar eventos de tarefas e serviços no console do Amazon ECS. Essa visualização fornece uma visão coesa da integridade de suas tarefas e serviços.

Cada tarefa em um serviço do Amazon ECS tem fluxos de log dedicados para cada contêiner. Se um serviço aumentar de escala verticalmente, cada instância de tarefa terá seu próprio conjunto de fluxos de log. A convenção de nomenclatura para fluxos de logs segue o padrão:

```
log-group-prefix/container-name/task-id
```

Uma única tarefa grava nos mesmos fluxos de logs durante sua vida útil. O fluxo de logs contém mensagens dos contêineres dessa tarefa e também qualquer saída do código da sua aplicação. Cada mensagem tem um registro de data e hora, incluindo logs personalizados.

> **Note**
>
> As sessões Live Tail acumulam custos por tempo de uso da sessão, por minuto. Para obter mais informações sobre a definição de preço, consulte Preços do Amazon CloudWatch.

Permissões obrigatórias

As seguintes permissões são necessárias para que o usuário do console do IAM inicie e interrompa as sessões do CloudWatch Logs Live Tail:

- `logs:DescribeLogGroups`
- `logs:StartLiveTail`
- `logs:StopLiveTail`

Procedimento

1. Abra o console em https://console.aws.amazon.com/ecs/v2.

2. Determinar o recurso que você deseja visualizar

Recurso	Etapas
Tarefas	a. Na página Clusters, escolha o cluster. A página de detalhes do cluster é exibida. b. Escolha a guia Tasks (Tarefas). A página de detalhes da tarefa é exibida. c. Escolha a guia Logs.
Serviços	a. Na página Clusters, escolha o cluster. A página de detalhes do cluster é exibida. b. Selecione o serviço. A página de detalhes do serviço é exibida. c. Escolha a guia Logs.

3. Escolha CloudWatch Logs Live Tail e, em seguida, Iniciar.
4. (Opcional) Para filtrar os fluxos, em Filtrar, em Selecionar grupos de logs, escolha o grupo de logs.

References da AWS CLI

Você também pode usar a AWS CLI para iniciar sessões do Live Tail para o CloudWatch Logs. Os seguintes recursos fornecem informações detalhadas sobre o uso da AWS CLI com o Live Tail:

- Referência do comando start-live-tail: sintaxe completa do comando, parâmetros e exemplos do comando `aws logs start-live-tail`.

- Guia do usuário do CloudWatch Logs Live Tail: guia abrangente que inclui o uso da AWS CLI nos modos somente para impressão e interativo.
- Exemplos do SDK do StartLiveTail: exemplos programáticos para usar a API StartLiveTail com vários SDKs da AWS.

Monitoramento de workloads usando metadados do Amazon ECS

É possível usar os metadados de tarefa e de contêiner para solucionar problemas relacionados às workloads e fazer alterações na configuração com base no ambiente de runtime.

Os metadados incluem as seguintes categorias:

- Atributos no nível da tarefa que fornecem informações sobre o local de execução da tarefa.
- Atributos no nível do contêiner que fornecem o ID, o nome e os detalhes da imagem do Docker.

  Isso fornece visibilidade do contêiner.
- Configurações de rede, como endereços IP, sub-redes e modo de rede.

  Isso ajuda na configuração da rede e na solução de problemas.
- Status e integridade de uma tarefa

  Isso permite que você saiba se as tarefas estão em execução.

É possível visualizar metadados usando qualquer um dos seguintes métodos:

- Arquivo de metadados de contêiner

  A partir da versão 1.15.0 do agente de contêiner do Amazon ECS, vários metadados de contêiner estão disponíveis em seus contêineres ou na instância de contêiner do host. Ao habilitar esse recurso, é possível consultar as informações sobre uma tarefa, um contêiner e uma instância de contêiner de dentro do contêiner ou da instância de contêiner do host. O arquivo de metadados é criado na instância do host e montado no contêiner como um volume do Docker e, portanto, não fica disponível quando uma tarefa é hospedada no AWS Fargate.
- Endpoint de metadados de tarefas

  O agente de contêiner do Amazon ECS injeta uma variável de ambiente em cada contêiner, conhecida como endpoint de metadados de tarefas, que fornece vários metadados de tarefas e dados estatísticos do Docker ao contêiner.

- Introspecção de contêiner

  O agente de contêiner do Amazon ECS fornece uma operação de API para reunir detalhes sobre a instância de contêiner na qual o agente está sendo executado e as tarefas associadas executadas nesta instância.

Variáveis de ambiente do Amazon ECS

As variáveis de ambiente controlam o comportamento de uma tarefa, como a capacidade usada sua execução. O Amazon ECS define as seguintes variáveis de ambiente para as tarefas:

- `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI`: o endpoint do URL HTTP completo para o SDK usar ao fazer uma solicitação de credenciais. Isso inclui o esquema e o host. Para obter mais informações, consulte Container credential provider no AWS SDKs Reference Guide.
- `ECS_CONTAINER_METADATA_URI_V4`: o endereço dos metadados da tarefa versão 4. Para obter mais informações, consulte Endpoint de metadados de tarefas do Amazon ECS versão 4.
- `ECS_CONTAINER_METADATA_URI`: o endereço dos metadados da tarefa versão 3. Para obter mais informações, consulte Endpoint de metadados de tarefas do Amazon ECS versão 3.
- `ECS_AGENT_URI`: o endereço de base dos diferentes endpoints compatíveis com o Fargate. Para obter mais informações, consulte:
  - Endpoint de Proteção de Tarefa na Redução de Escala do Amazon ECS
  - Endpoints para injeção de falhas do Amazon ECS
- `AWS_EXECUTION_ENV`: as informações sobre o tipo de inicialização em que a tarefa é executada.
  - Para o Fargate, o Amazon ECS define esse parâmetro como `AWS_ECS_FARGATE`.
  - Para o EC2, o Amazon ECS define esse parâmetro como `AWS_ECS_EC2`.
- `AWS_DEFAULT_REGION`: o padrão Região da AWS para o Conta da AWS. Esta é a região padrão em que a tarefa é executada.
- `AWS_REGION`: a região em que a tarefa é executada. Se estiver definido, esse valor substituirá `AWS_DEFAULT_REGION`.

Você pode visualizar as variáveis de ambiente nos metadados da tarefa. Para obter mais informações, consulte um dos tópicos a seguir:

- Fargate: Metadados de tarefas do Amazon ECS disponíveis para tarefas no Fargate
- EC2: Metadados de tarefas disponíveis para tarefas do Amazon ECS no EC2

Arquivo de metadados de contêiner do Amazon ECS

A partir da versão 1.15.0 do agente de contêiner do Amazon ECS, vários metadados de contêiner estão disponíveis em seus contêineres ou na instância de contêiner do host. Ao habilitar esse recurso, é possível consultar as informações sobre uma tarefa, um contêiner e uma instância de contêiner de dentro do contêiner ou da instância de contêiner do host. O arquivo de metadados é criado na instância do host e montado no contêiner como um volume do Docker e, portanto, não fica disponível quando uma tarefa é hospedada no AWS Fargate.

O arquivo de metadados do contêiner é limpo na instância do host quando o contêiner é limpo. É possível ajustar quando isso acontece com a variável do agente de contêiner `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION`. Para obter mais informações, consulte Limpeza automática de tarefas e de imagens do Amazon ECS.

Tópicos

- Locais do arquivo de metadados do contêiner
- Ativação de metadados de contêiner do Amazon ECS
- Formato do arquivo de metadados de contêiner do Amazon ECS

Locais do arquivo de metadados do contêiner

Por padrão, o arquivo de metadados do contêiner é gravado nos seguintes caminhos de host e contêiner.

- Para instâncias do Linux:
  - Caminho do host: `/var/lib/ecs/data/metadata/`*`cluster_name`*`/`*`task_id`*`/`*`container_name`*`/ecs-container-metadata.json`

> **Note**
>
> O caminho do host Linux pressupõe que o caminho padrão de montagem do diretório de dados (`/var/lib/ecs/data`) é usado quando o agente é iniciado. Se você não estiver usando a AMI otimizada para Amazon ECS (ou o pacote `ecs-init` para iniciar e manter o agente de contêiner), defina a variável de configuração do agente `ECS_HOST_DATA_DIR` para o caminho do host em que está localizado o arquivo de

estado do agente de contêiner. Para obter mais informações, consulte Configuração do agente de contêiner do Amazon ECS.

  - Caminho do contêiner: `/opt/ecs/metadata/random_ID/ecs-container-metadata.json`
- Para instâncias Windows:
  - Caminho do host: `C:\ProgramData\Amazon\ECS\data\metadata\task_id\container_name\ecs-container-metadata.json`
  - Caminho do contêiner: `C:\ProgramData\Amazon\ECS\metadata\random_ID\ecs-container-metadata.json`

No entanto, para acesso fácil, o local do arquivo de metadados do contêiner é definido na variável de ambiente `ECS_CONTAINER_METADATA_FILE`, dentro do contêiner. É possível ler o conteúdo do arquivo de dentro de contêiner com o comando a seguir:

- Para instâncias do Linux:

```
cat $ECS_CONTAINER_METADATA_FILE
```

- Para instâncias do Windows (PowerShell):

```
Get-Content -path $env:ECS_CONTAINER_METADATA_FILE
```

Ativação de metadados de contêiner do Amazon ECS

É possível ativar os metadados de contêiner no nível de instância de contêiner definindo a variável do agente de contêiner `ECS_ENABLE_CONTAINER_METADATA` como `true`. É possível definir essa variável no arquivo de configuração `/etc/ecs/ecs.config` e reiniciar o agente. Também é possível defini-la como uma variável de ambiente do Docker no runtime quando o contêiner do agente for iniciado. Para obter mais informações, consulte Configuração do agente de contêiner do Amazon ECS.

Se o `ECS_ENABLE_CONTAINER_METADATA` for definido como `true` quando o agente for iniciado, os arquivos de metadados serão criados para quaisquer contêineres criados a partir desse ponto em diante. O agente de contêiner do Amazon ECS não pode criar arquivos de metadados para contêineres que foram criados antes que a variável de agente de contêiner

`ECS_ENABLE_CONTAINER_METADATA` tenha sido definida como `true`. Para garantir que todos os contêineres recebam arquivos de metadados, você deve definir essa variável de agente na execução da instância de contêiner. Veja a seguir um exemplo de script de dados do usuário que definirá essa variável, bem como registrará sua instância de contêiner em seu cluster.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=your_cluster_name
ECS_ENABLE_CONTAINER_METADATA=true
EOF
```

Formato do arquivo de metadados de contêiner do Amazon ECS

As informações a seguir são armazenadas no arquivo JSON dos metadados de contêiner.

`Cluster`

O nome do cluster onde a tarefa do contêiner está sendo executada.

`ContainerInstanceARN`

O nome de recurso da Amazon (ARN) da instância de contêiner do host.

`TaskARN`

O nome de recurso da Amazon (ARN) da tarefa à qual o contêiner pertence.

`TaskDefinitionFamily`

O nome da família de definições de tarefa que o contêiner está usando.

`TaskDefinitionRevision`

A revisão de definição de tarefa que o contêiner está usando.

`ContainerID`

O ID do contêiner do Docker (e não o ID do contêiner do Amazon ECS) para o contêiner.

`ContainerName`

O nome do contêiner da definição de tarefa do Amazon ECS para o contêiner.

`DockerContainerName`

O nome do contêiner que o daemon do Docker usa para o contêiner (por exemplo, o nome que aparece no resultado do comando docker ps).

`ImageID`

O resumo SHA para a imagem do Docker usada para iniciar o contêiner.

`ImageName`

O nome da imagem e a tag para a imagem do Docker usada para iniciar o contêiner.

`PortMappings`

Todos os mapeamentos de porta associados ao contêiner.

`ContainerPort`

A porta no contêiner que é exposto.

`HostPort`

A porta na instância de contêiner do host que é exposto.

`BindIp`

O endereço IP associado atribuído ao contêiner pelo Docker. Esse endereço IP só é aplicado com o modo de rede `bridge` e só pode ser acessado pela instância de contêiner.

`Protocol`

O protocolo de rede usado para o mapeamento da porta.

`Networks`

O modo de rede e o endereço IP do contêiner.

`NetworkMode`

O modo de rede da tarefa à qual o contêiner pertence.

`IPv4Addresses`

Os endereços IP associados ao contêiner.

> **Important**
>
> Se a sua tarefa estiver usando o modo de rede `awsvpc`, o endereço IP do contêiner não será retornado. Nesse caso, você pode recuperar o endereço IP lendo o arquivo /etc/hosts com o seguinte comando:
>
> ```
> tail -1 /etc/hosts | awk '{print $1}'
> ```

`MetadataFileStatus`

O status do arquivo de metadados. Quando o status for `READY`, o arquivo de metadados estará atualizado e concluído. Se o arquivo ainda não estiver pronto (por exemplo, no momento em que a tarefa é iniciada), uma versão truncada do formato de arquivo estará disponível. Para evitar uma provável condição de corrida na qual o contêiner foi iniciado, mas os metadados ainda não foram criados, você pode analisar o arquivo de metadados e aguardar até que esse parâmetro seja definido como `READY` antes, dependendo dos metadados. Isso geralmente fica disponível menos de 1 segundo depois que o contêiner é iniciado.

`AvailabilityZone`

A zona de disponibilidade na qual a instância de contêiner do host reside.

`HostPrivateIPv4Address`

O endereço IP privado da tarefa à qual o contêiner pertence.

`HostPublicIPv4Address`

O endereço IP público da tarefa à qual o contêiner pertence.

Example Arquivo de metadados do contêiner do Amazon ECS (**`READY`**)

O exemplo a seguir mostra um arquivo de metadados do contêiner com status `READY`.

```
{
    "Cluster":"arn:aws:ecs:us-east-1:123456789012:cluster/MyCluster",
    "TaskARN":"arn:aws:ecs:us-east-1:123456789012:task/MyCluster/
b593651c4d6b44a6b2b583f45c957e15",
    "Family":"curltest-container",
    "Revision":"2",
    "DesiredStatus":"RUNNING",
    "KnownStatus":"RUNNING",
    "Limits":
        {
            "CPU":0.25,
            "Memory":512
        },
    "PullStartedAt":"2025-01-17T20:56:17.394610044Z",
    "PullStoppedAt":"2025-01-17T20:56:25.282708213Z",
    "AvailabilityZone":"us-east-1b",
    "LaunchType":"FARGATE",
    "Containers":[
```

```
        {
            "DockerId":"b593651c4d6b44a6b2b583f45c957e15-3356213583",
            "Name":"curltest","DockerName":"curltest",
            "Image":"public.ecr.aws/amazonlinux/amazonlinux:latest",

 "ImageID":"sha256:7f371357694782356b65c7fd60dd1ca124c47bd5ed1b1ffe7c0e17f562898367",
            "Labels":
                {
                    "com.amazonaws.ecs.cluster":"arn:aws:ecs:us-
east-1:123456789012:cluster/MyCluster",
                    "com.amazonaws.ecs.container-name":"curltest",
                    "com.amazonaws.ecs.task-arn":"arn:aws:ecs:us-
east-1:123456789012:task/MyCluster/b593651c4d6b44a6b2b583f45c957e15",
                    "com.amazonaws.ecs.task-definition-family":"curltest-
container","com.amazonaws.ecs.task-definition-version":"2"
                },
            "DesiredStatus":"RUNNING",
            "KnownStatus":"RUNNING",
            "Limits":
                {
                    "CPU":2
                },
            "CreatedAt":"2025-01-17T20:56:26.180347056Z",
            "StartedAt":"2025-01-17T20:56:26.180347056Z",
            "Type":"NORMAL",
            "LogDriver":"awslogs",
            "LogOptions":
                {
                    "awslogs-create-group":"true",
                    "awslogs-group":"/ecs/curltest-container",
                    "awslogs-region":"us-east-1",
                    "awslogs-stream":"ecs/curltest/b593651c4d6b44a6b2b583f45c957e15"
                       },
            "ContainerARN":"arn:aws:ecs:us-east-1:123456789012:container/MyCluster/
b593651c4d6b44a6b2b583f45c957e15/934575e8-5bdb-478f-b763-2341a85b690e",
            "Networks":[
                {
                    "NetworkMode":"awsvpc",
                    "IPv4Addresses":["10.0.1.58"]
                }
            ],
            "Snapshotter":"overlayfs"
        }
    ],
```

```
    "ClockDrift":
        {
            "ClockErrorBound":0.487801,"ReferenceTimestamp":"2025-01-17T20:56:02Z",
            "ClockSynchronizationStatus":"SYNCHRONIZED"
        },
    "FaultInjectionEnabled":false
}
```

Example Arquivo incompleto de metadados do contêiner do Amazon ECS (ainda não **`READY`**)

O exemplo a seguir mostra um arquivo de metadados de contêiner que ainda não atingiu o status `READY`. As informações no arquivo são limitadas a alguns parâmetros conhecidos da definição de tarefa. O arquivo de metadados do contêiner deve estar pronto em até 1 segundo após o contêiner ser iniciado.

```
{
    "Cluster": "default",
    "ContainerInstanceARN": "arn:aws:ecs:us-west-2:012345678910:container-instance/
default/1f73d099-b914-411c-a9ff-81633b7741dd",
    "TaskARN": "arn:aws:ecs:us-west-2:012345678910:task/default/
d90675f8-1a98-444b-805b-3d9cabb6fcd4",
    "ContainerName": "metadata"
}
```

Metadados de tarefas disponíveis para tarefas do Amazon ECS no EC2

O agente de contêiner do Amazon ECS fornece um método para recuperar diversos metadados de tarefas e dados estatísticos do Docker. Isso é denominado endpoint de metadados de tarefas. As seguintes versões estão disponíveis:

- Endpoint de metadados de tarefas versão 4: fornece uma variedade de metadados e dados estatísticos do Docker para contêineres. Também pode fornecer dados de taxa de rede. Disponível para tarefas do Amazon ECS inicializadas em instâncias Linux do Amazon EC2 executando pelo menos a versão `1.39.0` do agente de contêiner do Amazon ECS. Para instâncias do Windows do Amazon EC2 que usam o modo de rede `awsvpc`, o agente de contêiner do Amazon ECS deve ter pelo menos a versão `1.54.0`. Para obter mais informações, consulte Endpoint de metadados de tarefas do Amazon ECS versão 4.
- Endpoint de metadados de tarefas versão 3: fornece uma variedade de metadados e dados estatísticos do Docker para contêineres. Disponível para tarefas do Amazon ECS inicializadas

em instâncias Linux do Amazon EC2 executando pelo menos a versão `1.21.0` do agente de contêiner do Amazon ECS. Para instâncias do Windows do Amazon EC2 que usam o modo de rede `awsvpc`, o agente de contêiner do Amazon ECS deve ter pelo menos a versão `1.54.0`. Para obter mais informações, consulte Endpoint de metadados de tarefas do Amazon ECS versão 3.

- Endpoint de metadados de tarefas versão 2: disponível para tarefas do Amazon ECS inicializadas em instâncias Linux do Amazon EC2 executando pelo menos a versão `1.17.0` do agente de contêiner do Amazon ECS. Para instâncias do Windows do Amazon EC2 que usam o modo de rede `awsvpc`, o agente de contêiner do Amazon ECS deve ter pelo menos a versão `1.54.0`. Para obter mais informações, consulte Endpoint de metadados de tarefas do Amazon ECS versão 2.

Se sua tarefa do Amazon ECS estiver hospedada no Amazon EC2, você também poderá acessar os metadados do host de tarefas usando o endpoint do serviço de metadados de instância (IMDS). O comando a seguir, quando executado de dentro da instância que hospeda a tarefa, lista o ID da instância host.

```
curl http://169.254.169.254/latest/meta-data/instance-id
```

As informações que podem ser obtidas do endpoint são divididas em categorias, como *instance-id*. Para obter mais informações sobre as diferentes categorias de metadados da instância hospedeira que podem ser obtidas usando o endpoint, consulte Categorias de metadados da instância.

Endpoint de metadados de tarefas do Amazon ECS versão 4

O agente de contêiner do Amazon ECS injeta uma variável de ambiente em cada contêiner, conhecida como endpoint de metadados de tarefas, que fornece vários metadados de tarefas e dados estatísticos do Docker ao contêiner.

Os metadados de tarefas os dados estatísticos de taxa de rede são enviados para o CloudWatch Container Insights e podem ser visualizados no AWS Management Console. Para obter mais informações, consulte Monitorar contêineres do Amazon ECS utilizando o Container Insights com observabilidade aprimorada.

> **Note**
>
> O Amazon ECS fornece versões anteriores do endpoint de metadados de tarefas. Para evitar a necessidade de criar novas versões de endpoint de metadados de tarefas no futuro,

os metadados adicionais podem ser adicionados à saída da versão 4. Não removeremos quaisquer metadados existentes nem alteraremos os nomes dos campos de metadados.

A variável de ambiente é injetada por padrão nos contêineres das tarefas do Amazon ECS inicializadas em instâncias do Linux do Amazon EC2 que estão executando pelo menos a versão `1.39.0` do agente de contêiner do Amazon ECS. Para instâncias do Windows do Amazon EC2 que usam o modo de rede `awsvpc`, o agente de contêiner do Amazon ECS deve ter pelo menos a versão `1.54.0`. Para obter mais informações, consulte Gerenciamento de instâncias de contêiner do Linux no Amazon ECS.

> **Note**
>
> É possível adicionar suporte a esse recurso em instâncias do Amazon EC2 usando versões mais antigas do agente de contêiner do Amazon ECS atualizando o agente para a versão mais recente. Para obter mais informações, consulte Atualizar o agente de contêiner do Amazon ECS.

Para obter exemplos de saída de metadados de tarefas, consulte Exemplos de metadados de tarefas v4 no Amazon ECS.

Caminhos do endpoint de metadados de tarefas versão 4

Os seguintes caminhos de endpoints de metadados de tarefas estão disponíveis para contêineres.

`${ECS_CONTAINER_METADATA_URI_V4}`

Esse caminho retorna metadados para o contêiner.

`${ECS_CONTAINER_METADATA_URI_V4}/task`

Esse caminho retorna metadados para a tarefa, incluindo uma lista dos nomes e IDs de todos os contêineres associados à tarefa. Para obter mais informações sobre a resposta para esse endpoint, consulte Resposta JSON para metadados de tarefas v4 no Amazon ECS.

`${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`

Esse caminho retorna os metadados para a tarefa incluída no endpoint `/task` além das etiquetas de tarefa e de instância de contêiner que podem ser recuperadas usando a API

`ListTagsForResource`. Todos os erros recebidos ao serem recuperados os metadados da etiqueta serão incluídos no campo `Errors` da resposta.

> **Note**
>
> O campo `Errors` existe apenas na resposta para tarefas hospedadas em instâncias do Linux do Amazon EC2 executando pelo menos a versão `1.50.0` do agente de contêiner. Para instâncias do Windows do Amazon EC2 que usam o modo de rede `awsvpc`, o agente de contêiner do Amazon ECS deve ter pelo menos a versão `1.54.0`
> Esse endpoint requer a permissão `ecs.ListTagsForResource`.

> **Important**
>
> Ao usar o endpoint `${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`, esteja ciente de que cada chamada faz até duas solicitações de API para `ecs:ListTagsForResource` (uma para tags de instâncias de contêineres e outra para tags de tarefas) e que qualquer contêiner sidecar na tarefa pode fazer essas chamadas em seu nome. Chamadas frequentes de endpoint podem resultar no controle de utilização de API.
> Considere implementar estratégias de armazenamento em cache ou de processamento em lote para reduzir a frequência das chamadas, especialmente em aplicações de alto tráfego, e depurar problemas de controle de utilização de API usando o AWS CloudTrail. Para obter informações sobre os limites de controles de utilização para a API `ListTagsForResource`, consulte Request throttling for the Amazon ECS API na Referência de APIs do Amazon Elastic Container Service. Para obter informações sobre como depurar chamadas de API do Amazon ECS usando o AWS CloudTrail, consulte Registro em log das chamadas de API do Amazon ECS usando o AWS CloudTrail.

`${ECS_CONTAINER_METADATA_URI_V4}/stats`

Esse caminho retorna dados estatísticos do Docker para o contêiner específico. Para obter mais informações sobre cada uma das estatísticas retornadas, consulte ContainerStats na documentação da API do Docker.

Para tarefas do Amazon ECS que usam o modo de rede `awsvpc` ou `bridge` hospedado em instâncias do Linux do Amazon EC2 executando pelo menos a versão `1.43.0` do agente de

contêiner, haverá dados estatísticos adicionais de taxa de rede incluídos na resposta. Para todas as outras tarefas, a resposta só incluirá os dados estatísticos de rede cumulativos.

`${ECS_CONTAINER_METADATA_URI_V4}/task/stats`

Esse caminho retorna os dados estatísticos do Docker para todos os contêineres associados à tarefa. Isso pode ser usado por contêineres de arquivos associados para extrair métricas de rede. Para obter mais informações sobre cada uma das estatísticas retornadas, consulte ContainerStats na documentação da API do Docker.

Para tarefas do Amazon ECS que usam o modo de rede `awsvpc` ou `bridge` hospedado em instâncias do Linux do Amazon EC2 executando pelo menos a versão `1.43.0` do agente de contêiner, haverá dados estatísticos adicionais de taxa de rede incluídos na resposta. Para todas as outras tarefas, a resposta só incluirá os dados estatísticos de rede cumulativos.

Resposta JSON para metadados de tarefas v4 no Amazon ECS

As seguintes informações são retornadas da resposta em JSON (`${ECS_CONTAINER_METADATA_URI_V4}/task`) do endpoint de metadados de tarefas. Isso inclui metadados associados à tarefa, além dos metadados de cada contêiner dentro da tarefa.

`Cluster`

O nome do recurso da Amazon (ARN) ou nome curto do cluster do Amazon ECS ao qual a tarefa pertence.

`ServiceName`

O nome do serviço ao qual a tarefa pertence. Será exibido ServiceName para instâncias de contêiner do Amazon EC2 e do Amazon ECS Anywhere, se a tarefa estiver associada a um serviço.

> **Note**
>
> Os metadados `ServiceName` são incluídos somente quando é usada a versão `1.63.1` ou posterior do agente de contêiner do Amazon ECS.

`VPCID`

O ID da VPC da instância de contêiner do Amazon EC2. Esse campo é exibido somente para instâncias do Amazon EC2.

> **Note**
>
> Os metadados `VPCID` são incluídos somente quando é usada a versão `1.63.1` ou posterior do agente de contêiner do Amazon ECS.

`TaskARN`

O nome do recurso da Amazon (ARN) da tarefa à qual o contêiner pertence.

`Family`

A família da definição de tarefa do Amazon ECS para a tarefa.

`Revision`

A revisão da definição de tarefa do Amazon ECS para a tarefa.

`DesiredStatus`

O status desejado para a tarefa do Amazon ECS.

`KnownStatus`

O status conhecido para a tarefa do Amazon ECS.

`Limits`

Os limites de recursos especificados no nível da tarefa, como CPU (expresso em vCPUs) e memória. Esse parâmetro será omitido se não houver nenhum limite de recurso definido.

`PullStartedAt`

O timestamp de quando começou a primeira extração de imagem do contêiner.

`PullStoppedAt`

O timestamp de quando a última extração de imagem do contêiner terminou.

`AvailabilityZone`

A zona de disponibilidade em que a tarefa está.

> **Note**
>
> Os metadados da zona de disponibilidade estão disponíveis apenas para tarefas do Fargate que usam a versão 1.4 ou posterior (Linux) ou 1.0.0 (Windows) da plataforma.

`LaunchType`

O tipo de inicialização que a tarefa está usando. Ao usar provedores de capacidade de cluster, isso indica se a tarefa está usando a infraestrutura do Fargate ou do EC2.

> **Note**
>
> Esses metadados `LaunchType` estão incluídos somente quando é usada a versão `1.45.0` ou posterior (Linux) ou 1.0.0 ou posterior (Windows) do agente de contêiner do Linux do Amazon ECS.

`Containers`

Uma lista de metadados de contêiner para cada contêiner associado com a tarefa.

`DockerId`

O ID do Docker do contêiner.

Quando você usa Fargate, o id é um hexadecimal de 32 dígitos seguido por um número de 10 dígitos.

`Name`

O nome do contêiner, conforme especificado na definição da tarefa.

`DockerName`

O nome do contêiner fornecido para o Docker. O agente de contêiner do Amazon ECS gera um nome exclusivo para o contêiner para evitar colisões de nomes quando várias cópias da mesma definição de tarefa são executadas em uma única instância.

`Image`

A imagem para o contêiner.

`ImageID`

O resumo SHA-256 do manifesto da imagem. Esse é o resumo que pode ser usado para extrair a imagem usando o formato `repository-url/image@sha256:digest`.

`Ports`

Todas as portas expostas para o contêiner. Esse parâmetro será omitido se não houver portas expostas.

`Labels`

Todos os rótulos aplicados ao contêiner. Esse parâmetro será omitido se não houver rótulos aplicados.

`DesiredStatus`

O status desejado para o contêiner do Amazon ECS.

`KnownStatus`

O status conhecido para o contêiner do Amazon ECS.

`ExitCode`

O código de saída para o contêiner. Esse parâmetro é omitido se o contêiner não foi encerrado.

`Limits`

Os limites de recursos especificados no nível do contêiner, como CPU (expresso em unidades de CPU) e memória. Esse parâmetro será omitido se não houver nenhum limite de recurso definido.

`CreatedAt`

O time stamp de quando o contêiner foi criado. Esse parâmetro será omitido se o contêiner ainda não tiver sido criado.

`StartedAt`

O time stamp de quando o contêiner foi iniciado. Esse parâmetro será omitido se o contêiner ainda não tiver sido iniciado.

`FinishedAt`

O time stamp de quando o contêiner foi interrompido. Esse parâmetro será omitido se o contêiner ainda não tiver sido interrompido.

`Type`

O tipo do contêiner. Os contêineres que são especificados em sua definição de tarefa são do tipo `NORMAL`. É possível ignorar outros tipos de contêineres, que são usados para o provisionamento de recursos de tarefas internas pelo agente de contêiner do Amazon ECS.

`LogDriver`

O driver de log que o contêiner está usando.

> **Note**
>
> Esses metadados `LogDriver` estão incluídos somente quando é usada a versão `1.45.0` ou posterior do agente de contêiner do Linux do Amazon ECS.

`LogOptions`

As opções de driver de log definidas para o contêiner.

> **Note**
>
> Esses metadados `LogOptions` estão incluídos somente quando é usada a versão `1.45.0` ou posterior do agente de contêiner do Linux do Amazon ECS.

`ContainerARN`

O nome do recurso da Amazon (ARN) do contêiner.

> **Note**
>
> Esses metadados `ContainerARN` estão incluídos somente quando é usada a versão `1.45.0` ou posterior do agente de contêiner do Linux do Amazon ECS.

`Networks`

As informações de rede para o contêiner, como o modo de rede e o endereço IP. Esse parâmetro será omitido se não houver informações de rede definidas.

`RestartCount`

O número de vezes que um contêiner foi reiniciado.

> **Note**
>
> Os metadados de `RestartCount` são incluídos somente se uma política de reinicialização estiver habilitada para o contêiner. Para obter mais informações, consulte Reiniciar contêineres individuais em tarefas do Amazon ECS com políticas de reinicialização de contêineres.

`ExecutionStoppedAt`

O time stamp de quando o `DesiredStatus` da tarefa mudou para `STOPPED`. Isso ocorre quando um contêiner essencial muda para `STOPPED`.

Exemplos de metadados de tarefas v4 no Amazon ECS

Os exemplos a seguir mostram exemplos de saídas de cada endpoint de metadados de tarefas.

Exemplo de resposta de metadados de contêineres

Ao consultar o endpoint `${ECS_CONTAINER_METADATA_URI_V4}`, são retornados apenas metadados sobre o próprio contêiner. A seguir está um exemplo de saída de uma tarefa executada como parte de um serviço (MyService).

```
{
    "DockerId": "ea32192c8553fbff06c9340478a2ff089b2bb5646fb718b4ee206641c9086d66",
    "Name": "curl",
    "DockerName": "ecs-curltest-24-curl-cca48e8dcadd97805600",
    "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
    "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
    "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/8f03e41243824aea923aca126495f665",
        "com.amazonaws.ecs.task-definition-family": "curltest",
        "com.amazonaws.ecs.task-definition-version": "24"
    },
```

```
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "Limits": {
        "CPU": 10,
        "Memory": 128
    },
    "CreatedAt": "2020-10-02T00:15:07.620912337Z",
    "StartedAt": "2020-10-02T00:15:08.062559351Z",
    "Type": "NORMAL",
    "LogDriver": "awslogs",
    "LogOptions": {
        "awslogs-create-group": "true",
        "awslogs-group": "/ecs/metadata",
        "awslogs-region": "us-west-2",
        "awslogs-stream": "ecs/curl/8f03e41243824aea923aca126495f665"
    },
    "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/0206b271-
b33f-47ab-86c6-a0ba208a70a9",
    "Networks": [
        {
            "NetworkMode": "awsvpc",
            "IPv4Addresses": [
                "10.0.2.100"
            ],
            "AttachmentIndex": 0,
            "MACAddress": "0e:9e:32:c7:48:85",
            "IPv4SubnetCIDRBlock": "10.0.2.0/24",
            "PrivateDNSName": "ip-10-0-2-100.us-west-2.compute.internal",
            "SubnetGatewayIpv4Address": "10.0.2.1/24"
        }
    ]
}
```

Exemplo de resposta de metadados de tarefas

Ao consultar o endpoint `${ECS_CONTAINER_METADATA_URI_V4}/task`, são retornados metadados sobre a tarefa da qual o contêiner faz parte, e também os metadados de cada contêiner na tarefa. Veja a seguir um exemplo de saída.

```
{
    "Cluster": "default",
    "TaskARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
```

```
    "Family": "curltest",
    "ServiceName": "MyService",
    "Revision": "26",
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "PullStartedAt": "2020-10-02T00:43:06.202617438Z",
    "PullStoppedAt": "2020-10-02T00:43:06.31288465Z",
    "AvailabilityZone": "us-west-2d",
    "VPCID": "vpc-1234567890abcdef0",
    "LaunchType": "EC2",
    "Containers": [
        {
            "DockerId":
 "598cba581fe3f939459eaba1e071d5c93bb2c49b7d1ba7db6bb19deeb70d8e38",
            "Name": "~internal~ecs~pause",
            "DockerName": "ecs-curltest-26-internalecspause-e292d586b6f9dade4a00",
            "Image": "amazon/amazon-ecs-pause:0.1.0",
            "ImageID": "",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RESOURCES_PROVISIONED",
            "KnownStatus": "RESOURCES_PROVISIONED",
            "Limits": {
                "CPU": 0,
                "Memory": 0
            },
            "CreatedAt": "2020-10-02T00:43:05.602352471Z",
            "StartedAt": "2020-10-02T00:43:06.076707576Z",
            "Type": "CNI_PAUSE",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
```

```
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        },
        {
            "DockerId":
 "ee08638adaaf009d78c248913f629e38299471d45fe7dc944d1039077e3424ca",
            "Name": "curl",
            "DockerName": "ecs-curltest-26-curl-a0e7dba5aca6d8cb2e00",
            "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
            "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "curl",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Limits": {
                "CPU": 10,
                "Memory": 128
            },
            "CreatedAt": "2020-10-02T00:43:06.326590752Z",
            "StartedAt": "2020-10-02T00:43:06.767535449Z",
            "Type": "NORMAL",
            "LogDriver": "awslogs",
            "LogOptions": {
                "awslogs-create-group": "true",
                "awslogs-group": "/ecs/metadata",
                "awslogs-region": "us-west-2",
                "awslogs-stream": "ecs/curl/158d1c8083dd49d6b527399fd6414f5c"
            },
            "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/
abb51bdd-11b4-467f-8f6c-adcfe1fe059d",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
```

```
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        }
    ]
}
```

Exemplo de tarefa com resposta de metadados de tarefas

Ao consultar o endpoint `${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`, são retornados metadados sobre a tarefa da qual o contêiner faz parte, incluindo a tarefa e as etiquetas de instância de contêiner. Veja a seguir um exemplo de saída.

```
{
    "Cluster": "default",
    "TaskARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
    "Family": "curltest",
    "ServiceName": "MyService",
    "Revision": "26",
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "PullStartedAt": "2020-10-02T00:43:06.202617438Z",
    "PullStoppedAt": "2020-10-02T00:43:06.31288465Z",
    "AvailabilityZone": "us-west-2d",
    "VPCID": "vpc-1234567890abcdef0",
    "TaskTags": {
        "tag-use": "task-metadata-endpoint-test"
    },
    "ContainerInstanceTags":{
        "tag_key":"tag_value"
    },
    "LaunchType": "EC2",
    "Containers": [
        {
            "DockerId":
 "598cba581fe3f939459eaba1e071d5c93bb2c49b7d1ba7db6bb19deeb70d8e38",
            "Name": "~internal~ecs~pause",
```

```
            "DockerName": "ecs-curltest-26-internalecspause-e292d586b6f9dade4a00",
            "Image": "amazon/amazon-ecs-pause:0.1.0",
            "ImageID": "",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RESOURCES_PROVISIONED",
            "KnownStatus": "RESOURCES_PROVISIONED",
            "Limits": {
                "CPU": 0,
                "Memory": 0
            },
            "CreatedAt": "2020-10-02T00:43:05.602352471Z",
            "StartedAt": "2020-10-02T00:43:06.076707576Z",
            "Type": "CNI_PAUSE",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        },
        {
            "DockerId":
 "ee08638adaaf009d78c248913f629e38299471d45fe7dc944d1039077e3424ca",
            "Name": "curl",
            "DockerName": "ecs-curltest-26-curl-a0e7dba5aca6d8cb2e00",
            "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
            "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
```

```
                "com.amazonaws.ecs.container-name": "curl",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Limits": {
                "CPU": 10,
                "Memory": 128
            },
            "CreatedAt": "2020-10-02T00:43:06.326590752Z",
            "StartedAt": "2020-10-02T00:43:06.767535449Z",
            "Type": "NORMAL",
            "LogDriver": "awslogs",
            "LogOptions": {
                "awslogs-create-group": "true",
                "awslogs-group": "/ecs/metadata",
                "awslogs-region": "us-west-2",
                "awslogs-stream": "ecs/curl/158d1c8083dd49d6b527399fd6414f5c"
            },
            "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/
abb51bdd-11b4-467f-8f6c-adcfe1fe059d",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        }
    ]
}
```

Exemplo de tarefa com uma resposta de metadados de erro

Ao consultar o endpoint `${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`, são retornados metadados sobre a tarefa da qual o contêiner faz parte, incluindo a tarefa e as etiquetas de instância de contêiner. Se houver um erro na recuperação dos dados de marcação, o erro será retornado na resposta. Veja a seguir um exemplo de saída para quando a função do IAM associada à instância de contêiner não tem a permissão `ecs:ListTagsForResource` concedida.

```
{
    "Cluster": "default",
    "TaskARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
    "Family": "curltest",
    "ServiceName": "MyService",
    "Revision": "26",
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "PullStartedAt": "2020-10-02T00:43:06.202617438Z",
    "PullStoppedAt": "2020-10-02T00:43:06.31288465Z",
    "AvailabilityZone": "us-west-2d",
    "VPCID": "vpc-1234567890abcdef0",
    "Errors": [
        {
            "ErrorField": "ContainerInstanceTags",
            "ErrorCode": "AccessDeniedException",
            "ErrorMessage": "User: arn:aws:sts::111122223333:assumed-
role/ecsInstanceRole/i-0744a608689EXAMPLE is not authorized to perform:
 ecs:ListTagsForResource on resource: arn:aws:ecs:us-west-2:111122223333:container-
instance/default/2dd1b186f39845a584488d2ef155c131",
            "StatusCode": 400,
            "RequestId": "cd597ef0-272b-4643-9bd2-1de469870fa6",
            "ResourceARN": "arn:aws:ecs:us-west-2:111122223333:container-instance/
default/2dd1b186f39845a584488d2ef155c131"
        },
        {
            "ErrorField": "TaskTags",
            "ErrorCode": "AccessDeniedException",
            "ErrorMessage": "User: arn:aws:sts::111122223333:assumed-
role/ecsInstanceRole/i-0744a608689EXAMPLE is not authorized to perform:
 ecs:ListTagsForResource on resource: arn:aws:ecs:us-west-2:111122223333:task/
default/9ef30e4b7aa44d0db562749cff4983f3",
            "StatusCode": 400,
            "RequestId": "862c5986-6cd2-4aa6-87cc-70be395531e1",
```

```
            "ResourceARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/9ef30e4b7aa44d0db562749cff4983f3"
        }
    ],
    "LaunchType": "EC2",
    "Containers": [
        {
            "DockerId":
 "598cba581fe3f939459eaba1e071d5c93bb2c49b7d1ba7db6bb19deeb70d8e38",
            "Name": "~internal~ecs~pause",
            "DockerName": "ecs-curltest-26-internalecspause-e292d586b6f9dade4a00",
            "Image": "amazon/amazon-ecs-pause:0.1.0",
            "ImageID": "",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RESOURCES_PROVISIONED",
            "KnownStatus": "RESOURCES_PROVISIONED",
            "Limits": {
                "CPU": 0,
                "Memory": 0
            },
            "CreatedAt": "2020-10-02T00:43:05.602352471Z",
            "StartedAt": "2020-10-02T00:43:06.076707576Z",
            "Type": "CNI_PAUSE",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        },
```

```
        {
            "DockerId":
 "ee08638adaaf009d78c248913f629e38299471d45fe7dc944d1039077e3424ca",
            "Name": "curl",
            "DockerName": "ecs-curltest-26-curl-a0e7dba5aca6d8cb2e00",
            "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
            "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "curl",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Limits": {
                "CPU": 10,
                "Memory": 128
            },
            "CreatedAt": "2020-10-02T00:43:06.326590752Z",
            "StartedAt": "2020-10-02T00:43:06.767535449Z",
            "Type": "NORMAL",
            "LogDriver": "awslogs",
            "LogOptions": {
                "awslogs-create-group": "true",
                "awslogs-group": "/ecs/metadata",
                "awslogs-region": "us-west-2",
                "awslogs-stream": "ecs/curl/158d1c8083dd49d6b527399fd6414f5c"
            },
            "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/
abb51bdd-11b4-467f-8f6c-adcfe1fe059d",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
```

```
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        }
    ]
}
```

Exemplo de resposta de dados estatísticos de contêiner

Ao consultar o endpoint `${ECS_CONTAINER_METADATA_URI_V4}/stats`, são retornadas métricas de rede para o contêiner. Para tarefas do Amazon ECS que usam o modo de rede `awsvpc` ou `bridge` hospedado em instâncias do Amazon EC2 executando pelo menos a versão `1.43.0` do agente de contêiner, haverá dados estatísticos adicionais de taxa de rede incluídos na resposta. Para todas as outras tarefas, a resposta só incluirá os dados estatísticos de rede cumulativos.

Veja a seguir um exemplo de saída de uma tarefa do Amazon ECS no Amazon EC2 que usa o modo de rede `bridge`.

```
{
    "read": "2020-10-02T00:51:13.410254284Z",
    "preread": "2020-10-02T00:51:12.406202398Z",
    "pids_stats": {
        "current": 3
    },
    "blkio_stats": {
        "io_service_bytes_recursive": [

        ],
        "io_serviced_recursive": [

        ],
        "io_queue_recursive": [

        ],
        "io_service_time_recursive": [

        ],
        "io_wait_time_recursive": [

        ],
        "io_merged_recursive": [

        ],
```

```
        "io_time_recursive": [

        ],
        "sectors_recursive": [

        ]
    },
    "num_procs": 0,
    "storage_stats": {

    },
    "cpu_stats": {
        "cpu_usage": {
            "total_usage": 360968065,
            "percpu_usage": [
                182359190,
                178608875
            ],
            "usage_in_kernelmode": 40000000,
            "usage_in_usermode": 290000000
        },
        "system_cpu_usage": 13939680000000,
        "online_cpus": 2,
        "throttling_data": {
            "periods": 0,
            "throttled_periods": 0,
            "throttled_time": 0
        }
    },
    "precpu_stats": {
        "cpu_usage": {
            "total_usage": 360968065,
            "percpu_usage": [
                182359190,
                178608875
            ],
            "usage_in_kernelmode": 40000000,
            "usage_in_usermode": 290000000
        },
        "system_cpu_usage": 13937670000000,
        "online_cpus": 2,
        "throttling_data": {
            "periods": 0,
            "throttled_periods": 0,
```

```
            "throttled_time": 0
        }
    },
    "memory_stats": {
        "usage": 1806336,
        "max_usage": 6299648,
        "stats": {
            "active_anon": 606208,
            "active_file": 0,
            "cache": 0,
            "dirty": 0,
            "hierarchical_memory_limit": 134217728,
            "hierarchical_memsw_limit": 268435456,
            "inactive_anon": 0,
            "inactive_file": 0,
            "mapped_file": 0,
            "pgfault": 4185,
            "pgmajfault": 0,
            "pgpgin": 2926,
            "pgpgout": 2778,
            "rss": 606208,
            "rss_huge": 0,
            "total_active_anon": 606208,
            "total_active_file": 0,
            "total_cache": 0,
            "total_dirty": 0,
            "total_inactive_anon": 0,
            "total_inactive_file": 0,
            "total_mapped_file": 0,
            "total_pgfault": 4185,
            "total_pgmajfault": 0,
            "total_pgpgin": 2926,
            "total_pgpgout": 2778,
            "total_rss": 606208,
            "total_rss_huge": 0,
            "total_unevictable": 0,
            "total_writeback": 0,
            "unevictable": 0,
            "writeback": 0
        },
        "limit": 134217728
    },
    "name": "/ecs-curltest-26-curl-c2e5f6e0cf91b0bead01",
    "id": "5fc21e5b015f899d22618f8aede80b6d70d71b2a75465ea49d9462c8f3d2d3af",
```

```
    "networks": {
        "eth0": {
            "rx_bytes": 84,
            "rx_packets": 2,
            "rx_errors": 0,
            "rx_dropped": 0,
            "tx_bytes": 84,
            "tx_packets": 2,
            "tx_errors": 0,
            "tx_dropped": 0
        }
    },
    "network_rate_stats": {
        "rx_bytes_per_sec": 0,
        "tx_bytes_per_sec": 0
    }
}
```

Exemplo de resposta de dados estatísticos de tarefas

Ao consultar o endpoint `${ECS_CONTAINER_METADATA_URI_V4}/task/stats`, são retornadas métricas de rede sobre a tarefa da qual o contêiner faz parte. Veja a seguir um exemplo de saída.

```
{
    "01999f2e5c6cf4df3873f28950e6278813408f281c54778efec860d0caad4854": {
        "read": "2020-10-02T00:51:32.51467703Z",
        "preread": "2020-10-02T00:51:31.50860463Z",
        "pids_stats": {
            "current": 1
        },
        "blkio_stats": {
            "io_service_bytes_recursive": [

            ],
            "io_serviced_recursive": [

            ],
            "io_queue_recursive": [

            ],
            "io_service_time_recursive": [

            ],
```

```
            "io_wait_time_recursive": [

            ],
            "io_merged_recursive": [

            ],
            "io_time_recursive": [

            ],
            "sectors_recursive": [

            ]
        },
        "num_procs": 0,
        "storage_stats": {

        },
        "cpu_stats": {
            "cpu_usage": {
                "total_usage": 177232665,
                "percpu_usage": [
                    13376224,
                    163856441
                ],
                "usage_in_kernelmode": 0,
                "usage_in_usermode": 160000000
            },
            "system_cpu_usage": 13977820000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
        },
        "precpu_stats": {
            "cpu_usage": {
                "total_usage": 177232665,
                "percpu_usage": [
                    13376224,
                    163856441
                ],
                "usage_in_kernelmode": 0,
                "usage_in_usermode": 160000000
```

```
            },
            "system_cpu_usage": 13975800000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
        },
        "memory_stats": {
            "usage": 532480,
            "max_usage": 6279168,
            "stats": {
                "active_anon": 40960,
                "active_file": 0,
                "cache": 0,
                "dirty": 0,
                "hierarchical_memory_limit": 9223372036854771712,
                "hierarchical_memsw_limit": 9223372036854771712,
                "inactive_anon": 0,
                "inactive_file": 0,
                "mapped_file": 0,
                "pgfault": 2033,
                "pgmajfault": 0,
                "pgpgin": 1734,
                "pgpgout": 1724,
                "rss": 40960,
                "rss_huge": 0,
                "total_active_anon": 40960,
                "total_active_file": 0,
                "total_cache": 0,
                "total_dirty": 0,
                "total_inactive_anon": 0,
                "total_inactive_file": 0,
                "total_mapped_file": 0,
                "total_pgfault": 2033,
                "total_pgmajfault": 0,
                "total_pgpgin": 1734,
                "total_pgpgout": 1724,
                "total_rss": 40960,
                "total_rss_huge": 0,
                "total_unevictable": 0,
                "total_writeback": 0,
                "unevictable": 0,
```

```
                "writeback": 0
            },
            "limit": 4073377792
        },
        "name": "/ecs-curltest-26-internalecspause-a6bcc3dbadfacfe85300",
        "id": "01999f2e5c6cf4df3873f28950e6278813408f281c54778efec860d0caad4854",
        "networks": {
            "eth0": {
                "rx_bytes": 84,
                "rx_packets": 2,
                "rx_errors": 0,
                "rx_dropped": 0,
                "tx_bytes": 84,
                "tx_packets": 2,
                "tx_errors": 0,
                "tx_dropped": 0
            }
        },
        "network_rate_stats": {
            "rx_bytes_per_sec": 0,
            "tx_bytes_per_sec": 0
        }
    },
    "5fc21e5b015f899d22618f8aede80b6d70d71b2a75465ea49d9462c8f3d2d3af": {
        "read": "2020-10-02T00:51:32.512771349Z",
        "preread": "2020-10-02T00:51:31.510597736Z",
        "pids_stats": {
            "current": 3
        },
        "blkio_stats": {
            "io_service_bytes_recursive": [

            ],
            "io_serviced_recursive": [

            ],
            "io_queue_recursive": [

            ],
            "io_service_time_recursive": [

            ],
            "io_wait_time_recursive": [
```

```
            ],
            "io_merged_recursive": [

            ],
            "io_time_recursive": [

            ],
            "sectors_recursive": [

            ]
        },
        "num_procs": 0,
        "storage_stats": {

        },
        "cpu_stats": {
            "cpu_usage": {
                "total_usage": 379075681,
                "percpu_usage": [
                    191355275,
                    187720406
                ],
                "usage_in_kernelmode": 60000000,
                "usage_in_usermode": 310000000
            },
            "system_cpu_usage": 13977800000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
        },
        "precpu_stats": {
            "cpu_usage": {
                "total_usage": 378825197,
                "percpu_usage": [
                    191104791,
                    187720406
                ],
                "usage_in_kernelmode": 60000000,
                "usage_in_usermode": 310000000
            },
            "system_cpu_usage": 13975800000000,
```

```
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
        },
        "memory_stats": {
            "usage": 1814528,
            "max_usage": 6299648,
            "stats": {
                "active_anon": 606208,
                "active_file": 0,
                "cache": 0,
                "dirty": 0,
                "hierarchical_memory_limit": 134217728,
                "hierarchical_memsw_limit": 268435456,
                "inactive_anon": 0,
                "inactive_file": 0,
                "mapped_file": 0,
                "pgfault": 5377,
                "pgmajfault": 0,
                "pgpgin": 3613,
                "pgpgout": 3465,
                "rss": 606208,
                "rss_huge": 0,
                "total_active_anon": 606208,
                "total_active_file": 0,
                "total_cache": 0,
                "total_dirty": 0,
                "total_inactive_anon": 0,
                "total_inactive_file": 0,
                "total_mapped_file": 0,
                "total_pgfault": 5377,
                "total_pgmajfault": 0,
                "total_pgpgin": 3613,
                "total_pgpgout": 3465,
                "total_rss": 606208,
                "total_rss_huge": 0,
                "total_unevictable": 0,
                "total_writeback": 0,
                "unevictable": 0,
                "writeback": 0
            },
```

```
            "limit": 134217728
        },
        "name": "/ecs-curltest-26-curl-c2e5f6e0cf91b0bead01",
        "id": "5fc21e5b015f899d22618f8aede80b6d70d71b2a75465ea49d9462c8f3d2d3af",
        "networks": {
            "eth0": {
                "rx_bytes": 84,
                "rx_packets": 2,
                "rx_errors": 0,
                "rx_dropped": 0,
                "tx_bytes": 84,
                "tx_packets": 2,
                "tx_errors": 0,
                "tx_dropped": 0
            }
        },
        "network_rate_stats": {
            "rx_bytes_per_sec": 0,
            "tx_bytes_per_sec": 0
        }
    }
}
```


Endpoint de metadados de tarefas do Amazon ECS versão 3

> **Important**
>
> O endpoint da versão 3 dos metadados da tarefa não está mais sendo mantido ativamente. Recomendamos que você atualize o endpoint versão 4 de metadados da tarefa para obter as informações mais recentes do endpoint de metadados. Para obter mais informações, consulte the section called “Endpoint de metadados de tarefas versão 4”.
> Se você estiver usando tarefas do Amazon ECS hospedadas no AWS Fargate, consulte Endpoint de metadados de tarefas do Amazon ECS versão 3 para tarefas no Fargate.

A partir da versão 1.21.0 do agente de contêineres do Amazon ECS, o agente injeta uma variável de ambiente denominada `ECS_CONTAINER_METADATA_URI` em cada contêiner de uma tarefa. Quando você consultar o endpoint de metadados de tarefas versão 3, vários metadados de tarefas e estatísticas do Docker estarão disponíveis para tarefas. Para tarefas que usam o modo de rede `bridge`, as métricas de rede estão disponíveis ao consultar os endpoints `/stats`.

O recurso de endpoint de metadados de tarefas versão 3 é habilitado por padrão para tarefas que usam o tipo de inicialização do Fargate na versão v1.3.0 ou posterior da plataforma e tarefas que usam o tipo de inicialização do EC2 e são iniciadas na infraestrutura do Linux do Amazon EC2 executando pelo menos a versão 1.21.0 do agente de contêiner do Amazon ECS ou na infraestrutura do Windows do Amazon EC2 executando pelo menos a versão `1.54.0` do agente de contêiner do Amazon ECS e usa o modo de rede `awsvpc`. Para obter mais informações, consulte Gerenciamento de instâncias de contêiner do Linux no Amazon ECS.

É possível adicionar suporte a esse recurso nas instâncias de contêiner mais antigas atualizando o agente para a versão mais recente. Para obter mais informações, consulte Atualizar o agente de contêiner do Amazon ECS.

> **Important**
>
> Para tarefas que usam o tipo de execução do Fargate e versões anteriores à v1.3.0 da plataforma, a versão 2 do endpoint de metadados de tarefas é compatível. Para obter mais informações, consulte Endpoint de metadados de tarefas do Amazon ECS versão 2.

Caminhos do endpoint de metadados de tarefas versão 3

Os seguintes endpoints de metadados de tarefas estão disponíveis para os contêineres:

`${ECS_CONTAINER_METADATA_URI}`

Esse caminho retorna o JSON de metadados para o contêiner.

`${ECS_CONTAINER_METADATA_URI}/task`

Esse caminho retorna o JSON de metadados para a tarefa, incluindo uma lista dos nomes e IDs de todos os contêineres associados à tarefa. Para obter mais informações sobre a resposta para esse endpoint, consulte Resposta JSON para metadados de tarefas v3 no Amazon ECS.

`${ECS_CONTAINER_METADATA_URI}/taskWithTags`

Esse caminho retorna os metadados para a tarefa incluída no endpoint `/task` além das etiquetas de tarefa e de instância de contêiner que podem ser recuperadas usando a API `ListTagsForResource`.

`${ECS_CONTAINER_METADATA_URI}/stats`

Esse caminho retorna o JSON de estatísticas do Docker para o contêiner do Docker específico. Para obter mais informações sobre cada uma das estatísticas retornadas, consulte ContainerStats na documentação da API do Docker.

`${ECS_CONTAINER_METADATA_URI}/task/stats`

Esse caminho retorna o JSON de estatísticas do Docker para todos os contêineres associados à tarefa. Para obter mais informações sobre cada uma das estatísticas retornadas, consulte ContainerStats na documentação da API do Docker.

Resposta JSON para metadados de tarefas v3 no Amazon ECS

As seguintes informações são retornadas da resposta em JSON (`${ECS_CONTAINER_METADATA_URI}/task`) do endpoint de metadados de tarefas.

`Cluster`

O nome do recurso da Amazon (ARN) ou nome curto do cluster do Amazon ECS ao qual a tarefa pertence.

`TaskARN`

O nome do recurso da Amazon (ARN) da tarefa à qual o contêiner pertence.

`Family`

A família da definição de tarefa do Amazon ECS para a tarefa.

`Revision`

A revisão da definição de tarefa do Amazon ECS para a tarefa.

`DesiredStatus`

O status desejado para a tarefa do Amazon ECS.

`KnownStatus`

O status conhecido para a tarefa do Amazon ECS.

`Limits`

Os limites de recursos especificados no nível da tarefa, como CPU (expresso em vCPUs) e memória. Esse parâmetro será omitido se não houver nenhum limite de recurso definido.

`PullStartedAt`

O timestamp de quando começou a primeira extração de imagem do contêiner.

`PullStoppedAt`

O timestamp de quando a última extração de imagem do contêiner terminou.

`AvailabilityZone`

A zona de disponibilidade em que a tarefa está.

> **Note**
>
> Os metadados da zona de disponibilidade estão disponíveis apenas para tarefas do Fargate que usam a versão 1.4 ou posterior (Linux) ou 1.0.0 ou posterior (Windows) da plataforma.

`Containers`

Uma lista de metadados de contêiner para cada contêiner associado com a tarefa.

`DockerId`

O ID do Docker do contêiner.

`Name`

O nome do contêiner, conforme especificado na definição da tarefa.

`DockerName`

O nome do contêiner fornecido para o Docker. O agente de contêiner do Amazon ECS gera um nome exclusivo para o contêiner para evitar colisões de nomes quando várias cópias da mesma definição de tarefa são executadas em uma única instância.

`Image`

A imagem para o contêiner.

`ImageID`

O resumo SHA-256 do manifesto da imagem. Esse é o resumo que pode ser usado para extrair a imagem usando o formato `repository-url/image@sha256:digest`.

`Ports`

Todas as portas expostas para o contêiner. Esse parâmetro será omitido se não houver portas expostas.

`Labels`

Todos os rótulos aplicados ao contêiner. Esse parâmetro será omitido se não houver rótulos aplicados.

`DesiredStatus`

O status desejado para o contêiner do Amazon ECS.

`KnownStatus`

O status conhecido para o contêiner do Amazon ECS.

`ExitCode`

O código de saída para o contêiner. Esse parâmetro é omitido se o contêiner não foi encerrado.

`Limits`

Os limites de recursos especificados no nível do contêiner, como CPU (expresso em unidades de CPU) e memória. Esse parâmetro será omitido se não houver nenhum limite de recurso definido.

`CreatedAt`

O time stamp de quando o contêiner foi criado. Esse parâmetro será omitido se o contêiner ainda não tiver sido criado.

`StartedAt`

O time stamp de quando o contêiner foi iniciado. Esse parâmetro será omitido se o contêiner ainda não tiver sido iniciado.

`FinishedAt`

O time stamp de quando o contêiner foi interrompido. Esse parâmetro será omitido se o contêiner ainda não tiver sido interrompido.

`Type`

O tipo do contêiner. Os contêineres que são especificados em sua definição de tarefa são do tipo `NORMAL`. É possível ignorar outros tipos de contêineres, que são usados para o provisionamento de recursos de tarefas internas pelo agente de contêiner do Amazon ECS.

`Networks`

As informações de rede para o contêiner, como o modo de rede e o endereço IP. Esse parâmetro será omitido se não houver informações de rede definidas.

`ClockDrift`

A informação sobre a diferença entre o tempo de referência e a hora do sistema. Isso se aplica ao sistema operacional Linux. Esse recurso usa o Serviço de Sincronização Temporal da Amazon para medir a precisão do relógio e fornecer o erro de relógio vinculado aos contêineres. Para obter mais informações, consulte Definir a hora da instância do Linux no Guia do usuário do Amazon EC2 para instâncias do Linux.

`ReferenceTime`

A base da precisão do relógio. O Amazon ECS usa o padrão global Coordinated Universal Time (UTC — Tempo universal coordenado) por meio do NTP, por exemplo, `2021-09-07T16:57:44Z`.

`ClockErrorBound`

A medida do erro do relógio, definida como o deslocamento para UTC. Esse erro é a diferença em milissegundos entre o tempo de referência e a hora do sistema.

`ClockSynchronizationStatus`

Indica se a tentativa de sincronização mais recente entre a hora do sistema e o horário de referência obteve êxito.

Os valores válidos são `SYNCHRONIZED` e `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

O time stamp de quando o `DesiredStatus` da tarefa mudou para `STOPPED`. Isso ocorre quando um contêiner essencial muda para `STOPPED`.

Exemplos de metadados de tarefas v3 no Amazon ECS

Os exemplos a seguir mostram saídas de exemplo dos endpoints dos metadados da tarefa.

Exemplo de resposta de metadados do contêiner

Ao consultar o endpoint `${ECS_CONTAINER_METADATA_URI}`, são retornados apenas metadados sobre o próprio contêiner. Veja a seguir um exemplo de saída.


```
{
    "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
    "Name": "nginx-curl",
    "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
    "Image": "nrdlngr/nginx-curl",
    "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
    "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
    },
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "Limits": {
        "CPU": 512,
        "Memory": 512
    },
    "CreatedAt": "2018-02-01T20:55:10.554941919Z",
    "StartedAt": "2018-02-01T20:55:11.064236631Z",
    "Type": "NORMAL",
    "Networks": [
        {
            "NetworkMode": "awsvpc",
            "IPv4Addresses": [
                "10.0.2.106"
            ]
        }
    ]
}
```


Exemplo de resposta de metadados de tarefas

Ao consultar o endpoint `${ECS_CONTAINER_METADATA_URI}/task`, são retornados metadados sobre a tarefa da qual o contêiner faz parte. Veja a seguir um exemplo de saída.

A seguinte resposta em JSON apresenta uma tarefa de contêiner único.


```
{
  "Cluster": "default",
  "TaskARN": "arn:aws:ecs:us-east-2:012345678910:task/9781c248-0edd-4cdb-9a93-
f63cb662a5d3",
  "Family": "nginx",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Containers": [
    {
      "DockerId": "731a0d6a3b4210e2448339bc7015aaa79bfe4fa256384f4102db86ef94cbbc4c",
      "Name": "~internal~ecs~pause",
      "DockerName": "ecs-nginx-5-internalecspause-acc699c0cbf2d6d11700",
      "Image": "amazon/amazon-ecs-pause:0.1.0",
      "ImageID": "",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RESOURCES_PROVISIONED",
      "KnownStatus": "RESOURCES_PROVISIONED",
      "Limits": {
        "CPU": 0,
        "Memory": 0
      },
      "CreatedAt": "2018-02-01T20:55:08.366329616Z",
      "StartedAt": "2018-02-01T20:55:09.058354915Z",
      "Type": "CNI_PAUSE",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    },
    {
```

```
      "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
      "Name": "nginx-curl",
      "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
      "Image": "nrdlngr/nginx-curl",
      "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": {
        "CPU": 512,
        "Memory": 512
      },
      "CreatedAt": "2018-02-01T20:55:10.554941919Z",
      "StartedAt": "2018-02-01T20:55:11.064236631Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    }
  ],
  "PullStartedAt": "2018-02-01T20:55:09.372495529Z",
  "PullStoppedAt": "2018-02-01T20:55:10.552018345Z",
  "AvailabilityZone": "us-east-2b"
}
```

Endpoint de metadados de tarefas do Amazon ECS versão 2

> **Important**
>
> O endpoint da versão 2 dos metadados da tarefa não está mais sendo mantido ativamente. Recomendamos que você atualize o endpoint versão 4 de metadados da tarefa para obter as informações mais recentes do endpoint de metadados. Para obter mais informações, consulte the section called “Endpoint de metadados de tarefas versão 4”.

A partir da versão 1.17.0 do agente de contêiner do Amazon ECS, vários metadados de tarefas e dados estatísticos do Docker estão disponíveis para tarefas que usam o modo de rede `awsvpc` em um endpoint HTTP fornecido pelo agente de contêiner do Amazon ECS.

Todos os contêineres que pertencem a tarefas que são executadas no modo de rede `awsvpc` recebem um endereço IPv4 local dentro de um intervalo pré-definido de endereços locais de link. Quando um contêiner consulta o endpoint de metadados, o agente de contêiner do Amazon ECS pode determinar a qual tarefa o contêiner pertence com base no seu endereço IP exclusivo, e são retornados metadados e dados estatísticos dessa tarefa.

Habilitar metadados de tarefas

O recurso dos metadados de tarefas versão 2 é ativado por padrão para o seguinte:

- Tarefas que usam o tipo de inicialização do Fargate e a versão v1.1.0 ou posterior da plataforma. Para obter mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.
- Tarefas que usam o tipo de inicialização do EC2 que também usam o modo de rede `awsvpc` são iniciadas na infraestrutura do Linux do Amazon EC2 executando pelo menos a versão 1.17.0 do agente de contêiner do Amazon ECS ou na infraestrutura do Windows do Amazon EC2 executando pelo menos a versão `1.54.0` do agente de contêiner do Amazon ECS. Para obter mais informações, consulte Gerenciamento de instâncias de contêiner do Linux no Amazon ECS.

É possível adicionar suporte a esse recurso nas instâncias de contêiner mais antigas atualizando o agente para a versão mais recente. Para obter mais informações, consulte Atualizar o agente de contêiner do Amazon ECS.

Caminhos do endpoint de metadados de tarefas

Os seguintes endpoints de API estão disponíveis para os contêineres:

`169.254.170.2/v2/metadata`

Esse endpoint retorna o JSON de metadados para a tarefa, incluindo uma lista dos nomes e IDs de todos os contêineres associados à tarefa. Para obter mais informações sobre a resposta para esse endpoint, consulte Resposta do JSON de metadados de tarefas.

`169.254.170.2/v2/metadata/<container-id>`

Esse endpoint retorna o JSON de metadados para o ID de contêiner do Docker especificado.

`169.254.170.2/v2/metadata/taskWithTags`

Esse caminho retorna os metadados para a tarefa incluída no endpoint `/task` além das etiquetas de tarefa e de instância de contêiner que podem ser recuperadas usando a API `ListTagsForResource`.

`169.254.170.2/v2/stats`

Esse endpoint retorna o JSON de estatísticas do Docker para todos os contêineres associados à tarefa. Para obter mais informações sobre cada uma das estatísticas retornadas, consulte ContainerStats na documentação da API do Docker.

`169.254.170.2/v2/stats/<container-id>`

Esse endpoint retorna o JSON de estatísticas do Docker para o ID de contêiner do Docker especificado. Para obter mais informações sobre cada uma das estatísticas retornadas, consulte ContainerStats na documentação da API do Docker.

Resposta do JSON de metadados de tarefas

As seguintes informações são retornadas da resposta em JSON (`169.254.170.2/v2/metadata`) do endpoint de metadados de tarefas.

`Cluster`

O nome do recurso da Amazon (ARN) ou nome curto do cluster do Amazon ECS ao qual a tarefa pertence.

`TaskARN`

O nome do recurso da Amazon (ARN) da tarefa à qual o contêiner pertence.

`Family`

A família da definição de tarefa do Amazon ECS para a tarefa.

`Revision`

A revisão da definição de tarefa do Amazon ECS para a tarefa.

`DesiredStatus`

O status desejado para a tarefa do Amazon ECS.

`KnownStatus`

O status conhecido para a tarefa do Amazon ECS.

`Limits`

Os limites de recursos especificados no nível da tarefa, como CPU (expresso em vCPUs) e memória. Esse parâmetro será omitido se não houver nenhum limite de recurso definido.

`PullStartedAt`

O timestamp de quando começou a primeira extração de imagem do contêiner.

`PullStoppedAt`

O timestamp de quando a última extração de imagem do contêiner terminou.

`AvailabilityZone`

A zona de disponibilidade em que a tarefa está.

> **Note**
>
> Os metadados da zona de disponibilidade estão disponíveis apenas para tarefas do Fargate que usam a versão 1.4 ou posterior (Linux) ou 1.0.0 ou posterior (Windows) da plataforma.

`Containers`

Uma lista de metadados de contêiner para cada contêiner associado com a tarefa.

`DockerId`

O ID do Docker do contêiner.

`Name`

O nome do contêiner, conforme especificado na definição da tarefa.

`DockerName`

O nome do contêiner fornecido para o Docker. O agente de contêiner do Amazon ECS gera um nome exclusivo para o contêiner para evitar colisões de nomes quando várias cópias da mesma definição de tarefa são executadas em uma única instância.

`Image`

A imagem para o contêiner.

`ImageID`

O resumo SHA-256 do manifesto da imagem. Esse é o resumo que pode ser usado para extrair a imagem usando o formato `repository-url/image@sha256:digest`.

`Ports`

Todas as portas expostas para o contêiner. Esse parâmetro será omitido se não houver portas expostas.

`Labels`

Todos os rótulos aplicados ao contêiner. Esse parâmetro será omitido se não houver rótulos aplicados.

`DesiredStatus`

O status desejado para o contêiner do Amazon ECS.

`KnownStatus`

O status conhecido para o contêiner do Amazon ECS.

`ExitCode`

O código de saída para o contêiner. Esse parâmetro é omitido se o contêiner não foi encerrado.

`Limits`

Os limites de recursos especificados no nível do contêiner, como CPU (expresso em unidades de CPU) e memória. Esse parâmetro será omitido se não houver nenhum limite de recurso definido.

`CreatedAt`

O time stamp de quando o contêiner foi criado. Esse parâmetro será omitido se o contêiner ainda não tiver sido criado.

`StartedAt`

O time stamp de quando o contêiner foi iniciado. Esse parâmetro será omitido se o contêiner ainda não tiver sido iniciado.

`FinishedAt`

O time stamp de quando o contêiner foi interrompido. Esse parâmetro será omitido se o contêiner ainda não tiver sido interrompido.

`Type`

O tipo do contêiner. Os contêineres que são especificados em sua definição de tarefa são do tipo `NORMAL`. É possível ignorar outros tipos de contêineres, que são usados para o provisionamento de recursos de tarefas internas pelo agente de contêiner do Amazon ECS.

`Networks`

As informações de rede para o contêiner, como o modo de rede e o endereço IP. Esse parâmetro será omitido se não houver informações de rede definidas.

`ClockDrift`

A informação sobre a diferença entre o tempo de referência e a hora do sistema. Isso se aplica ao sistema operacional Linux. Esse recurso usa o Serviço de Sincronização Temporal da Amazon para medir a precisão do relógio e fornecer o erro de relógio vinculado aos contêineres. Para obter mais informações, consulte Definir a hora da instância do Linux no Guia do usuário do Amazon EC2 para instâncias do Linux.

`ReferenceTime`

A base da precisão do relógio. O Amazon ECS usa o padrão global Coordinated Universal Time (UTC — Tempo universal coordenado) por meio do NTP, por exemplo, `2021-09-07T16:57:44Z`.

`ClockErrorBound`

A medida do erro do relógio, definida como o deslocamento para UTC. Esse erro é a diferença em milissegundos entre o tempo de referência e a hora do sistema.

`ClockSynchronizationStatus`

Indica se a tentativa de sincronização mais recente entre a hora do sistema e o horário de referência obteve êxito.

Os valores válidos são `SYNCHRONIZED` e `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

O time stamp de quando o `DesiredStatus` da tarefa mudou para `STOPPED`. Isso ocorre quando um contêiner essencial muda para `STOPPED`.

Exemplo de resposta de metadados de tarefas

A seguinte resposta em JSON apresenta uma tarefa de contêiner único.

```
{
  "Cluster": "default",
  "TaskARN": "arn:aws:ecs:us-east-2:012345678910:task/9781c248-0edd-4cdb-9a93-
f63cb662a5d3",
  "Family": "nginx",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Containers": [
    {
      "DockerId": "731a0d6a3b4210e2448339bc7015aaa79bfe4fa256384f4102db86ef94cbbc4c",
      "Name": "~internal~ecs~pause",
      "DockerName": "ecs-nginx-5-internalecspause-acc699c0cbf2d6d11700",
      "Image": "amazon/amazon-ecs-pause:0.1.0",
      "ImageID": "",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RESOURCES_PROVISIONED",
      "KnownStatus": "RESOURCES_PROVISIONED",
      "Limits": {
        "CPU": 0,
        "Memory": 0
      },
      "CreatedAt": "2018-02-01T20:55:08.366329616Z",
      "StartedAt": "2018-02-01T20:55:09.058354915Z",
      "Type": "CNI_PAUSE",
      "Networks": [
        {
```

```
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    },
    {
      "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
      "Name": "nginx-curl",
      "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
      "Image": "nrdlngr/nginx-curl",
      "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": {
        "CPU": 512,
        "Memory": 512
      },
      "CreatedAt": "2018-02-01T20:55:10.554941919Z",
      "StartedAt": "2018-02-01T20:55:11.064236631Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    }
  ],
  "PullStartedAt": "2018-02-01T20:55:09.372495529Z",
  "PullStoppedAt": "2018-02-01T20:55:10.552018345Z",
  "AvailabilityZone": "us-east-2b"
```

```
}
```

Metadados de tarefas do Amazon ECS disponíveis para tarefas no Fargate

O Amazon ECS no Fargate fornece um método para recuperar vários metadados, métricas de rede e dados estatísticos do Docker sobre os contêineres e as tarefas das quais eles fazem parte.. Isso é denominado endpoint de metadados de tarefas. As seguintes versões de endpoint de metadados de tarefas estão disponíveis para tarefas do Amazon ECS e do Fargate:

- Endpoint de metadados de tarefas versão 4: disponível para tarefas que usam a versão de plataforma 1.4.0 ou posterior.
- Endpoint de metadados de tarefas versão 3: disponível para tarefas que usam a versão de plataforma 1.1.0 ou posterior.

Todos os contêineres que pertencem a tarefas que são executadas no modo de rede `awsvpc` recebem um endereço IPv4 local dentro de um intervalo pré-definido de endereços locais de link. Quando um contêiner consulta o endpoint de metadados, o agente de contêiner pode determinar a qual tarefa o contêiner pertence com base em seu endereço IP exclusivo, e os metadados e as estatísticas dessa tarefa são retornados.

Tópicos

- Endpoint de metadados de tarefas do Amazon ECS versão 4 para tarefas no Fargate
- Endpoint de metadados de tarefas do Amazon ECS versão 3 para tarefas no Fargate

Endpoint de metadados de tarefas do Amazon ECS versão 4 para tarefas no Fargate

> **Important**
>
> Se estiver usando tarefas do Amazon ECS hospedadas em instâncias do Amazon EC2, consulte Amazon ECS task metadata endpoint.

Começando com a versão `1.4.0` da plataforma do Fargate, uma variável de ambiente denominada `ECS_CONTAINER_METADATA_URI_V4` é injetada em cada contêiner de uma tarefa. Quando você consulta o endpoint de metadados de tarefas versão 4, vários metadados de tarefas e dados estatísticos do Docker estão disponíveis para tarefas.

O endpoint de metadados de tarefas versão 4 funciona como o endpoint versão 3, mas fornece metadados de rede adicionais para contêineres e tarefas. Métricas de rede adicionais também estão disponíveis ao consultar os endpoints `/stats`.

O endpoint de metadados de tarefa é ativado por padrão para todas as tarefas do Amazon ECS executadas no AWS Fargate que usam a versão `1.4.0` ou posterior da plataforma.

> **Note**
>
> Para evitar a necessidade de criar novas versões de endpoint de metadados de tarefas no futuro, os metadados adicionais podem ser adicionados à saída da versão 4. Não removeremos quaisquer metadados existentes nem alteraremos os nomes dos campos de metadados.

Caminhos do endpoint de metadados de tarefas do Fargate versão 4

Os seguintes endpoints de metadados de tarefas estão disponíveis para os contêineres:

`${ECS_CONTAINER_METADATA_URI_V4}`

Esse caminho retorna metadados para o contêiner.

`${ECS_CONTAINER_METADATA_URI_V4}/task`

Esse caminho retorna metadados para a tarefa, incluindo uma lista dos nomes e IDs de todos os contêineres associados à tarefa. Para obter mais informações sobre a resposta para esse endpoint, consulte Resposta JSON para metadados de tarefas v4 no Amazon ECS para tarefas no Fargate.

`${ECS_CONTAINER_METADATA_URI_V4}/stats`

Esse caminho retorna dados estatísticos do Docker para o contêiner do Docker. Para obter mais informações sobre cada uma das estatísticas retornadas, consulte ContainerStats na documentação da API do Docker.

> **Note**
>
> Tarefas do Amazon ECS no AWS Fargate exigem que o contêiner seja executado por ~1 segundo antes de retornar os dados estatísticos do contêiner.

`${ECS_CONTAINER_METADATA_URI_V4}/task/stats`

Esse caminho retorna os dados estatísticos do Docker para todos os contêineres associados à tarefa. Para obter mais informações sobre cada uma das estatísticas retornadas, consulte ContainerStats na documentação da API do Docker.

> **Note**
>
> Tarefas do Amazon ECS no AWS Fargate exigem que o contêiner seja executado por ~1 segundo antes de retornar os dados estatísticos do contêiner.

Resposta JSON para metadados de tarefas v4 no Amazon ECS para tarefas no Fargate

Os seguintes metadados são retornados na resposta em formato JSON (`${ECS_CONTAINER_METADATA_URI_V4}/task`) do endpoint de metadados da tarefa.

`Cluster`

O nome do recurso da Amazon (ARN) ou nome curto do cluster do Amazon ECS ao qual a tarefa pertence.

`VPCID`

O ID da VPC da instância de contêiner do Amazon EC2. Esse campo é exibido somente para instâncias do Amazon EC2.

> **Note**
>
> Os metadados `VPCID` são incluídos somente quando é usada a versão `1.63.1` ou posterior do agente de contêiner do Amazon ECS.

`TaskARN`

O nome do recurso da Amazon (ARN) da tarefa à qual o contêiner pertence.

`Family`

A família da definição de tarefa do Amazon ECS para a tarefa.

`Revision`

A revisão da definição de tarefa do Amazon ECS para a tarefa.

`DesiredStatus`

O status desejado para a tarefa do Amazon ECS.

`KnownStatus`

O status conhecido para a tarefa do Amazon ECS.

`Limits`

Os limites de recursos especificados nos níveis da tarefa, como CPU (expresso em vCPUs) e memória. Esse parâmetro será omitido se não houver nenhum limite de recurso definido.

`PullStartedAt`

O timestamp de quando começou a primeira extração de imagem do contêiner.

`PullStoppedAt`

O timestamp de quando a última extração de imagem do contêiner terminou.

`AvailabilityZone`

A zona de disponibilidade em que a tarefa está.

> **Note**
>
> Os metadados da zona de disponibilidade estão disponíveis apenas para tarefas do Fargate que usam a versão 1.4 ou posterior (Linux) ou 1.0.0 (Windows) da plataforma.

`LaunchType`

O tipo de inicialização que a tarefa está usando. Ao usar provedores de capacidade de cluster, isso indica se a tarefa está usando a infraestrutura do Fargate ou do EC2.

> **Note**
>
> Esses metadados `LaunchType` estão incluídos somente quando é usada a versão `1.45.0` ou posterior (Linux) ou 1.0.0 ou posterior (Windows) do agente de contêiner do Linux do Amazon ECS.

`EphemeralStorageMetrics`

O tamanho reservado e o uso atual do armazenamento temporário dessa tarefa.

> **Note**
>
> O Fargate reserva espaço no disco. Esse espaço é usado apenas pelo Fargate. Você não é cobrado por isso. Ele não é mostrado nessas métricas. Porém, você pode ver esse armazenamento adicional em outras ferramentas, como o `df`.

`Utilized`

O uso atual de armazenamento temporário (em MiB) dessa tarefa.

`Reserved`

O armazenamento temporário reservado (em MiB) dessa tarefa. O tamanho do armazenamento temporário não pode ser alterado em uma tarefa em execução. É possível especificar o objeto `ephermalStorage` na sua definição de tarefa para alterar a quantidade de armazenamento temporário. O `ephermalStorage` é especificado em GiB, não em MiB. O `ephermalStorage` e as `EphemeralStorageMetrics` só estão disponíveis na versão 1.4.0 ou posterior da plataforma Linux do Fargate.

`Containers`

Uma lista de metadados de contêiner para cada contêiner associado com a tarefa.

`DockerId`

O ID do Docker do contêiner.

Quando você usa Fargate, o id é um hexadecimal de 32 dígitos seguido por um número de 10 dígitos.

`Name`

O nome do contêiner, conforme especificado na definição da tarefa.

`DockerName`

O nome do contêiner fornecido para o Docker. O agente de contêiner do Amazon ECS gera um nome exclusivo para o contêiner para evitar colisões de nomes quando várias cópias da mesma definição de tarefa são executadas em uma única instância.

`Image`

A imagem para o contêiner.

`ImageID`

O resumo SHA-256 do manifesto da imagem. Esse é o resumo que pode ser usado para extrair a imagem usando o formato `repository-url/image@sha256:digest`.

`Ports`

Todas as portas expostas para o contêiner. Esse parâmetro será omitido se não houver portas expostas.

`Labels`

Todos os rótulos aplicados ao contêiner. Esse parâmetro será omitido se não houver rótulos aplicados.

`DesiredStatus`

O status desejado para o contêiner do Amazon ECS.

`KnownStatus`

O status conhecido para o contêiner do Amazon ECS.

`ExitCode`

O código de saída para o contêiner. Esse parâmetro é omitido se o contêiner não foi encerrado.

`Limits`

Os limites de recursos especificados no nível do contêiner, como CPU (expresso em unidades de CPU) e memória. Esse parâmetro será omitido se não houver nenhum limite de recurso definido.

`CreatedAt`

O time stamp de quando o contêiner foi criado. Esse parâmetro será omitido se o contêiner ainda não tiver sido criado.

`StartedAt`

O time stamp de quando o contêiner foi iniciado. Esse parâmetro será omitido se o contêiner ainda não tiver sido iniciado.

`FinishedAt`

O time stamp de quando o contêiner foi interrompido. Esse parâmetro será omitido se o contêiner ainda não tiver sido interrompido.

`Type`

O tipo do contêiner. Os contêineres que são especificados em sua definição de tarefa são do tipo `NORMAL`. É possível ignorar outros tipos de contêineres, que são usados para o provisionamento de recursos de tarefas internas pelo agente de contêiner do Amazon ECS.

`LogDriver`

O driver de log que o contêiner está usando.

> **Note**
>
> Esses metadados `LogDriver` estão incluídos somente quando é usada a versão `1.45.0` ou posterior do agente de contêiner do Linux do Amazon ECS.

`LogOptions`

As opções de driver de log definidas para o contêiner.

> **Note**
>
> Esses metadados `LogOptions` estão incluídos somente quando é usada a versão `1.45.0` ou posterior do agente de contêiner do Linux do Amazon ECS.

`ContainerARN`

O nome do recurso da Amazon (ARN) do contêiner.

> **Note**
>
> Esses metadados `ContainerARN` estão incluídos somente quando é usada a versão `1.45.0` ou posterior do agente de contêiner do Linux do Amazon ECS.

`Networks`

As informações de rede para o contêiner, como o modo de rede e o endereço IP. Esse parâmetro será omitido se não houver informações de rede definidas.

`Snapshotter`

O snapshotter que foi usado pelo containerd para baixar essa imagem do contêiner. Os valores válidos são `overlayfs`, que é o padrão, e `soci`, usado no carregamento lento com um índice SOCI. Este parâmetro só está disponível para tarefas executadas na versão da plataforma Linux `1.4.0`.

`RestartCount`

O número de vezes que um contêiner foi reiniciado.

> **Note**
>
> Os metadados de `RestartCount` são incluídos somente se uma política de reinicialização estiver habilitada para o contêiner. Para obter mais informações, consulte Reiniciar contêineres individuais em tarefas do Amazon ECS com políticas de reinicialização de contêineres.

`ClockDrift`

A informação sobre a diferença entre o tempo de referência e a hora do sistema. Esse recurso usa o Serviço de Sincronização Temporal da Amazon para medir a precisão do relógio e fornecer o erro de relógio vinculado aos contêineres. Para obter mais informações, consulte Definir a hora da instância do Linux no Guia do usuário do Amazon EC2 para instâncias do Linux.

`ReferenceTime`

A base da precisão do relógio. O Amazon ECS usa o padrão global Coordinated Universal Time (UTC — Tempo universal coordenado) por meio do NTP, por exemplo, `2021-09-07T16:57:44Z`.

`ClockErrorBound`

A medida do erro do relógio, definida como o deslocamento para UTC. Esse erro é a diferença em milissegundos entre o tempo de referência e a hora do sistema.

`ClockSynchronizationStatus`

Indica se a tentativa de sincronização mais recente entre a hora do sistema e o horário de referência obteve êxito.

Os valores válidos são `SYNCHRONIZED` e `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

O time stamp de quando o `DesiredStatus` da tarefa mudou para `STOPPED`. Isso ocorre quando um contêiner essencial muda para `STOPPED`.

Exemplos de metadados de tarefas v4 no Amazon ECS para tarefas no Fargate

A seguir, são mostrados exemplos de saídas dos endpoints de metadados de tarefas do Amazon ECS executadas no AWS Fargate.

No contêiner, você pode usar o curl seguido pelo endpoint de metadados da tarefa para consultar o endpoint, por exemplo, `curl ${ECS_CONTAINER_METADATA_URI_V4}/task`.

Exemplo de resposta de metadados de contêineres

Ao consultar o endpoint `${ECS_CONTAINER_METADATA_URI_V4}`, são retornados apenas metadados sobre o próprio contêiner. Veja a seguir um exemplo de saída.

```
{
    "DockerId": "cd189a933e5849daa93386466019ab50-2495160603",
    "Name": "curl",
    "DockerName": "curl",
    "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
    "ImageID":
 "sha256:25f3695bedfb454a50f12d127839a68ad3caf91e451c1da073db34c542c4d2cb",
    "Labels": {
        "com.amazonaws.ecs.cluster": "arn:aws:ecs:us-west-2:111122223333:cluster/
default",
        "com.amazonaws.ecs.container-name": "curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/default/
cd189a933e5849daa93386466019ab50",
        "com.amazonaws.ecs.task-definition-family": "curltest",
        "com.amazonaws.ecs.task-definition-version": "2"
    },
    "DesiredStatus": "RUNNING",
```

```
    "KnownStatus": "RUNNING",
    "Limits": {
        "CPU": 10,
        "Memory": 128
    },
    "CreatedAt": "2020-10-08T20:09:11.44527186Z",
    "StartedAt": "2020-10-08T20:09:11.44527186Z",
    "Type": "NORMAL",
    "Networks": [
        {
            "NetworkMode": "awsvpc",
            "IPv4Addresses": [
                "192.0.2.3"
            ],
            "AttachmentIndex": 0,
            "MACAddress": "0a:de:f6:10:51:e5",
            "IPv4SubnetCIDRBlock": "192.0.2.0/24",
            "DomainNameServers": [
                "192.0.2.2"
            ],
            "DomainNameSearchList": [
                "us-west-2.compute.internal"
            ],
            "PrivateDNSName": "ip-10-0-0-222.us-west-2.compute.internal",
            "SubnetGatewayIpv4Address": "192.0.2.0/24"
        }
    ],
    "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/05966557-
f16c-49cb-9352-24b3a0dcd0e1",
    "LogOptions": {
        "awslogs-create-group": "true",
        "awslogs-group": "/ecs/containerlogs",
        "awslogs-region": "us-west-2",
        "awslogs-stream": "ecs/curl/cd189a933e5849daa93386466019ab50"
    },
    "LogDriver": "awslogs",
    "Snapshotter": "overlayfs"
}
```


Exemplos de metadados de tarefas v4 no Amazon ECS para tarefas no Fargate

Ao consultar o endpoint `${ECS_CONTAINER_METADATA_URI_V4}/task`, são retornados metadados sobre a tarefa da qual o contêiner faz parte. Veja a seguir um exemplo de saída.

```
{
  "Cluster": "arn:aws:ecs:us-east-1:123456789012:cluster/MyEmptyCluster",
  "TaskARN": "arn:aws:ecs:us-east-1:123456789012:task/MyEmptyCluster/
bfa2636268144d039771334145e490c5",
  "Family": "sample-fargate",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Limits": {
    "CPU": 0.25,
    "Memory": 512
  },
  "PullStartedAt": "2023-07-21T15:45:33.532811081Z",
  "PullStoppedAt": "2023-07-21T15:45:38.541068435Z",
  "AvailabilityZone": "us-east-1d",
  "Containers": [
    {
      "DockerId": "bfa2636268144d039771334145e490c5-1117626119",
      "Name": "curl-image",
      "DockerName": "curl-image",
      "Image": "curlimages/curl",
      "ImageID":
 "sha256:daf3f46a2639c1613b25e85c9ee4193af8a1d538f92483d67f9a3d7f21721827",
      "Labels": {
        "com.amazonaws.ecs.cluster": "arn:aws:ecs:us-east-1:123456789012:cluster/
MyEmptyCluster",
        "com.amazonaws.ecs.container-name": "curl-image",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-east-1:123456789012:task/
MyEmptyCluster/bfa2636268144d039771334145e490c5",
        "com.amazonaws.ecs.task-definition-family": "sample-fargate",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": { "CPU": 128 },
      "CreatedAt": "2023-07-21T15:45:44.91368314Z",
      "StartedAt": "2023-07-21T15:45:44.91368314Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": ["172.31.42.189"],
          "AttachmentIndex": 0,
```

```
          "MACAddress": "0e:98:9f:33:76:d3",
          "IPv4SubnetCIDRBlock": "172.31.32.0/20",
          "DomainNameServers": ["172.31.0.2"],
          "DomainNameSearchList": ["ec2.internal"],
          "PrivateDNSName": "ip-172-31-42-189.ec2.internal",
          "SubnetGatewayIpv4Address": "172.31.32.1/20"
        }
      ],
      "ContainerARN": "arn:aws:ecs:us-east-1:123456789012:container/MyEmptyCluster/
bfa2636268144d039771334145e490c5/da6cccf7-1178-400c-afdf-7536173ee209",
      "Snapshotter": "overlayfs"
    },
    {
      "DockerId": "bfa2636268144d039771334145e490c5-3681984407",
      "Name": "fargate-app",
      "DockerName": "fargate-app",
      "Image": "public.ecr.aws/docker/library/httpd:latest",
      "ImageID":
 "sha256:8059bdd0058510c03ae4c808de8c4fd2c1f3c1b6d9ea75487f1e5caa5ececa02",
      "Labels": {
        "com.amazonaws.ecs.cluster": "arn:aws:ecs:us-east-1:123456789012:cluster/
MyEmptyCluster",
        "com.amazonaws.ecs.container-name": "fargate-app",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-east-1:123456789012:task/
MyEmptyCluster/bfa2636268144d039771334145e490c5",
        "com.amazonaws.ecs.task-definition-family": "sample-fargate",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": { "CPU": 2 },
      "CreatedAt": "2023-07-21T15:45:44.954460255Z",
      "StartedAt": "2023-07-21T15:45:44.954460255Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": ["172.31.42.189"],
          "AttachmentIndex": 0,
          "MACAddress": "0e:98:9f:33:76:d3",
          "IPv4SubnetCIDRBlock": "172.31.32.0/20",
          "DomainNameServers": ["172.31.0.2"],
          "DomainNameSearchList": ["ec2.internal"],
          "PrivateDNSName": "ip-172-31-42-189.ec2.internal",
```

```
          "SubnetGatewayIpv4Address": "172.31.32.1/20"
        }
      ],
      "ContainerARN": "arn:aws:ecs:us-east-1:123456789012:container/MyEmptyCluster/
bfa2636268144d039771334145e490c5/f65b461d-aa09-4acb-a579-9785c0530cbc",
      "Snapshotter": "overlayfs"
    }
  ],
  "LaunchType": "FARGATE",
  "ClockDrift": {
    "ClockErrorBound": 0.446931,
    "ReferenceTimestamp": "2023-07-21T16:09:17Z",
    "ClockSynchronizationStatus": "SYNCHRONIZED"
  },
  "EphemeralStorageMetrics": {
    "Utilized": 261,
    "Reserved": 20496
  }
}
```

Exemplo de resposta de dados estatísticos de tarefas

Ao consultar o endpoint `${ECS_CONTAINER_METADATA_URI_V4}/task/stats`, são retornadas métricas de rede sobre a tarefa da qual o contêiner faz parte. Veja a seguir um exemplo de saída.

```
{
  "3d1f891cded94dc795608466cce8ddcf-464223573": {
    "read": "2020-10-08T21:24:44.938937019Z",
    "preread": "2020-10-08T21:24:34.938633969Z",
    "pids_stats": {},
    "blkio_stats": {
      "io_service_bytes_recursive": [
        {
          "major": 202,
          "minor": 26368,
          "op": "Read",
          "value": 638976
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Write",
          "value": 0
```

```
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Sync",
          "value": 638976
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Async",
          "value": 0
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Total",
          "value": 638976
        }
      ],
      "io_serviced_recursive": [
        {
          "major": 202,
          "minor": 26368,
          "op": "Read",
          "value": 12
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Write",
          "value": 0
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Sync",
          "value": 12
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Async",
          "value": 0
```

```
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Total",
          "value": 12
        }
      ],
      "io_queue_recursive": [],
      "io_service_time_recursive": [],
      "io_wait_time_recursive": [],
      "io_merged_recursive": [],
      "io_time_recursive": [],
      "sectors_recursive": []
    },
    "num_procs": 0,
    "storage_stats": {},
    "cpu_stats": {
      "cpu_usage": {
        "total_usage": 1137691504,
        "percpu_usage": [
          696479228,
          441212276,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0
        ],
        "usage_in_kernelmode": 80000000,
        "usage_in_usermode": 810000000
      },
      "system_cpu_usage": 9393210000000,
      "online_cpus": 2,
      "throttling_data": {
        "periods": 0,
```

```
        "throttled_periods": 0,
        "throttled_time": 0
      }
    },
    "precpu_stats": {
      "cpu_usage": {
        "total_usage": 1136624601,
        "percpu_usage": [
          695639662,
          440984939,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0
        ],
        "usage_in_kernelmode": 80000000,
        "usage_in_usermode": 810000000
      },
      "system_cpu_usage": 9373330000000,
      "online_cpus": 2,
      "throttling_data": {
        "periods": 0,
        "throttled_periods": 0,
        "throttled_time": 0
      }
    },
    "memory_stats": {
      "usage": 6504448,
      "max_usage": 8458240,
      "stats": {
        "active_anon": 1675264,
        "active_file": 557056,
        "cache": 651264,
        "dirty": 0,
        "hierarchical_memory_limit": 536870912,
```

```
        "hierarchical_memsw_limit": 9223372036854772000,
        "inactive_anon": 0,
        "inactive_file": 3088384,
        "mapped_file": 430080,
        "pgfault": 11034,
        "pgmajfault": 5,
        "pgpgin": 8436,
        "pgpgout": 7137,
        "rss": 4669440,
        "rss_huge": 0,
        "total_active_anon": 1675264,
        "total_active_file": 557056,
        "total_cache": 651264,
        "total_dirty": 0,
        "total_inactive_anon": 0,
        "total_inactive_file": 3088384,
        "total_mapped_file": 430080,
        "total_pgfault": 11034,
        "total_pgmajfault": 5,
        "total_pgpgin": 8436,
        "total_pgpgout": 7137,
        "total_rss": 4669440,
        "total_rss_huge": 0,
        "total_unevictable": 0,
        "total_writeback": 0,
        "unevictable": 0,
        "writeback": 0
      },
      "limit": 9223372036854772000
    },
    "name": "curltest",
    "id": "3d1f891cded94dc795608466cce8ddcf-464223573",
    "networks": {
      "eth1": {
        "rx_bytes": 2398415937,
        "rx_packets": 1898631,
        "rx_errors": 0,
        "rx_dropped": 0,
        "tx_bytes": 1259037719,
        "tx_packets": 428002,
        "tx_errors": 0,
        "tx_dropped": 0
      }
    },
```

```
    "network_rate_stats": {
      "rx_bytes_per_sec": 43.298687872232854,
      "tx_bytes_per_sec": 215.39347269466413
    }
  }
}
```

Endpoint de metadados de tarefas do Amazon ECS versão 3 para tarefas no Fargate

> **Important**
>
> O endpoint da versão 3 dos metadados da tarefa não está mais sendo mantido ativamente. Recomendamos que você atualize o endpoint versão 4 de metadados da tarefa para obter as informações mais recentes do endpoint de metadados. Para obter mais informações, consulte the section called “Endpoint de metadados de tarefas versão 4 para tarefas no Fargate”.

Começando com a versão `1.1.0` da plataforma do Fargate, uma variável de ambiente denominada `ECS_CONTAINER_METADATA_URI` é injetada em cada contêiner de uma tarefa. Quando você consultar o endpoint de metadados de tarefas versão 3, vários metadados de tarefas e estatísticas do Docker estarão disponíveis para tarefas.

O endpoint de metadados de tarefas é habilitado por padrão para todas as tarefas do Amazon ECS hospedadas no Fargate que usam a versão `1.1.0` ou posterior da plataforma. Para obter mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.

Caminhos de endpoint de metadados de tarefas para tarefas no Fargate

Os seguintes endpoints de API estão disponíveis para os contêineres:

`${ECS_CONTAINER_METADATA_URI}`

Esse caminho retorna o JSON de metadados para o contêiner.

`${ECS_CONTAINER_METADATA_URI}/task`

Esse caminho retorna o JSON de metadados para a tarefa, incluindo uma lista dos nomes e IDs de todos os contêineres associados à tarefa. Para obter mais informações sobre a resposta para esse endpoint, consulte Resposta JSON para metadados de tarefas v3 no Amazon ECS para tarefas no Fargate.

`${ECS_CONTAINER_METADATA_URI}/stats`

Esse caminho retorna o JSON de estatísticas do Docker para o contêiner do Docker específico. Para obter mais informações sobre cada uma das estatísticas retornadas, consulte ContainerStats na documentação da API do Docker.

`${ECS_CONTAINER_METADATA_URI}/task/stats`

Esse caminho retorna o JSON de estatísticas do Docker para todos os contêineres associados à tarefa. Para obter mais informações sobre cada uma das estatísticas retornadas, consulte ContainerStats na documentação da API do Docker.

Resposta JSON para metadados de tarefas v3 no Amazon ECS para tarefas no Fargate

As seguintes informações são retornadas da resposta em JSON (`${ECS_CONTAINER_METADATA_URI}/task`) do endpoint de metadados de tarefas.

`Cluster`

O nome do recurso da Amazon (ARN) ou nome curto do cluster do Amazon ECS ao qual a tarefa pertence.

`TaskARN`

O nome do recurso da Amazon (ARN) da tarefa à qual o contêiner pertence.

`Family`

A família da definição de tarefa do Amazon ECS para a tarefa.

`Revision`

A revisão da definição de tarefa do Amazon ECS para a tarefa.

`DesiredStatus`

O status desejado para a tarefa do Amazon ECS.

`KnownStatus`

O status conhecido para a tarefa do Amazon ECS.

`Limits`

Os limites de recursos especificados no nível da tarefa, como CPU (expresso em vCPUs) e memória. Esse parâmetro será omitido se não houver nenhum limite de recurso definido.

`PullStartedAt`

O timestamp de quando começou a primeira extração de imagem do contêiner.

`PullStoppedAt`

O timestamp de quando a última extração de imagem do contêiner terminou.

`AvailabilityZone`

A zona de disponibilidade em que a tarefa está.

> **Note**
>
> Os metadados da zona de disponibilidade estão disponíveis apenas para tarefas do Fargate que usam a versão 1.4 ou posterior (Linux) ou 1.0.0 ou posterior (Windows) da plataforma.

`Containers`

Uma lista de metadados de contêiner para cada contêiner associado com a tarefa.

`DockerId`

O ID do Docker do contêiner.

`Name`

O nome do contêiner, conforme especificado na definição da tarefa.

`DockerName`

O nome do contêiner fornecido para o Docker. O agente de contêiner do Amazon ECS gera um nome exclusivo para o contêiner para evitar colisões de nomes quando várias cópias da mesma definição de tarefa são executadas em uma única instância.

`Image`

A imagem para o contêiner.

`ImageID`

O resumo SHA-256 do manifesto da imagem. Esse é o resumo que pode ser usado para extrair a imagem usando o formato `repository-url/image@sha256:digest`.

`Ports`

Todas as portas expostas para o contêiner. Esse parâmetro será omitido se não houver portas expostas.

`Labels`

Todos os rótulos aplicados ao contêiner. Esse parâmetro será omitido se não houver rótulos aplicados.

`DesiredStatus`

O status desejado para o contêiner do Amazon ECS.

`KnownStatus`

O status conhecido para o contêiner do Amazon ECS.

`ExitCode`

O código de saída para o contêiner. Esse parâmetro é omitido se o contêiner não foi encerrado.

`Limits`

Os limites de recursos especificados no nível do contêiner, como CPU (expresso em unidades de CPU) e memória. Esse parâmetro será omitido se não houver nenhum limite de recurso definido.

`CreatedAt`

O time stamp de quando o contêiner foi criado. Esse parâmetro será omitido se o contêiner ainda não tiver sido criado.

`StartedAt`

O time stamp de quando o contêiner foi iniciado. Esse parâmetro será omitido se o contêiner ainda não tiver sido iniciado.

`FinishedAt`

O time stamp de quando o contêiner foi interrompido. Esse parâmetro será omitido se o contêiner ainda não tiver sido interrompido.

`Type`

O tipo do contêiner. Os contêineres que são especificados em sua definição de tarefa são do tipo `NORMAL`. É possível ignorar outros tipos de contêineres, que são usados para o provisionamento de recursos de tarefas internas pelo agente de contêiner do Amazon ECS.

`Networks`

As informações de rede para o contêiner, como o modo de rede e o endereço IP. Esse parâmetro será omitido se não houver informações de rede definidas.

`ClockDrift`

A informação sobre a diferença entre o tempo de referência e a hora do sistema. Isso se aplica ao sistema operacional Linux. Esse recurso usa o Serviço de Sincronização Temporal da Amazon para medir a precisão do relógio e fornecer o erro de relógio vinculado aos contêineres. Para obter mais informações, consulte Definir a hora da instância do Linux no Guia do usuário do Amazon EC2 para instâncias do Linux.

`ReferenceTime`

A base da precisão do relógio. O Amazon ECS usa o padrão global Coordinated Universal Time (UTC — Tempo universal coordenado) por meio do NTP, por exemplo, `2021-09-07T16:57:44Z`.

`ClockErrorBound`

A medida do erro do relógio, definida como o deslocamento para UTC. Esse erro é a diferença em milissegundos entre o tempo de referência e a hora do sistema.

`ClockSynchronizationStatus`

Indica se a tentativa de sincronização mais recente entre a hora do sistema e o horário de referência obteve êxito.

Os valores válidos são `SYNCHRONIZED` e `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

O time stamp de quando o `DesiredStatus` da tarefa mudou para `STOPPED`. Isso ocorre quando um contêiner essencial muda para `STOPPED`.

Exemplos de metadados de tarefas v3 no Amazon ECS para tarefas no Fargate

A seguinte resposta em JSON apresenta uma tarefa de contêiner único.

```
{
  "Cluster": "default",
  "TaskARN": "arn:aws:ecs:us-east-2:012345678910:task/9781c248-0edd-4cdb-9a93-
f63cb662a5d3",
  "Family": "nginx",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Containers": [
    {
      "DockerId": "731a0d6a3b4210e2448339bc7015aaa79bfe4fa256384f4102db86ef94cbbc4c",
      "Name": "~internal~ecs~pause",
      "DockerName": "ecs-nginx-5-internalecspause-acc699c0cbf2d6d11700",
      "Image": "amazon/amazon-ecs-pause:0.1.0",
      "ImageID": "",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RESOURCES_PROVISIONED",
      "KnownStatus": "RESOURCES_PROVISIONED",
      "Limits": {
        "CPU": 0,
        "Memory": 0
      },
      "CreatedAt": "2018-02-01T20:55:08.366329616Z",
      "StartedAt": "2018-02-01T20:55:09.058354915Z",
      "Type": "CNI_PAUSE",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    },
    {
      "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
```

```
      "Name": "nginx-curl",
      "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
      "Image": "nrdlngr/nginx-curl",
      "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": {
        "CPU": 512,
        "Memory": 512
      },
      "CreatedAt": "2018-02-01T20:55:10.554941919Z",
      "StartedAt": "2018-02-01T20:55:11.064236631Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    }
  ],
  "PullStartedAt": "2018-02-01T20:55:09.372495529Z",
  "PullStoppedAt": "2018-02-01T20:55:10.552018345Z",
  "AvailabilityZone": "us-east-2b"
}
```


Introspecção de contêiner do Amazon ECS

O agente de contêiner do Amazon ECS fornece uma operação de API para reunir detalhes sobre a instância de contêiner na qual o agente está sendo executado e as tarefas associadas executadas nesta instância. É possível usar o comando curl na instância de contêiner para consultar o agente

de contêiner do Amazon ECS (porta 51678) e retornar metadados da instância de contêiner ou informações sobre tarefas.

> **Important**
>
> A instância de contêiner deve ter uma função do IAM que conceda acesso ao Amazon ECS para recuperar os metadados. Para obter mais informações, consulte Função do IAM de instância de contêiner do Amazon ECS.

Para exibir os metadados da instância de contêiner, faça login na instância de contêiner via SSH e execute o comando a seguir. Os metadados incluem o ID da instância de contêiner, o cluster do Amazon ECS no qual a instância de contêiner está registrada e as informações sobre versão do agente de contêiner do Amazon ECS.

```
curl -s http://localhost:51678/v1/metadata | python3 -mjson.tool
```

Saída:

```
{
    "Cluster": "cluster_name",
    "ContainerInstanceArn": "arn:aws:ecs:region:aws_account_id:container-
instance/cluster_name/container_instance_id",
    "Version": "Amazon ECS Agent - v1.30.0 (02ff320c)"
}
```

Para exibir informações sobre todas as tarefas que estão sendo executadas em uma instância de contêiner, efetue login na instância de contêiner via SSH e execute o comando a seguir:

```
curl http://localhost:51678/v1/tasks
```

Saída:

```
{
    "Tasks": [
        {
            "Arn": "arn:aws:ecs:us-west-2:012345678910:task/default/example5-58ff-46c9-
ae05-543f8example",
```

```
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Family": "hello_world",
            "Version": "8",
            "Containers": [
                {
          "DockerId":
 "9581a69a761a557fbfce1d0f6745e4af5b9dbfb86b6b2c5c4df156f1a5932ff1",
          "DockerName": "ecs-hello_world-8-mysql-fcae8ac8f9f1d89d8301",
          "Name": "mysql",
          "CreatedAt": "2023-10-08T20:09:11.44527186Z",
          "StartedAt": "2023-10-08T20:09:11.44527186Z",
          "ImageID":
 "sha256:2ae34abc2ed0a22e280d17e13f9c01aaf725688b09b7a1525d1a2750e2c0d1de"
        },
        {
          "DockerId":
 "bf25c5c5b2d4dba68846c7236e75b6915e1e778d31611e3c6a06831e39814a15",
          "DockerName": "ecs-hello_world-8-wordpress-e8bfddf9b488dff36c00",
          "Name": "wordpress"
        }
    ]
}
    ]
}
```


É possível exibir informações sobre uma tarefa específica que esteja em execução em uma instância de contêiner. Para especificar uma tarefa ou um contêiner, acrescente um dos seguintes itens à solicitação:

- O Nome de região da Amazon (ARN) da tarefa (`?taskarn=`*`task_arn`*)
- O ID do Docker de um contêiner (`?dockerid=`*`docker_id`*)

Para obter informações sobre tarefas com um ID de Docker de um contêiner, faça login na instância de contêiner via SSH e execute o comando a seguir.

> **Note**
>
> Os agentes de contêiner do Amazon ECS antes da versão 1.14.2 requerem IDs de contêiner do Docker completos para a API de introspecção, e não a versão curta mostrada com docker

ps. É possível obter o ID de docker completo de um contêiner executando o comando docker ps --no-trunc na instância de contêiner.

```
curl http://localhost:51678/v1/tasks?dockerid=79c796ed2a7f
```

Saída:

```
{
    "Arn": "arn:aws:ecs:us-west-2:012345678910:task/default/e01d58a8-151b-40e8-
bc01-22647b9ecfec",
    "Containers": [
        {
            "DockerId":
 "79c796ed2a7f864f485c76f83f3165488097279d296a7c05bd5201a1c69b2920",
            "DockerName": "ecs-nginx-efs-2-nginx-9ac0808dd0afa495f001",
            "Name": "nginx",
            "CreatedAt": "2023-10-08T20:09:11.44527186Z",
            "StartedAt": "2023-10-08T20:09:11.44527186Z",
            "ImageID":
 "sha256:2ae34abc2ed0a22e280d17e13f9c01aaf725688b09b7a1525d1a2750e2c0d1de"
        }
    ],
    "DesiredStatus": "RUNNING",
    "Family": "nginx-efs",
    "KnownStatus": "RUNNING",
    "Version": "2"
}
```

Identificação de comportamentos não autorizados usando o monitoramento de runtime

O Amazon GuardDuty é um serviço de detecção de ameaças que ajuda a proteger contas, contêineres, workloads e dados no ambiente da AWS. Usando modelos de machine learning (ML) e recursos de detecção de anomalias e ameaças, o GuardDuty monitora continuamente diferentes fontes de log e atividades de runtime para identificar e priorizar possíveis riscos de segurança e atividades maliciosas no seu ambiente.

O monitoramento de runtime no GuardDuty protege workloads em execução nas instâncias de contêiner do Fargate e do EC2, monitorando continuamente as atividades de log e rede da AWS para identificar comportamentos maliciosos ou não autorizados. O monitoramento de runtime usa um agente de segurança do GuardDuty leve e totalmente gerenciado que analisa o comportamento no host, como acesso a arquivos, execução de processos e conexões de rede. Isso inclui problemas como escalação de privilégios, uso de credenciais expostas, comunicação com endereços IP ou domínios maliciosos e a presença de malware nas instâncias e workloads de contêiner do Amazon EC2. Para obter mais informações, consulte GuardDuty Runtime Monitoring no Guia do usuário do GuardDuty.

Seu administrador de segurança habilita o monitoramento de runtime para uma ou várias contas no AWS Organizations para o GuardDuty. Ele também seleciona se o GuardDuty implanta automaticamente o agente de segurança do GuardDuty ao usar o Fargate. Todos os clusters são protegidos automaticamente, e o GuardDuty gerencia o agente de segurança em seu nome.

Você também pode configurar manualmente o agente de segurança do GuardDuty nos seguintes casos:

- Você usa instâncias de contêiner do EC2
- Você precisa de controle granular para habilitar o monitoramento de runtime no nível do cluster

Para usar o monitoramento de runtime, você deve configurar os clusters protegidos, bem como instalar e gerenciar o agente de segurança do GuardDuty nas instâncias de contêiner do EC2.

Como o monitoramento de runtime funciona com o Amazon ECS

O monitoramento de runtime usa um agente de segurança leve do GuardDuty que monitora a atividade de workload do Amazon ECS para saber como as aplicações estão solicitando, obtendo acesso e consumindo os recursos subjacentes do sistema.

Para tarefas do Fargate, o agente de segurança do GuardDuty é executado como um contêiner auxiliar em cada tarefa.

Para instâncias de contêiner do EC2, o agente de segurança do GuardDuty é executado como um processo na instância.

O agente de segurança do GuardDuty coleta dados dos recursos a seguir e envia os dados ao GuardDuty para processamento. Você pode visualizar as descobertas no console do GuardDuty.

Você também pode enviá-los a outros Serviços da AWS, como o AWS Security Hub, ou a um fornecedor de segurança terceiro para agregação e remediação. Para obter informações sobre como visualizar e gerenciar descobertas, consulte Managing Amazon GuardDuty findings no Guia do usuário do Amazon GuardDuty.

- Respostas das seguintes chamadas de API do Amazon ECS:
  - DescribeClusters

    Os parâmetros de resposta incluem a tag do monitoramento de runtime (quando a tag é definida) ao usar a opção `--include TAGS`.
  - DescribeTasks

    Para o tipo de execução do Fargate, os parâmetros de resposta incluem o contêiner auxiliar do GuardDuty.
  - ListAccountSettings

    Os parâmetros de resposta incluem a configuração da conta do monitoramento de runtime, definida pelo administrador de segurança.
- Os dados de introspecção do agente de contêiner. Para obter mais informações, consulte Introspecção de contêiner do Amazon ECS.
- O endpoint de metadados da tarefa para o tipo de execução:
  - Endpoint de metadados de tarefas do Amazon ECS versão 4
  - Endpoint de metadados de tarefas do Amazon ECS versão 4 para tarefas no Fargate

Considerações

Considere o seguinte ao usar o monitoramento de runtime:

- O monitoramento de runtime tem um custo associado a ele. Para obter mais informações, consulte Amazon GuardDuty Pricing.
- O monitoramento de runtime não é compatível com o Amazon ECS Anywhere.
- O monitoramento de runtime não é compatível com o sistema operacional Windows.
- Ao usar o Amazon ECS Exec no Fargate, você deve especificar o nome do contêiner, porque o agente de segurança do GuardDuty é executado como um contêiner auxiliar.
- Você não pode usar o Amazon ECS Exec no contêiner auxiliar do agente de segurança do GuardDuty.

- O usuário do IAM que controla o monitoramento de runtime no nível do cluster deve ter as permissões apropriadas do IAM para marcação. Para obter mais informações, consulte Tutorial do IAM: Definir permissões para acessar recursos da AWS com base em tags no Guia do usuário do IAM.
- As tarefas do Fargate devem usar um perfil de execução de tarefas. Esse perfil concede às tarefas permissão para recuperar, atualizar e gerenciar o agente de segurança do GuardDuty, que é armazenado em um repositório privado do Amazon ECR, em seu nome.

Utilização de recursos

A tag adicionada ao cluster é contabilizada na cota de tags do cluster.

O contêiner auxiliar do agente do GuardDuty não é contabilizado na cota de definição de contêineres por tarefa.

Como acontece com a maioria dos softwares de segurança, há uma pequena sobrecarga no GuardDuty. Para obter informações sobre os limites de memória do Fargate, consulte CPU and memory limits no Guia do usuário do GuardDuty. Para obter informações sobre os limites de memória do Amazon EC2, consulte CPU and memory limit for GuardDuty agent.

Monitoramento de runtime para workloads do Fargate no Amazon ECS

Caso use instâncias de contêiner do EC2, você deve configurar manualmente o monitoramento de runtime. Para obter mais informações, consulte Monitoramento de runtime para workloads do EC2 no Amazon ECS.

Você pode fazer com que o GuardDuty gerencie o agente de segurança nas instâncias de contêiner. Esta opção está disponível somente para o Fargate. Esta opção (gerenciamento de agente por parte do GuardDuty) está disponível no GuardDuty.

Ao usar o gerenciamento de agente do GuardDuty, o GuardDuty executa as seguintes operações:

- Cria endpoints da VPC do GuardDuty em cada VPC que hospeda um cluster.
- Recupera e instala o agente de segurança do GuardDuty mais recente como contêiner auxiliar em todas as novas tarefas autônomas do Fargate e em novas implantações de serviços.

  Uma nova implantação de serviço acontece na primeira vez que você executa um serviço ou ao atualizar um serviço existente com a opção forçar nova implantação.

Ativação do monitoramento de runtime para o Amazon ECS

É possível configurar o GuardDuty para gerenciar automaticamente o agente de segurança para todos os clusters do Fargate.

Pré-requisitos

Estes são os pré-requisitos para usar o monitoramento de runtime:

- A versão da plataforma do Fargate deve ser `1.4.0` ou posterior para Linux.
- Permissões e perfis do IAM para Amazon ECS:
  - As tarefas do Fargate devem usar um perfil de execução de tarefas. Esse perfil concede às tarefas permissão para recuperar, atualizar e gerenciar o agente de segurança do GuardDuty em seu nome. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.
  - Você controla o monitoramento de runtime de um cluster com uma tag predefinida. Se suas políticas de acesso restringirem o acesso com base em tags, você deve conceder permissões explícitas aos usuários do IAM para marcar clusters. Para obter mais informações, consulte Tutorial do IAM: Definir permissões para acessar recursos da AWS com base em tags no Guia do usuário do IAM.
- Conectar-se ao repositório do Amazon ECR:

  O agente de segurança do GuardDuty é armazenado em um repositório do Amazon ECR. Cada tarefa autônoma e de serviço deve ter acesso ao repositório. Você pode usar uma das opções a seguir:
  - Para tarefas em sub-redes públicas, você pode usar um endereço IP público para a tarefa ou criar um endpoint da VPC para o Amazon ECR na sub-rede em que a tarefa é executada. Para obter mais informações, consulte Endpoints da VPC de interface do Amazon ECR (AWS PrivateLink) no Guia do usuário do Amazon Elastic Container Registry.
  - Para tarefas em sub-redes privadas, você pode usar um gateway de conversão de endereços de rede (NAT) ou criar um endpoint da VPC para o Amazon ECR na sub-rede em que a tarefa é executada.

    Para obter mais informações, consulte Sub-rede privada e gateway de NAT.
- É necessário ter o perfil `AWSServiceRoleForAmazonGuardDuty` do GuardDuty. Para obter mais informações, consulte Service-linked role permissions for GuardDuty no Guia do usuário do Amazon GuardDuty.

- Todos os arquivos que deseja proteger com o monitoramento de runtime devem estar acessíveis pelo usuário-raiz. Caso tenha alterado manualmente as permissões de um arquivo, você deve configurá-lo como `755`.

Estes são os pré-requisitos para usar o monitoramento de runtime em instâncias de contêiner do EC2:

- Você deve usar a versão `20230929` ou posterior da AMI do Amazon ECS.
- Você deve executar o agente do Amazon ECS para a versão `1.77` ou posterior nas instâncias de contêiner.
- Você deve usar a versão `5.10` ou posterior do kernel.
- Para obter informações sobre os sistemas operacionais e arquiteturas Linux compatíveis, consulte Which operating models and workloads does GuardDuty Runtime Monitoring support.
- Você pode usar o Systems Manager para gerenciar as instâncias de contêiner. Para obter mais informações, consulte Configuração do Systems Manager para instâncias do EC2 no Guia do usuário do AWS Systems Manager Session Manager.

Procedimento

Você habilita o monitoramento de runtime no GuardDuty. Para obter informações sobre como habilitar o recurso, consulte Enabling Runtime Monitoring no Guia do usuário do Amazon GuardDuty.

Adição do monitoramento de runtime para tarefas existentes do Fargate no Amazon ECS

Ao ativar o monitoramento de runtime, todas as novas tarefas independentes e novas implantações de serviços no cluster são protegidas automaticamente. Para preservar a restrição de imutabilidade, as tarefas existentes não serão afetadas.

Pré-requisitos

1. Ative o Monitoramento de runtime. Para obter mais informações, consulte Ativação do monitoramento de runtime para o Amazon ECS.
2. As tarefas do Fargate devem usar um perfil de execução de tarefas. Esse perfil concede às tarefas permissão para recuperar, atualizar e gerenciar o agente de segurança do GuardDuty em seu nome. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.

Procedimento

- Para proteger uma tarefa imediatamente, você deve executar uma das seguintes ações:
  - Para tarefas autônomas, interrompa as tarefas e inicie-as. Para ter mais informações, consulte Interrupção de uma tarefa do Amazon ECS e Execução de uma aplicação como uma tarefa do Amazon ECS
  - Para as tarefas que fazem parte de um serviço, atualize o serviço com a opção “forçar nova implantação”. Para obter mais informações, consulte Atualizar um serviço do Amazon ECS.

Remoção do monitoramento de runtime de um cluster do Amazon ECS

Talvez você queira excluir determinados clusters da proteção, por exemplo, os que são usados para testes. Isso faz com que o GuardDuty execute as operações a seguir nos recursos do cluster:

- Não implantar mais o agente de segurança do GuardDuty em novas tarefas autônomas do Fargate ou em novas implantações de serviços.

  Para preservar a restrição de imutabilidade, as tarefas e implantações existentes com o monitoramento de runtime habilitado não são afetadas.
- Parar de emitir cobranças e não aceitar mais eventos de runtime para tarefas.

Procedimento

Execute as etapas apresentadas a seguir para remover o monitoramento de runtime de um cluster.

1. Use o console do Amazon ECS ou a AWS CLI para definir a chave de tag `GuardDutyManaged` no cluster como `false`. Para obter mais informações, consulte Updating a cluster ou Working with tags using the CLI or API. Use os valores a seguir para a tag.

   > **Note**
   >
   > A chave e o valor diferenciam maiúsculas de minúsculas e devem corresponder exatamente às strings.

   Chave = `GuardDutyManaged`, Valor = `false`

2. Exclua o endpoint da VPC do GuardDuty do cluster. Para obter mais informações sobre como excluir endpoints da VPC, consulte Delete an interface endpoint no Guia do usuário do AWS PrivateLink.

Remoção do monitoramento de runtime do Amazon ECS de uma conta

Quando não quiser mais usar o monitoramento de runtime, desabilite o recurso no GuardDuty. Para obter informações sobre como desabilitar o recurso, consulte Enabling Runtime Monitoring no Guia do usuário do Amazon GuardDuty.

O GuardDuty realiza as seguintes operações:

- Exclui os endpoints da VPC do GuardDuty para cada VPC que hospeda um cluster.
- Não implanta mais o agente de segurança do GuardDuty em novas tarefas autônomas do Fargate ou em novas implantações de serviços.

  Para preservar a restrição de imutabilidade, as tarefas e implantações existentes não são afetadas até serem interrompidas, replicadas ou ajustadas na escala.
- Interrompe a cobrança e não aceita mais eventos de runtime para tarefas.

Monitoramento de runtime para workloads do EC2 no Amazon ECS

Use esta opção ao utilizar instâncias do EC2 para sua capacidade ou quando precisar de controle granular do monitoramento de runtime no nível do cluster no Fargate.

Você provisiona os clusters para o monitoramento de runtime adicionando uma tag predefinida.

Para instâncias de contêiner do EC2, você baixa, instala e gerencia o agente de segurança do GuardDuty.

No Fargate, o GuardDuty gerencia o agente de segurança em seu nome.

Ativação do monitoramento de runtime para o Amazon ECS

É possível ativar o monitoramento de runtime para clusters com instâncias do EC2 ou quando precisar de controle granular do monitoramento de runtime no nível do cluster no Fargate.

Estes são os pré-requisitos para usar o monitoramento de runtime:

- A versão da plataforma do Fargate deve ser `1.4.0` ou posterior para Linux.
- Permissões e perfis do IAM para Amazon ECS:
  - As tarefas do Fargate devem usar um perfil de execução de tarefas. Esse perfil concede às tarefas permissão para recuperar, atualizar e gerenciar o agente de segurança do GuardDuty em seu nome. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.
  - Você controla o monitoramento de runtime de um cluster com uma tag predefinida. Se suas políticas de acesso restringirem o acesso com base em tags, você deve conceder permissões explícitas aos usuários do IAM para marcar clusters. Para obter mais informações, consulte Tutorial do IAM: Definir permissões para acessar recursos da AWS com base em tags no Guia do usuário do IAM.
- Conectar-se ao repositório do Amazon ECR:

  O agente de segurança do GuardDuty é armazenado em um repositório do Amazon ECR. Cada tarefa autônoma e de serviço deve ter acesso ao repositório. Você pode usar uma das opções a seguir:
  - Para tarefas em sub-redes públicas, você pode usar um endereço IP público para a tarefa ou criar um endpoint da VPC para o Amazon ECR na sub-rede em que a tarefa é executada. Para obter mais informações, consulte Endpoints da VPC de interface do Amazon ECR (AWS PrivateLink) no Guia do usuário do Amazon Elastic Container Registry.
  - Para tarefas em sub-redes privadas, você pode usar um gateway de conversão de endereços de rede (NAT) ou criar um endpoint da VPC para o Amazon ECR na sub-rede em que a tarefa é executada.

    Para obter mais informações, consulte Sub-rede privada e gateway de NAT.
- É necessário ter o perfil `AWSServiceRoleForAmazonGuardDuty` do GuardDuty. Para obter mais informações, consulte Service-linked role permissions for GuardDuty no Guia do usuário do Amazon GuardDuty.
- Todos os arquivos que deseja proteger com o monitoramento de runtime devem estar acessíveis pelo usuário-raiz. Caso tenha alterado manualmente as permissões de um arquivo, você deve configurá-lo como `755`.

Estes são os pré-requisitos para usar o monitoramento de runtime em instâncias de contêiner do EC2:

- Você deve usar a versão `20230929` ou posterior da AMI do Amazon ECS.

- Você deve executar o agente do Amazon ECS para a versão `1.77` ou posterior nas instâncias de contêiner.
- Você deve usar a versão `5.10` ou posterior do kernel.
- Para obter informações sobre os sistemas operacionais e arquiteturas Linux compatíveis, consulte Which operating models and workloads does GuardDuty Runtime Monitoring support.
- Você pode usar o Systems Manager para gerenciar as instâncias de contêiner. Para obter mais informações, consulte Configuração do Systems Manager para instâncias do EC2 no Guia do usuário do AWS Systems Manager Session Manager.

Você ativa o monitoramento de runtime no GuardDuty. Para obter informações sobre como habilitar o recurso, consulte Enabling Runtime Monitoring no Guia do usuário do Amazon GuardDuty.

Adição do monitoramento de runtime para a um cluster do Amazon ECS

Configure o monitoramento de runtime para o cluster e instale o agente de segurança do GuardDuty nas instâncias de contêiner do EC2.

Pré-requisitos

1. Ative o Monitoramento de runtime. Para obter mais informações, consulte Ativação do monitoramento de runtime para o Amazon ECS.
2. Você controla o monitoramento de runtime de um cluster com uma tag predefinida. Se suas políticas de acesso restringirem o acesso com base em tags, você deve conceder permissões explícitas aos usuários do IAM para marcar clusters. Para obter mais informações, consulte Tutorial do IAM: Definir permissões para acessar recursos da AWS com base em tags no Guia do usuário do IAM.

Procedimento

Execute as operações a seguir para adicionar o monitoramento de runtime a um cluster.

1. Crie um endpoint da VPC do GuardDuty para cada VPC do cluster. Para obter mais informações, consulte Creating Amazon VPC endpoint manually no Guia do usuário do GuardDuty.
2. Configure as instâncias de contêiner do EC2.

a. Atualize o agente do Amazon ECS para a versão `1.77` ou posterior nas instâncias de contêiner do EC2 no cluster. Para obter mais informações, consulte Atualizar o agente de contêiner do Amazon ECS.

b. Instale o agente de segurança do GuardDuty nas instâncias de contêiner do EC2 no cluster. Para obter mais informações, consulte Managing the security agent on an Amazon EC2 instance manually no Guia do usuário do GuardDuty.

   Todas as tarefas e implantações novas e existentes são protegidas imediatamente, pois o agente de segurança do GuardDuty é executado como um processo na instância de contêiner do EC2.

3. Use o console do Amazon ECS ou a AWS CLI para definir a chave de tag `GuardDutyManaged` no cluster como `true`. Para obter mais informações, consulte Updating a cluster ou Working with tags using the CLI or API. Use os valores a seguir para a tag.

   > **Note**
   >
   > A chave e o valor diferenciam maiúsculas de minúsculas e devem corresponder exatamente às strings.

   Chave = `GuardDutyManaged`, Valor = `true`

Adição do monitoramento de runtime para tarefas existentes do Amazon ECS

Ao ativar o monitoramento de runtime, todas as novas tarefas independentes e novas implantações de serviços no cluster são protegidas automaticamente. Para preservar a restrição de imutabilidade, as tarefas existentes não serão afetadas.

Pré-requisitos

- Ative o Monitoramento de runtime. Para obter mais informações, consulte Ativação do monitoramento de runtime para o Amazon ECS.

Procedimento

- Para proteger uma tarefa imediatamente, você deve executar uma das seguintes ações:

- Para tarefas autônomas, interrompa as tarefas e inicie-as. Para obter mais informações, consulte Interrupção de uma tarefa do Amazon ECS e Execução de uma aplicação como uma tarefa do Amazon ECS.
- Para as tarefas que fazem parte de um serviço, atualize o serviço com a opção “forçar nova implantação”. Para obter mais informações, consulte Atualizar um serviço do Amazon ECS.

Remoção do monitoramento de runtime de um cluster do Amazon ECS

Você pode remover o monitoramento de runtime de um cluster. Isso faz com que o GuardDuty pare de monitorar todos os recursos no cluster.

Para remover o Monitoramento de runtime de um cluster

1. Use o console do Amazon ECS ou a AWS CLI para definir a chave de tag `GuardDutyManaged` no cluster como `false`. Para obter mais informações, consulte Updating a cluster ou Working with tags using the CLI or API.

   > **Note**
   >
   > A chave e o valor diferenciam maiúsculas de minúsculas e devem corresponder exatamente às strings.

   Chave = `GuardDutyManaged`, Valor = `false`
2. Desinstale o agente de segurança do GuardDuty nas instâncias de contêiner do EC2 no cluster.

   Para obter mais informações, consulte Uninstalling the security agent manually no Guia do usuário do GuardDuty.
3. Exclua o endpoint da VPC do GuardDuty para cada VPC do cluster. Para obter mais informações sobre como excluir endpoints da VPC, consulte Delete an interface endpoint no Guia do usuário do AWS PrivateLink.

Atualização do agente de segurança do GuardDuty nas instâncias de contêiner do Amazon ECS

Para obter informações sobre como atualizar o agente de segurança do GuardDuty nas instâncias de contêiner do EC2, consulte Updating GuardDuty security agent no Guia do usuário do Amazon GuardDuty.

Remoção do monitoramento de runtime do Amazon ECS de uma conta

Quando não quiser mais usar o monitoramento de runtime, desabilite o recurso no GuardDuty. Para obter informações sobre como desabilitar o recurso, consulte Enabling Runtime Monitoring no Guia do usuário do Amazon GuardDuty.

Remova o monitoramento de runtime de todos os clusters. Para obter mais informações, consulte Remoção do monitoramento de runtime de um cluster do Amazon ECS.

Solução de problemas de monitoramento de runtime

Talvez seja necessário solucionar problemas ou verificar se o monitoramento de runtime está habilitado e em execução nas tarefas e nos contêineres.

Tópicos

- Como posso saber se o monitoramento de runtime está ativo na minha conta?
- Como posso saber se o monitoramento de runtime está ativo em um cluster?
- Como posso saber se o agente de segurança do GuardDuty está sendo executado em uma tarefa do Fargate?
- Como posso saber se o agente de segurança do GuardDuty está sendo executado em uma instância de contêiner do EC2?
- O que acontece quando não há perfil de execução de tarefas para uma tarefa em execução no cluster?
- Como posso saber se tenho as permissões corretas para marcar clusters no monitoramento de runtime?
- O que acontece quando não há conexão com o Amazon ECR?
- Como soluciono erros de falta de memória nas tarefas do Fargate depois de habilitar o monitoramento de runtime?

Como posso saber se o monitoramento de runtime está ativo na minha conta?

No console do Amazon ECS, as informações estão na página Configurações da conta.

Você também pode executar `list-account-settings` com a opção `effective-settings`.

```
aws ecs list-account-settings --effective-settings
```

Saída

A configuração com nome definido como `guardDutyActivate` e valor definido como on indica que a conta está configurada. Você deve verificar com o administrador do GuardDuty se o gerenciamento é automático ou manual.

```
{
    "setting": {
        "name": "guardDutyActivate",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:root",
        "type": "aws-managed"
    }
}
```

Como posso saber se o monitoramento de runtime está ativo em um cluster?

Você pode revisar as estatísticas de cobertura no console do GuardDuty. Isso inclui informações para os recursos do Amazon ECS associados à sua própria conta ou às contas-membro e a porcentagem de clusters íntegros dentre todos os clusters na Região da AWS selecionada. Isso inclui a cobertura para os clusters que usam os tipos de inicialização do Fargate e do EC2. Para obter mais informações, consulte Reviewing coverage statistics no Amazon GuardDuty User Guide.

Como posso saber se o agente de segurança do GuardDuty está sendo executado em uma tarefa do Fargate?

O agente de segurança do GuardDuty é executado como contêiner auxiliar em tarefas do Fargate.

No console do Amazon ECS, o auxiliar é exibido em Contêineres na página Detalhes da tarefa.

Você pode executar `describe-tasks` e procurar o contêiner com um nome definido como `aws-gd-agent` e lastStatus definido como `RUNNING`.

O exemplo a seguir mostra a saída do cluster padrão para a tarefa `aws:ecs:us-east-1:123456789012:task/0b69d5c0-d655-4695-98cd-5d2d5EXAMPLE`.

```
aws ecs describe-tasks --cluster default --tasks aws:ecs:us-
east-1:123456789012:task/0b69d5c0-d655-4695-98cd-5d2d5EXAMPLE
```

Saída

O contêiner denominado `gd-agent` está no estado `RUNNING`.

```
"containers": [
      {
        "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/4df26bb4-
f057-467b-a079-96167EXAMPLE",
        "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/0b69d5c0-
d655-4695-98cd-5d2d5EXAMPLE",
        "lastStatus": "RUNNING",
        "healthStatus": "UNKNOWN",
        "memory": "string",
        "name": "aws-gd-agent"
      }
    ]
```

Como posso saber se o agente de segurança do GuardDuty está sendo executado em uma instância de contêiner do EC2?

Execute o seguinte comando para visualizar o status:

```
sudo systemctl status amazon-guardduty-agent
```

O arquivo de log está no seguinte local:

```
/var/log/amzn-guardduty-agent
```

O que acontece quando não há perfil de execução de tarefas para uma tarefa em execução no cluster?

Para tarefas do Fargate, a tarefa começa sem o contêiner auxiliar do agente de segurança do GuardDuty. O painel do GuardDuty mostrará que a tarefa está sem proteção no painel de estatísticas de cobertura.

Como posso saber se tenho as permissões corretas para marcar clusters no monitoramento de runtime?

Para marcar um cluster, você deve ter a ação `ecs:TagResource` para `CreateCluster` e `UpdateCluster`.

Veja a seguir um trecho de um exemplo de política.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ecs:TagResource"
      ],
      "Resource": "*",
      "Condition": {
         "StringEquals": {
             "ecs:CreateAction" : "CreateCluster",
             "ecs:CreateAction" : "UpdateCluster",
          }
       }
    }
  ]
}
```

O que acontece quando não há conexão com o Amazon ECR?

Para tarefas do Fargate, a tarefa começa sem o contêiner auxiliar do agente de segurança do GuardDuty. O painel do GuardDuty mostrará que a tarefa está sem proteção no painel de estatísticas de cobertura.

Como soluciono erros de falta de memória nas tarefas do Fargate depois de habilitar o monitoramento de runtime?

O agente de segurança do GuardDuty é um processo leve. No entanto, o processo ainda consome recursos de acordo com o tamanho da workload. Recomendamos usar ferramentas de rastreamento de recursos de contêineres, como o Amazon CloudWatch Container Insights, para preparar as implantações do GuardDuty no cluster. Essas ferramentas ajudam você a descobrir o perfil de consumo do agente de segurança do GuardDuty para as aplicações. Em seguida, você pode ajustar o tamanho da tarefa do Fargate, se necessário, para evitar possíveis condições de falta de memória.

Monitoramento de contêineres do Amazon ECS com o ECS Exec

Com o Amazon ECS Exec, você pode interagir diretamente com contêineres sem precisar interagir primeiro com o sistema operacional do contêiner do host, abrir portas de entrada ou gerenciar chaves SSH. É possível usar o ECS Exec para executar comandos em ou obter um shell para um contêiner em execução em uma instância do Amazon EC2 ou no AWS Fargate. Isso facilita a coleta de informações de diagnóstico e a solução rápida de problemas. Por exemplo, em um contexto de desenvolvimento, você pode usar o ECS Exec para interagir facilmente com vários processos nos contêineres e solucionar problemas das aplicações. Além disso, em cenários de produção, você pode usá-lo para obter acesso imediato aos contêineres para depurar problemas.

É possível executar comandos em um contêiner Linux ou Windows em execução usando o ECS Exec na API do Amazon ECS, no AWS Command Line Interface (AWS CLI), nos AWS SDKs ou na CLI do AWS Copilot. Para obter detalhes sobre como usar o ECS Exec, além de uma demonstração em vídeo, utilizando a CLI do AWS Copilot, consulte a documentação do GitHub sobre o Copilot.

Além disso, é possível usar o ECS Exec para manter as políticas de controle de acesso mais rígidas. Ao ativar seletivamente esse recurso, você pode controlar quem pode executar comandos e em quais tarefas eles podem executar estes comandos. Com um log de cada comando e a saída correspondente, é possível usar o ECS Exec para visualizar quais tarefas foram executadas e usar o CloudTrail para auditar as pessoas que acessaram um contêiner.

Considerações

Para este tópico, você deve se familiarizar com os seguintes aspectos envolvidos com o uso do ECS Exec:

- O ECS Exec, no momento, não é compatível com o uso do AWS Management Console.

  A página de detalhes dos clusters exibe o registro em log e a chave do AWS KMS gerenciada pelo cliente utiliza para criptografar os dados entre o cliente local e o contêiner.
- O ECS Exec pode não funcionar conforme o esperado quando executado em sistemas operacionais que não são compatíveis com o Systems Manager. Para obter mais informações sobre os sistemas operacionais compatíveis, consulte Tipos de sistemas operacionais no Guia do usuário do AWS Systems Manager.
- Há suporte para o ECS Exec para tarefas executadas na infraestrutura a seguir:
  - Contêineres Linux no Amazon EC2 em qualquer AMI otimizada para Amazon ECS, incluindo Bottlerocket

  - Contêineres do Linux e Windows em instâncias externas (Amazon ECS Anywhere)
  - Contêineres Linux e Windows no AWS Fargate
  - Contêineres Windows no Amazon EC2 nas AMIs otimizadas para Amazon ECS do Windows (com a versão do agente de contêiner `1.56` ou posterior):
    - AMI do Windows Server 2022 Full otimizada para Amazon ECS
    - AMI do Windows Server 2022 Core otimizada para Amazon ECS
    - AMI do Windows Server 2019 Full otimizada para Amazon ECS
    - AMI do Windows Server 2019 Core otimizada para Amazon ECS
    - AMI do Windows Server 20H2 Core otimizada para Amazon ECS
- Se você configurou um proxy HTTP para sua tarefa, defina a variável de ambiente `NO_PROXY` como `"NO_PROXY=169.254.169.254,169.254.170.2"` para ignorar o proxy para metadados da instância do EC2 e tráfego de perfil do IAM. Se você não configurar a variável de ambiente `NO_PROXY`, poderá haver falhas ao recuperar metadados da instância ou credenciais do perfil do IAM do endpoint de metadados dentro do contêiner. Definir a variável de ambiente `NO_PROXY` conforme recomendado filtra os metadados e o tráfego do IAM para que as solicitações a `169.254.169.254 and 169.254.170.2` não passem pelo proxy `HTTP`.
- ECS Exec e Amazon VPC
  - Se você estiver usando endpoints da Amazon VPC de interface com o Amazon ECS, deverá criar os endpoints da Amazon VPC da interface para o Gerenciador de Sessões do Systems Manager (`ssmmessages`). Para obter mais informações sobre endpoints da VPC do Systems Manager, consulte Uso de AWS PrivateLink para configurar um endpoint da VPC para o Gerenciador de Sessões no Guia do usuário do AWS Systems Manager.
  - Se você estiver usando endpoints do Amazon VPC de interface com o Amazon ECS e estiver usando o AWS KMS key para criptografia, deverá criar o endpoint do Amazon VPC da interface para o AWS KMS key. Para obter mais informações, consulte Conexão com o AWS KMS key usando um endpoint da VPC no Guia do desenvolvedor do AWS Key Management Service.
  - Quando tiver tarefas executadas em instâncias do Amazon EC2, use o modo de rede `awsvpc`. Caso não tenha acesso à internet, como ausência de configuração para usar um gateway NAT, você deve criar os endpoints de interface da Amazon VPC para o Gerenciador de Sessões do Systems Manager (`ssmmessages`). Para obter mais informações sobre considerações a respeito do modo de rede `awsvpc`, consulte Considerações. Para obter mais informações sobre endpoints da VPC do Systems Manager, consulte Uso de AWS PrivateLink para configurar um endpoint da VPC para o Gerenciador de Sessões no Guia do usuário do AWS Systems Manager.

- ECS Exec e SSM
  - Quando um usuário executa comandos em um contêiner usando o ECS Exec, estes comandos são executados como o usuário `root`. O SSM Agent e seus processos filho são executados como raiz mesmo quando você especifica um ID de usuário para o contêiner.
  - O agente SSM exige que o sistema de arquivos de contêiner possa ser gravado para criar os diretórios e arquivos necessários. Portanto, não existe suporte para a ação de tornar o sistema de arquivos raiz somente leitura usando o parâmetro de definição de tarefa `readonlyRootFilesystem` ou qualquer outro método.
  - Embora iniciar sessões do SSM fora da ação `execute-command` seja possível, isso fará com que as sessões não sejam registradas e contadas em relação ao limite da sessão. Recomendamos limitar esse acesso ao negar a ação `ssm:start-session` usando uma política do IAM. Para obter mais informações, consulte Limitar o acesso à ação Iniciar sessão.
- Os recursos a seguir são executados como contêiner auxiliar. Portanto, você deve especificar o nome do contêiner no qual executar o comando.
  - Monitoramento de runtime
  - Service Connect
- Os usuários podem executar todos os comandos disponíveis no contexto do contêiner. As seguintes ações podem resultar em processos órfãos e zumbis: encerramento do processo principal do contêiner, encerramento do agente de comando e exclusão de dependências. Para limpar os processos zumbis, recomendamos a adição do sinalizador `initProcessEnabled` à definição da tarefa.
- O ECS Exec usa um pouco de CPU e memória. Recomendamos que isso seja previsto quando você especificar as alocações de recursos de CPU e memória na definição da tarefa.
- Você deve estar usando a AWS CLI versão `1.22.3` ou posterior ou a AWS CLI versão `2.3.6` ou posterior. Para obter informações sobre como atualizar a AWS CLI, consulte Instalação ou atualização da versão mais recente da AWS CLI no Guia do usuário da AWS Command Line Interface versão 2.
- É possível ter somente uma sessão do ECS Exec por namespace de ID de processo (PID). Se você estiver compartilhando um namespace de PID em uma tarefa, só poderá iniciar sessões do ECS Exec em um contêiner.
- A sessão do ECS Exec tem um tempo limite de ociosidade de 20 minutos. Esse valor não pode ser alterado.
- Não é possível ativar o ECS Exec para tarefas existentes. Ele só pode ser ativado para novas tarefas.

- Não é possível usar o ECS Exec ao utilizar `run-task` para executar uma tarefa em um cluster que usa escalabilidade gerenciada com posicionamento assíncrono (executar uma tarefa sem instância).
- Não é possível executar o ECS Exec em contêineres do Microsoft Nano Server.

Pré-requisitos

Antes de começar a usar o ECS Exec, conclua as seguintes ações:

- Instale e configure a AWS CLI. Para obter mais informações, consulte Conceitos básicos do AWS CLI.
- Instale o plugin do gerenciador de sessão para a AWS CLI. Para obter mais informações, consulte Instalar o plugin do gerenciador de sessão para a AWS CLI.
- Você deve usar um perfil de tarefa com as permissões apropriadas para o ECS Exec. Para obter mais informações, consulte Perfil do IAM de tarefa.
- O ECS Exec tem requisitos de versão que dependem das suas tarefas estarem hospedadas no Amazon EC2 ou no AWS Fargate:
  - Se você estiver usando o Amazon EC2, deverá usar uma AMI otimizada para Amazon ECS que tenha sido lançada após 20 de janeiro de 2021, com a versão 1.50.2 ou superior do agente. Para obter mais informações consulte AMIs otimizadas para Amazon ECS.
  - Se você estiver usando o AWS Fargate, deverá usar a versão `1.4.0` ou superior (Linux) ou `1.0.0` (Windows) da plataforma. Para obter mais informações, consulte Versões da plataforma do AWS Fargate.

Arquitetura

O ECS Exec faz uso do gerenciador de sessão do AWS Systems Manager (SSM) para estabelecer uma conexão com o contêiner em execução e usa políticas do AWS Identity and Access Management (IAM) para controlar o acesso a comandos em execução em um contêiner em execução. Isso é possível por meio de uma montagem bind dos binários necessários do SSM Agent no contêiner. O Amazon ECS ou o agente do AWS Fargate é responsável por iniciar o agente central do SSM no interior do contêiner juntamente com o código da aplicação. Para obter mais informações, consulte Gerenciador de sessão do Systems Manager.

É possível auditar qual usuário acessou o contêiner por meio do evento `ExecuteCommand` no AWS CloudTrail e registrar em log cada comando (e suas saídas) no Amazon S3 ou no Amazon

CloudWatch Logs. Para criptografar dados entre o cliente local e o contêiner com sua própria chave de criptografia, você deve fornecer a chave do AWS Key Management Service (AWS KMS).

Usar o ECS Exec

Alterações opcionais de definição de tarefa

Se você definir o parâmetro de definição de tarefa `initProcessEnabled` como `true`, o processo de inicialização dentro do contêiner será iniciado. Isso remove todos os processos filho do agente SSM zumbi encontrados. A seguir, é fornecido um exemplo.

```
{
    "taskRoleArn": "ecsTaskRole",
    "networkMode": "awsvpc",
    "requiresCompatibilities": [
        "EC2",
        "FARGATE"
    ],
    "executionRoleArn": "ecsTaskExecutionRole",
    "memory": ".5 gb",
    "cpu": ".25 vcpu",
    "containerDefinitions": [
        {
            "name": "amazon-linux",
            "image": "amazonlinux:latest",
            "essential": true,
            "command": ["sleep","3600"],
            "linuxParameters": {
                "initProcessEnabled": true
            }
        }
    ],
    "family": "ecs-exec-task"
}
```

Ativar o ECS Exec para tarefas e serviços

É possível ativar o recurso ECS Exec para seus serviços e tarefas autônomas especificando o sinalizador `--enable-execute-command` quando usar um dos comandos a seguir da AWS CLI: `create-service`, `update-service`, `start-task` ou `run-task`.

Por exemplo, se você executar o comando a seguir, o recurso ECS Exec será ativado em um serviço recém-criado executado no Fargate. Para obter mais informações sobre como criar serviços, consulte create-service..

```
aws ecs create-service \
    --cluster cluster-name \
    --task-definition task-definition-name \
    --enable-execute-command \
    --service-name service-name \
    --launch-type FARGATE \
     --network-configuration
 "awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp=EN
 \
    --desired-count 1
```

Depois de ativar o ECS Exec para uma tarefa, execute o comando a seguir para confirmar se a tarefa está pronta para ser usada. Se a propriedade `lastStatus` do `ExecuteCommandAgent` estiver listada como `RUNNING` e a propriedade `enableExecuteCommand` estiver definida como `true`, a tarefa estará pronta.

```
aws ecs describe-tasks \
    --cluster cluster-name \
    --tasks task-id
```

O seguinte trecho de saída é um exemplo do que pode ser visualizado.

```
{
    "tasks": [
        {
            ...
            "containers": [
                {
                    ...
                    "managedAgents": [
                        {
                            "lastStartedAt": "2021-03-01T14:49:44.574000-06:00",
                            "name": "ExecuteCommandAgent",
                            "lastStatus": "RUNNING"
                        }
                    ]
                }
            ],
```

```
            ...
            "enableExecuteCommand": true,
            ...
        }
    ]
}
```

Executar comandos usando ECS Exec

Depois de ter confirmado que o `ExecuteCommandAgent` está em execução, você pode abrir um shell interativo no contêiner usando o comando a seguir. Se a tarefa contiver vários contêineres, você deverá especificar o nome do contêiner usando o sinalizador `--container`. O Amazon ECS só oferece suporte à inicialização de sessões interativas Portanto, você deve usar o sinalizador `--interactive`.

O comando a seguir executará um comando `/bin/sh` interativo em um contêiner denominado *`container-name`* para uma tarefa com o ID *`task-id`*.

O *`task-id`* é o nome do recurso da Amazon (ARN) da tarefa.

```
aws ecs execute-command --cluster cluster-name \
    --task task-id \
    --container container-name \
    --interactive \
    --command "/bin/sh"
```

Registro em log usando o ECS Exec

Ativação do registro em log em tarefas e serviços

> **Important**
>
> Para obter mais informações sobre os preços do CloudWatch, consulte Preço do CloudWatch. O Amazon ECS também fornece métricas de monitoramento, fornecidas sem custo adicional. Para obter mais informações, consulte Monitoramento do Amazon ECS usando o CloudWatch.

O Amazon ECS fornece uma configuração padrão para registrar em log os comandos executados usando o ECS Exec. O padrão é enviar logs ao CloudWatch Logs utilizando o driver de log `awslogs`

configurado na definição da tarefa. Se você quiser fornecer uma configuração personalizada, a AWS CLI oferecerá suporte ao sinalizador `--configuration` para os comandos `create-cluster` e `update-cluster`. A imagem de contêiner exige que `script` e `cat` estejam instalados para que os logs de comando sejam carregados corretamente no Amazon S3 ou no CloudWatch Logs. Para obter mais informações sobre como criar clusters, consulte create-cluster.

> **Note**
>
> Essa configuração só lida com o registro da sessão `execute-command`. Isso não afeta o registro da aplicação.

O exemplo a seguir cria um cluster e, em seguida, registra a saída no seu LogGroup do CloudWatch Logs denominado `cloudwatch-log-group-name` e no bucket do Amazon S3 denominado `s3-bucket-name`.

Você deve usar uma chave gerenciada pelo cliente do AWS KMS para criptografar o grupo de log quando definir a opção `CloudWatchEncryptionEnabled` como `true`. Para obter informações sobre como criptografar o grupo de log, consulte Criptografe dados de log no CloudWatch Logs usando AWS Key Management Service, no Manual usuário do Amazon CloudWatch Logs.

```
aws ecs create-cluster \
    --cluster-name cluster-name \
    --configuration executeCommandConfiguration="{ \
        kmsKeyId=string, \
        logging=OVERRIDE, \
        logConfiguration={ \
            cloudWatchLogGroupName=cloudwatch-log-group-name, \
            cloudWatchEncryptionEnabled=true, \
            s3BucketName=s3-bucket-name, \
            s3EncryptionEnabled=true, \
            s3KeyPrefix=demo \
        } \
    }"
```

A propriedade `logging` determina o comportamento da capacidade de registro do ECS Exec:

- `NONE`: o registro em log está desativado.
- `DEFAULT`: os logs são enviados ao driver `awslogs` configurado. Se o driver não estiver configurado, nenhum log será salvo.

- `OVERRIDE`: os logs são enviados ao LogGroup do Amazon CloudWatch Logs fornecido, ao bucket do Amazon S3 ou a ambos.

Permissões do IAM necessárias para registros no Amazon CloudWatch Logs ou no Amazon S3

Para ativar o registro, a função de tarefa do Amazon ECS referenciada na definição de tarefa precisa ter permissões adicionais. Essas permissões adicionais podem ser acrescentadas como uma política para o perfil de tarefa. Elas são diferentes, dependendo de você direcionar seus logs para o Amazon CloudWatch Logs ou para o Amazon S3.

Amazon CloudWatch Logs

O exemplo de política a seguir adiciona as permissões necessárias do Amazon CloudWatch Logs.

JSON

```
{
   "Version": "2012-10-17",
   "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
               "logs:CreateLogStream",
               "logs:DescribeLogStreams",
               "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:arn:aws:: 111122223333:log-group:/aws/
ecs/cloudwatch-log-group-name:*"
        }
   ]
}
```

Amazon S3

O exemplo de política a seguir adiciona as permissões necessárias do Amazon S3.

JSON

```
{
   "Version": "2012-10-17",
   "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "s3:GetBucketLocation"
            ],
            "Resource": "*"
        },
        {
           "Effect": "Allow",
           "Action": [
               "s3:GetEncryptionConfiguration"
           ],
           "Resource": "arn:aws:s3:::s3-bucket-name"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::s3-bucket-name/*"
        }
    ]
   }
```

Permissões do IAM necessárias para criptografia usando sua própria AWS KMS key (chave do KMS)

Por padrão, os dados transferidos entre o cliente local e o contêiner usam a criptografia TLS 1.2 que a AWS fornece. Para criptografar ainda mais os dados usando sua própria chave do KMS, crie uma chave do KMS e adicione a permissão `kms:Decrypt` à função do IAM da tarefa. Essa permissão é

usada pelo contêiner para descriptografar os dados. Para obter mais informações sobre a criação de uma chave do KMS, consulte Criar chaves.

Você adiciona a seguinte política em linha ao perfil do IAM de tarefa, que requer as permissões AWS KMS. Para obter mais informações, consulte Permissões do ECS Exec.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource":
 "arn:aws:kms:arn:aws::111122223333:key/12345678-1234-1234-1234-123456789012"
        }
    ]
}
```

Para que os dados sejam criptografados com a própria chave do KMS, o usuário ou grupo que está usando a ação `execute-command` deve receber a permissão `kms:GenerateDataKey`.

O exemplo de política a seguir para o usuário ou grupo contém a permissão necessária pra o uso da própria chave do KMS. É necessário especificar o nome do recurso da Amazon (ARN) da chave do KMS.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey"
            ],
            "Resource": "arn:aws:kms:arn:aws::111122223333:key/key"
```

```
        }
    ]
}
```

Uso de políticas do IAM para limitar o acesso ao ECS Exec

Você limita o acesso do usuário à ação da API execute-command usando uma ou mais das seguintes chaves de condição de política do IAM:

- `aws:ResourceTag/clusterTagKey`
- `ecs:ResourceTag/clusterTagKey`
- `aws:ResourceTag/taskTagKey`
- `ecs:ResourceTag/taskTagKey`
- `ecs:container-name`
- `ecs:cluster`
- `ecs:task`
- `ecs:enable-execute-command`

Com o exemplo de política do IAM a seguir, os usuários podem executar comandos em contêineres que estão sendo executados em tarefas com uma etiqueta que tenha a chave `environment` e o valor `development` em um cluster denominado `cluster-name`.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:ExecuteCommand",
                "ecs:DescribeTasks"
            ],
            "Resource": [
                  "arn:aws:ecs:arn:aws::111122223333:task/cluster-name/*",
                  "arn:aws:ecs:arn:aws::111122223333:cluster/cluster-name"
            ],
```

```
            "Condition": {
                "StringEquals": {
                    "ecs:ResourceTag/environment": "development"
                }
            }
        }
    ]
}
```

Com o exemplo de política do IAM a seguir, os usuários não podem usar a API `execute-command` quando o nome do contêiner é `production-app`.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "ecs:ExecuteCommand"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ecs:container-name": "production-app"
                }
            }
        }
    ]
}
```

Com a seguinte política do IAM, os usuários só podem iniciar tarefas quando o ECS Exec estiver desativado.

JSON

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:RunTask",
                "ecs:StartTask",
                "ecs:CreateService",
                "ecs:UpdateService"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ecs:enable-execute-command": "false"
                }
            }
        }
    ]
}
```

> **Note**
>
> Como ação da API `execute-command` contém apenas recursos de tarefa e cluster em uma solicitação, somente etiquetas de cluster e tarefa são avaliadas.

Para obter mais informações sobre essas chaves de condição de política do IAM, consulte Ações, recursos e chaves de condição do Amazon Elastic Container Service na Referência de autorização do serviço.

Limitar o acesso à ação Iniciar sessão

Embora seja possível iniciar sessões do SSM no contêiner fora do ECS Exec, isso poderá fazer com que as sessões não sejam registradas. As sessões iniciadas fora do ECS Exec também são consideradas na cota de sessão. Recomendamos que esse acesso seja limitado negando diretamente a ação `ssm:start-session` para as tarefas do Amazon ECS que usam uma política do IAM. É possível negar acesso a todas as tarefas do Amazon ECS ou a tarefas específicas com base nas etiquetas usadas.

Veja a seguir um exemplo de política do IAM que nega acesso à ação `ssm:start-session` para tarefas em todas as regiões com um nome de cluster especificado. Opcionalmente, é possível incluir um curinga com o *`cluster-name`*.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:StartSession",
            "Resource": [
            "arn:aws:ecs:arn:aws:;: 111122223333:task/cluster-name/*"
            ]
        }
    ]
}
```

Veja a seguir um exemplo de política do IAM que nega acesso à ação `ssm:start-session` em recursos em todas as regiões marcadas com chave de etiqueta `Task-Tag-Key` e valor de etiqueta `Exec-Task`.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:StartSession",
            "Resource": "arn:aws:ecs:*:*:task/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Task-Tag-Key": "Exec-Task"
                }
            }
        }
    ]
```

```
}
```

Para obter ajuda com quaisquer problemas que você possa encontrar ao usar o Amazon ECS Exec, consulte Troubleshooting issues with Exec.

Recomendações do AWS Compute Optimizer para o Amazon ECS

O AWS Compute Optimizer gera recomendações para tamanhos de tarefas e contêineres do Amazon ECS. Para obter mais informações, consulte O que é o AWS Compute Optimizer? no Guia do usuário do AWS Compute Optimizer.

Recomendações de tamanho de tarefas e contêineres para serviços do Amazon ECS no AWS Fargate

O AWS Compute Optimizer gera recomendações para serviços do Amazon ECS no AWS Fargate. O AWS Compute Optimizer recomenda o tamanho da CPU da tarefa e da memória da tarefa, além dos tamanhos da CPU do contêiner, da memória do contêiner e de reserva da memória do contêiner. Essas recomendações são exibidas nas páginas do console do Compute Optimizer abaixo.

- Página Recomendações para serviços do Amazon ECS no Fargate
- Página Detalhes de serviços do Amazon ECS no Fargate

Para obter mais informações, consulte Visualização de recomendações para serviços do Amazon ECS no Fargate no Guia do usuário do AWS Compute Optimizer.

Solução de problemas do Amazon ECS

Talvez você precise solucionar problemas dos balanceadores de carga, das tarefas, dos serviços ou das instâncias de contêiner. Este capítulo ajuda você a encontrar informações de diagnóstico do agente de contêiner do Amazon ECS, do daemon do Docker na instância de contêiner e do log de eventos de serviço no console do Amazon ECS.

Para obter informações sobre tarefas interrompidas, consulte um dos tópicos a seguir.

Ação	Saiba mais
Resolver erros de tarefa interrompida.	Visualizar erros de tarefa interrompida do Amazon ECS
Visualizar erros de tarefa interrompida.	Resolver erros de tarefa interrompida do Amazon ECS
Revisar códigos de erro de tarefa interrompida.	Mensagens de erro de tarefa interrompida do Amazon ECS
Revisar erros de tarefa CannotPullContainer.	Erros de tarefa CannotPullContainer no Amazon ECS
Visualizar solicitações de perfil do IAM de tarefas.	Visualização de solicitações de perfil do IAM para tarefas do Amazon ECS

Para obter informações sobre erros de serviço, consulte um dos tópicos a seguir.

Ação	Saiba mais
Visualizar mensagens de eventos de serviço.	Visualizar mensagens de eventos de serviço do Amazon ECS
Revisar mensagens de eventos de serviço.	Mensagens de eventos do serviço do Amazon ECS

Ação	Saiba mais	
Revisar problemas de balanceador de carga.	Solução de problemas relacionados aos balanceadores de carga de serviço no Amazon ECS	
Revisar problemas de ajuste de escala automático de serviço.	Solução de problemas relacionados ao ajuste de escala automático de serviço no Amazon ECS	

Para obter informações sobre erros de definição de tarefa, consulte um dos tópicos a seguir.

Ação	Saiba mais	
Resolver erro de memória de definição de tarefa.	Solucionar problemas causados por erros de CPU ou memória inválida na definição de tarefa do Amazon ECS	

Para obter informações sobre erros do agente do Amazon ECS, consulte um dos tópicos a seguir.

Ação	Saiba mais	
Visualizar logs do agente de contêiner do Amazon ECS.	Visualização de logs do agente de contêiner do Amazon ECS	
Saiba como coletar logs do Amazon ECS.	Coleta de logs de contêiner com o coletor de logs do Amazon ECS	

Ação	Saiba mais	
Recuperar detalhes de diagnóstico com o agente do Amazon ECS.	Recuperar detalhes de diagnóstico do Amazon ECS com introspecção do agente	

Para obter informações sobre o Docker, consulte um dos tópicos a seguir.

Ação	Saiba mais	
Usar diagnósticos do Docker.	Diagnóstico do Docker no Amazon ECS	
Ativar o modo de depuração do Docker.	Configuração da saída detalhada do daemon do Docker no Amazon ECS	
Solucionar o erro 500 da API do Docker.	Solução de problemas relacionados a `API error (500): devmapper` do Docker no Amazon ECS	

Para obter informações sobre erros do ECS Exec e do Amazon ECS Anywhere, consulte um dos tópicos a seguir.

Ação	Saiba mais	
Solucionar problemas do ECS Exec.	Solução de problemas do Amazon ECS Exec	
Solucionar problemas do Amazon ECS Anywhere.	Solução de problemas do Amazon ECS Anywhere	

Para obter informações sobre problemas de controle de utilização, consulte um dos tópicos a seguir.

Ação	Saiba mais	
Saiba mais sobre as cotas do controle de utilização do Fargate.	Cotas de controle de utilização do AWS Fargate	
Saiba mais sobre as práticas recomendadas para controle de utilização do Amazon ECS.	Lidar com problemas de controle de utilização do Amazon ECS	

Para obter informações sobre erros de API, consulte um dos tópicos a seguir.

Ação	Saiba mais	
Resolver erros de API.	Motivos de falha da API no Amazon ECS	

Resolver erros de tarefa interrompida do Amazon ECS

Quando não for possível iniciar a tarefa, você verá uma mensagem de erro no console e nos parâmetros de saída de `describe-tasks` (`stoppedReason` e `stoppedCode`).

É possível visualizar tarefas interrompidas no console por uma hora. Para visualizar tarefas interrompidas, é necessário alterar a opção de filtro. Para obter mais informações, consulte Visualizar erros de tarefa interrompida do Amazon ECS.

As páginas a seguir fornecem informações sobre tarefas interrompidas.

- Saiba mais sobre mensagens de erros de tarefa interrompida.

  Atualizações de mensagens de erro de tarefa interrompida do Amazon ECS
- Visualize as tarefas interrompidas para obter informações sobre a causa da interrupção.

  Visualizar erros de tarefa interrompida do Amazon ECS
- Saiba mais sobre as mensagens de erro de tarefa interrompida e os possíveis motivos dos erros.

  Mensagens de erro de tarefa interrompida do Amazon ECS

- Saiba como verificar conectividade de tarefa interrompida e corrigir os erros.

  Verificar conectividade de tarefa interrompida do Amazon ECS

Atualizações de mensagens de erro de tarefa interrompida do Amazon ECS

A partir de 14 de junho de 2024, a equipe do Amazon ECS está alterando as mensagens de erro de tarefa interrompida conforme descrito nas tabelas a seguir. O `stopCode` não vai mudar. Se seus aplicativos dependerem de strings de mensagens de erro exatas, você deverá atualizar seus aplicativos com as novas strings de caracteres. Para obter ajuda com dúvidas ou problemas, entre em contato com AWS Support.

> **Note**
>
> Recomendamos que você não confie nas mensagens de erro para a automação, pois elas estão sujeitas a alterações.

CannotPullContainerError

Mensagem de erro antiga	Nova mensagem de erro	
CannotPullContainerError: resposta de erro do daemon: acesso para extração negado para o *repositório* , o repositório não existe ou pode exigir 'login do docker': negado: usuário: *roleARN*	• CannotPullContainerError: a tarefa não consegue extrair a imagem. Verifique se o perfil tem as permissões para extrair imagens do registro. Resposta de erro do daemon: acesso para extração negado para o *repositório* , o repositór io não existe ou pode exigir 'login do docker': negado: o usuário: *roleARN* não está autorizado a executar: ecr:BatchGetImage no recurso: *imagem* porque	

Mensagem de erro antiga	Nova mensagem de erro	
	nenhuma política baseada em identidades permite a ação ecr:BatchGetImage. • CannotPullContainerError: a tarefa não consegue extrair a imagem. Verifique se a imagem existe. Resposta de erro do daemon: acesso para extração negado para o *repositório* , o repositório não existe ou pode exigir 'login do docker': negado: o acesso solicitado ao recurso foi negado.	
CannotPullContainerError: resposta de erro do daemon: obter imageURI: net/http: solicitação cancelada enquanto aguardava conexão	CannotPullContainerError: a tarefa não consegue extrair a imagem. Verifique a configuração da rede. Resposta de erro do daemon: obter *imagem*: net/http: solicitação cancelada enquanto aguardava conexão (Client.Timeout excedido enquanto aguardava cabeçalhos)	

Mensagem de erro antiga	Nova mensagem de erro	
CannotPullContainerError: ref pull foi tentado novamente 5 vezes: falha ao copiar: httpReadSeeker: falha ao abrir: falha ao fazer a solicitação: Get *registry-uri* : dial tcp <ip>:<port>: tempo limite de E/S excedido	CannotPullContainerError: a tarefa não pode extrair *image-uri* do registro *registry-uri* . Há um problema de conexão entre a tarefa e o registro. Verifique a configuração de rede da tarefa. : falha ao copiar: httpReadSeeker: falha ao abrir: falha ao fazer a solicitação: Obter *registry-uri* : dial tcp <ip>:<port>: tempo limite de E/S excedido	

ResourceNotFoundException

Mensagem de erro antiga	Nova mensagem de erro	
Buscar dados secretos do AWS Secrets Manager na região *região*: *sercretARN* do segredo: ResourceNotFoundException: o Secrets Manager não consegue encontrar o segredo especificado.	ResourceNotFoundException: a tarefa não consegue recuperar o segredo com o ARN '*sercretARN* ' do AWS Secrets Manager. Verifique se o segredo existe na região especificada. ResourceNotFoundException: a busca de dados secretos do AWS Secrets Manager na região *região*: segredo *sercretARN* : ResourceNotFoundException: o Secrets Manager não consegue encontrar o segredo especificado.	

ResourceInitializationError

Mensagem de erro antiga	Nova mensagem de erro	
ResourceInitializationError: não foi possível extrair segredos ou autenticação do registro: falha ao recuperar recurso de execução: não é possível recuperar a autenticação do registro ecr: a chamada de serviço foi repetida 3 vezes: RequestError: falha na solicitação de envio causada por: Post "https://api.ecr.us-east-1.amazonaws.com/": dial tcp <ip>:<port>: tempo limite de E/S excedido. Verifique sua configuração de rede da tarefa.	ResourceInitializationError: não é possível extrair segredos ou a autenticação do registro: a tarefa não consegue extrair a autenticação do registro do Amazon ECR. Há um problema de conexão entre a tarefa e o Amazon ECR. Verifique a configuração de rede da tarefa. RequestError: falha no envio da solicitação causada por: Post "https://api.ecr.us-east-1.amazonaws.com": dial tcp <ip>:<port>: tempo limite de E/S excedido	
ResourceInitializationError: não é possível extrair segredos ou autenticação do registro: falha na recuperação do recurso em execução: não é possível recuperar segredos do ssm: a chamada de serviço foi repetida 5 vezes: requestCanceled: contexto da solicitação cancelado causado por: prazo do contexto excedido	ResourceInitializationError: não é possível extrair segredos ou autenticação do registro: não é possível recuperar segredos do ssm: A tarefa não pode extrair segredos de AWS Systems Manager. Há um problema de conexão entre a tarefa e o AWS Systems Manager Parameter Store. Verifique a configuração de rede da tarefa. RequestCanceled: solicitação de contexto	

Mensagem de erro antiga	Nova mensagem de erro	
	cancelada devido a: prazo de contexto excedido	
ResourceInitializationError: falha ao baixar arquivos env: comando de download de arquivo: fluxo de erro não vazio: RequestCanceled: solicitação de contexto cancelada devido a: prazo de contexto excedido	ResourceInitializationError: falha ao baixar arquivos env: a tarefa não consegue baixar os arquivos de variáveis de ambiente do Amazon S3. Há um problema de conexão entre a tarefa e o Amazon S3. Verifique a configura ção de rede da sua tarefa. A chamada de serviço foi repetida 5 vezes: RequestCa nceled: solicitação de contexto cancelada devido a: prazo do contexto excedido	
ResourceInitializationError: falha ao validar os argumentos do logger::signal:killed	ResourceInitializationError: falha ao validar os argumento s do logger::signal:killed: a tarefa não consegue encontrar o grupo de logs do Amazon CloudWatch definido na tarefa. Há um problema de conexão entre a tarefa e o Amazon CloudWatch. Verifique a configuração da rede. : signal: killed	

Mensagem de erro antiga	Nova mensagem de erro	
ResourceInitializationError : não é possível extrair segredos ou autenticação do registro: falha do comando pull: : signal: killed	ResourceInitializationError : não é possível extrair segredos ou autenticação do registro: Verifique a configura ção de rede da tarefa. : signal: killed	

Visualizar erros de tarefa interrompida do Amazon ECS

Se você tiver problemas ao iniciar uma tarefa, sua tarefa poderá ser interrompida devido a erros na aplicação ou na configuração. Por exemplo, você executa a tarefa, ela exibe um status `PENDING` e, em seguida, desaparece.

Se sua tarefa tiver sido criada por um serviço do Amazon ECS, as ações que o Amazon ECS executará para manter o serviço serão publicadas nos eventos do serviço. É possível visualizar os eventos no AWS Management Console, na AWS CLI, em AWS SDKs, na API do Amazon ECS ou nas ferramentas que usam os SDKs e a API. Esses eventos incluem a interrupção e a substituição de uma tarefa pelo Amazon ECS porque os contêineres da tarefa interromperam a execução ou apresentaram falha em muitas verificações de integridade do Elastic Load Balancing.

Caso a tarefa tenha sido executada em uma instância de contêiner no Amazon EC2 ou em computadores externos, você também pode verificar os logs do runtime do contêiner e do agente do Amazon ECS. Esses logs estão na instância do Amazon EC2 do host ou no computador externo. Para obter mais informações, consulte Visualização de logs do agente de contêiner do Amazon ECS.

Procedimento

Console

AWS Management Console

As etapas a seguir podem ser usadas para verificar se há erros nas tarefas interrompidas usando o console. Para visualizar tarefas interrompidas, é necessário alterar a opção de filtro.

As tarefas interrompidas só aparecem no console por 1 hora.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.

2. No painel de navegação, escolha Clusters.
3. Na página Clusters, escolha o cluster.
4. Na página Cluster : ***name***, escolha a guia Tasks (Tarefas).
5. Configure o filtro para exibir tarefas interrompidas. Em Filtrar status desejado, escolha Interrompido.

   A opção Interrompida exibe suas tarefas interrompidas e Qualquer status desejado exibe todas as suas tarefas.
6. Escolha a tarefa interrompida para inspecionar.
7. Na linha das suas tarefas interrompidas, na coluna Último status, escolha Interrompida.

   Uma janela pop-up exibe o motivo da interrupção.

AWS CLI

1. Liste as tarefas interrompidas no cluster. A saída contém o nome do recurso da Amazon (ARN) da tarefa, necessário para a descrição da tarefa.

```
aws ecs list-tasks \
     --cluster cluster_name \
     --desired-status STOPPED \
     --region region
```

2. Descreva a tarefa interrompida para recuperar as informações. Para obter mais informações, consulte describe-tasks na AWS Command Line Interface Reference.

```
aws ecs describe-tasks \
     --cluster cluster_name \
     --tasks arn:aws:ecs:region:account_id:task/cluster_name/task_ID \
     --region region
```

Use os seguintes parâmetros de saída.

- `stopCode`: o código de interrupção indica por que uma tarefa foi interrompida, por exemplo ResourceInitializationError
- `StoppedReason`: o motivo pelo qual a tarefa foi interrompida.

- `reason` (na estrutura de `containers`): o motivo fornece detalhes adicionais sobre o contêiner interrompido.

Próximas etapas

Visualize as tarefas interrompidas para obter informações sobre a causa da interrupção. Para obter mais informações, consulte Mensagens de erro de tarefa interrompida do Amazon ECS.

Mensagens de erro de tarefa interrompida do Amazon ECS

A seguir estão as possíveis mensagens de erro que você pode receber quando sua tarefa é interrompida inesperadamente.

Para verificar se há uma mensagem de erro nas tarefas interrompidas usando o AWS Management Console, consulte Visualizar erros de tarefa interrompida do Amazon ECS.

Os códigos de erros de tarefa interrompida têm uma categoria associada a eles, por exemplo, "ResourceInitializationError". Para obter mais informações sobre cada categoria, consulte o seguinte:

Categoria	Saiba mais
TaskFailedToStart	Solução de problemas causados por erros TaskFailedToStart do Amazon ECS
ResourceInitializationError	Solução de problemas causados por erros ResourceInitializationError do Amazon ECS
ResourceNotFoundException	Solução de problemas causados por erros ResourceNotFoundException do Amazon ECS
SpotInterruptionError	Solução de problemas causados por erros SpotInterruption do Amazon ECS

Categoria	Saiba mais
InternalError	Solução de problemas causados por erros InternalError do Amazon ECS
OutOfMemoryError	Solução de problemas causados por erros OutOfMemoryError do Amazon ECS
ContainerRuntimeError	Solução de problemas causados por erros ContainerRuntimeError do Amazon ECS
ContainerRuntimeTimeoutError	Solução de problemas causados por erros ContainerRuntimeTimeoutError do Amazon ECS
CannotStartContainerError	Solução de problemas causados por erros CannotStartContainerError do Amazon ECS
CannotStopContainerError	Solução de problemas causados por erros CannotStopContainerError do Amazon ECS
CannotInspectContainerError	Solução de problemas causados por erros CannotInspectContainerError do Amazon ECS

Categoria	Saiba mais
CannotCreateVolumeError	Solução de problemas causados por erros CannotCreateVolumeError do Amazon ECS
CannotPullContainer	Erros de tarefa CannotPullContainer no Amazon ECS

Solução de problemas causados por erros TaskFailedToStart do Amazon ECS

A seguir estão algumas mensagens do erro `TaskFailedToStart` e ações que podem ser executadas para corrigi-lo.

Para verificar se há uma mensagem de erro nas tarefas interrompidas usando o AWS Management Console, consulte Visualizar erros de tarefa interrompida do Amazon ECS.

Erro inesperado do EC2 ao tentar criar interface de rede com atribuição de IP público habilitada na sub-rede '***subnet-id***'

Isso acontece quando uma tarefa do Fargate usa o modo de rede `awsvpc` e é executada em uma sub-rede com um endereço IP público, e a sub-rede não tem endereços IP suficientes.

O número de endereços IP disponíveis é fornecido na página de detalhes da sub-rede no console do Amazon EC2 ou usando `describe-subnets`. Para obter mais informações, consulte View your subnet no Guia do usuário da Amazon VPC.

Para corrigir esse problema, você pode criar uma sub-rede para executar a tarefa.

InternalError: ***<motivo>***

Esse erro ocorre quando um anexo da ENI é solicitado. O Amazon EC2 gerencia o provisionamento da ENI de forma assíncrona. O processo de provisionamento leva tempo. O Amazon ECS tem um tempo limite, caso haja longos tempos de espera ou falhas não relatadas. Há momentos em que a ENI é provisionada, mas o relatório chega ao Amazon ECS após a falha exceder o tempo limite. Nesse caso, o Amazon ECS vê a falha relatada na tarefa com uma ENI em uso.

A definição de tarefa selecionada não é compatível com a estratégia de computação selecionada

Esse erro ocorre quando você escolhe uma definição de tarefa com um tipo de execução que não corresponde ao tipo de capacidade do cluster. Para obter mais informações, consulte Tipos de inicialização do Amazon ECS. Você precisa selecionar uma definição de tarefa que corresponda ao provedor de capacidade atribuído ao cluster.

Não é possível conectar a interface de rede ao índice de dispositivos não utilizados

Este erro ocorre ao usar o tipo de rede `awsvpc` e não há CPU/memória suficiente para a tarefa. Primeiro, verifique se há instâncias na CPU. Para obter mais informações, consulte as especificações de tipo de instância do Amazon EC2 em Tipos de instância do Amazon EC2. Pegue o valor da CPU para a instância e multiplique-o pelo número de ENIs da instância. Use esse valor e na definição de tarefa.

AGENT

A instância de contêiner na qual você tentou iniciar uma tarefa tem um agente que está desconectado, no momento. Para evitar tempos de espera prolongados para a tarefa, a solicitação foi rejeitada.

Para saber mais sobre como solucionar um agente que está desconectado, consulte Como solucionar problemas com um agente desconectado do Amazon ECS.

Solução de problemas causados por erros ResourceInitializationError do Amazon ECS

A seguir estão algumas mensagens do erro `ResourceInitialization` e ações que podem ser executadas para corrigi-lo.

Para verificar se há uma mensagem de erro nas tarefas interrompidas usando o AWS Management Console, consulte Visualizar erros de tarefa interrompida do Amazon ECS.

Erros

- A tarefa não pode extrair a autenticação do registro do Amazon ECR. Há um problema de conexão entre a tarefa e o Amazon ECR. Verifique a configuração de rede da tarefa.
- A tarefa não consegue baixar os arquivos de variáveis de ambiente do Amazon S3. Há um problema de conexão entre a tarefa e o Amazon S3. Verifique a configuração de rede da tarefa.
- A tarefa consegue extrair segredos do AWS Systems Manager Parameter Store. Verifique a conexão de rede entre a tarefa e o AWS Systems Manager.

- A tarefa não consegue extrair segredos do AWS Secrets Manager. Há um problema de conexão entre a tarefa e o Secrets Manager. Verifique a configuração de rede da tarefa.
- A tarefa não consegue extrair o segredo do Secrets Manager. A tarefa não consegue recuperar o segredo com ARN "secretARN"' do Secrets Manager. Verifique se o segredo existe na região especificada.
- Falha do comando pull: não é possível extrair segredos ou autenticação do registro. Verifique a configuração de rede da sua tarefa.
- A tarefa não consegue encontrar o grupo de logs do Amazon CloudWatch definido na tarefa. Há um problema de conexão entre a tarefa e o Amazon CloudWatch. Verifique a configuração da rede.
- falha ao inicializar o driver de registro em log
- falha ao invocar comandos do EFS utils para configurar volumes do EFS

A tarefa não pode extrair a autenticação do registro do Amazon ECR. Há um problema de conexão entre a tarefa e o Amazon ECR. Verifique a configuração de rede da tarefa.

Esse erro indica que a tarefa não consegue se conectar ao Amazon ECR.

Verifique a conexão entre a tarefa e o Amazon ECR. Para mais informações, consulte Verificar conectividade de tarefa interrompida do Amazon ECS.

A tarefa não consegue baixar os arquivos de variáveis de ambiente do Amazon S3. Há um problema de conexão entre a tarefa e o Amazon S3. Verifique a configuração de rede da tarefa.

Esse erro ocorre quando a tarefa não consegue baixar o arquivo de ambiente do Amazon S3.

Verifique a conexão entre a tarefa e o endpoint da VPC do Amazon S3. Para mais informações, consulte Verificar conectividade de tarefa interrompida do Amazon ECS.

A tarefa consegue extrair segredos do AWS Systems Manager Parameter Store. Verifique a conexão de rede entre a tarefa e o AWS Systems Manager.

Esse erro ocorre quando a tarefa não consegue extrair a imagem definida na definição de tarefa usando as credenciais do Systems Manager.

Verifique a conexão entre a tarefa e o endpoint da VPC do Systems Manager. Para mais informações, consulte Verificar conectividade de tarefa interrompida do Amazon ECS.

A tarefa não consegue extrair segredos do AWS Secrets Manager. Há um problema de conexão entre a tarefa e o Secrets Manager. Verifique a configuração de rede da tarefa.

Esse erro ocorre quando a tarefa não consegue extrair a imagem definida na definição de tarefa usando as credenciais do Secrets Manager.

O erro indica que existe um problema de conectividade de rede entre o endpoint da VPC do Systems Manager e a tarefa.

Para obter informações sobre como verificar a conectividade entre a tarefa e o endpoint, consulte Verificar conectividade de tarefa interrompida do Amazon ECS.

A tarefa não consegue extrair o segredo do Secrets Manager. A tarefa não consegue recuperar o segredo com ARN "***secretARN'***" do Secrets Manager. Verifique se o segredo existe na região especificada.

Esse erro ocorre quando a tarefa não consegue extrair a imagem definida na definição de tarefa usando as credenciais do Secrets Manager.

Esse problema é causado por um ou mais dos seguintes motivos:

Causa do erro	Fazer isso...	
Problema de conectividade de rede entre o endpoint da VPC do Secrets Manager e a tarefa. Existe um problema de rede quando você vê uma das seguintes strings na mensagem de erro: • dial tcp • dial udp • <ip>:<port>: tempo limite de e/s esgotado	Verifique a conectividade entre a tarefa e o endpoint do Secrets Manager. Para obter mais informações, consulte Verificar conectividade de tarefa interrompida do Amazon ECS.	

Causa do erro	Fazer isso...	
• net/http: tempo limite de handshake do TLS esgotado • ler: tempo limite de conexão esgotado • Client.Timeout esgotado enquanto aguardava os cabeçalhos • net/http: solicitação cancelada enquanto aguardava conexão • sinal: eliminado • prazo de contexto esgotado		
O perfil de execução de tarefa definido na definição de tarefa não tem as permissões para o Secrets Manager.	Adicione as permissões necessárias do Secrets Manager ao perfil de execução de tarefa. Para obter mais informações, consulte Permissões do Secrets Manager ou do Systems Manager.	
O ARN do segredo não existe	Verifique se o ARN existe no Secrets Manager. Para obter informações sobre visualização de imagens, consulte Find secrets in Secrets Manager no Secrets Manager Developer Guide.	

Falha do comando pull: não é possível extrair segredos ou autenticação do registro. Verifique a configuração de rede da sua tarefa.

Esse erro ocorre quando a tarefa não consegue se conectar ao Amazon ECR, Systems Manager ou Secrets Manager. Isso ocorre quando há uma configuração incorreta na rede.

Para corrigir o problema, verifique a conectividade entre a tarefa e o Amazon ECR. Também é necessário verificar a conectividade entre sua tarefa e o serviço que armazena seu segredo (Systems Manager ou Secrets Manager). Para obter mais informações, consulte Verificar conectividade de tarefa interrompida do Amazon ECS.

A tarefa não consegue encontrar o grupo de logs do Amazon CloudWatch definido na tarefa. Há um problema de conexão entre a tarefa e o Amazon CloudWatch. Verifique a configuração da rede.

Esse erro ocorre quando a tarefa não consegue encontrar o grupo de logs do CloudWatch especificado na definição da tarefa.

O erro indica que existe um problema de conectividade de rede entre o endpoint da VPC do CloudWatch e a tarefa.

Para obter informações sobre como verificar a conectividade entre a tarefa e o endpoint, consulte Verificar conectividade de tarefa interrompida do Amazon ECS.

falha ao inicializar o driver de registro em log

Esse erro ocorre quando a tarefa não consegue encontrar o grupo de logs do CloudWatch especificado na definição da tarefa.

O erro indica que o grupo do CloudWatch da definição de tarefa não existe.

Use as etapas a seguir para encontrar o CloudWatch ausente.

1. Execute o comando a seguir para obter as informações de definição da tarefa.

```
aws ecs describe-task-definition \
    --task-definition task-def-name
```

   Observe a saída de cada contêiner e anote o valor `awslogs-group`.

```
"logConfiguration": {
```

```
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/example-group",
                    "awslogs-create-group": "true",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "ecs"
                },
```

2. Verifique se o grupo existe no CloudWatch. Para obter mais informações, consulte Como trabalhar com grupos de logs e fluxos de logs no Guia do usuário do Amazon CloudWatch Logs.

   O problema é que o grupo especificado na definição da tarefa está incorreto ou o grupo de logs não existe.

3. Corrija o problema.

O problema é...	Fazer isso...
O grupo de logs incorreto está especificado na definição da tarefa.	Atualize a definição da tarefa para incluir a configuração do grupo de logs na definição do contêiner. Para obter mais informações sobre atualização da definição de tarefa, consulte Atualizar uma definição de tarefa do Amazon ECS usando o console ou RegisterTaskDefinition na Referência da API do Amazon Elastic Container Service.
O grupo de logs não existe no CloudWatch	Crie o grupo de logs. Para obter mais informações, consulte Criar um grupo de logs no CloudWatch Logs no Guia do usuário do Amazon CloudWatch Logs.

falha ao invocar comandos do EFS utils para configurar volumes do EFS

Os seguintes problemas podem impedir que você monte os volumes do Amazon EFS nas tarefas:

- O sistema de arquivos do Amazon EFS não está configurado corretamente.
- A tarefa não tem as permissões necessárias.
- Há problemas relacionados às configurações de rede e VPC.

Para obter informações sobre como depurar e corrigir esse problema, consulte Por que não consigo montar meus volumes do Amazon EFS em minhas tarefas do AWS Fargate? no AWS re:Post.

Solução de problemas causados por erros ResourceNotFoundException do Amazon ECS

A seguir estão algumas mensagens do erro `ResourceNotFoundException` e ações que podem ser executadas para corrigi-lo.

Para verificar se há uma mensagem de erro nas tarefas interrompidas usando o AWS Management Console, consulte Visualizar erros de tarefa interrompida do Amazon ECS.

A tarefa não consegue recuperar o segredo com o ARN '***sercretARN***' do AWS Secrets Manager. Verifique se o segredo existe na região especificada.

Esse erro ocorre quando a tarefa não consegue recuperar o segredo do Secrets Manager. Isso significa que o segredo especificado na definição de tarefa (e contido na mensagem de erro) não existe no Secrets Manager.

A região está na mensagem de erro.

Buscar dados secretos do AWS Secrets Manager na região *região*: *sercretARN* do segredo: ResourceNotFoundException: o Secrets Manager não consegue encontrar o segredo especificado.

Para obter informações sobre como encontrar um segredo, consulte Find secrets in AWS Secrets Manager no AWS Secrets Manager User Guide.

Use a tabela a seguir para determinar qual é o erro e resolvê-lo.

Problema	Ações
O segredo está em uma região diferente da definição de tarefa.	a. Crie o segredo na mesma região que a tarefa. Para obter mais informações, consulte Criar um segredo no AWS Secrets Manager. b. Atualize a definição de tarefa com o novo segredo. Para obter mais informaçõ es, consulte Atualizar uma definição de tarefa do Amazon ECS usando o console ou RegisterT askDefinition na Referênci a da API do Amazon Elastic Container Service.
A definição de tarefa tem o ARN do segredo incorreto. O segredo correto encontra-se no Secrets Manager.	Atualize a definição de tarefa com o segredo correto. Para obter mais informaçõ es, consulte Atualizar uma definição de tarefa do Amazon ECS usando o console ou RegisterTaskDefinition na Referência da API do Amazon Elastic Container Service.
O segredo não existe mais.	a. Crie o segredo na mesma região que a tarefa. Para obter mais informações, consulte Criar um segredo no AWS Secrets Manager. b. Atualize a definição de tarefa com o novo segredo. Para obter mais informaçõ

Problema	Ações	
	es, consulte Atualizar uma definição de tarefa do Amazon ECS usando o console ou RegisterTaskDefinition na Referência da API do Amazon Elastic Container Service.	

Solução de problemas causados por erros SpotInterruption do Amazon ECS

O erro `SpotInterruption` tem motivos diferentes para os tipos de inicialização do Fargate e do EC2.

Para verificar se há uma mensagem de erro nas tarefas interrompidas usando o AWS Management Console, consulte Visualizar erros de tarefa interrompida do Amazon ECS.

Tipo de inicialização do Fargate

O erro `SpotInterruption` ocorre quando não há capacidade spot do Fargate ou quando o Fargate recupera capacidade spot.

Você pode executar as tarefas em várias zonas de disponibilidade para permitir mais capacidade.

Tipo de inicialização do EC2

Esse erro ocorre quando não há instâncias spot disponíveis ou quando o EC2 recupera a capacidade spot.

Você pode executar as instâncias em várias zonas de disponibilidade para permitir mais capacidade.

Solução de problemas causados por erros InternalError do Amazon ECS

Aplica-se a: tipo de inicialização do Fargate

Para verificar se há uma mensagem de erro nas tarefas interrompidas usando o AWS Management Console, consulte Visualizar erros de tarefa interrompida do Amazon ECS.

O erro `InternalError` ocorre quando o agente encontra um erro interno inesperado não relacionado ao runtime.

Esse erro só ocorre se estiver sendo usada a versão `1.4` ou posterior da plataforma.

Para obter informações sobre como depurar e solucionar esse problema, consulte Mensagens de erro de tarefa interrompida do Amazon ECS.

Solução de problemas causados por erros OutOfMemoryError do Amazon ECS

A seguir estão algumas mensagens de erros OutOfMemoryError e medidas que podem ser tomadas para corrigi-los.

Para verificar se há uma mensagem de erro nas tarefas interrompidas usando o AWS Management Console, consulte Visualizar erros de tarefa interrompida do Amazon ECS.

contêiner eliminado devido ao uso de memória

Esse erro ocorre quando um contêiner fecha devido a processos que consomem mais memória do que a alocada na definição de tarefa ou devido a restrições do host ou do sistema operacional.

Solução de problemas causados por erros ContainerRuntimeError do Amazon ECS

A seguir estão algumas mensagens de erros ContainerRuntimeError e medidas que podem ser tomadas para corrigi-los.

Para verificar se há uma mensagem de erro nas tarefas interrompidas usando o AWS Management Console, consulte Visualizar erros de tarefa interrompida do Amazon ECS.

ContainerRuntimeError

Esse erro ocorre quando o agente recebe um erro inesperado do `containerd` para uma operação específica do runtime. Esse erro geralmente é causado por uma falha interna no agente ou no runtime do `containerd`.

Esse erro ocorre somente se você usa a versão da plataforma `1.4.0` ou posterior (Linux) ou `1.0.0` ou posterior (Windows).

Para obter informações sobre como depurar e corrigir esse problema, consulte Por que minha tarefa do Amazon ECS parou? no AWS re:Post.

Solução de problemas causados por erros ContainerRuntimeTimeoutError do Amazon ECS

A seguir estão algumas mensagens de erros ContainerRuntimeTimeoutError e medidas que podem ser tomadas para corrigi-los.

Para verificar se há uma mensagem de erro nas tarefas interrompidas usando o AWS Management Console, consulte Visualizar erros de tarefa interrompida do Amazon ECS.

Não foi possível passar para a execução; tempo limite esgotado esperando 1m ou erro de tempo limite esgotado do Docker

Esse erro ocorre quando um contêiner não consegue fazer a transição para um estado `RUNNING` ou `STOPPED` dentro no tempo limite. O motivo e o valor de tempo limite são fornecidos na mensagem de erro.

Solução de problemas causados por erros CannotStartContainerError do Amazon ECS

A seguir estão algumas mensagens de erro CannotStartContainerError e medidas que podem ser tomadas para corrigi-los.

Para verificar se há uma mensagem de erro nas tarefas interrompidas usando o AWS Management Console, consulte Visualizar erros de tarefa interrompida do Amazon ECS.

falha ao obter o status do contêiner: ***<motivo>***

O erro ocorre quando não é possível iniciar um contêiner.

Caso seu contêiner tente exceder a memória especificada, o mesmo será interrompido. Aumente a memória apresentada ao contêiner. Esse é o parâmetro `memory` na definição da tarefa. Para obter mais informações, consulte the section called “Memória”.

Solução de problemas causados por erros CannotStopContainerError do Amazon ECS

A seguir estão algumas mensagens de erro CannotStopContainerError e medidas que podem ser tomadas para corrigi-los.

Para verificar se há uma mensagem de erro nas tarefas interrompidas usando o AWS Management Console, consulte Visualizar erros de tarefa interrompida do Amazon ECS.

CannotStopContainerError

Este erro ocorre quando não é possível interromper um contêiner.

Para obter informações sobre como depurar e corrigir esse problema, consulte Por que minha tarefa do Amazon ECS parou? no AWS re:Post.

Solução de problemas causados por erros CannotInspectContainerError do Amazon ECS

A seguir estão algumas mensagens de erro CannotInspectContainerError e medidas que podem ser tomadas para corrigi-los.

Para verificar se há uma mensagem de erro nas tarefas interrompidas usando o AWS Management Console, consulte Visualizar erros de tarefa interrompida do Amazon ECS.

CannotInspectContainerError

Esse erro ocorre quando o agente de contêiner não consegue descrever o contêiner no runtime do contêiner.

Ao usar a versão `1.3` da plataforma ou anterior, o agente do Amazon ECS retorna o motivo do Docker.

Ao usar a versão `1.4.0` da plataforma ou posterior (Linux) ou `1.0.0` ou posterior (Windows), o agente do Fargate retorna o motivo do `containerd`.

Para obter informações sobre como depurar e corrigir esse problema, consulte Por que minha tarefa do Amazon ECS parou? no AWS re:Post.

Solução de problemas causados por erros CannotCreateVolumeError do Amazon ECS

A seguir estão algumas mensagens de erro CannotCreateVolumeError e medidas que podem ser tomadas para corrigi-los.

Para verificar se há uma mensagem de erro nas tarefas interrompidas usando o AWS Management Console, consulte Visualizar erros de tarefa interrompida do Amazon ECS.

CannotCreateVolumeError

Esse erro ocorre quando o agente não consegue criar a montagem de volume especificada na definição de tarefa.

Esse erro ocorre somente se você usa a versão da plataforma `1.4.0` ou posterior (Linux) ou `1.0.0` ou posterior (Windows).

Para obter informações sobre como depurar e corrigir esse problema, consulte Por que minha tarefa do Amazon ECS parou? no AWS re:Post.

Erros de tarefa CannotPullContainer no Amazon ECS

Os erros a seguir indicam que a tarefa falhou ao iniciar porque o Amazon ECS não consegue recuperar a imagem do contêiner especificada.

> **Note**
>
> A versão 1.4 da plataforma do Fargate trunca mensagens de erro longas.

Para verificar se há uma mensagem de erro nas tarefas interrompidas usando o AWS Management Console, consulte Visualizar erros de tarefa interrompida do Amazon ECS.

Erros

- A tarefa não consegue extrair a imagem. Verifique se o perfil tem as permissões para extrair imagens do registro.
- A tarefa não pode extrair "image-name" do repositório "repository URI" do Amazon ECR. Há um problema de conexão entre a tarefa e o Amazon ECR. Verifique a configuração de rede da tarefa.
- A tarefa não consegue extrair a imagem. Verificar a configuração da rede
- CannotPullContainerError: o manifesto de extração da imagem foi repetido 5 vez(es): falha ao resolver referência
- Erro de API (500): obter https://111122223333.dkr.ecr.us-east-1.amazonaws.com/v2/: net/http: solicitação cancelada enquanto aguardava conexão
- Erro de API
- gravar /var/lib/docker/tmp/GetImageBlob111111111: nenhum espaço restante no dispositivo
- ERRO: toomanyrequests: excesso de solicitações ou você atingiu o limite de taxa de extração.
- Resposta de erro do daemon: obter url: net/http: solicitação cancelada enquanto aguardava conexão
- 1 nova tentativa de extração de ref.: falha ao copiar: httpReaderSeeker: falha ao abrir: código de status inesperado
- acesso para extração negado
- falha no comando pull: pânico: erro de runtime: endereço de memória inválido ou desreferência de ponteiro nulo
- erro ao extrair conf. de imagem/erro ao extrair configuração de imagem

- Contexto cancelado

A tarefa não consegue extrair a imagem. Verifique se o perfil tem as permissões para extrair imagens do registro.

Esse erro indica que a tarefa não consegue extrair a imagem especificada na definição de tarefa devido a problemas de permissão.

Para resolver esse problema:

1. Verifique se a imagem existe no *repositório*. Para obter informações sobre visualização das imagens, consulte Viewing image details in Amazon ECR no Amazon Elastic Container Registry User Guide.
2. Verifique se o *role-arn* tem as permissões corretas para extrair a imagem.

   Para obter informações sobre como atualizar perfis, consulte Atualizar permissões para um perfil no Guia do usuário do AWS Identity and Access Management.

   A tarefa usa um dos seguintes perfis:

   - Para tarefas com o tipo de inicialização do Fargate, esse é o perfil de execução de tarefa. Para obter informações sobre as permissões adicionais para o Amazon ECR, Tarefas do Fargate que extraem imagens do Amazon ECR pelos endpoints da interface.
   - Para tarefas com o tipo de inicialização do EC2, esse é o perfil de instância de contêiner. Para obter informações sobre as permissões adicionais para o Amazon ECR, Permissões do Amazon ECR.

A tarefa não pode extrair "***image-name***" do repositório "***repository URI***" do Amazon ECR. Há um problema de conexão entre a tarefa e o Amazon ECR. Verifique a configuração de rede da tarefa.

Esse erro indica que a tarefa não consegue se conectar ao Amazon ECR. Verifique a conexão com o repositório *repository URI*.

Para obter informações sobre como verificar e resolver o problema, consulte Verificar conectividade de tarefa interrompida do Amazon ECS.

A tarefa não consegue extrair a imagem. Verificar a configuração da rede

Esse erro indica que a tarefa não consegue se conectar ao Amazon ECR.

Para obter informações sobre como verificar e resolver o problema, consulte Verificar conectividade de tarefa interrompida do Amazon ECS.

CannotPullContainerError: o manifesto de extração da imagem foi repetido 5 vez(es): falha ao resolver referência

Esse erro indica que a tarefa não consegue extrair a imagem.

Para resolver isso, você pode:

- Verifique se a imagem especificada na definição de tarefa corresponde à imagem no repositório.
- O Amazon ECS força a estabilidade da versão da imagem. Se a imagem original não estiver mais disponível, você receberá esse erro. A tag de imagem faz parte da imposição desse comportamento. Altere a imagem na definição da tarefa de usar :latest como tag para uma versão específica. Para obter mais informações, consulte Resolução da imagem do contêiner.

Para obter informações sobre como verificar e resolver o problema, consulte Verificar conectividade de tarefa interrompida do Amazon ECS.

Erro de API (500): obter https://111122223333.dkr.ecr.us-east-1.amazonaws.com/v2/: net/http: solicitação cancelada enquanto aguardava conexão

Esse erro indica que a conexão atingiu o tempo limite porque não existe uma rota para a Internet.

Para resolver esse problema, você pode:

- Para tarefas em sub-redes públicas, especifique ENABLED (Habilitado) para Auto-assign public IP (Atribuir IP público automaticamente) ao iniciar a tarefa. Para obter mais informações, consulte Execução de uma aplicação como uma tarefa do Amazon ECS.
- Para tarefas em sub-redes privadas, especifique DISABLED (DESABILITADO) em Auto-assign public IP (Atribuir IP público automaticamente) ao iniciar a tarefa e configure um Gateway NAT na VPC para encaminhar solicitações para a Internet. Para obter mais informações, consulte Gateways NAT no Guia do usuário da Amazon VPC.

Erro de API

Esse erro indica que há um problema de conexão com o endpoint do Amazon ECR.

Para obter informações sobre como resolver esse problema, consulte Como posso resolver o erro do Amazon ECR “CannotPullContainerError: API error” no Amazon ECS? no site do Suporte.

gravar /var/lib/docker/tmp/***GetImageBlob111111111***: nenhum espaço restante no dispositivo

Esse erro indica que não há espaço suficiente em disco.

Para resolver esse problema, libere espaço em disco.

Se você estiver usando a API otimizada para o Amazon ECS, utilize o seguinte comando para recuperar os 20 maiores arquivos do sistema de arquivos:

```
du -Sh / | sort -rh | head -20
```

Resultado do exemplo:

```
5.7G    /var/lib/docker/
containers/50501b5f4cbf90b406e0ca60bf4e6d4ec8f773a6c1d2b451ed8e0195418ad0d2
1.2G    /var/log/ecs
594M    /var/lib/docker/devicemapper/mnt/
c8e3010e36ce4c089bf286a623699f5233097ca126ebd5a700af023a5127633d/rootfs/data/logs
...
```

Em alguns casos, o volume raiz pode ser preenchido por um contêiner em execução. Se o contêiner estiver usando o driver de log `json-file` padrão sem um limite `max-size`, o arquivo de log poderá ser responsável pela maioria do espaço usado. É possível usar o comando `docker ps` para verificar qual contêiner está usando o espaço mapeando o nome do diretório da saída acima para o ID do contêiner. Por exemplo:

```
CONTAINER ID   IMAGE                                COMMAND             CREATED
 STATUS              PORTS                              NAMES
50501b5f4cbf   amazon/amazon-ecs-agent:latest   "/agent"            4 days ago
 Up 4 days                                          ecs-agent
```

Por padrão, ao usar o driver de log `json-file`, o Docker registra a saída padrão (e o erro padrão) de todos os contêineres e a grava em arquivos usando o formato JSON. É possível definir o `max-size` como uma opção de driver de log, que impede que o arquivo de log ocupe muito espaço. Para obter mais informações, consulte Driver de registro em log de arquivo JSON na documentação do Docker.

Este é um trecho de definição de contêiner que mostra como usar essa opção:

```
{
```

```
    "log-driver": "json-file",
    "log-opts": {
        "max-size": "256m"
    }
}
```

Uma alternativa caso os logs de contêiner estejam ocupando muito espaço em disco é usar o driver de log `awslogs`. O driver de log `awslogs` envia os logs para o CloudWatch, que libera o espaço em disco que seria usado para os logs de contêiner na instância de contêiner. Para obter mais informações, consulte Envio de logs do Amazon ECS para o CloudWatch .

Você pode precisar atualizar o tamanho do disco que o Docker pode acessar.

Para obter mais informações, consulte CannotPullContainerError: no space left on device.

ERRO: toomanyrequests: excesso de solicitações ou você atingiu o limite de taxa de extração.

Esse erro indica que há uma limitação de taxa do Docker Hub.

Se você receber um dos seguintes erros, você provavelmente está atingindo os limites de taxa do Docker Hub:

Para obter mais informações sobre os limites de taxa do Docker Hub, consulte Entender a limitação de taxa do Docker Hub.

Se você tiver aumentado o limite da taxa do Docker Hub e precisar autenticar as extrações do Docker para as instâncias de contêiner, consulte Autenticação de registro privado para instâncias de contêiner.

Resposta de erro do daemon: obter ***`url`***: net/http: solicitação cancelada enquanto aguardava conexão

Esse erro indica que a conexão atingiu o tempo limite porque não existe uma rota para a Internet.

Para resolver esse problema, você pode:

- Para tarefas em sub-redes públicas, especifique ENABLED (Habilitado) para Auto-assign public IP (Atribuir IP público automaticamente) ao iniciar a tarefa. Para obter mais informações, consulte Execução de uma aplicação como uma tarefa do Amazon ECS.
- Para tarefas em sub-redes privadas, especifique DISABLED (DESABILITADO) em Auto-assign public IP (Atribuir IP público automaticamente) ao iniciar a tarefa e configure um Gateway NAT

na VPC para encaminhar solicitações para a Internet. Para obter mais informações, consulte Gateways NAT no Guia do usuário da Amazon VPC.

1 nova tentativa de extração de ref.: falha ao copiar: httpReaderSeeker: falha ao abrir: código de status inesperado

Esse erro indica que houve uma falha ao copiar uma imagem.

Para resolver esse problema, consulte um dos seguintes artigos:

- Para tarefas do Fargate, consulte Como resolvo o erro “cannotpullcontainererror” para minhas tarefas do Amazon ECS no Fargate.
- Para outras tarefas, consulte Como resolvo o erro “cannotpullcontainererror” para minhas tarefas do Amazon ECS.

acesso para extração negado

Esse erro indica que não há acesso à imagem.

Para resolver esse problema, talvez seja necessário autenticar o cliente Docker com o Amazon ECR. Para obter mais informações, consulte Private registry authentication no Guia do usuário do Amazon ECR.

falha no comando pull: pânico: erro de runtime: endereço de memória inválido ou desreferência de ponteiro nulo

Esse erro indica que não há acesso à imagem devido a um endereço de memória inválido ou à desreferência do ponteiro nulo.

Para resolver esse problema:

- Verifique se você tem as regras do grupo de segurança para acessar o Amazon S3.
- Ao usar endpoints de gateway, você deve adicionar uma rota na tabela de rotas para acessar o endpoint.

erro ao extrair conf. de imagem/erro ao extrair configuração de imagem

Este erro indica que um limite de taxa foi atingido ou que há um erro de rede:

Para resolver esse problema, consulte Como posso resolver o erro “CannotPullContainerError” na minha tarefa de tipo de inicialização do EC2 do Amazon ECS.

Contexto cancelado

Esse erro indica que o contexto foi cancelado.

Esse erro comumente ocorre porque a VPC que sua tarefa está usando não tem uma rota para extrair a imagem de contêiner do Amazon ECR.

Verificar conectividade de tarefa interrompida do Amazon ECS

Algumas vezes uma tarefa é interrompida devido a um problema de conectividade de rede. Esse problema pode ser intermitente, mas a causa mais provável é a tarefa não conseguir se conectar a um endpoint.

Testar a conectividade da tarefa

Você pode usar o runbook `AWSSupport-TroubleshootECSTaskFailedToStart` para testar a conectividade da tarefa. Para usar o runbook, você precisa das seguintes informações do recurso:

- O ID da tarefa

  Use o ID da tarefa mais recente em que houve falha.
- O cluster em que a tarefa estava

Para obter informações sobre como usar o runbook, consulte `AWSSupport-TroubleshootECSTaskFailedToStart` na Referência do runbook do AWS Systems Manager Automation.

O runbook analisa a tarefa. Você pode visualizar os resultados na seção Saída para os seguintes problemas que podem impedir que uma tarefa seja iniciada:

- Conectividade de rede com o registro de contêiner configurado
- Conectividade do serviço de endpoint da VPC
- Configuração de regras do grupo de segurança

Corrigir problemas de endpoint da VPC

Quando o resultado do runbook `AWSSupport-TroubleshootECSTaskFailedToStart` indicar um problema do endpoint da VPC, verifique a seguinte configuração:

- A VPC em que você cria o endpoint precisa usar DNS privado.
- Certifique-se de ter um endpoint AWS PrivateLink para o serviço ao qual a tarefa não consegue se conectar na mesma VPC da tarefa. Para obter mais informações, consulte um dos seguintes tópicos:

Serviço	Informações sobre o endpoint da VPC para o serviço
Amazon ECR	Endpoints da VPC de interface do Amazon ECS (AWS PrivateLink)
Systems Manager	Melhorar a segurança das instâncias do EC2 usando endpoints da VPC para o Systems Manager
Secrets Manager	Usar um endpoint da VPC do AWS Secrets Manager
CloudWatch	Endpoint da VPC do CloudWatch
Amazon S3	AWS PrivateLink for Amazon S3

- Configure uma regra de saída para a sub-rede da tarefa que permita HTTPS no tráfego DNS da porta 443 (TCP). Para obter mais informações, consulte Configurar regras de grupo de segurança no Guia do usuário da Amazon Elastic Compute Cloud.
- Se você usa um servidor de nome de domínio personalizado, confirme as configurações da consulta ao DNS. A consulta deve ter acesso de saída na porta 53 e usar os protocolos UDP e TCP. Além disso, ela deve ter acesso HTTPS na porta 443. Para obter mais informações, consulte Configurar regras de grupo de segurança no Guia do usuário da Amazon Elastic Compute Cloud.

- Se a sub-rede tiver uma ACL de rede, as seguintes regras de ACL serão necessárias:
  - Uma regra de saída que permita tráfego nas portas 1024-65535.
  - Adicione uma regra de entrada que permita o tráfego TCP na porta 443.

  Para obter informações sobre como configurar regras, consulte Controlar o tráfego para sub-redes com ACLs de rede no Guia do usuário da Amazon Virtual Private Cloud.

Corrigir problemas de rede

Quando o resultado do runbook `AWSSupport-TroubleshootECSTaskFailedToStart` indicar um problema rede, verifique a seguinte configuração:

Tarefas que usam o modo de rede awsvpc em uma sub-rede pública

Faça a configuração a seguir baseada no runbook:

- Para tarefas em sub-redes públicas, especifique ENABLED (Habilitado) para Auto-assign public IP (Atribuir IP público automaticamente) ao iniciar a tarefa. Para obter mais informações, consulte Execução de uma aplicação como uma tarefa do Amazon ECS.
- Você precisa de um gateway para lidar com o tráfego da Internet. A tabela de rotas para a sub-rede da tarefa precisa ter uma rota para o tráfego destinado ao gateway.

  Para obter mais informações, consulte Adicionar e remover rotas de uma tabela de rotas no Manual do usuário da Amazon Virtual Private Cloud.

Tipo de gateway	Destino da tabela de rotas	Alvo da tabela de rotas
NAT	0.0.0.0/0	ID do gateway NAT
Gateway da Internet	0.0.0.0/0	ID do gateway da Internet

- Se a sub-rede da tarefa tiver uma ACL de rede, as seguintes regras de ACL serão necessárias:
  - Uma regra de saída que permite tráfego nas portas de 1024 a 65535.
  - Adicione uma regra de entrada que permita o tráfego TCP na porta 443.

  Para obter informações sobre como configurar regras, consulte Controlar o tráfego para sub-redes com ACLs de rede no Guia do usuário da Amazon Virtual Private Cloud.

Tarefas que usam o modo de rede awsvpc em uma sub-rede privada

Faça a configuração a seguir baseada no runbook:

- Escolha DESABILITADO para atribuir automaticamente um IP público ao iniciar a tarefa.
- Configure um gateway NAT na VPC para rotear solicitações para a Internet. Para obter mais informações, consulte Gateways NAT no Guia do usuário do Amazon Virtual Private Cloud.
- A tabela de rotas para a sub-rede da tarefa precisa ter uma rota para o tráfego destinado ao gateway NAT.

  Para obter mais informações, consulte Adicionar e remover rotas de uma tabela de rotas no Manual do usuário da Amazon Virtual Private Cloud.

Tipo de gateway	Destino da tabela de rotas	Alvo da tabela de rotas
NAT	0.0.0.0/0	ID do gateway NAT

- Se a sub-rede da tarefa tiver uma ACL de rede, as seguintes regras de ACL serão necessárias:
  - Uma regra de saída que permite tráfego nas portas de 1024 a 65535.
  - Adicione uma regra de entrada que permita o tráfego TCP na porta 443.

  Para obter informações sobre como configurar regras, consulte Controlar o tráfego para sub-redes com ACLs de rede no Guia do usuário da Amazon Virtual Private Cloud.

Tarefas que não usam o modo de rede awsvpc em uma sub-rede pública

Faça a configuração a seguir baseada no runbook:

- Escolha Ativar para Atribuir IP automaticamente em Rede para instâncias do Amazon EC2 ao criar o cluster.

  Essa opção atribui um endereço IP público à interface de rede primária da instância.
- Você precisa de um gateway para lidar com o tráfego da Internet. A tabela de rotas para a sub-rede da instância precisa ter uma rota para o tráfego destinado ao gateway.

  Para obter mais informações, consulte Adicionar e remover rotas de uma tabela de rotas no Manual do usuário da Amazon Virtual Private Cloud.

Tipo de gateway	Destino da tabela de rotas	Alvo da tabela de rotas
NAT	0.0.0.0/0	ID do gateway NAT
Gateway da Internet	0.0.0.0/0	ID do gateway da Internet

- Se a sub-rede da instância tiver uma ACL de rede, as seguintes regras de ACL serão necessárias:
  - Uma regra de saída que permite tráfego nas portas de 1024 a 65535.
  - Adicione uma regra de entrada que permita o tráfego TCP na porta 443.

  Para obter informações sobre como configurar regras, consulte Controlar o tráfego para sub-redes com ACLs de rede no Guia do usuário da Amazon Virtual Private Cloud.

Tarefas que não usam o modo de rede awsvpc em uma sub-rede privada

Faça a configuração a seguir baseada no runbook:

- Escolha Desativar para Atribuir IP automaticamente em Redes para instâncias do Amazon EC2 ao criar o cluster.
- Configure um gateway NAT na VPC para rotear solicitações para a Internet. Para obter mais informações, consulte Gateways NAT no Guia do usuário da Amazon VPC.
- A tabela de rotas para a sub-rede da instância precisa ter uma rota para o tráfego destinado ao gateway NAT.

  Para obter mais informações, consulte Adicionar e remover rotas de uma tabela de rotas no Manual do usuário da Amazon Virtual Private Cloud.

Tipo de gateway	Destino da tabela de rotas	Alvo da tabela de rotas
NAT	0.0.0.0/0	ID do gateway NAT

- Se a sub-rede da tarefa tiver uma ACL de rede, as seguintes regras de ACL serão necessárias:
  - Uma regra de saída que permite tráfego nas portas de 1024 a 65535.
  - Adicione uma regra de entrada que permita o tráfego TCP na porta 443.

  Para obter informações sobre como configurar regras, consulte Controlar o tráfego para sub-redes com ACLs de rede no Guia do usuário da Amazon Virtual Private Cloud.

Visualização de solicitações de perfil do IAM para tarefas do Amazon ECS

Quando você usa um provedor para as credenciais de tarefa em um perfil do IAM, as solicitações do provedor são salvas em um log de auditoria. O log de auditoria herda as mesmas configurações de rotação de log que o log do agente de contêiner. As variáveis de configuração do agente de contêiner `ECS_LOG_ROLLOVER_TYPE`, `ECS_LOG_MAX_FILE_SIZE_MB` e `ECS_LOG_MAX_ROLL_COUNT` podem ser definidas para afetar o comportamento do log de auditoria. Para obter mais informações, consulte Parâmetros de configuração do log do agente de contêiner do Amazon ECS.

Para o agente de contêiner versão 1.36.0 e posteriores, o log de auditoria está localizado em `/var/log/ecs/audit.log`. Quando o log é girado, um timestamp no formato *`YYYY-MM-DD-HH`* é adicionado ao final do nome do arquivo de log.

Para o agente de contêiner versão 1.35.0 e anteriores, o log de auditoria está localizado em `/var/log/ecs/audit.log.`*`YYYY-MM-DD-HH`*.

O formato de entrada do log é o seguinte:

- Timestamp
- Código de resposta HTTP
- Endereço IP e número de porta de origem da solicitação
- URI relativo do provedor de credencial
- O agente do usuário que fez a solicitação
- O ARN da tarefa ao qual o contêiner solicitante pertence
- O nome e o número da versão da API `GetCredentials`
- O nome do cluster do Amazon ECS no qual a instância de contêiner é registrada
- O ARN de instância do contêiner

É possível usar o comando a seguir para visualizar os arquivos de log.

```
cat /var/log/ecs/audit.log.2016-07-13-16
```

Saída:

```
2016-07-13T16:11:53Z 200 172.17.0.5:52444 "/v1/credentials" "python-requests/2.7.0
 CPython/2.7.6 Linux/4.4.14-24.50.amzn1.x86_64" TASK_ARN GetCredentials
 1 CLUSTER_NAME CONTAINER_INSTANCE_ARN
```

Visualizar mensagens de eventos de serviço do Amazon ECS

Se você estiver solucionando um problema em um serviço, o primeiro lugar em que você deverá procurar informações de diagnóstico é o log de eventos do serviço. É possível visualizar eventos de serviço usando a API `DescribeServices`, a AWS CLI a opção AWS Management Console.

Ao visualizar mensagens de evento de serviço usando a API do Amazon ECS, somente os eventos do programador de serviço são retornados. Esses eventos incluem o posicionamento de tarefas e eventos de integridade da instância mais recentes. No entanto, o console do Amazon ECS exibe eventos de serviço das fontes a seguir.

- Posicionamento de tarefas e eventos de integridade da instância do programador de serviço do Amazon ECS. Esses eventos têm o prefixo serviço ***(service-name)***. Para garantir que esta visualização de eventos seja útil, mostramos somente os `100` eventos mais recentes. As mensagens de evento de serviço duplicadas são omitidas até que a causa seja resolvida ou depois de transcorridas seis horas. Se a causa não for resolvida em até seis horas, você receberá outra mensagem de evento do serviço referente a essa causa.
- Eventos de Auto Scaling do serviço. Esses eventos têm um prefixo Message e ocorrem apenas quando um serviço é configurado com uma política de escalação do Application Auto Scaling.

Use as etapas a seguir para visualizar as mensagens de evento de serviço atuais.

Console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Clusters.
3. Na página Clusters, escolha o cluster.
4. Escolha o serviço a ser inspecionado.
5. Escolha Deployments and events (Implantações e eventos), em Events (Eventos), visualize as mensagens.

AWS CLI

Use o comando describe-serviços para visualizar as mensagens de evento de serviço para um serviço especificado.

O exemplo de AWS CLI a seguir descreve o serviço *service-name* no cluster *padrão*, que fornece as mensagens de evento de serviço mais recentes.

```
aws ecs describe-services \
    --cluster default \
    --services service-name \
    --region us-west-2
```

Mensagens de eventos do serviço do Amazon ECS

Veja a seguir exemplos de mensagens de eventos de serviço que você pode ver no console do Amazon ECS.

O serviço (*service-name*) atingiu um estado estável.

O programador de serviços envia um evento de serviço `service (service-name) has reached a steady state.` quando o serviço está íntegro e no número desejado de tarefas, atingindo assim um estado estável.

O programador de serviços relata o status periodicamente, portanto, você pode receber essa mensagem várias vezes.

O serviço (*service-name*) não conseguiu fazer uma tarefa porque nenhuma instância de contêiner atendeu a todos os requisitos.

O programador de serviços envia essa mensagem de evento quando não consegue encontrar os recursos disponíveis para adicionar outra tarefa. As causas possíveis para isso são:

Use provedores de capacidade para escalar automaticamente suas instâncias do EC2. Para obter mais informações, consulte Provedores de capacidade do Amazon ECS para o tipo de inicialização do EC2 .

Se você pretende usar um provedor de capacidade, certifique-se de estar passando uma estratégia de provedor de capacidade ou de ter uma estratégia de provedor de capacidade

padrão associada ao seu cluster e não esteja passando o tipo de inicialização e a estratégia do provedor de capacidade como entrada

Não foram encontradas instâncias de contêiner no cluster

Se não houver instâncias de contêiner registradas no cluster no qual tenta executar uma tarefa, você receberá esse erro. Você deve adicionar instâncias de contêiner ao seu cluster. Para obter mais informações, consulte Iniciar uma instância de contêiner do Linux do Amazon ECS.

Não há portas suficientes

Caso a tarefa use um mapeamento de porta host fixo (por exemplo, a tarefa usa a porta 80 no host para um servidor da web), você deve ter pelo menos uma instância de contêiner por tarefa, porque apenas um contêiner pode usar uma única porta host por vez. Você deve adicionar instâncias de contêiner ao cluster ou reduzir o número de tarefas desejadas.

Muitas portas registradas

A instância de contêiner com maior correspondência para o posicionamento de tarefas não pode exceder o limite máximo permitido de porta reservada de cem portas de host por instância de contêiner. O uso do mapeamento de porta de host dinâmico pode corrigir o problema.

Porta já em uso

A definição de tarefa dessa tarefa usa a mesma porta no mapeamento de porta que uma tarefa já em execução na instância de contêiner escolhida. A mensagem de evento de serviço teria o ID da instância de contêiner escolhido como parte da mensagem abaixo.

```
The closest matching container-instance is already using a port required by your
 task.
```

Não há memória suficiente

Caso a definição de tarefa especifique 1.000 MiB de memória e as instâncias de contêiner no cluster tenham 1.024 MiB de memória cada, você só pode executar uma cópia dessa tarefa por instância de contêiner. É possível experimentar com menos memória na definição de tarefa, de maneira que possa ativar mais de uma tarefa por instância de contêiner ou ativar mais instâncias de contêiner no cluster.

> **Note**
>
> Se você deseja maximizar a utilização de recursos fornecendo às tarefas o máximo de memória possível para um determinado tipo de instância, consulte Reserva de memória da instância de contêiner do Linux no Amazon ECS.

Não há CPU suficiente

Uma instância de contêiner tem 1.024 unidades de CPU para cada núcleo de CPU. Caso a definição de tarefa especifique 1.000 unidades de CPU e as instâncias de contêiner no cluster tenham 1.024 unidades de CPU, você só pode executar uma cópia dessa tarefa por instância de contêiner. É possível experimentar com um número menor de unidades de CPU na definição de tarefa, de maneira que possa ativar mais de uma tarefa por instância de contêiner ou ativar mais instâncias de contêineres no cluster.

Não há número suficiente de pontos de conexão ENI disponíveis

Tarefas que usam o modo de rede `awsvpc` recebem cada uma sua própria interface de rede elástica (ENI), que é conectada à instância de contêiner que as hospeda. As instâncias do Amazon EC2 têm um limite para o número de ENIs que podem ser anexadas a elas e não há instâncias de contêiner no cluster que tenham capacidade de ENI disponível.

O limite de ENI para instâncias de contêiner individuais depende das seguintes condições:

- Se você não tiver aceitado a configuração de conta `awsvpcTrunking`, o limite de ENI para cada instância de contêiner dependerá do tipo de instância. Para obter mais informações, consulte Endereços IP por interface de rede por tipo de instância no Manual do usuário do Amazon EC2.
- Se você aceitou a configuração de conta `awsvpcTrunking`, mas não executou novas instâncias de contêiner usando um tipo de instância com suporte depois de aceitar a configuração, o limite de ENI para cada instância de contêiner ainda está no valor padrão. Para obter mais informações, consulte Endereços IP por interface de rede por tipo de instância no Manual do usuário do Amazon EC2.
- Se você aceitou a configuração de conta `awsvpcTrunking` e executou novas instâncias de contêiner usando um tipo suportado depois de aceitar a configuração, estarão disponíveis outras ENIs. Para obter mais informações, consulte Instâncias com suporte para o aumento de interfaces de rede de contêineres do Amazon ECS.

Para obter mais informações sobre como optar pela configuração de conta `awsvpcTrunking`, consulte Aumento das interfaces de rede de instâncias de contêiner do Linux no Amazon ECS.

É possível adicionar instâncias de contêiner ao seu cluster para fornecer mais adaptadores de rede disponíveis.

Instância de contêiner sem o atributo obrigatório

Alguns parâmetros de definição de tarefa exigem uma versão de API remota do Docker específica a ser instalada na instância de contêiner. Outros, como as opções de driver de registro em log, exigem que as instâncias de contêiner registrem esses drivers de log com a variável de configuração do agente `ECS_AVAILABLE_LOGGING_DRIVERS`. Caso a definição de tarefa contenha um parâmetro que exija um atributo de instância de contêiner específico e você não tenha instâncias de contêiner disponíveis que possam atender a esse requisito, a tarefa não poderá ser realizada.

Uma causa comum desse erro é o serviço estar usando tarefas que utilizam o modo de rede `awsvpc` e o tipo de execução do EC2. O cluster especificado não tem uma instância de contêiner registrada na mesma sub-rede especificada em `awsvpcConfiguration` quando o serviço foi criado.

É possível usar o runbook AWSSupport-TroubleshootecsContainerInstance para solucionar problemas. O runbook analisa se os dados do usuário da instância contêm as informações corretas do cluster, se o perfil de instância contém as permissões necessárias e problemas de configuração de rede. Para obter mais informações, consulte AWSSupport-TroubleshootECSContainerInstance no Guia do usuário de referência do runbook do AWS Systems Manager Automation.

Para obter mais informações sobre quais atributos são obrigatórios para parâmetros de definição de tarefa específicos e variáveis de configuração do agente, consulte Parâmetros de definição de tarefa do Amazon ECS para o tipo de inicialização do Fargate e Configuração do agente de contêiner do Amazon ECS.

O serviço (*service-name*) não conseguiu fazer uma tarefa porque nenhuma instância de contêiner atendeu a todos os requisitos. O *container-instance-id* de instância de contêiner correspondente mais próximo tem unidades de CPU insuficientes disponíveis.

A instância de contêiner com maior correspondência para o posicionamento de tarefas não contém unidades de CPU suficientes para atender aos requisitos na definição de tarefa. Revise os requisitos de CPU nos parâmetros de definição de contêiner e tamanho de tarefa da definição de tarefa.

O serviço (*service-name*) não conseguiu fazer uma tarefa porque nenhuma instância de contêiner atendeu a todos os requisitos. O container-instance mais próximo *container-instance-id* encontrou um erro "AGENT".

O agente de contêiner do Amazon ECS na instância de contêiner com correspondência mais próxima para a realização da tarefa está desconectado. Caso possa se conectar à instância de contêiner usando SSH, você pode examinar os logs de agente. Para obter mais informações, consulte Parâmetros de configuração do log do agente de contêiner do Amazon ECS. Você também deve verificar se o agente está em execução na instância. Se você estiver usando a AMI otimizada para Amazon ECS, poderá tentar interromper e reiniciar o agente com o comando a seguir.

- Para a AMI do Amazon Linux 2 otimizada para o Amazon ECS e a AMI do Amazon Linux 2023 otimizada para o Amazon ECS

  ```
  sudo systemctl restart ecs
  ```

- Para a AMI do Amazon Linux otimizada para o Amazon ECS

  ```
  sudo stop ecs && sudo start ecs
  ```

O serviço (*service-name*) (tarefa *task-id*) (instância *instance-id*) está sem integridade em (elb *elb-name*) por causa de (motivo da falha na instância no mínimo no número de verificações de integridade UnhealthyThreshold consecutivas.)

O serviço é registrado com um load balancer, e as verificações de integridade do load balancer apresentam falhas. A mensagem inclui o ID da tarefa para ajudar a identificar qual tarefa específica está falhando nas verificações de integridade. Para obter mais informações, consulte Solução de problemas relacionados aos balanceadores de carga de serviço no Amazon ECS.

O serviço (***service-name***) não consegue iniciar tarefas com êxito de maneira consistente.

Esse serviço contém tarefas que deixaram de ser iniciadas após tentativas consecutivas. Nesse ponto, o programador de serviço começa a aumentar incrementalmente o tempo entre as novas tentativas. Você deve solucionar o motivo pelo qual suas tarefas falham ao iniciar. Para obter mais informações, consulte Lógica de controle de utilização do serviço do Amazon ECS.

Depois que o serviço estiver atualizado, por exemplo, com uma definição de tarefa atualizada, o programador de serviços retomará o comportamento normal.

As operações do serviço (***service-name***) estão tendo a utilização controlada. Tentará novamente mais tarde.

Este serviço não consegue iniciar mais tarefas devido aos limites do controle de utilização da API. Como o programador de serviço é capaz de iniciar mais tarefas, ele é retomado.

Para solicitar um aumento na cota de limite da taxa de API, abra a página AWS Support Center, faça login, se necessário, e escolha Create case (Criar caso). Escolha Service limit increase (Aumento de limite do serviço). Preencha e envie o formulário.

O serviço (***service-name***) não conseguiu interromper ou iniciar tarefas durante uma implantação devido à configuração de implantação do serviço. Atualize o valor minimumHealthyPercent ou maximumPercent e tente novamente.

Esse serviço não conseguiu interromper ou iniciar tarefas durante uma implantação de serviço devido à configuração da implantação. A configuração de implantação consiste nos valores `minimumHealthyPercent` e `maximumPercent`, que são definidos quando o serviço é criado. Esses valores também podem ser atualizados em um serviço existente.

O valor `minimumHealthyPercent` representa o limite inferior do número de tarefas que devem estar sendo executadas em um serviço durante uma implantação ou quando uma instância de contêiner está sendo drenada. É uma porcentagem do número de tarefas desejadas para o serviço. Esse valor é arredondado para cima. Por exemplo, se a porcentagem mínima de integridade for `50` e a contagem de tarefas desejadas for quatro, o programador poderá interromper duas tarefas existentes antes de iniciar duas novas tarefas. Da mesma forma, se a porcentagem mínima de integridade é 75% e a contagem de tarefas desejada é dois, o programador não pode parar quaisquer tarefas porque o valor resultante também é dois.

O valor `maximumPercent` representa o limite superior do número de tarefas que devem estar sendo executadas em um serviço durante uma implantação ou quando uma instância de contêiner está sendo drenada. É uma porcentagem do número de tarefas desejadas para um serviço. Esse valor é arredondado para baixo. Por exemplo, se a porcentagem máxima for `200` e a contagem de tarefas desejadas for quatro, o programador poderá iniciar quatro novas tarefas antes de interromper quatro tarefas existentes. Da mesma forma, se a porcentagem máxima de integridade é `125` e a contagem de tarefas desejada é três, o programador não pode iniciar quaisquer tarefas porque o valor resultante também é três.

Ao definir um percentual mínimo de integridade ou um percentual máximo, você deve garantir que o programador possa interromper ou iniciar pelo menos uma tarefa quando uma implantação é acionada.

O serviço (***service-name***) não conseguiu posicionar uma tarefa. Motivo: você atingiu o limite do número de tarefas que podem ser executadas simultaneamente

É possível solicitar um aumento de cota para o recurso que causou o erro. Para obter mais informações, consulte Cotas de serviço. Para solicitar um aumento da cota, consulte Requesting a quota increase no Guia do usuário do Service Quotas.

O serviço (***service-name***) não conseguiu posicionar uma tarefa. Motivo: erro interno.

O possível motivo para esse erro é o seguinte:

O serviço não consegue iniciar uma tarefa devido a uma sub-rede estar em uma zona de disponibilidade sem suporte.

Para obter mais informações sobre as regiões do Fargate e zonas de disponibilidade com suporte, consulte the section called “Regiões do AWS Fargate”.

Para obter informações sobre como visualizar a zona de disponibilidade de sub-rede, consulte Visualizar sua sub-rede no Guia do usuário da Amazon VPC.

O serviço (***service-name***) não conseguiu posicionar uma tarefa. Motivo: a configuração de CPU solicitada está acima do limite.

É possível solicitar um aumento de cota para o recurso que causou o erro. Para obter mais informações, consulte Cotas de serviço. Para solicitar um aumento da cota, consulte Requesting a quota increase no Guia do usuário do Service Quotas.

O serviço (*service-name*) não conseguiu posicionar uma tarefa. Motivo: A configuração MEMORY solicitada está acima do limite.

É possível solicitar um aumento de cota para o recurso que causou o erro. Para obter mais informações, consulte Cotas de serviço. Para solicitar um aumento da cota, consulte Requesting a quota increase no Guia do usuário do Service Quotas.

O serviço (*service-name*) não conseguiu posicionar uma tarefa. Motivo: você atingiu o limite do número de vCPUs que pode executar simultaneamente

O AWS Fargate está fazendo a transição de cotas baseadas em contagem de tarefas para cotas baseadas em vCPU.

É possível solicitar um aumento de cota para a cota baseada em vCPU do Fargate. Para obter mais informações, consulte Cotas de serviço. Para solicitar o aumento da cota do Fargate, consulte Requesting a Quota Increase (Solicitar um aumento de cota) no Guia do usuário do Service Quotas.

O serviço (*service-name*) não conseguiu atingir o estado estacionário porque o conjunto de tarefas (*taskSet-ID*) não conseguiu reduzir a escala horizontalmente. Motivo: o número de tarefas protegidas é maior do que a contagem desejada de tarefas

O serviço tem mais tarefas protegidas do que o número desejado de tarefas. É possível executar uma das ações a seguir:

- Aguarde até que a proteção das tarefas atuais expire, permitindo que elas sejam encerradas.
- Determine quais tarefas podem ser interrompidas e use a API `UpdateTaskProtection` com a opção `protectionEnabled` definida como `false` para cancelar a proteção dessas tarefas.
- Aumente a contagem de tarefas desejada do serviço para mais do que o número de tarefas protegidas.

O serviço (*nome-do-serviço*) foi incapaz de atingir o estado estável. Motivo: nenhuma instância de contêiner foi encontrada em seu provedor de capacidade.

O programador de serviços envia essa mensagem de evento quando não consegue encontrar os recursos disponíveis para adicionar outra tarefa. As causas possíveis para isso são:

Não há provedor de capacidade associado ao cluster

Use `describe-services` para verificar se há um provedor de capacidade associado ao cluster. Você pode atualizar a estratégia do provedor de capacidade do serviço.

Verifique se há capacidade disponível no provedor de capacidade. No caso do tipo de execução do EC2, certifique-se de que as instâncias de contêiner atendam aos requisitos de definição de tarefas.

Não foram encontradas instâncias de contêiner no cluster

Se não houver instâncias de contêiner registradas no cluster no qual tenta executar uma tarefa, você receberá esse erro. Você deve adicionar instâncias de contêiner ao seu cluster. Para obter mais informações, consulte Iniciar uma instância de contêiner do Linux do Amazon ECS.

Não há portas suficientes

Caso a tarefa use um mapeamento de porta de host fixo (por exemplo, a tarefa usa a porta 80 no host de um servidor Web), você deve ter pelo menos uma instância de contêiner por tarefa. Somente um contêiner pode usar uma única porta de host por vez. Você deve adicionar instâncias de contêiner ao cluster ou reduzir o número de tarefas desejadas.

Muitas portas registradas

A instância de contêiner com maior correspondência para o posicionamento de tarefas não pode exceder o limite máximo permitido de porta reservada de cem portas de host por instância de contêiner. O uso do mapeamento de porta de host dinâmico pode corrigir o problema.

Porta já em uso

A definição de tarefa dessa tarefa usa a mesma porta no mapeamento de porta que uma tarefa já em execução na instância de contêiner escolhida. A mensagem de evento de serviço teria o ID da instância de contêiner escolhido como parte da mensagem abaixo.

```
The closest matching container-instance is already using a port required by your
 task.
```

Não há memória suficiente

Caso a definição de tarefa especifique 1.000 MiB de memória e as instâncias de contêiner no cluster tenham 1.024 MiB de memória cada, você só pode executar uma cópia dessa tarefa por instância de contêiner. É possível experimentar com menos memória na definição de tarefa, de

maneira que possa ativar mais de uma tarefa por instância de contêiner ou ativar mais instâncias de contêiner no cluster.

> **Note**
>
> Se você deseja maximizar a utilização de recursos fornecendo às tarefas o máximo de memória possível para um determinado tipo de instância, consulte Reserva de memória da instância de contêiner do Linux no Amazon ECS.

Não há número suficiente de pontos de conexão ENI disponíveis

Tarefas que usam o modo de rede `awsvpc` recebem cada uma sua própria interface de rede elástica (ENI), que é conectada à instância de contêiner que as hospeda. As instâncias do Amazon EC2 têm um limite para o número de ENIs que podem ser anexadas a elas e não há instâncias de contêiner no cluster com capacidade de ENI disponível.

O limite de ENI para instâncias de contêiner individuais depende das seguintes condições:

- Se você não tiver aceitado a configuração de conta `awsvpcTrunking`, o limite de ENI para cada instância de contêiner dependerá do tipo de instância. Para obter mais informações, consulte Endereços IP por interface de rede por tipo de instância no Manual do usuário do Amazon EC2.
- Se você aceitou a configuração de conta `awsvpcTrunking`, mas não executou novas instâncias de contêiner usando um tipo de instância com suporte depois de aceitar a configuração, o limite de ENI para cada instância de contêiner ainda está no valor padrão. Para obter mais informações, consulte Endereços IP por interface de rede por tipo de instância no Manual do usuário do Amazon EC2.
- Se você aceitou a configuração de conta `awsvpcTrunking` e executou novas instâncias de contêiner usando um tipo suportado depois de aceitar a configuração, estarão disponíveis outras ENIs. Para obter mais informações, consulte Instâncias com suporte para o aumento de interfaces de rede de contêineres do Amazon ECS.

Para obter mais informações sobre como optar pela configuração de conta `awsvpcTrunking`, consulte Aumento das interfaces de rede de instâncias de contêiner do Linux no Amazon ECS.

É possível adicionar instâncias de contêiner ao seu cluster para fornecer mais adaptadores de rede disponíveis.

Instância de contêiner sem o atributo obrigatório

Alguns parâmetros de definição de tarefa exigem uma versão de API remota do Docker específica a ser instalada na instância de contêiner. Outros, como as opções de driver de registro em log, exigem que as instâncias de contêiner registrem esses drivers de log com a variável de configuração do agente `ECS_AVAILABLE_LOGGING_DRIVERS`. Caso a definição de tarefa contenha um parâmetro que exija um atributo de instância de contêiner específico e você não tenha instâncias de contêiner disponíveis que possam atender a esse requisito, a tarefa não poderá ser realizada.

Uma causa comum desse erro é o serviço estar usando tarefas que utilizam o modo de rede `awsvpc` e o tipo de execução do EC2, e o cluster especificado não ter uma instância de contêiner registrada na mesma sub-rede especificada na `awsvpcConfiguration` quando o serviço foi criado.

É possível usar o runbook AWSSupport-TroubleshootecsContainerInstance para solucionar problemas. O runbook analisa se os dados do usuário da instância contêm as informações corretas do cluster, se o perfil de instância contém as permissões necessárias e problemas de configuração de rede. Para obter mais informações, consulte AWSSupport-TroubleshootECSContainerInstance no Guia do usuário de referência do runbook do AWS Systems Manager Automation.

Para obter mais informações sobre quais atributos são obrigatórios para parâmetros de definição de tarefa específicos e variáveis de configuração do agente, consulte Parâmetros de definição de tarefa do Amazon ECS para o tipo de inicialização do Fargate e Configuração do agente de contêiner do Amazon ECS.

O serviço (***`service-name`***) não conseguiu posicionar uma tarefa. Motivo: a capacidade não está disponível no momento. Tente novamente mais tarde ou em uma zona de disponibilidade diferente.

No momento, não há capacidade disponível para executar seu serviço.

É possível executar uma das ações a seguir:

- Aguarde até que a capacidade do Fargate ou as instâncias de contêiner do EC2 estejam disponíveis.
- Reinicie o serviço e especifique sub-redes adicionais.

falha na implantação do serviço (*service-name*): houve falha nas tarefas ao iniciar.

Houve falha ao iniciar as tarefas no serviço.

Para obter informações sobre como depurar tarefas interrompidas, consulte Mensagens de erro de tarefa interrompida do Amazon ECS.

O serviço (*service-name*) atingiu o tempo limite aguardando o início do agente do Amazon ECS. Verifique os logs em /var/log/ecs/ecs-agent.log".

O agente de contêiner do Amazon ECS na instância de contêiner com correspondência mais próxima para a realização da tarefa está desconectado. Caso possa se conectar à instância de contêiner usando SSH, você pode examinar os logs de agente. Para obter mais informações, consulte Parâmetros de configuração do log do agente de contêiner do Amazon ECS. Você também deve verificar se o agente está em execução na instância. Se você estiver usando a AMI otimizada para Amazon ECS, poderá tentar interromper e reiniciar o agente com o comando a seguir.

- Para a AMI do Amazon Linux 2 otimizada para o Amazon ECS

```
sudo systemctl restart ecs
```

- Para a AMI do Amazon Linux otimizada para o Amazon ECS

```
sudo stop ecs && sudo start ecs
```

O conjunto de tarefas (*taskSet-ID*) (tarefa *task-id*) do serviço (*service-name*) não está íntegro no grupo de destino (*targetGroup-ARN*) devido a **TARGET GROUP IS NOT FOUND**.

A tarefa definida para o serviço está falhando nas verificações de integridade porque o grupo de destino não foi encontrado. A mensagem inclui o ID da tarefa para ajudar a identificar qual tarefa específica está falhando nas verificações de integridade. Você precisa excluir e recriar o serviço. Não exclua nenhum grupo de destino do Elastic Load Balancing, a menos que o serviço correspondente do Amazon ECS já tenha sido excluído.

O conjunto de tarefas (***taskSet-ID***) (tarefa ***task-id***) do serviço (***service-name***) não está íntegro no grupo de destino (***targetGroup-ARN***) devido a **`TARGET IS NOT FOUND`**.

A tarefa definida para o serviço está falhando nas verificações de integridade porque o destino não foi encontrado. A mensagem inclui o ID da tarefa para ajudar a identificar qual tarefa específica está falhando nas verificações de integridade.

Mensagens de evento não íntegro de serviço do Amazon ECS

Os exemplos a seguir são mensagens de evento não íntegro de serviço.

EC2: (serviço ***service-name***) (tarefa ***task-id***) (instância ***instance-id***) (porta ***port-number***) não está íntegro em (grupo de destino ***target-group***) devido a (razão ***failure-reason***)

Essa mensagem indica que uma tarefa em execução em uma instância do EC2 está falhando nas verificações de integridade. Para obter mais informações, consulte:

- Como faço para que minhas tarefas do Amazon ECS que usam o tipo de execução do Amazon EC2 sejam aprovadas na verificação de integridade do Application Load Balancer?

EC2 com taskSet: (serviço ***service-name***, taskSet ***taskSet-id***) (tarefa ***task-id***) (instância ***instance-id***) (porta ***port-number***) não está íntegro em (grupo de destino ***target-group-name***) devido a (razão ***failure-reason***)

Essa mensagem indica que uma tarefa em um conjunto de tarefas em execução em uma instância do EC2 está falhando nas verificações de integridade. Para obter mais informações, consulte:

- Como faço para que minhas tarefas do Amazon ECS que usam o tipo de execução do Amazon EC2 sejam aprovadas na verificação de integridade do Application Load Balancer?

Fargate: (serviço ***service-name***) (tarefa ***task-id***) (porta ***port-number***) não está íntegro em (grupo de destino ***target-group-name***) devido a (razão ***failure-reason***)

Essa mensagem indica que uma tarefa do Fargate está falhando nas verificações de integridade.

Para obter mais informações sobre como solucionar falhas de verificação de integridade em tarefas do Fargate, consulte Como solucionar falhas na verificação de integridade de tarefas do Amazon ECS no Fargate?.

Fargate com taskSet: (serviço ***service-name***, taskSet ***taskSet-id***) (tarefa ***task-id***) (porta ***port-number***) não está íntegro em (grupo de destino ***target-group-name***) devido a (razão ***failure-reason***)

Essa mensagem indica que uma tarefa em um conjunto de tarefas em execução no Fargate está falhando nas verificações de integridade.

Para obter mais informações sobre como solucionar falhas de verificação de integridade em tarefas do Fargate, consulte Como solucionar falhas na verificação de integridade de tarefas do Amazon ECS no Fargate?.

Mensagens de eventos do serviço de rebalanceamento do serviço de zonas de disponibilidade do Amazon ECS

Os exemplos a seguir são mensagens de eventos de serviços que poderão ser mostradas.

O serviço (***service-name***) não está balanceado em AZ com ***number-tasks*** tarefas na ***Zona de disponibilidade 1***, ***number-tasks*** na ***Zona de disponibilidade 2*** e ***number-tasks*** na ***Zona de disponibilidade 3***. Rebalanceamento de AZ em andamento.

O agendador de serviços envia um evento de serviço do `service (service-name) is not AZ balanced` quando o número de tarefas não está distribuído uniformemente pelas zonas de disponibilidade. Não há nenhuma ação a ser realizada. Este é um evento informativo.

O serviço (***service-name***) está balanceado em AZ com ***number-tasks*** tarefas na ***Zona de disponibilidade 1***, ***number-tasks*** na ***Zona de disponibilidade 2*** e ***number-tasks*** na ***Zona de disponibilidade 3***.

O agendador de serviços envia um evento de serviço do `service (service-name) is AZ balanced` quando o rebalanceamento do serviço em zonas de disponibilidade é concluído. Não há nenhuma ação a ser realizada. Este é um evento informativo.

O *service-name* iniciou *number-tasks* tarefas na *Zona de disponibilidade* para rebalancear AZ: *task-ids*.

O agendador de serviços envia um evento de serviço "*service-name*/*task-set-name* iniciou *number* tarefas na *Zona de disponibilidade*" ao iniciar tarefas na zona de disponibilidade devido ao rebalanceamento de serviços. Não há nenhuma ação a ser realizada. Este é um evento informativo.

O *service-name* interrompeu *number-tasks* tarefas na *Zona de disponibilidade* devido ao rebalanceamento de AZ: *task-id*.

O agendador de serviços envia um evento de serviço *service-name*/*task-set-name* interrompeu *number* tarefas na *Zona de disponibilidade* quando ele interrompe tarefas na zona de disponibilidade por causa do rebalanceamento de serviços. Não há nenhuma ação a ser realizada. Este é um evento informativo.

O serviço (*service-name*) não conseguiu colocar uma tarefa na *Zona de disponibilidade* porque nenhuma instância de contêiner atendeu a todos os requisitos.

O agendador de serviços envia um evento de serviço "O *service-name* não conseguiu colocar uma tarefa na *Zona de disponibilidade*" porque nenhuma instância de contêiner atendeu a todos os requisitos. Para resolver o problema, execute instâncias na zona de disponibilidade.

O serviço (*service-name*) não conseguiu colocar uma tarefa na *Zona de disponibilidade*.

O agendador de serviços envia um evento de serviço "O *service-name* não conseguiu colocar uma tarefa na *Zona de disponibilidade* quando você usa o tipo de inicialização do Fargate e não há capacidade disponível.

É possível adicionar outras sub-redes na zona de disponibilidade na mensagem de erro ou entrar em contato com o Suporte para obter capacidade adicional.

O serviço (*service-name*) não conseguiu fazer o rebalanceamento de AZ porque não foi possível escalar *task-set-name* devido a *reason*.

O agendador de serviços envia um evento de serviço "O *service-name* não conseguiu fazer o rebalanceamento de AZ porque não foi possível escalar *task-set-name* devido a *reason*" quando a proteção de redução de escala é usada.

É possível executar uma das ações a seguir:

- Aguarde até que a proteção das tarefas atuais expire, permitindo que elas sejam encerradas.
- Determine quais tarefas podem ser interrompidas e use a API `UpdateTaskProtection` com a opção `protectionEnabled` definida como `false` para cancelar a proteção dessas tarefas.
- Aumente a contagem de tarefas desejada do serviço para mais do que o número de tarefas protegidas.

O serviço (*service-name*) interrompeu o rebalanceamento de AZ.

O agendador de serviços envia um evento de serviço "O *service-name* interrompeu o rebalance de AZ" quando a operação de rebalanceamento de zona de disponibilidade é interrompida. Este é um evento informativo. O Amazon ECS envia eventos adicionais que fornecem mais informações.

Solução de problemas relacionados aos balanceadores de carga de serviço no Amazon ECS

Os serviços do Amazon ECS podem registrar tarefas com um balanceador de carga do Elastic Load Balancing. Erros de configuração do load balancer são causas comuns de tarefas interrompidas. Caso as tarefas interrompidas tenham sido iniciadas por serviços que usam um load balancer, leve em consideração as possíveis causas a seguir.

O perfil vinculado ao serviço do Amazon ECS não existe

> A função vinculada ao serviço do Amazon ECS permite que os serviços do Amazon ECS registrem instâncias de contêiner com balanceadores de carga do Elastic Load Balancing. A função vinculada ao serviço deve ser criada na sua conta. Para obter mais informações, consulte Uso de perfis vinculados ao serviço para o Amazon ECS.

Grupo de segurança da instância de contêiner

Caso o contêiner seja mapeado para a porta 80 na instância de contêiner, o grupo de segurança da instância de contêiner deve permitir o tráfego de entrada na porta 80 para que as verificações de integridade do load balancer sejam aprovadas.

O balanceador de carga do Elastic Load Balancing não está configurado em todas as zonas de disponibilidade

O load balancer deve ser configurado para usar todas as zonas de disponibilidade em uma região, ou pelo menos todas as zonas de disponibilidade em que as instâncias de contêiner residem. Se um serviço usar um balanceador de carga e iniciar uma tarefa em uma instância de contêiner que resida em uma zona de disponibilidade que o balanceador de carga não está configurado para usar, a tarefa jamais será aprovada na verificação de integridade. Isso resulta na eliminação da tarefa.

Verificação de integridade do balanceador de carga do Elastic Load Balancing com configuração incorreta

Os parâmetros de verificação de integridade do balanceador de carga podem ser excessivamente restritivos ou apontar para recursos que não existem. Caso não seja considerada íntegra, a instância de contêiner é removida do balanceador de carga. Não se esqueça de verificar se os parâmetros a seguir estão configurados corretamente para o load balancer de serviço.

Ping Port

O valor Ping Port de uma verificação de integridade do load balancer é a porta nas instâncias de contêiner que o load balancer verifica para determinar se ela é íntegra. Se essa porta estiver mal configurada, o balanceador de carga provavelmente cancelará o registro da instância de contêiner. Essa porta deve ser configurada para usar o valor `hostPort` para o contêiner na definição de tarefa do serviço que você está usando com a verificação de integridade.

Ping Path

Isso faz parte da verificação de integridade do balanceador de carga. É um endpoint na aplicação que pode retornar um código de status de bem-sucedido (por exemplo, 200) quando a aplicação está íntegra. Esse valor costuma ser definido como `index.html`. Porém, caso o serviço não responda a essa solicitação, a verificação de integridade apresentará falha. Se o contêiner não tiver um arquivo `index.html`, será possível defini-lo como `/` para direcionar o URL base da instância de contêiner.

Tempo limite de resposta

Este é o tempo que o contêiner tem para retornar uma resposta para o ping de verificação de integridade. Caso o valor seja menor que o tempo necessário a uma resposta, a verificação de integridade falhará.

Intervalo de verificação de integridade

Este é o tempo entre os pings de verificação de integridade. Quanto menor for o intervalo de verificação de integridade, mais rapidamente a instância de contêiner poderá atingir Unhealthy Threshold (Limite não íntegro).

Limite não íntegro

Este é o número de vezes em que a verificação de integridade pode falhar até a instância de contêiner ser considerada não íntegra. Caso você tenha um limite não íntegro de 2 e um intervalo de verificação de integridade de 30 segundos, a tarefa tem 60 segundos para responder ao ping de verificação de integridade antes de ser considerada não íntegra. É possível aumentar o limite não íntegro ou o intervalo de verificação de integridade para dar às tarefas mais tempo para responder.

Não foi possível atualizar o serviço *`servicename`*: o nome do contêiner do balanceador de carga ou a porta sofreu alteração na definição da tarefa

Se seu serviço usa um balanceador de carga, será possível usar a AWS CLI ou o SDK para modificar a configuração do balanceador de carga. Para obter informações sobre como modificar a configuração, consulte UpdateService na Referência da API do Amazon Elastic Container Service. Se você atualizar a definição de tarefa para o serviço, o nome do contêiner e a porta do contêiner especificados na configuração do balanceador de carga deverão permanecer na definição da tarefa.

Você atingiu o limite do número de tarefas que podem ser executadas simultaneamente.

Para uma nova conta, suas cotas podem ser menores que as cotas de serviço. A cota de serviço da conta pode ser visualizada no console do Service Quotas. Para solicitar um aumento da cota, consulte Requesting a quota increase no Guia do usuário do Service Quotas.

Solução de problemas relacionados ao ajuste de escala automático de serviço no Amazon ECS

O Auto Scaling da aplicação desativa processos de redução horizontal da escala enquanto as implantações do Amazon ECS estão em andamento. Esses processos são retomados assim que a implantação é concluída. No entanto, processos de aumento continuam a ocorrer, a menos que sejam suspensos, durante uma implantação. Para obter mais informações, consulte Suspender e retomar a escalabilidade do Application Auto Scaling.

Solucionar problemas causados por erros de CPU ou memória inválida na definição de tarefa do Amazon ECS

Ao registrar uma definição de tarefa usando a API do Amazon ECS ou a AWS CLI, se você especificar um valor inválido de `cpu` ou `memory`, será retornado o erro a seguir.

```
An error occurred (ClientException) when calling the RegisterTaskDefinition operation:
 Invalid 'cpu' setting for task.
```

> **Note**
>
> Ao usar o Terraform, o erro a seguir pode ser retornado.
>
> ```
> Error: ClientException: No Fargate configuration exists for given values.
> ```

Para resolver esse problema, você deve especificar um valor compatível para a CPU e a memória em sua definição da tarefa. O valor de `cpu` pode ser expresso em unidades de CPU ou vCPUs em uma definição de tarefa. Ele é convertido em um número inteiro que indica as unidades de CPU quando a definição da tarefa é registrada. O valor de `memory` pode ser expresso em MiB ou GB em uma definição de tarefa. Ele é convertido em um número inteiro que indica o valor em MiB quando a definição da tarefa é registrada.

Em definições de tarefa que especificam o `FARGATE` para o parâmetro `requiresCompatibilities` (mesmo se o `EC2` também estiver especificado), você deve usar um dos valores da tabela a seguir. Esses valores determinam sua faixa de valores compatíveis com o parâmetro de CPU e memória.

Para tarefas hospedadas no Fargate, a tabela a seguir mostra as combinações válidas de CPU e memória. Os valores de memória no arquivo JSON são especificados em MiB. É possível converter o valor de GB em MiB multiplicando o valor por 1024. Por exemplo, 1 GB = 1024 MiB.

Valor de CPU	Valor de memória	Sistemas operacionais com suporte para o AWS Fargate
256 (0,25 vCPU)	512 MiB, 1 GB, 2 GB	Linux
512 (0,5 vCPU)	1 GB, 2 GB, 3 GB, 4 GB	Linux
1024 (1 vCPU)	2 GB, 3 GB, 4 GB, 5 GB, 6 GB, 7 GB, 8 GB	Linux, Windows
2048 (2 vCPU)	Entre 4 GB e 16 GB em incrementos de 1 GB	Linux, Windows
4096 (4 vCPU)	Entre 8 GB e 30 GB em incrementos de 1 GB	Linux, Windows
8192 (8 vCPU) **Note** Essa opção requer a plataforma Linux `1.4.0` ou posterior.	Entre 16 GB e 60 GB em incrementos de 4 GB	Linux
16384 (16 vCPU) **Note** Essa opção requer a plataforma Linux `1.4.0` ou posterior.	Entre 32 GB e 120 GB em incrementos de 8 GB	Linux

Em tarefas hospedadas no Amazon EC2, os valores de CPU da tarefa compatível estão entre 0,25 vCPUs e 192 vCPUs.

O mecanismo de controle de CPU difere entre os tipos de execução do Amazon EC2 e do Fargate:

- Para tarefas hospedadas no Amazon EC2: o Amazon ECS usa o período da CPU e a cota de CPU para controlar os limites rígidos de CPU do tamanho da tarefa. Quando você especifica a vCPU na definição da tarefa, o Amazon ECS traduz o valor para o período da CPU e as configurações de cota de CPU que se aplicam a `cgroup`.
- Em tarefas hospedadas no Fargate: o Amazon ECS usa compartilhamentos de CPU para controlar a alocação de CPU. Os valores da cota e do período da CPU não são usados para limitar a CPU em tarefas do Fargate.

Para tarefas do Amazon EC2, a cota de CPU controla a quantidade de tempo de CPU concedida a `cgroup` durante um determinado período de CPU. Ambas as configurações são expressas em termos de microssegundos. Quando a cota de CPU é igual ao período da CPU, isso significa que `cgroup` pode executar até 100% em uma vCPU (ou qualquer outra fração que totalize 100% para várias vCPUs). A cota de CPU tem um máximo de 1.000.000 microssegundos e o período de CPU tem um mínimo de 1 ms. Você pode usar esses valores para definir os limites da contagem de CPUs. Quando você altera o período da CPU sem alterar a cota de CPU, você passa a ter limites efetivos diferentes dos especificados na definição da tarefa.

O período de 100 ms permite vCPUs que variam de 0,125 a 10.

> **Note**
>
> Os parâmetros de CPU e memória em nível de tarefa são ignorados para contêineres do Windows.

Visualização de logs do agente de contêiner do Amazon ECS

O Amazon ECS armazena logs na pasta `/var/log/ecs` das instâncias de contêiner. Existem logs disponibilizados pelo agente de contêiner do Amazon ECS e pelo serviço `ecs-init` que controla o estado do agente (iniciar/interromper) na instância de contêiner. É possível visualizar esses arquivos de log conectando-se a uma instância de contêiner usando SSH.

> **Note**
>
> Se você não tiver certeza de como coletar todos os logs nas instâncias de contêiner, poderá usar o coletor de logs do Amazon ECS. Para obter mais informações, consulte Coleta de logs de contêiner com o coletor de logs do Amazon ECS .

Sistema operacional Linux

O processo `ecs-init` armazena logs em `/var/log/ecs/ecs-init.log`.

O arquivo `ecs-init.log` contém informações sobre o gerenciamento, a configuração e a inicialização do ciclo de vida do agente de contêiner.

É possível usar o comando a seguir para visualizar os arquivos de log.

```
cat /var/log/ecs/ecs-init.log
```

Saída:

```
2018-02-16T18:13:54Z [INFO] pre-start
2018-02-16T18:13:56Z [INFO] start
2018-02-16T18:13:56Z [INFO] No existing agent container to remove.
2018-02-16T18:13:56Z [INFO] Starting Amazon Elastic Container Service Agent
```

Sistema operacional Windows

Você pode usar o coletor de logs do Amazon ECS para Windows. Para obter mais informações, consulte Amazon ECS Logs Collector for Windows no Github.

1. Conecte-se à sua instância.
2. Abra o PowerShell e execute os comandos a seguir com privilégios administrativos. Os comandos baixam o script e coletam os logs.

   ```
   Invoke-WebRequest -OutFile ecs-logs-collector.ps1 https://
   raw.githubusercontent.com/awslabs/aws-ecs-logs-collector-for-windows/master/ecs-
   logs-collector.ps1
   .\ecs-logs-collector.ps1
   ```

Você pode ativar o registro em log de depuração para o agente do Amazon ECS e o daemon do Docker. Essa opção permite que o script colete os logs antes de ativar o modo de depuração. O script reinicia o daemon do Docker e o agente do Amazon ECS, depois encerra todos os contêineres em execução na instância. Antes de executar o comando a seguir, esgote a instância de contêiner e mova todas as tarefas importantes para outras instâncias de contêiner.

Execute o comando a seguir para ativar o registro em log.

```
.\ecs-logs-collector.ps1 -RunMode debug
```

Coleta de logs de contêiner com o coletor de logs do Amazon ECS

Se você não tiver certeza de como coletar todos os vários logs nas instâncias de contêiner, poderá usar o coletor de logs do Amazon ECS. Ele está disponível no GitHub para os sistemas Linux e Windows. O script coleta logs gerais do sistema operacional geral, além dos logs do agente de contêiner do Docker e do Amazon ECS, que podem ser úteis para solucionar problemas de casos do AWS Support. Em seguida, compacta e arquiva os dados coletados em um único arquivo que pode ser facilmente compartilhado para fins de diagnóstico. Também oferece suporte à habilitação do modo de depuração do daemon do Docker e do agente de contêiner do Amazon ECS em variantes do Amazon Linux, como a AMI otimizada para Amazon ECS. No momento, o coletor de logs do Amazon ECS oferece suporte aos seguintes sistemas operacionais:

- Amazon Linux
- Red Hat Enterprise Linux 7
- Debian 8
- Ubuntu 14.04
- Ubuntu 16.04
- Ubuntu 18.04
- Windows Server 2016

> **Note**
>
> O código fonte do coletor de logs do Amazon ECS está disponível no GitHub para Linux e Windows. Incentivamos você a enviar solicitações de envio para alterações que você

gostaria de ter incluído. No entanto, o Amazon Web Services atualmente não oferece suporte para a execução de cópias modificadas desse software.

Para baixar e executar o coletor de logs do Amazon ECS para Linux

1. Conecte-se à sua instância de contêiner.
2. Baixe o script do coletor de logs do Amazon ECS.

```
curl -O https://raw.githubusercontent.com/awslabs/ecs-logs-collector/master/ecs-logs-collector.sh
```

3. Execute o script para coletar os logs e criar o arquivo.

> **Note**
>
> Para habilitar o modo de depuração no daemon do Docker e no agente de contêiner do Amazon ECS, adicione a opção `--mode=enable-debug` ao comando a seguir. Isso pode reiniciar o daemon do Docker, o que elimina todos os contêineres em execução na instância. Considere drenar a instância de contêiner e mover todas as tarefas importantes para outras instâncias de contêiner antes de ativar o modo de depuração. Para obter mais informações, consulte Drenagem de instâncias de contêiner do Amazon ECS.

```
[ec2-user ~]$ sudo bash ./ecs-logs-collector.sh
```

> **Important**
>
> Recomendamos editar os logs e remover todos os dados confidenciais dos arquivos. Você pode pesquisar dados conhecidos e também pesquisar variáveis de ambiente, como `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` e `AWS_SESSION_TOKEN` no arquivo.

Depois que tiver implementado o script, será possível examinar os logs coletados na pasta `collect` criada pelo script. O arquivo `collect.tgz` é um arquivo compactado de todos os logs, que você pode compartilhar com o AWS Support para receber ajuda no diagnóstico.

Para baixar e executar o coletor de logs do Amazon ECS para Windows

1. Conecte-se à sua instância de contêiner. Para obter mais informações, consulte Conectar à instância do Windows usando um cliente RDP no Guia do usuário do Amazon EC2.
2. Baixe o script do coletor de logs do Amazon ECS usando o PowerShell.

```
Invoke-WebRequest -OutFile ecs-logs-collector.ps1 https://
raw.githubusercontent.com/awslabs/aws-ecs-logs-collector-for-windows/master/ecs-
logs-collector.ps1
```

3. Execute o script para coletar os logs e criar o arquivo.

> **Note**
>
> Para habilitar o modo de depuração no daemon do Docker e no agente de contêiner do Amazon ECS, adicione a opção `-RunMode debug` ao comando a seguir. Isso reinicia o daemon do Docker, o que elimina todos os contêineres em execução na instância. Considere drenar a instância de contêiner e mover todas as tarefas importantes para outras instâncias de contêiner antes de ativar o modo de depuração. Para obter mais informações, consulte Drenagem de instâncias de contêiner do Amazon ECS.

```
.\ecs-logs-collector.ps1
```

> **Important**
>
> Recomendamos editar os logs e remover todos os dados confidenciais dos arquivos. Você pode pesquisar dados conhecidos e também pesquisar variáveis de ambiente, como `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` e `AWS_SESSION_TOKEN` no arquivo.

Depois que tiver implementado o script, será possível examinar os logs coletados na pasta `collect` criada pelo script. O arquivo `collect.tgz` é um arquivo compactado de todos os logs, que você pode compartilhar com a AWS Support para receber ajuda no diagnóstico.

Recuperar detalhes de diagnóstico do Amazon ECS com introspecção do agente

A API de introspecção do agente do Amazon ECS fornece informações sobre o estado geral do agente e das instâncias de contêiner do Amazon ECS.

Por exemplo, você pode usar a API de introspecção do agente para obter o ID do Docker para um contêiner na tarefa. É possível usar a API de introspecção do agente se conectando a uma instância de contêiner usando SSH.

> **Important**
>
> A instância de contêiner deve ter uma função do IAM que concede acesso ao Amazon ECS para alcançar a API de introspecção. Para obter mais informações, consulte Função do IAM de instância de contêiner do Amazon ECS.

O exemplo a seguir mostra duas tarefas, uma em execução no momento e uma interrompida.

> **Note**
>
> O comando a seguir é direcionado por meio de python -mjson.tool para mais legibilidade.

```
curl http://localhost:51678/v1/tasks | python -mjson.tool
```

Saída:

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1095  100  1095    0     0   117k      0 --:--:-- --:--:-- --:--:--  133k
{
    "Tasks": [
        {
            "Arn": "arn:aws:ecs:us-west-2:aws_account_id:task/090eff9b-1ce3-4db6-848a-
a8d14064fd24",
            "Containers": [
                {
```

```
                    "DockerId":
 "189a8ff4b5f04affe40e5160a5ffadca395136eb5faf4950c57963c06f82c76d",
                    "DockerName": "ecs-console-sample-app-static-6-simple-
app-86caf9bcabe3e9c61600",
                    "Name": "simple-app"
                },
                {
                    "DockerId":
 "f7f1f8a7a245c5da83aa92729bd28c6bcb004d1f6a35409e4207e1d34030e966",
                    "DockerName": "ecs-console-sample-app-static-6-busybox-
ce83ce978a87a890ab01",
                    "Name": "busybox"
                }
            ],
            "Family": "console-sample-app-static",
            "KnownStatus": "STOPPED",
            "Version": "6"
        },
        {
            "Arn": "arn:aws:ecs:us-west-2:aws_account_id:task/1810e302-eaea-4da9-
a638-097bea534740",
            "Containers": [
                {
                    "DockerId":
 "dc7240fe892ab233dbbcee5044d95e1456c120dba9a6b56ec513da45c38e3aeb",
                    "DockerName": "ecs-console-sample-app-static-6-simple-app-
f0e5859699a7aecfb101",
                    "Name": "simple-app"
                },
                {
                    "DockerId":
 "096d685fb85a1ff3e021c8254672ab8497e3c13986b9cf005cbae9460b7b901e",
                    "DockerName": "ecs-console-sample-app-static-6-
busybox-92e4b8d0ecd0cce69a01",
                    "Name": "busybox"
                }
            ],
            "DesiredStatus": "RUNNING",
            "Family": "console-sample-app-static",
            "KnownStatus": "RUNNING",
            "Version": "6"
        }
    ]
```

```
}
```

No exemplo anterior, a tarefa interrompida (*090eff9b-1ce3-4db6-848a-a8d14064fd24*) tem dois contêineres. É possível usar docker inspect ***container-ID*** para exibir informações detalhadas sobre cada contêiner. Para obter mais informações, consulte Introspecção de contêiner do Amazon ECS.

Diagnóstico do Docker no Amazon ECS

O Docker oferece várias ferramentas de diagnóstico que ajudam a solucionar problemas com os contêineres e as tarefas. Para obter mais informações sobre todos os utilitários de linha de comando do Docker, consulte o tópico Referência da linha de comando do Docker na documentação do Docker. É possível acessar os utilitários de linha de comando do Docker se conectando a uma instância de contêiner usando SSH.

Os códigos de saída que os contêineres do Docker relatam também podem fornecer algumas informações de diagnóstico (por exemplo, código de saída 137 significa que o contêiner recebeu um sinal `SIGKILL`). Para obter mais informações, consulte Status de saída na documentação do Docker.

Listagem de contêineres do Docker no Amazon ECS

É possível usar o comando docker ps na instância de contêiner para listar os contêineres em execução. No exemplo a seguir, somente o agente de contêiner do Amazon ECS está em execução. Para obter mais informações, consulte docker ps na documentação do Docker.

```
docker ps
```

Saída:

```
CONTAINER ID        IMAGE                            COMMAND             CREATED
      STATUS              PORTS                          NAMES
cee0d6986de0        amazon/amazon-ecs-agent:latest   "/agent"            22 hours ago
      Up 22 hours         127.0.0.1:51678->51678/tcp   ecs-agent
```

É possível usar o comando docker ps -a para ver todos os contêineres (até mesmo contêineres parados ou encerrados). Isso é útil para listar contêineres que estejam parando inesperadamente. No exemplo a seguir, o contêiner `f7f1f8a7a245` saiu há 9 segundos. Portanto, ele não aparece em uma saída docker ps sem o sinalizador `-a`.

```
docker ps -a
```

Saída:

```
CONTAINER ID        IMAGE                                          COMMAND
 CREATED             STATUS                        PORTS                        NAMES
db4d48e411b1        amazon/ecs-emptyvolume-base:autogenerated   "not-applicable"
  19 seconds ago                                                                ecs-
console-sample-app-static-6-internalecs-emptyvolume-source-c09288a6b0cba8a53700
f7f1f8a7a245        busybox:buildroot-2014.02                   "\"sh -c '/bin/sh -c
  22 hours ago        Exited (137) 9 seconds ago                                ecs-
console-sample-app-static-6-busybox-ce83ce978a87a890ab01
189a8ff4b5f0        httpd:2                                     "httpd-foreground"
  22 hours ago        Exited (137) 40 seconds ago                               ecs-
console-sample-app-static-6-simple-app-86caf9bcabe3e9c61600
0c7dca9321e3        amazon/ecs-emptyvolume-base:autogenerated   "not-applicable"
  22 hours ago                                                                  ecs-
console-sample-app-static-6-internalecs-emptyvolume-source-90fefaa68498a8a80700
cee0d6986de0        amazon/amazon-ecs-agent:latest              "/agent"
  22 hours ago        Up 22 hours                   127.0.0.1:51678->51678/tcp   ecs-
agent
```

Visualização de logs do Docker no Amazon ECS

É possível visualizar os fluxos `STDOUT` e `STDERR` para um contêiner com o comando docker logs. Neste exemplo, os logs são exibidos para o contêiner *`dc7240fe892a`* e direcionados por meio do comando head para agilizar. Para obter mais informações, vá até docker logs na documentação do Docker.

> **Note**
>
> Os logs do Docker estarão disponíveis na instância do contêiner apenas se você estiver usando o driver de logs `json` padrão. Se você tiver configurado as tarefas para usar o driver de logs `awslogs`, os logs do contêiner estarão disponíveis no CloudWatch Logs. Para obter mais informações, consulte Envio de logs do Amazon ECS para o CloudWatch .

```
docker logs dc7240fe892a | head
```

Saída:

```
AH00558: httpd: Could not reliably determine the server's fully qualified domain name,
 using 172.17.0.11. Set the 'ServerName' directive globally to suppress this message
AH00558: httpd: Could not reliably determine the server's fully qualified domain name,
 using 172.17.0.11. Set the 'ServerName' directive globally to suppress this message
[Thu Apr 23 19:48:36.956682 2015] [mpm_event:notice] [pid 1:tid 140327115417472]
 AH00489: Apache/2.4.12 (Unix) configured -- resuming normal operations
[Thu Apr 23 19:48:36.956827 2015] [core:notice] [pid 1:tid 140327115417472] AH00094:
 Command line: 'httpd -D FOREGROUND'
10.0.1.86 - - [23/Apr/2015:19:48:59 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:48:59 +0000] "GET / HTTP/1.1" 200 348
10.0.1.86 - - [23/Apr/2015:19:49:28 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:49:29 +0000] "GET / HTTP/1.1" 200 348
10.0.1.86 - - [23/Apr/2015:19:49:50 +0000] "-" 408 -
10.0.0.154 - - [23/Apr/2015:19:49:50 +0000] "-" 408 -
10.0.1.86 - - [23/Apr/2015:19:49:58 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:49:59 +0000] "GET / HTTP/1.1" 200 348
10.0.1.86 - - [23/Apr/2015:19:50:28 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:50:29 +0000] "GET / HTTP/1.1" 200 348
time="2015-04-23T20:11:20Z" level="fatal" msg="write /dev/stdout: broken pipe"
```

Inspeção de contêineres do Docker no Amazon ECS

Caso tenha o ID do Docker de um contêiner, você pode inspecioná-lo com o comando docker inspect. A inspeção de contêineres apresenta a visão mais detalhada do ambiente no qual um contêiner foi ativado. Para obter mais informações, consulte docker inspect na documentação do Docker.

```
docker inspect dc7240fe892a
```

Saída:

```
[{
    "AppArmorProfile": "",
    "Args": [],
    "Config": {
        "AttachStderr": false,
        "AttachStdin": false,
        "AttachStdout": false,
        "Cmd": [
            "httpd-foreground"
        ],
        "CpuShares": 10,
```

```
        "Cpuset": "",
        "Domainname": "",
        "Entrypoint": null,
        "Env": [
            "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/
local/apache2/bin",
            "HTTPD_PREFIX=/usr/local/apache2",
            "HTTPD_VERSION=2.4.12",
            "HTTPD_BZ2_URL=https://www.apache.org/dist/httpd/httpd-2.4.12.tar.bz2"
        ],
        "ExposedPorts": {
            "80/tcp": {}
        },
        "Hostname": "dc7240fe892a",
...
```

Configuração da saída detalhada do daemon do Docker no Amazon ECS

Caso enfrente problemas com contêineres ou imagens do Docker, você pode ativar o modo de depuração no daemon do Docker. O uso da depuração fornece uma saída mais detalhada do daemon. É possível usar esse recurso para recuperar mensagens de erro enviadas de registros de contêiner, como o Amazon ECR.

> **Important**
>
> Esse procedimento foi escrito para a AMI do Amazon Linux otimizada para Amazon ECS. Para outros sistemas operacionais, consulte Enable debugging e Control and configure Docker with systemd na documentação do Docker.

Para usar o modo de depuração do daemon do Docker na AMI do Amazon Linux otimizada para o Amazon ECS

1. Conecte-se à sua instância de contêiner.
2. Abra o arquivo de opções do Docker com um editor de texto, como vi. Para a AMI do Amazon Linux otimizada para Amazon ECS, o arquivo de opções do Docker está em `/etc/sysconfig/docker`.
3. Encontre a instrução de opções do Docker e adicione a opção `-D` à string, entre aspas.

> **Note**
>
> Se a instrução de opções do Docker começar com um #, remova esse caractere para cancelar o comentário da instrução e ativar as opções.

Para a AMI do Amazon Linux otimizada para Amazon ECS, a instrução de opções do Docker é denominada `OPTIONS`. Por exemplo:

```
# Additional startup options for the Docker daemon, for example:
# OPTIONS="--ip-forward=true --iptables=true"
# By default we limit the number of open files per container
OPTIONS="-D --default-ulimit nofile=1024:4096"
```

4. Salve o arquivo e saia do seu editor de texto.
5. Reinicie o daemon do Docker.

```
sudo service docker restart
```

A saída é a seguinte:

```
Stopping docker:                                           [  OK  ]
Starting docker: .                                   [  OK  ]
```

6. Reinicie o agente do Amazon ECS.

```
sudo service ecs restart
```

Os logs do Docker já devem mostrar uma saída mais detalhada.

```
time="2015-12-30T21:48:21.907640838Z" level=debug msg="Unexpected response from
 server: \"{\\\"errors\\\":[{\\\"code\\\":\\\"DENIED\\\",\\\"message\\\":\\\"User:
 arn:aws:sts::1111:assumed-role/ecrReadOnly/i-abcdefg is not authorized to perform:
 ecr:InitiateLayerUpload on resource: arn:aws:ecr:us-east-1:1111:repository/nginx_test
\\\"}]}\\n\" http.Header{\"Connection\":[]string{\"keep-alive\"}, \"Content-Type\":
[]string{\"application/json; charset=utf-8\"}, \"Date\":[]string{\"Wed, 30 Dec 2015
 21:48:21 GMT\"}, \"Docker-Distribution-Api-Version\":[]string{\"registry/2.0\"},
 \"Content-Length\":[]string{\"235\"}}"
```

Solução de problemas relacionados a **`API error (500): devmapper`** do Docker no Amazon ECS

O seguinte erro do Docker indica que o armazenamento de grupos thin em sua instância de contêiner está cheio e que o daemon do Docker não pode criar novos contêineres:

```
CannotCreateContainerError: API error (500): devmapper: Thin Pool has 4350 free data
 blocks which is less than minimum required 4454 free data blocks. Create more free
 space in thin pool or use dm.min_free_space option to change behavior
```

Por padrão. as AMIs do Amazon Linux otimizadas para o Amazon ECS da versão `2015.09.d` e posterior são iniciadas com um volume de 8 GiB para o sistema operacional anexado a `/dev/xvda` e montado como a raiz do sistema de arquivos. Há um volume de 22 GiB adicionais anexado a `/dev/xvdcz` que o Docker usa em armazenamento de imagens e metadados. Caso o espaço de armazenamento esteja cheio, o daemon do Docker não pode criar novos contêineres.

A maneira mais fácil de adicionar armazenamento às instâncias de contêiner é encerrar as instâncias existentes e executar novas com volumes de armazenamento físico de dados maiores. Porém, se você não puder fazer isso, poderá adicionar armazenamento ao grupo de volumes que o Docker usa e estender o volume lógico seguindo os procedimentos em AMIs do Linux otimizadas para o Amazon ECS.

Caso o armazenamento de instância de contêiner esteja enchendo muito rapidamente, existem algumas ações que você pode tomar para reduzir esse efeito:

- Para visualizar as informações de pesquisas detalhadas, execute o seguinte comando na instância de contêiner:

  ```
  docker info
  ```

- (Agente de contêiner 1.8.0 e posterior do Amazon ECS) Reduza o tempo em que contêineres interrompidos ou encerrados permanecem nas instâncias de contêiner. A variável de configuração do agente `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION` define a duração da espera desde quando uma tarefa é parada até o contêiner do Docker ser removido (por padrão, esse valor é de 3 horas). Isso remove os dados de contêiner do Docker. Caso esse valor seja definido muito baixo, você não pode inspecionar os contêineres interrompidos ou visualizar os logs antes de serem removidos. Para obter mais informações, consulte Configuração do agente de contêiner do Amazon ECS.

- É possível remover contêineres que não estejam em execução e imagens inutilizadas das instâncias de contêiner. É possível usar os comandos de exemplo a seguir para remover manualmente os contêineres parados e as imagens inutilizadas. Os contêineres excluídos não podem ser inspecionados posteriormente e as imagens excluídas devem ser extraídas novamente antes que novos contêineres sejam iniciados a partir delas.

  Para remover contêineres que não estejam em execução, execute o seguinte comando na instância de contêiner:

  ```
  docker rm $(docker ps -aq)
  ```

  Para remover imagens não utilizadas, execute o seguinte comando na instância de contêiner:

  ```
  docker rmi $(docker images -q)
  ```

- É possível remover blocos de dados inutilizados em contêineres. É possível usar o comando a seguir para executar fstrim em qualquer contêiner em execução e descartar todos os blocos de dados inutilizados pelo sistema de arquivos do contêiner.

  ```
  sudo sh -c "docker ps -q | xargs docker inspect --format='{{ .State.Pid }}' | xargs -IZ fstrim /proc/Z/root/"
  ```

Solução de problemas do Amazon ECS Exec

Veja a seguir notas de solução de problemas para ajudar a diagnosticar o motivo pelo qual você pode estar recebendo um erro ao usar o ECS Exec.

Verificar usando o verificador do Exec

O script do Verificador do ECS Exec fornece uma maneira de examinar e validar se o cluster e a tarefa do Amazon ECS atenderam aos pré-requisitos para usar o recurso do ECS Exec. O script do Verificador do ECS Exec examina se o ambiente e o cluster da AWS CLI e as tarefas estão prontos para o ECS Exec, chamando várias APIs em seu nome. A ferramenta exige a versão mais recente da AWS CLI e que o `jq` esteja disponível. Para obter mais informações, consulte ECS Exec Checker no GitHub.

Erro ao chamar **`execute-command`**

Se ocorrer um erro `The execute command failed`, veja a seguir as possíveis causas.

- A tarefa não tem as permissões necessárias. Verifique se a definição de tarefa usada para iniciar a tarefa tem uma função do IAM de tarefa definida e se a função tem as permissões necessárias. Para obter mais informações, consulte Permissões do ECS Exec.
- O agente SSM não está instalado ou não está em execução.
- Existe um endpoint da Amazon VPC de interface para o Amazon ECS, mas não há um para o gerenciador de sessões do Systems Manager.

Solução de problemas do Amazon ECS Anywhere

O Amazon ECS Anywhere fornece suporte para registrar uma instância externa, como um servidor on-premises ou uma máquina virtual (VM), no cluster do Amazon ECS. Veja a seguir os problemas comuns que você pode encontrar e as recomendações gerais para solução destes problemas.

Tópicos

- Problemas de registro de instância externa
- Problemas de rede de instâncias externas
- Problemas na execução de tarefas ne instância externa

Problemas de registro de instância externa

Ao registrar uma instância externa no cluster do Amazon ECS, os seguintes requisitos devem ser atendidos:

- Uma ativação do AWS Systems Manager, que consiste em um ID de ativação e em um código de ativação, deve ser recuperada. Você a utiliza para registrar a instância externa como uma instância gerenciada do Systems Manager. Quando uma ativação do Systems Manager é solicitada, especifique um limite de registro e uma data de validade. O limite de registro especifica o número máximo de instâncias que podem ser registradas usando a ativação. O valor padrão do limite de registro é a instância 1. A data de validade especifica quando a validade da ativação. O valor padrão é 24 horas. Se a ativação do Systems Manager que você está usando para registrar a instância externa não estiver válida, solicite uma nova. Para obter mais informações, consulte Registro de uma instância externa para um cluster do Amazon ECS.

- Uma política do IAM é usada para fornecer à instância externa as permissões necessárias para se comunicar com operações de API da AWS. Se essa política gerenciada não for criada corretamente e não contiver as permissões necessárias, o registro da instância externa apresentará falha. Para obter mais informações, consulte Perfil do IAM para o Amazon ECS Anywhere.
- O Amazon ECS fornece um script de instalação que instala o Docker, o agente de contêiner do Amazon ECS e o Systems Manager Agent na instância externa. Se o script de instalação apresentar falha, é provável que ele não possa ser executado novamente na mesma instância sem a ocorrência de erro. Se isso acontecer, siga o processo de limpeza dos recursos da AWS da instância para que o script de instalação possa ser executado novamente. Para obter mais informações, consulte Cancelamento do registro de uma instância externa do Amazon ECS.

  > **Note**
  >
  > Saiba que, se o script de instalação tiver solicitado e usado com êxito a ativação do Systems Manager, a execução, pela segunda vez, do script de instalação usará novamente a ativação do Systems Manager. Isso pode, por sua vez, fazer com que você atinja o limite de registro para essa ativação. Se esse limite for atingido, você deverá criar uma nova ativação.

- Ao executar o script de instalação em uma instância externa para workloads de GPU, ocorrerá um erro se o driver NVIDIA não for detectado ou configurado corretamente. O script de instalação usa o comando `nvidia-smi` para confirmar a existência do driver NVIDIA.

Problemas de rede de instâncias externas

Para comunicar quaisquer alterações, a instância externa requer uma conexão de rede com a AWS. Se sua instância externa perder sua conexão de rede com a AWS, as tarefas que estão sendo executadas nas instâncias continuarão sendo executadas de qualquer maneira, a menos que sejam interrompidas manualmente. Após a restauração da conexão com a AWS, as credenciais da AWS usadas pelo agente de contêiner do Amazon ECS e pelo Systems Manager Agent na instância externa são renovadas automaticamente. Para obter mais informações sobre os domínios da AWS que são usados para comunicação entre a instância externa e a AWS, consulte Redes .

Problemas na execução de tarefas ne instância externa

Se as tarefas ou contêineres não forem executados na instância externa, as causas mais comuns serão relacionadas à rede ou à permissão. Se os contêineres estiverem extraindo imagens do Amazon ECR ou estiverem configurados para enviar logs de contêiner para o CloudWatch Logs, sua definição de tarefa deverá especificar uma função do IAM de execução de tarefa válida. Sem uma função do IAM de execução de tarefa válida, os contêineres apresentarão falha na inicialização. Para obter mais informações sobre problemas relacionados à rede, consulte Problemas de rede de instâncias externas.

> **Important**
>
> O Amazon ECS fornece a ferramenta de coleta de logs do Amazon ECS. É possível usá-la para coletar logs das instâncias externas para fins de solução de problemas. Para obter mais informações, consulte Coleta de logs de contêiner com o coletor de logs do Amazon ECS .

Cotas de controle de utilização do AWS Fargate

O AWS Fargate limita as tarefas do Amazon ECS e as taxas de início dos pods do Amazon EKS a cotas (anteriormente chamadas de limites) usando um algoritmo de bucket de token para cada conta da AWS por região. Com esse algoritmo, sua conta tem um bucket que contém um número específico de tokens. O número de tokens no bucket representa sua cota de taxa a qualquer segundo. Cada conta de cliente tem um bucket de tokens de tarefas e pods que se esgota com base no número de tarefas e pods iniciados pela conta do cliente. Esse bucket de tokens tem o tamanho máximo de bucket que permite fazer periodicamente um número maior de solicitações e uma taxa de recarga que permite manter uma taxa constante de solicitações pelo tempo necessário.

Por exemplo, o tamanho do bucket de tokens de tarefas e pods para uma conta de cliente do Fargate é de 100 tokens e a taxa de recarga é de 20 tokens por segundo. Portanto, é possível iniciar imediatamente até 100 tarefas do Amazon ECS e pods do Amazon EKS por conta de cliente, com uma taxa de lançamento sustentada de 20 tarefas do Amazon ECS e pods do Amazon EKS por segundo.

Ações	Capacidade máxima do bucket (ou taxa de intermitência)	Taxa de reabastecimento do bucket (ou taxa sustentada)
Cota de taxa de recursos do Fargate para tarefas sob demanda do Amazon ECS e pods do Amazon EKS[1]	100	20
Cota de taxa de recursos do Fargate para tarefas spot do Amazon ECS	100	20

[1]As contas que iniciam apenas pods do Amazon EKS têm uma taxa de intermitência de 20, com uma taxa de início de pod sustentada de 20 inícios de pod por segundo ao usar as versões da plataforma chamadas em Versões da plataforma do Amazon EKS.

Controle de utilização da API **RunTask** no Fargate

Além disso, o Fargate limita a taxa de solicitações ao iniciar tarefas usando a API `RunTask` do Amazon ECS com uma cota separada. O Fargate limita as solicitações da API `RunTask` do Amazon ECS para cada conta da AWS de acordo com a região. Cada solicitação feita remove um token do bucket. Fazemos isso para ajudar na performance do serviço e para garantir o uso justo para todos os clientes da Fargate. As chamadas de API estão sujeitas às cotas de solicitação, independentemente de serem originadas no console do Amazon Elastic Conteiner Service, em uma ferramenta da linha de comando ou em uma aplicação de terceiros. A cota de taxa para chamadas para a API `RunTask` do Amazon ECS é de 20 chamadas por segundo (intermitência e sustentada). Contudo, cada chamada para essa API pode iniciar até 10 tarefas. Isso significa que é possível iniciar 100 tarefas em um segundo fazendo 10 chamadas para essa API, solicitando que 10 tarefas sejam iniciadas em cada chamada. Da mesma forma, também é possível fazer 20 chamadas para essa API, solicitando que 5 tarefas sejam iniciadas em cada chamada. Para obter mais informações sobre o controle de utilização de APIs para a API `RunTask` do Amazon ECS, consulte API request throttling na Referência de APIs do Amazon ECS.

Na prática, as taxas de início de tarefas e pods também dependem de outras considerações, como imagens de contêiner a serem baixadas e descompactadas, verificações de integridade e outras integrações habilitadas, como registrar tarefas ou pods com um balanceador de carga. Os clientes consultam variações nas taxas de execução de tarefas e pods em comparação com as cotas representadas anteriormente com base nos recursos habilitados pelos clientes.

Ajuste de cotas tarifárias no Fargate

É possível solicitar um aumento da cotas de controle de utilização da taxa do Fargate para a sua conta da AWS. Para obter mais informações, consulte Solicitar um aumento de cota no Guia do usuário do Service Quotas.

Motivos de falha da API no Amazon ECS

Quando uma ação de API que você acionou por meio da API do Amazon ECS, do console ou da AWS CLI é encerrada com uma mensagem de erro `failures`, a ação a seguir pode ajudar a reparar a causa. A falha retorna o motivo e o nome do recurso da Amazon (ARN) do recurso associado à falha.

Muitos recursos são específicos da região. Portanto, ao usar o console, certifique-se de ter definido a região correta para os recursos. Ao usar a AWS CLI, certifique-se de que os comandos da AWS CLI estejam sendo enviados para a região correta com o parâmetro `--region` *`region`*.

Para obter mais informações sobre a estrutura do tipo de dados da `Failure`, consulte Falha na Referência da API do Amazon Elastic Service.

Veja a seguir exemplos de mensagens de falha que você pode receber ao executar comandos da API.

Ação da API	Motivo da falha ou Motivo da interrupção	Causa
`DescribeClusters`	`MISSING`	O cluster especificado não foi encontrado. Verifique a ortografia do nome do cluster.
`DescribeInstances`	`MISSING`	A instância de contêiner especificada não foi encontrada. Verifique se você especificou o cluster no qual a instância de contêiner está registrada e se o ARN e o ID da instância de contêiner estão corretos.

Ação da API	Motivo da falha ou Motivo da interrupção	Causa
`DescribeServices`	`MISSING`	O serviço especificado não foi encontrado. Verifique se o cluster ou a região correta está especificada e se o ARN ou o nome do serviço é válido.
`DescribeTasks`	`MISSING`	A tarefa especificada não foi encontrada. Verifique se o cluster ou a região correta está especificada e se o ARN ou o ID da tarefa é válido.

Ação da API	Motivo da falha ou Motivo da interrupção	Causa
`DescribeTasks`	`TaskFailedToStart: RESOURCE:*`	Em caso de erro de `RESOURCE:CPU` , o número de CPUs solicitadas pela tarefa está indisponível nas suas instâncias de contêiner. Isso costuma ocorrer quando o requisito de unidade de CPU na definição de tarefa é maior do que o tamanho da CPU das instâncias do Amazon EC2 definidas no grupo do Auto Scaling mapeado para o provedor de capacidad e. Você precisa verificar a configuração do seu provedor de capacidade. Em caso de erro de `RESOURCE:MEMORY` , a quantidade de memória solicitada pela tarefa está indisponível nas suas instâncias de contêiner. Isso costuma ocorrer quando o requisito de quantidade de memória na definição de tarefa é maior do que o suporte de memória nas instâncias do Amazon EC2 definidas no grupo do Auto Scaling mapeado para o provedor de capacidade. Você precisa verificar a configura

Ação da API	Motivo da falha ou Motivo da interrupção	Causa
		ção do seu provedor de capacidade.
	`TaskFailedToStart: AGENT`	A instância de contêiner na qual você tentou iniciar uma tarefa tem um agente que está desconectado, no momento. Para evitar tempos de espera prolongados para a tarefa, a solicitação foi rejeitada. Para saber mais sobre como solucionar um agente que está desconectado, consulte Como solucionar problemas com um agente desconectado do Amazon ECS.
	`TaskFailedToStart: MemberOf placement constraint unsatisfied`	Não há nenhuma instância de contêiner que atenda às restrições de posicionamento especificadas na definição da tarefa.

Ação da API	Motivo da falha ou Motivo da interrupção	Causa
	`TaskFailedToStart: ATTRIBUTE`	A definição de tarefa contém um parâmetro que exige um atributo de instância de contêiner específico que não está disponível nas instância s de contêiner. Por exemplo, se sua tarefa usar o modo de rede `awsvpc`, mas não houver instâncias nas suas sub-redes especificadas com o atributo `ecs.capability.tas k-eni` . Para obter mais informações sobre quais atributos são obrigatórios para parâmetros de definição de tarefa específicos e variáveis de configuração de agentes, consulte Parâmetros de definição de tarefa do Amazon ECS para o tipo de inicializ ação do Fargate e Configura ção do agente de contêiner do Amazon ECS.
	`TaskFailedToStart: NO ACTIVE INSTANCES`	Não há instâncias ativas em seu provedor de capacidad e. Para obter mais informaçõ es sobre como gerenciar seus grupos do Auto Scaling, consulte Grupos do Auto Scaling no Guia do usuário do Amazon EC2 Auto Scaling.

Ação da API	Motivo da falha ou Motivo da interrupção	Causa
	`TaskFailedToStart: EMPTY CAPACITY PROVIDER`	Não existem instâncias no seu cluster. Provavelmente, isso se deve a um provedor de capacidade vazio ou porque as instâncias no provedor de capacidade não estão registradas no cluster. Para obter mais informações sobre como gerenciar seus grupos do Auto Scaling, consulte Grupos do Auto Scaling no Guia do usuário do Amazon EC2 Auto Scaling.
`GetTaskProtection`	`MISSING`	A tarefa especificada não foi encontrada. Verifique se o nome ou o ARN do cluster e o ARN ou o ID da tarefa são válidos.
	`TASK_NOT_VALID`	A tarefa especificada não faz parte de um serviço do Amazon ECS. Somente tarefas gerenciadas por serviços do Amazon ECS podem ser protegidas. Verifique o ARN ou o ID da tarefa e tente novamente.

Ação da API	Motivo da falha ou Motivo da interrupção	Causa
`RunTask` ou `StartTask`	`RESOURCE:*`	O recurso ou os recursos solicitados pela tarefa estão indisponíveis nas instâncias de contêiner no cluster. Se o recurso for CPU, memória, portas ou interfaces de redes elásticas, poderá ser necessário adicionar instâncias de contêineres ao cluster. Para erros `RESOURCE:ENI`, o cluster não tem quaisquer pontos de conexão com a interface de rede elástica disponíveis, que são necessários para tarefas que usam o modo de rede `awsvpc`. As instâncias do Amazon EC2 têm um limite para o número de interfaces de rede que podem ser anexadas a elas, e a interface de rede primária conta como uma delas. Para obter mais informações sobre quantas interfaces de rede e endereços IP privados são compatíveis com cada tipo de instância, consulte Endereços IP por interface de rede por tipo de instância no Manual do usuário do Amazon EC2. Para erros `RESOURCE:GPU`, o número de GPUs

Ação da API	Motivo da falha ou Motivo da interrupção	Causa
		solicitadas pela tarefa não está disponível e talvez você precise adicionar instâncias de contêiner habilitadas para GPU ao cluster. Para obter mais informações, consulte Definições de tarefa do Amazon ECS para workloads de GPU.
	AGENT	A instância de contêiner na qual você tentou iniciar uma tarefa tem um agente que está desconectado, no momento. Para evitar tempos de espera prolongados para a tarefa, a solicitação foi rejeitada. Para saber mais sobre como solucionar um agente que está desconectado, consulte Como solucionar problemas com um agente desconectado do Amazon ECS.
	LOCATION	A instância de contêiner na qual você tentou iniciar uma tarefa está em uma zona de disponibilidade diferente das sub-redes especificadas em `awsVpcConfiguration` .

Ação da API	Motivo da falha ou Motivo da interrupção	Causa
	`ATTRIBUTE`	A definição de tarefa contém um parâmetro que exige um atributo de instância de contêiner específico que não está disponível nas instância s de contêiner. Por exemplo, se sua tarefa usar o modo de rede `awsvpc`, mas não houver instâncias nas suas sub-redes especificadas com o atributo `ecs.capability.tas k-eni` . Para obter mais informações sobre quais atributos são obrigatórios para parâmetros de definição de tarefa específicos e variáveis de configuração de agentes, consulte Parâmetros de definição de tarefa do Amazon ECS para o tipo de inicializ ação do Fargate e Configura ção do agente de contêiner do Amazon ECS.
`StartTask`	`MISSING`	Não foi possível encontrar a instância de contêiner na qual você tentou executar a tarefa. Verifique se o cluster ou a região errada foram especific ados ou se o ARN ou ID da instância de contêiner estão escritos incorretamente.

Ação da API	Motivo da falha ou Motivo da interrupção	Causa
	`INACTIVE`	O registro da instância de contêiner na qual você tentou iniciar uma tarefa foi cancelado anteriormente junto ao Amazon ECS e não pode ser usado.
`StopServiceDeployment`	`ECS deployment failed`	Uma conta fraudulenta executou a API `StopServiceDeployment` .
`TagResource`	`InvalidParameterException`	O ARN do serviço que você está marcando tem o formato curto. Você deve migrar para o formato longo. Para obter informações sobre como migrar o ARN, consulte Migrar um ARN curto de um serviço do Amazon ECS para um ARN longo.
`UpdateTaskProtection`	`DEPLOYMENT_BLOCKED`	Não é possível definir a proteção de tarefas, pois uma ou mais tarefas protegidas estão impedindo que a implantação do serviço atinja um estado estável. Cancele a proteção de tarefas existentes ou espere até que a proteção de tarefas expire.

Ação da API	Motivo da falha ou Motivo da interrupção	Causa
	`MISSING`	A tarefa especificada não foi encontrada. Verifique se o nome ou o ARN do cluster e o ARN ou o ID da tarefa são válidos.
	`TASK_NOT_VALID`	A tarefa especificada não faz parte de um serviço do Amazon ECS. Somente tarefas gerenciadas por serviços do Amazon ECS podem ser protegidas. Verifique o ARN ou o ID da tarefa e tente novamente.

> **Note**
>
> Além dos cenários de falha descritos aqui, as operações de APIs também podem apresentar falha devido a exceções, resultando em respostas de erro. Para obter uma lista dessas exceções, consulte Common Errors (Erros comuns).

Segurança no Amazon Elastic Container Service

A segurança da nuvem na AWS é a nossa maior prioridade. Como cliente da AWS, você contará com um data center e uma arquitetura de rede criados para atender aos requisitos das organizações com as maiores exigências de segurança.

A segurança é uma responsabilidade compartilhada entre a AWS e você. O modelo de responsabilidade compartilhada descreve isto como segurança da nuvem e segurança na nuvem:

- Segurança da nuvem: a AWS é responsável pela proteção da infraestrutura que executa produtos da AWS na Nuvem AWS. A AWS também fornece serviços que podem ser usados com segurança. Auditores de terceiros testam e verificam regularmente a eficácia da nossa segurança como parte dos programas de conformidade da AWS. Para saber mais sobre os programas de compatibilidade que se aplicam ao Amazon Elastic Container Service, consulte Serviços da AWS no escopo pelo programa de compatibilidade.
- Segurança na nuvem: sua responsabilidade é determinada pelo serviço da AWS que você usa. Você também é responsável por outros fatores, incluindo a confidencialidade de seus dados, os requisitos da empresa e as leis e regulamentos aplicáveis.

Esta documentação ajuda você a entender como aplicar o modelo de responsabilidade compartilhada quando usar o Amazon ECS. Os tópicos a seguir mostram como configurar o Amazon ECS para atender aos seus objetivos de segurança e compatibilidade. Você também aprenderá como usar outros serviços da AWS que ajudam você a monitorar e proteger os recursos do Amazon ECS.

Tópicos

- Gerenciamento de Identidade e Acesso para o Amazon Elastic Container Service
- Registrar e monitorar no Amazon Elastic Container Service
- Validação da conformidade do Amazon Elastic Container Service
- Padrão Federal de Processamento de Informações (FIPS-140) do AWS Fargate
- Segurança da infraestrutura no Amazon Elastic Container Service
- Modelo de responsabilidade compartilhada da AWS para o Amazon ECS
- Práticas recomendadas de segurança de rede para o Amazon ECS
- Práticas recomendadas de segurança para tarefas e contêineres do Amazon ECS

Gerenciamento de Identidade e Acesso para o Amazon Elastic Container Service

O AWS Identity and Access Management (IAM) é um serviço da AWS service (Serviço da AWS) que ajuda um administrador a controlar com segurança o acesso aos recursos da AWS. Os administradores do IAM controlam quem pode ser autenticado (conectado) e autorizado (ter permissões) para usar os recursos do Amazon ECS. O IAM é um AWS service (Serviço da AWS) que pode ser usado sem custo adicional.

Tópicos

- Público
- Autenticação com identidades
- Gerenciar o acesso usando políticas
- Como o Amazon Elastic Container Service funciona com o IAM
- Exemplos de políticas baseadas em identidade do Amazon Elastic Container Service
- Políticas gerenciadas pela AWS para o Amazon Elastic Container Service
- Uso de perfis vinculados ao serviço para o Amazon ECS
- Perfis do IAM para o Amazon ECS
- Permissões necessárias para usar o console do Amazon ECS
- Permissões necessárias para funções do Lambda em implantações azul/verde do Amazon ECS
- Permissões obrigatórias do IAM para o ajuste de escala automático do serviço Amazon ECS
- Conceder permissão para a atribuição de tags a recursos durante a criação
- Solução de problemas de identidade e acesso do Amazon Elastic Container Service
- Práticas recomendadas do IAM para o Amazon ECS

Público

A maneira como o AWS Identity and Access Management (IAM) é usado varia, dependendo do trabalho que é realizado no Amazon ECS.

Usuário do serviço: se você usar o serviço do Amazon ECS para fazer seu trabalho, o administrador fornecerá as credenciais e as permissões de que você precisa. À medida que mais recursos do Amazon ECS forem usados para realizar o trabalho, talvez sejam necessárias permissões

adicionais. Entender como o acesso é gerenciado pode ajudar a solicitar as permissões corretas ao administrador. Se você não puder acessar um recurso no Amazon ECS, consulte Solução de problemas de identidade e acesso do Amazon Elastic Container Service.

Administrador do serviço: se você for o responsável pelos recursos do AmazonECS em sua empresa, provavelmente terá acesso total ao Amazon ECS. Cabe a você determinar quais funcionalidades e recursos do Amazon ECS os usuários do seu serviço devem acessar. Envie as solicitações ao administrador do IAM para alterar as permissões dos usuários de serviço. Revise as informações nesta página para compreender os conceitos básicos do IAM. Para saber mais sobre como a empresa pode usar o IAM com o Amazon ECS, consulte Como o Amazon Elastic Container Service funciona com o IAM.

Administrador do IAM: se você é um administrador do IAM, talvez queira saber detalhes sobre como pode escrever políticas para gerenciar o acesso ao Amazon ECS. Para visualizar exemplos de políticas baseadas em identidade do Amazon ECS que podem ser usadas no IAM, consulte Exemplos de políticas baseadas em identidade do Amazon Elastic Container Service.

Autenticação com identidades

A autenticação é a forma como fazer login na AWS usando suas credenciais de identidade. É necessário ser autenticado (fazer login na AWS) como Usuário raiz da conta da AWS, como usuário do IAM, ou assumindo um perfil do IAM.

É possível fazer login na AWS como uma identidade federada usando credenciais fornecidas por uma fonte de identidades. AWS IAM Identity Center (Centro de Identidade do IAM), a autenticação única da empresa e as suas credenciais do Google ou do Facebook são exemplos de identidades federadas. Quando você faz login como identidade federada, o administrador já configurou anteriormente a federação de identidades usando perfis do IAM. Quando você acessa a AWS usando a federação, está indiretamente assumindo um perfil.

A depender do tipo de usuário, você pode fazer login no AWS Management Console ou no portal de acesso AWS. Para obter mais informações sobre como fazer login na AWS, consulte Como fazer login na Conta da AWS no Guia do usuário do Início de Sessão da AWS.

Se você acessar a AWS de forma programática, a AWS fornecerá um kit de desenvolvimento de software (SDK) e uma interface de linha de comandos (CLI) para você assinar de forma criptográfica as solicitações usando as suas credenciais. Se você não utilizar as ferramentas da AWS, deverá assinar as solicitações por conta própria. Para obter mais informações sobre como usar o método

recomendado para designar solicitações por conta própria, consulte Versão 4 do AWS Signature para solicitações de API no Guia do usuário do IAM.

Independente do método de autenticação usado, também pode ser necessário fornecer informações adicionais de segurança. Por exemplo, a AWS recomenda o uso da autenticação multifator (MFA) para aumentar a segurança de sua conta. Para saber mais, consulte Autenticação multifator no Guia do usuário do AWS IAM Identity Center e Usar a autenticação multifator da AWS no IAM no Guia do usuário do IAM.

Usuário-raiz Conta da AWS

Ao criar uma Conta da AWS, você começa com uma identidade de login com acesso completo a todos os Serviços da AWS e recursos na conta. Essa identidade, chamada usuário-raiz da Conta da AWS, é acessada por login com o endereço de e-mail e a senha usada para criar a conta. É altamente recomendável não usar o usuário-raiz para tarefas diárias. Proteja as credenciais do usuário-raiz e use-as para executar as tarefas que somente ele puder executar. Para obter a lista completa das tarefas que exigem login como usuário-raiz, consulte Tarefas que exigem credenciais de usuário-raiz no Guia do Usuário do IAM.

Identidade federada

Como prática recomendada, exija que os usuários, inclusive os que precisam de acesso de administrador, usem a federação com um provedor de identidades para acessar os Serviços da AWS usando credenciais temporárias.

Uma identidade federada é um usuário de seu diretório de usuários corporativos, um provedor de identidades da web, o AWS Directory Service, o diretório do Centro de Identidade ou qualquer usuário que acesse os Serviços da AWS usando credenciais fornecidas por meio de uma fonte de identidade. Quando as identidades federadas acessam Contas da AWS, elas assumem perfis que fornecem credenciais temporárias.

Para o gerenciamento de acesso centralizado, é recomendável usar o AWS IAM Identity Center. É possível criar usuários e grupos no IAM Identity Center ou conectar-se e sincronizar com um conjunto de usuários e grupos em sua própria fonte de identidade para uso em todas as suas Contas da AWS e aplicações. Para obter mais informações sobre o Centro de Identidade do IAM, consulte O que é o Centro de Identidade do IAM? no Guia do Usuário do AWS IAM Identity Center.

Usuários e grupos do IAM

Um usuário do IAM é uma identidade dentro da Conta da AWS que tem permissões específicas para uma única pessoa ou aplicação. Sempre que possível, é recomendável contar com credenciais temporárias em vez de criar usuários do IAM com credenciais de longo prazo, como senhas e chaves de acesso. No entanto, se você tiver casos de uso específicos que exijam credenciais de longo prazo com usuários do IAM, é recomendável alternar as chaves de acesso. Para obter mais informações, consulte Alternar as chaves de acesso regularmente para casos de uso que exijam credenciais de longo prazo no Guia do Usuário do IAM.

Um grupo do IAM é uma identidade que especifica uma coleção de usuários do IAM. Não é possível fazer login como um grupo. É possível usar grupos para especificar permissões para vários usuários de uma vez. Os grupos facilitam o gerenciamento de permissões para grandes conjuntos de usuários. Por exemplo, você pode ter um grupo chamado IAMAdmins e conceder a esse grupo permissões para administrar recursos do IAM.

Usuários são diferentes de perfis. Um usuário é exclusivamente associado a uma pessoa ou a uma aplicação, mas um perfil pode ser assumido por qualquer pessoa que precisar dele. Os usuários têm credenciais permanentes de longo prazo, mas os perfis fornecem credenciais temporárias. Para saber mais, consulte Casos de uso para usuários do IAM no Guia do usuário do IAM.

Perfis do IAM

Um perfil do IAM é uma identidade na Conta da AWS que tem permissões específicas. Ele é semelhante a um usuário do IAM, mas não está associado a uma pessoa específica. Para assumir temporariamente um perfil do IAM no AWS Management Console, você pode alternar de um usuário para um perfil do IAM (console). É possível presumir um perfil chamando uma operação de API da AWS CLI ou da AWS, ou usando um URL personalizado. Para obter mais informações sobre métodos para usar perfis, consulte Métodos para assumir um perfil no Guia do usuário do IAM.

Perfis do IAM com credenciais temporárias são úteis nas seguintes situações:

- Acesso de usuário federado: para atribuir permissões a identidades federadas, é possível criar um perfil e definir permissões para ele. Quando uma identidade federada é autenticada, essa identidade é associada ao perfil e recebe as permissões definidas por ele. Para ter mais informações sobre perfis para federação, consulte Criar um perfil para um provedor de identidade de terceiros (federação) no Guia do usuário do IAM. Se usar o Centro de Identidade do IAM, configure um conjunto de permissões. Para controlar o que suas identidades podem acessar após a autenticação, o Centro de Identidade do IAM correlaciona o conjunto de permissões a

um perfil no IAM. Para obter informações sobre conjuntos de permissões, consulte Conjuntos de Permissões no Guia do Usuário do AWS IAM Identity Center.

- Permissões temporárias para usuários do IAM: um usuário ou um perfil do IAM pode presumir um perfil do IAM para obter temporariamente permissões diferentes para uma tarefa específica.
- Acesso entre contas: é possível usar um perfil do IAM para permitir que alguém (uma entidade principal confiável) em outra conta acesse recursos em sua conta. Os perfis são a principal forma de conceder acesso entre contas. No entanto, alguns Serviços da AWS permitem anexar uma política diretamente a um recurso (em vez de usar um perfil como proxy). Para conhecer a diferença entre perfis e políticas baseadas em recurso para acesso entre contas, consulte Acesso a recursos entre contas no IAM no Guia do usuário do IAM.
- Acesso entre serviços: alguns Serviços da AWS usam recursos em outros Serviços da AWS. Por exemplo, quando você faz uma chamada em um serviço, é comum que esse serviço execute aplicações no Amazon EC2 ou armazene objetos no Amazon S3. Um serviço pode fazer isso usando as permissões da entidade principal da chamada, usando um perfil de serviço ou um perfil vinculado ao serviço.
  - Sessões de acesso direto (FAS): qualquer pessoa que utilizar um perfil ou usuário do IAM para executar ações na AWS é considerada uma entidade principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O recurso FAS usa as permissões da entidade principal chamando um AWS service (Serviço da AWS), bem como o AWS service (Serviço da AWS) solicitante, para fazer solicitações para serviços subsequentes. As solicitações de FAS são feitas somente quando um serviço recebe uma solicitação que exige interações com outros Serviços da AWS ou com recursos para serem concluídas. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte Sessões de acesso direto.
  - Perfil de serviço: um perfil de serviço é um perfil do IAM que um serviço assume para executar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte Criar um perfil para delegar permissões a um AWS service (Serviço da AWS) no Guia do Usuário do IAM.
  - Perfil vinculado a serviço: um perfil vinculado a serviço é um tipo de perfil de serviço vinculado a um AWS service (Serviço da AWS). O serviço pode presumir o perfil para executar uma ação em seu nome. Perfis vinculadas ao serviço aparecem em sua Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não editar as permissões para perfis vinculados a serviço.
- Aplicações em execução no Amazon EC2: é possível usar um perfil do IAM para gerenciar credenciais temporárias para aplicações em execução em uma instância do EC2 e fazer

solicitações da AWS CLI ou da API da AWS. É preferível fazer isso a armazenar chaves de acesso na instância do EC2. Para atribuir um perfil da AWS a uma instância do EC2 e disponibilizá-la para todas as suas aplicações, crie um perfil de instância que esteja anexado a ela. Um perfil de instância contém o perfil e permite que os programas em execução na instância do EC2 obtenham credenciais temporárias. Para mais informações, consulte Utilizar um perfil do IAM para conceder permissões a aplicações em execução nas instâncias do Amazon EC2 no Guia do usuário do IAM.

Gerenciar o acesso usando políticas

Você controla o acesso na AWS criando políticas e anexando-as a identidades ou recursos da AWS. Uma política é um objeto na AWS que, quando associado a uma identidade ou recurso, define suas permissões. A AWS avalia essas políticas quando uma entidade principal (usuário, usuário-raiz ou sessão de perfil) faz uma solicitação. As permissões nas políticas determinam se a solicitação será permitida ou negada. A maioria das políticas é armazenada na AWS como documentos JSON. Para obter mais informações sobre a estrutura e o conteúdo de documentos de políticas JSON, consulte Visão geral das políticas JSON no Guia do usuário do IAM.

Os administradores podem usar as políticas JSON da AWS para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

Por padrão, usuários e perfis não têm permissões. Para conceder permissão aos usuários para executar ações nos recursos que eles precisam, um administrador do IAM pode criar políticas do IAM. O administrador pode então adicionar as políticas do IAM aos perfis e os usuários podem assumir os perfis.

As políticas do IAM definem permissões para uma ação independentemente do método usado para executar a operação. Por exemplo, suponha que você tenha uma política que permite a ação `iam:GetRole`. Um usuário com essa política pode obter informações de perfis do AWS Management Console, da AWS CLI ou da API AWS.

Políticas baseadas em identidade

As políticas baseadas em identidade são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como usuário, grupo de usuários ou perfil do IAM. Essas políticas controlam quais ações os usuários e perfis podem realizar, em quais recursos e em que condições. Para saber como criar uma política baseada em identidade, consulte Definir permissões personalizadas do IAM com as políticas gerenciadas pelo cliente no Guia do Usuário do IAM.

As políticas baseadas em identidade podem ser categorizadas como políticas em linha ou políticas gerenciadas. As políticas em linha são anexadas diretamente a um único usuário, grupo ou perfil. As políticas gerenciadas são políticas independentes que podem ser anexadas a vários usuários, grupos e perfis na Conta da AWS. As políticas gerenciadas incluem políticas gerenciadas pela AWS e políticas gerenciadas pelo cliente. Para saber como escolher entre uma política gerenciada ou uma política em linha, consulte Escolher entre políticas gerenciadas e políticas em linha no Guia do usuário do IAM.

Políticas baseadas em recursos

Políticas baseadas em recursos são documentos de políticas JSON que você anexa a um recurso. São exemplos de políticas baseadas em recursos as políticas de confiança de perfil do IAM e as políticas de bucket do Amazon S3. Em serviços compatíveis com políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. Para o atributo ao qual a política está anexada, a política define quais ações uma entidade principal especificado pode executar nesse atributo e em que condições. Você deve especificar uma entidade principal em uma política baseada em recursos. As entidades principais podem incluir contas, usuários, funções, usuários federados ou Serviços da AWS.

Políticas baseadas em recursos são políticas em linha localizadas nesse serviço. Não é possível usar as políticas gerenciadas pela AWS do IAM em uma política baseada em atributos.

Listas de controle de acesso (ACLs)

As listas de controle de acesso (ACLs) controlam quais entidades principais (membros, usuários ou perfis da conta) têm permissões para acessar um recurso. As ACLs são semelhantes às políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

Amazon S3, AWS WAF e Amazon VPC são exemplos de serviços que oferecem compatibilidade com ACLs. Para saber mais sobre ACLs, consulte Visão geral da lista de controle de acesso (ACL) no Guia do Desenvolvedor do Amazon Simple Storage Service.

Outros tipos de política

A AWS oferece compatibilidade com tipos de política menos comuns. Esses tipos de política podem definir o máximo de permissões concedidas a você pelos tipos de política mais comuns.

- Limites de permissões: um limite de permissões é um recurso avançado no qual você define o máximo de permissões que uma política baseada em identidade pode conceder a uma entidade do IAM (usuário ou perfil do IAM). É possível definir um limite de permissões para uma entidade.

As permissões resultantes são a interseção das políticas baseadas em identidade de uma entidade com seus limites de permissões. As políticas baseadas em recurso que especificam o usuário ou o perfil no campo `Principal` não são limitadas pelo limite de permissões. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações sobre limites de permissões, consulte Limites de permissões para identidades do IAM no Guia do usuário do IAM.

- Políticas de controle de serviço (SCPs): SCPs são políticas JSON que especificam as permissões máximas para uma organização ou unidade organizacional (UO) no AWS Organizations. O AWS Organizations é um serviço que agrupa e gerencia centralmente várias Contas da AWS pertencentes a sua empresa. Se você habilitar todos os recursos em uma organização, poderá aplicar políticas de controle de serviço (SCPs) a qualquer uma ou a todas as contas. A SCP limita as permissões para entidades em contas de membros, o que inclui cada Usuário raiz da conta da AWS. Para obter mais informações sobre o Organizations e SCPs, consulte Service control policies no Guia do usuário do AWS Organizations.
- Políticas de controle de recursos (RCPs): RCPs são políticas JSON que podem ser usadas para definir o máximo de permissões disponíveis para recursos em suas contas sem atualizar as políticas do IAM anexadas a cada recurso que você possui. A RCP limita as permissões para recursos nas contas-membro e pode afetar as permissões efetivas para identidades, incluindo o Usuário raiz da conta da AWS, independentemente de pertencerem a sua organização. Consulte mais informações sobre o Organizations e as RCPs, incluindo uma lista de Serviços da AWS compatível com RCPs em Resource control policies (RCPs) no Guia do usuário do AWS Organizations.
- Políticas de sessão: são políticas avançadas que você transmite como um parâmetro quando cria de forma programática uma sessão temporária para um perfil ou um usuário federado. As permissões da sessão resultante são a interseção das políticas baseadas em identidade do usuário ou do perfil e das políticas de sessão. As permissões também podem ser provenientes de uma política baseada em recursos. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações, consulte Políticas de sessão no Guia do usuário do IAM.

Vários tipos de política

Quando vários tipos de política são aplicáveis a uma solicitação, é mais complicado compreender as permissões resultantes. Para saber como a AWS determina se deve permitir uma solicitação quando há vários tipos de política envolvidos, consulte Lógica da avaliação de políticas no Guia do usuário do IAM.

Como o Amazon Elastic Container Service funciona com o IAM

Antes de usar o IAM para gerenciar o acesso ao Amazon ECS, entenda que recursos do IAM estão disponíveis para uso com o Amazon ECS.

Recurso do IAM	Suporte ao Amazon ECS
Políticas baseadas em identidade	Sim
Políticas baseadas em recurso	Não
Ações de políticas	Sim
Recursos de políticas	Parcial
Chaves de condição de políticas	Sim
ACLs	Não
ABAC (tags em políticas)	Sim
Credenciais temporárias	Sim
Sessões de acesso direto (FAS)	Sim
Perfis de serviço	Sim
Perfis vinculados a serviço	Sim

Para obter uma visualização de alto nível de como o Amazon ECS e outros serviços da AWS funcionam com a maioria dos recursos do IAM, consulte Serviços da AWS compatíveis com o IAM no Guia do usuário do IAM.

Políticas baseadas em identidade do Amazon ECS

Suporte a políticas baseadas em identidades: sim

As políticas baseadas em identidade são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como usuário do IAM, grupo de usuários ou perfil. Essas

políticas controlam quais ações os usuários e perfis podem realizar, em quais recursos e em que condições. Para saber como criar uma política baseada em identidade, consulte Definir permissões personalizadas do IAM com as políticas gerenciadas pelo cliente no Guia do Usuário do IAM.

Com as políticas baseadas em identidade do IAM, é possível especificar ações e recursos permitidos ou negados, assim como as condições sob as quais as ações são permitidas ou negadas. Você não pode especificar a entidade principal em uma política baseada em identidade porque ela se aplica ao usuário ou perfil ao qual ela está anexada. Para saber mais sobre todos os elementos que podem ser usados em uma política JSON, consulte Referência de elemento de política JSON do IAM no Guia do usuário do IAM.

Exemplos de políticas baseadas em identidade para o Amazon ECS

Para visualizar exemplos de políticas baseadas em identidade do Amazon ECS, consulte Exemplos de políticas baseadas em identidade do Amazon Elastic Container Service.

Políticas baseadas em recursos do Amazon ECS

Suporte a políticas baseadas em recursos: não

Políticas baseadas em recursos são documentos de políticas JSON que você anexa a um recurso. São exemplos de políticas baseadas em recursos as políticas de confiança de perfil do IAM e as políticas de bucket do Amazon S3. Em serviços compatíveis com políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. Para o atributo ao qual a política está anexada, a política define quais ações uma entidade principal especificado pode executar nesse atributo e em que condições. Você deve especificar uma entidade principal em uma política baseada em recursos. As entidades principais podem incluir contas, usuários, perfis, usuários federados ou Serviços da AWS.

Para permitir o acesso entre contas, você pode especificar uma conta inteira ou as entidades do IAM em outra conta como a entidade principal em uma política baseada em recursos. Adicionar uma entidade principal entre contas à política baseada em recurso é apenas metade da tarefa de estabelecimento da relação de confiança. Quando a entidade principal e o recurso estão em diferentes Contas da AWS, um administrador do IAM da conta confiável também deve conceder à entidade principal (usuário ou perfil) permissão para acessar o recurso. Eles concedem permissão ao anexar uma política baseada em identidade para a entidade. No entanto, se uma política baseada em recurso conceder acesso a uma entidade principal na mesma conta, nenhuma política baseada em identidade adicional será necessária. Consulte mais informações em Acesso a recursos entre contas no IAM no Guia do usuário do IAM.

Ações de políticas para o Amazon ECS

Suporte a ações de políticas: sim

Os administradores podem usar as políticas JSON da AWS para especificar quem tem acesso a quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

O elemento `Action` de uma política JSON descreve as ações que podem ser usadas para permitir ou negar acesso em uma política. As ações de políticas geralmente têm o mesmo nome que a operação de API da AWS associada. Existem algumas exceções, como ações somente de permissão, que não têm uma operação de API correspondente. Algumas operações também exigem várias ações em uma política. Essas ações adicionais são chamadas de ações dependentes.

Incluem ações em uma política para conceder permissões para executar a operação associada.

Para ver uma lista das ações do Amazon ECS, consulte Ações definidas pelo Amazon Elastic Conteiner Service na Referência de autorização de serviço.

As ações de política no Amazon ECS usam o seguinte prefixo antes da ação:

```
ecs
```

Para especificar várias ações em uma única instrução, separe-as com vírgulas.

```
"Action": [
      "ecs:action1",
      "ecs:action2"
         ]
```

Você também pode especificar várias ações usando caracteres-curinga (*). Por exemplo, para especificar todas as ações que começam com a palavra `Describe`, inclua a seguinte ação:

```
"Action": "ecs:Describe*"
```

Para visualizar exemplos de políticas baseadas em identidade do Amazon ECS, consulte Exemplos de políticas baseadas em identidade do Amazon Elastic Container Service.

Recursos de políticas para o Amazon ECS

Suporte a recursos de políticas: parcial

Os administradores podem usar as políticas JSON da AWS para especificar quem tem acesso a quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

O elemento de política JSON `Resource` especifica o objeto ou os objetos aos quais a ação se aplica. As instruções devem incluir um elemento `Resource` ou `NotResource`. Como prática recomendada, especifique um recurso usando seu nome do recurso da Amazon (ARN). Isso pode ser feito para ações que oferecem compatibilidade com um tipo de recurso específico, conhecido como permissões em nível de recurso.

Para ações que não oferecem compatibilidade com permissões em nível de recurso, como operações de listagem, use um curinga (*) para indicar que a instrução se aplica a todos os recursos.

```
"Resource": "*"
```

Para ver uma lista dos tipos de recursos do Amazon ECS e seus ARNs, consulte Recursos definidos pelo Amazon Elastic Conteiner Service na Referência de autorização de serviço. Para saber com quais ações você pode especificar o ARN de cada recurso, consulte Ações definidas pelo Amazon Elastic Container Service.

Algumas ações da API do Amazon ECS oferecem suporte a vários recursos. Por exemplo, é possível fazer referência a vários clusters ao chamar a ação de API `DescribeClusters`. Para especificar vários recursos em uma única declaração, separe os ARNs com vírgulas.

```
"Resource": [
      "EXAMPLE-RESOURCE-1",
      "EXAMPLE-RESOURCE-2"
```

Por exemplo, o recurso de cluster do Amazon ECS tem o seguinte ARN:

```
arn:${Partition}:ecs:${Region}:${Account}:cluster/${clusterName}
```

Para especificar os clusters `my-cluster-1` e `my-cluster-2` em sua instrução, use os seguintes ARNs:

```
"Resource": [
        "arn:aws:ecs:us-east-1:123456789012:cluster/my-cluster-1",
        "arn:aws:ecs:us-east-1:123456789012:cluster/my-cluster-2"
```

Para especificar todos os clusters que pertencem a uma conta específica, use o caractere curinga (*):

```
"Resource": "arn:aws:ecs:us-east-1:123456789012:cluster/*"
```

Para definições de tarefa, você pode especificar a revisão mais recente ou uma revisão específica.

Para especificar todas as revisões da definição de tarefa, use o caractere curinga (*):

```
"Resource:arn:${Partition}:ecs:${Region}:${Account}:task-definition/
${TaskDefinitionFamilyName}:*"
```

Para especificar uma revisão de definição de tarefa específica, use ${TaskDefinitionRevisionNumber}:

```
"Resource:arn:${Partition}:ecs:${Region}:${Account}:task-definition/
${TaskDefinitionFamilyName}:${TaskDefinitionRevisionNumber}"
```

Para visualizar exemplos de políticas baseadas em identidade do Amazon ECS, consulte Exemplos de políticas baseadas em identidade do Amazon Elastic Container Service.

Chaves de condição de políticas para o Amazon ECS

Suporte a chaves de condição de políticas específicas do serviço: sim

Os administradores podem usar as políticas JSON da AWS para especificar quem tem acesso a quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

O elemento `Condition` (ou bloco `Condition`) permite que você especifique condições nas quais uma instrução estiver em vigor. O elemento `Condition` é opcional. É possível criar expressões condicionais que usem agentes de condição, como “igual a” ou “menor que”, para fazer a condição da política corresponder aos valores na solicitação.

Se você especificar vários elementos de `Condition` em uma declaração ou várias chaves em um único elemento de `Condition`, a AWS os avaliará usando uma operação lógica `AND`. Se você especificar vários valores para uma única chave de condição, a AWS avaliará a condição usando uma operação lógica `OR`. Todas as condições devem ser atendidas antes que as permissões da instrução sejam concedidas.

Você também pode usar variáveis de espaço reservado ao especificar condições. Por exemplo, é possível conceder a um usuário do IAM permissão para acessar um recurso somente se ele estiver

marcado com seu nome de usuário do IAM. Para obter mais informações, consulte Elementos da política do IAM: variáveis e tags no Guia do usuário do IAM.

A AWS oferece compatibilidade com chaves de condição globais e chaves de condição específicas do serviço. Para ver todas as chaves de condição globais da AWS, consulte Chaves de contexto de condição globais da AWS no Guia do usuário do IAM.

O Amazon ECS oferece suporte às seguintes chaves de condição específicas ao serviço que você pode usar para fornecer filtragem refinada para suas políticas do IAM:

Chave de condição	Descrição	Tipos de avaliação
aws:RequestTag/${TagKey}	Essa chave de contexto é formatada `"aws:RequestTag/ tag-key":"tag-value "` em que *tag-key* e *tag-value* são um par de chave e valor da tag. Verifica se o par de chave/valor da tag está presente em uma solicitação da AWS. Por exemplo, você pode verificar que a solicitação inclui a chave de tag `"Dept"` e se ela tem o valor `"Accounting"` .	String
aws:ResourceTag/${TagKey}	Essa chave de contexto é formatada `"aws:ResourceTag/ tag-key":"tag-value "` em que *tag-key* e *tag-value* são um par de chave e valor da tag. Verifica se a tag anexada ao recurso de identidade (usuário ou função) corresponde ao nome e ao valor da chave especificada.	String
aws:TagKeys	Essa chave de contexto é formatada como `"aws:TagKeys":" tag-key"`, onde *tag-key* é uma lista de chaves de tags sem valores (por exemplo, `["Dept","Cost-Center"]`). Verifica as chaves de tags que estão presentes em uma solicitação da AWS.	String

Chave de condição	Descrição	Tipos de avaliação
ecs:ResourceTag/${TagKey}	Essa chave de contexto é formatada "ecs:ResourceTag/ *tag-key*":"*tag-value* " em que *tag-key* e *tag-value* são um par de chave e valor da tag. Verifica se a tag anexada ao recurso de identidade (usuário ou função) corresponde ao nome e ao valor da chave especificada.	String
ecs:account-setting	A chave de contexto é formatada como "ecs:account-setting":" *account-setting* ", onde *account-setting* é o nome da configuração da conta.	String
ecs:auto-assign-public-ip	A chave de contexto é formatada como "ecs:auto-assign-public-ip":" *value*", onde *value-* é "true" ou "false".	String
ecs:capacity-provider	A chave de contexto é formatada como "ecs:capacity-provider":" *capacity-provider-arn* ", onde *capacity-provider-arn* é o ARN do provedor de capacidade.	ARN, nulo
ecs:cluster	A chave de contexto é formatada como "ecs:cluster":" *cluster-arn* ", onde *cluster-arn* é o ARN do cluster do Amazon ECS.	ARN, nulo
ecs:compute-compatability	A chave de contexto é formatada como "ecs:compute-compatability":" *compatability-type* ", em que *compatability-type* é “FARGATE”, “EC2” ou “EXTERNAL”.	String
ecs:container-instances	A chave de contexto é formatada como "ecs:container-instances":" *container-instance-arns*", onde *container-instance-arns* é um ou mais ARNs de instância de contêiner.	ARN, nulo

Chave de condição	Descrição	Tipos de avaliação
ecs:container-name	A chave de contexto é formatada como `"ecs:container-name":"` *container-name* `"`, onde *conteiner-instance-* é o nome de um contêiner do Amazon ECS especificado na definição da tarefa.	String
ecs:enable-execute-command	A chave de contexto é formatada como `"ecs:enable-execute-command":"` *value*`"`, onde *value-* é `"true"` ou `"false"`.	String
ecs:enable-service-connect	A chave de contexto é formatada como `"ecs:enable-service-connect":"` *value*`"`, onde *value* é `"true"` ou `"false"`.	String
ecs:enable-ebs-volumes	A chave de contexto é formatada como `"ecs:enable-ebs-volumes":"` *value*`"`, onde *value* é `"true"` ou `"false"`.	String
ecs:enable-managed-tags	A chave de contexto é formatada como `"ecs:enable-managed-tags":"` *value*`"`, onde *value* é `"true"` ou `"false"`.	String
ecs:enable-vpc-lattice	A chave de contexto é formatada como `"ecs:vpc-lattice":"` *value*`"`, onde *value* é `"true"` ou `"false"`.	String
ecs:fargate-ephemeral-storage-kms-key	A chave de contexto é formatada como, `"ecs:fargate-ephemeral-storage-kms-key":"` *key*`"` onde *key* é o ID da chave AWS KMS.	String
ecs:namespace	A chave de contexto é formatada como `"ecs:namespace":"` *namespace-arn* `"`, onde *namespace-arn* é o ARN para o namespace do AWS Cloud Map.	ARN, nulo

Chave de condição	Descrição	Tipos de avaliação
ecs:propagate-tags	A chave de contexto é formatada como "ecs:propagate-tags":" *value*", onde *value* é "TASK_DEFINITION" , "SERVICE" ou "NONE".	String
ecs:service	A chave de contexto é formatada como "ecs:service":" *service-arn* ", onde *service-arn* é o ARN para o serviço do Amazon ECS.	ARN, nulo
ecs:task-definition	A chave de contexto é formatada como "ecs:task-definition":" *task-definition-arn* ", onde *task-definition-arn* é o ARN da definição de tarefa do Amazon ECS.	ARN, nulo
ecs:subnet	A chave de contexto é formatada como "ecs:subnet":" *subnets*" onde *subnets* são as sub-redes de uma tarefa ou serviço que usa o modo awsvpc.	String
ecs:task	A chave de contexto é formatada como "ecs:task":" *task-arn*", onde *task-arn* é o ARN para o serviço do Amazon ECS.	ARN, nulo
ecs:task-cpu	A chave de contexto é formatada como "ecs:task-cpu":"*task-cpu*", onde *task-cpu* é a CPU da tarefa, como um número inteiro com 1024 = 1 vCPU.	Inteiro
ecs:task-memory	A chave de contexto é formatada como "ecs:task-memory":" *task-memory* ", onde *task-memory* é a memória da tarefa em MiB.	Inteiro

Para ver uma lista de chaves de condição do Amazon ECS, consulte Chaves de condição para o Amazon Elastic Conteiner Service na Referência de autorização de serviço. Para saber com quais ações e recursos você pode usar a chave de condição, consulte Ações definidas pelo Amazon Elastic Container Service.

Para visualizar exemplos de políticas baseadas em identidade do Amazon ECS, consulte Exemplos de políticas baseadas em identidade do Amazon Elastic Container Service.

Listas de controle de acesso (ACLs) no Amazon ECS

Suporte a ACLs: não

As listas de controle de acesso (ACLs) controlam quais entidades principais (membros, usuários ou perfis da conta) têm permissões para acessar um recurso. As ACLs são semelhantes às políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

Controle de acesso baseado em atributos (ABAC) com o Amazon ECS

> **Important**
>
> O Amazon ECS oferece suporte ao controle de acesso baseado em atributos para todos os recursos do Amazon ECS. Para determinar se é possível usar atributos para definir o escopo de uma ação, use a tabela Ações definidas pelo Amazon ECS em Referência da autorização de serviço. Primeiro, verifique se há um recurso na coluna Recurso. Em seguida, use a coluna Chaves de condição para ver as chaves da combinação de ação/recurso.

Suporte a ABAC (tags em políticas): sim

O controle de acesso por atributo (ABAC) é uma estratégia de autorização que define as permissões com base em atributos. Na AWS, esses atributos são chamados de tags. É possível anexar tags a entidades do IAM (usuários ou perfis) e a muitos recursos da AWS. Marcar de entidades e atributos é a primeira etapa do ABAC. Em seguida, você cria políticas de ABAC para permitir operações quando a tag da entidade principal corresponder à tag do recurso que ela estiver tentando acessar.

O ABAC é útil em ambientes que estão crescendo rapidamente e ajuda em situações em que o gerenciamento de políticas se torna um problema.

Para controlar o acesso baseado em tags, forneça informações sobre as tags no elemento de condição de uma política usando as `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`* ou chaves de condição `aws:TagKeys`.

Se um serviço for compatível com as três chaves de condição para cada tipo de recurso, o valor será Sim para o serviço. Se um serviço for compatível com as três chaves de condição somente para alguns tipos de recursos, o valor será Parcial

Para obter mais informações sobre o ABAC, consulte Definir permissões com autorização do ABAC no Guia do usuário do IAM. Para visualizar um tutorial com etapas para configurar o ABAC, consulte Usar controle de acesso baseado em atributos (ABAC) no Guia do usuário do IAM.

Para obter mais informações sobre recursos de marcação do Amazon ECS, consulte Marcação de recursos do Amazon ECS.

Para visualizar um exemplo de política baseada em identidade para limitar o acesso a um recurso baseado em tags desse recurso, consulte Descrever serviços do Amazon ECS com base em etiquetas.

Uso de credenciais temporárias com o Amazon ECS

Suporte a credenciais temporárias: sim

Alguns Serviços da AWS não funcionam quando você faz login usando credenciais temporárias. Para obter informações adicionais, como quais Serviços da AWS funcionam com credenciais temporárias, consulte Serviços da AWS que funcionem com o IAM no Guia do usuário do IAM.

Você está usando credenciais temporárias se fizer login no AWS Management Console por qualquer método, exceto nome de usuário e uma senha. Por exemplo, quando você acessa a AWS usando o link de autenticação única (SSO) da sua empresa, esse processo cria credenciais temporárias automaticamente. Você também cria automaticamente credenciais temporárias quando faz login no console como usuário e, em seguida, alterna perfis. Para obter mais informações sobre como alternar funções, consulte Alternar para um perfil do IAM (console) no Guia do usuário do IAM.

É possível criar credenciais temporárias manualmente usando a API AWS CLI ou AWS. Em seguida, você pode usar essas credenciais temporárias para acessar a AWS. A AWS recomenda que você gere credenciais temporárias dinamicamente em vez de usar chaves de acesso de longo prazo. Para obter mais informações, consulte Credenciais de segurança temporárias no IAM.

Sessões de acesso direto para o Amazon ECS

Suporte ao recurso de encaminhamento de sessões de acesso (FAS): sim

O usuário ou perfil do IAM usado para executar ações na AWS é considerado uma entidade principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O recurso FAS usa as permissões da entidade principal chamando um AWS service (Serviço da AWS), bem como o AWS service (Serviço da AWS) solicitante, para fazer solicitações para serviços subsequentes. As solicitações de FAS são feitas somente quando um serviço recebe uma solicitação que exige interações com outros Serviços da AWS ou com recursos

para serem concluídas. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte Sessões de acesso direto.

Funções de serviço para o Amazon ECS

Suporte a perfis de serviço: sim

O perfil de serviço é um perfil do IAM que um serviço assume para executar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte Criar um perfil para delegar permissões a um AWS service (Serviço da AWS) no Guia do Usuário do IAM.

> **Warning**
>
> A alteração das permissões de uma função de serviço pode interromper a funcionalidade do Amazon ECS. Edite funções de serviço somente quando o Amazon ECS fornecer orientação para isso.

Funções vinculadas ao serviço para o Amazon ECS

Suporte a perfis vinculados a serviços: sim

Um perfil vinculado a serviço é um tipo de perfil de serviço vinculado a um AWS service (Serviço da AWS). O serviço pode presumir o perfil de executar uma ação em seu nome. Perfis vinculadas ao serviço aparecem em sua Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não editar as permissões para perfis vinculados a serviço.

Para obter detalhes sobre como criar ou gerenciar funções vinculadas ao serviço do Amazon ECS, consulte Uso de perfis vinculados ao serviço para o Amazon ECS.

Exemplos de políticas baseadas em identidade do Amazon Elastic Container Service

Por padrão, os usuários e perfis não têm permissão para criar ou modificar recursos do Amazon ECS. Eles também não podem executar tarefas usando o AWS Management Console, a AWS Command Line Interface (AWS CLI) ou a API da AWS. Para conceder aos usuários permissões para executar ações nos recursos que eles precisam, um administrador do IAM pode criar políticas do IAM. O administrador pode então adicionar as políticas do IAM aos perfis e os usuários podem assumir os perfis.

Para saber como criar uma política baseada em identidade do IAM usando esses exemplos de documento de política JSON, consulte Criar políticas do IAM (console) no Guia do Usuário do IAM.

Para obter detalhes sobre ações e tipos de recurso definidos pelo Amazon ECS, inclusive o formato dos ARNs para cada um dos tipos de recurso, consulte Ações, recursos e chaves de condição do Amazon Elastic Container Service na Referência de autorização do serviço.

Tópicos

- Práticas recomendadas de políticas do Amazon ECS
- Permissões para que usuários do Amazon ECS visualizem as próprias permissões
- Exemplos de clusters do Amazon ECS
- Exemplos de instância de contêiner do Amazon ECS
- Exemplos de definição de tarefa do Amazon ECS
- Exemplo de tarefa de execução do Amazon ECS
- Exemplo de tarefa de inicialização do Amazon ECS
- Listar e descrever exemplos de tarefas do Amazon ECS
- Exemplo de criação de serviço do Amazon ECS
- Exemplo de serviço de atualização do Amazon ECS
- Descrever serviços do Amazon ECS com base em etiquetas
- Exemplo de negação da substituição de namespace do Service Connect do Amazon ECS

Práticas recomendadas de políticas do Amazon ECS

As políticas baseadas em identidade determinam se alguém pode criar, acessar ou excluir recursos do Amazon ECS em sua conta. Essas ações podem incorrer em custos para sua Conta da AWS. Ao criar ou editar políticas baseadas em identidade, siga estas diretrizes e recomendações:

- Comece com as políticas gerenciadas pela AWS e avance para as permissões de privilégio mínimo: para começar a conceder permissões a seus usuários e workloads, use as políticas gerenciadas pela AWS, que concedem permissões para muitos casos de uso comuns. Elas estão disponíveis em seus Conta da AWS. Recomendamos que você reduza ainda mais as permissões definindo políticas gerenciadas pelo cliente da AWS que são específicas para seus casos de uso. Para obter mais informações, consulte Políticas gerenciadas pela AWS ou Políticas gerenciadas pela AWS para funções de trabalho no Guia do usuário do IAM.

- Aplique permissões de privilégio mínimo: ao definir permissões com as políticas do IAM, conceda apenas as permissões necessárias para executar uma tarefa. Você faz isso definindo as ações que podem ser executadas em recursos específicos sob condições específicas, também conhecidas como permissões de privilégio mínimo. Para obter mais informações sobre como usar o IAM para aplicar permissões, consulte Políticas e permissões no IAM no Guia do usuário do IAM.
- Use condições nas políticas do IAM para restringir ainda mais o acesso: você pode adicionar uma condição às políticas para limitar o acesso a ações e recursos. Por exemplo, você pode escrever uma condição de política para especificar que todas as solicitações devem ser enviadas usando SSL. Também pode usar condições para conceder acesso a ações de serviço, se elas forem usadas por meio de um AWS service (Serviço da AWS) específico, como o AWS CloudFormation. Para obter mais informações, consulte Elementos da política JSON do IAM: condição no Guia do usuário do IAM.
- Use o IAM Access Analyzer para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais: o IAM Access Analyzer valida as políticas novas e existentes para que elas sigam a linguagem de política do IAM (JSON) e as práticas recomendadas do IAM. O IAM Access Analyzer oferece mais de cem verificações de política e recomendações práticas para ajudar a criar políticas seguras e funcionais. Para obter mais informações, consulte Validação de políticas do IAM Access Analyzer no Guia do Usuário do IAM.
- Exigir autenticação multifator (MFA): se houver um cenário que exija usuários do IAM ou um usuário raiz em sua Conta da AWS, ative a MFA para obter segurança adicional. Para exigir MFA quando as operações de API forem chamadas, adicione condições de MFA às suas políticas. Para obter mais informações, consulte Configuração de acesso à API protegido por MFA no Guia do Usuário do IAM.

Para obter mais informações sobre as práticas recomendadas do IAM, consulte Práticas recomendadas de segurança no IAM no Guia do usuário do IAM.

Permissões para que usuários do Amazon ECS visualizem as próprias permissões

Este exemplo mostra como criar uma política que permita que os usuários do IAM visualizem as políticas gerenciadas e em linha anexadas a sua identidade de usuário. Essa política inclui permissões para concluir essa ação no console ou de forma programática usando a AWS CLI ou a API da AWS.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Exemplos de clusters do Amazon ECS

A política do IAM a seguir possibilita que a permissão crie e liste clusters. Como as ações `CreateCluster` e `ListClusters` não aceitam recursos, a definição de recurso é definida como `*` para todos os recursos.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:CreateCluster",
```

```
                "ecs:ListClusters"
            ],
            "Resource": ["*"]
        }
    ]
}
```

A política do IAM a seguir possibilita que a permissão descreva e exclua um cluster específico. As ações `DescribeClusters` e `DeleteCluster` aceitam os ARNs de cluster como recursos.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:DescribeClusters",
                "ecs:DeleteCluster"
            ],
            "Resource": ["arn:aws:ecs:us-east-1:<aws_account_id>:cluster/
<cluster_name>"]
        }
    ]
}
```

A política do IAM a seguir pode ser anexada a um usuário ou grupo que só permitiria que esse usuário ou grupo realizasse operações em um cluster específico.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ecs:Describe*",
                "ecs:List*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "ecs:DeleteCluster",
```

```
                "ecs:DeregisterContainerInstance",
                "ecs:ListContainerInstances",
                "ecs:RegisterContainerInstance",
                "ecs:SubmitContainerStateChange",
                "ecs:SubmitTaskStateChange"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:ecs:us-east-1:<aws_account_id>:cluster/default"
        },
        {
            "Action": [
                "ecs:DescribeContainerInstances",
                "ecs:DescribeTasks",
                "ecs:ListTasks",
                "ecs:UpdateContainerAgent",
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:RunTask"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "ArnEquals": {"ecs:cluster": "arn:aws:ecs:us-
east-1:<aws_account_id>:cluster/default"}
            }
        }
    ]
}
```

Exemplos de instância de contêiner do Amazon ECS

O registro da instância de contêiner é processado pelo agente do Amazon ECS, mas pode haver momentos em que você queira permitir que um usuário cancele o registro de uma instância manualmente em um cluster. Talvez a instância de contêiner tenha sido registrada acidentalmente no cluster errado, ou a instância tenha sido encerrada com tarefas ainda em execução.

A política do IAM a seguir permite que um usuário liste e cancele o registro de instâncias de contêiner em um cluster especificado:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": [
                "ecs:DeregisterContainerInstance",
                "ecs:ListContainerInstances"
            ],
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:cluster/
<cluster_name>"]
        }
    ]
}
```

A política do IAM a seguir permite que um usuário descreva uma instância de contêiner especificada em um determinado cluster. Para abrir essa permissão a todas as instâncias de contêiner em um cluster, você pode substituir a UUID da instância de contêiner por *.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:DescribeContainerInstances"],
            "Condition": {
                "ArnEquals": {"ecs:cluster":
 "arn:aws:ecs:<region>:<aws_account_id>:cluster/<cluster_name>"}
            },
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:container-instance/
<cluster_name>/<container_instance_UUID>"]
        }
    ]
}
```

Exemplos de definição de tarefa do Amazon ECS

As políticas do IAM de definição de tarefa não dão suporte a permissões no nível do recurso, mas a política do IAM a seguir permite que um usuário registre, liste e descreva definições de tarefa.

Se você usar o console, será necessário adicionar `CloudFormation: CreateStack` como uma `Action`.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": [
                "ecs:RegisterTaskDefinition",
                "ecs:ListTaskDefinitions",
                "ecs:DescribeTaskDefinition"
            ],
            "Resource": ["*"]
        }
    ]
}
```

Exemplo de tarefa de execução do Amazon ECS

Os recursos de `RunTask` são definições de tarefa. Para limitar em quais clusters um usuário pode executar definições de tarefa, você pode especificá-los no bloco `Condition`. A vantagem é que você não precisa listar definições de tarefa e clusters nos recursos para permitir o acesso apropriado. É possível aplicar um, o outro ou ambos.

A política do IAM a seguir permite a execução de qualquer revisão de uma definição de tarefa específica em um cluster específico:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:RunTask"],
            "Condition": {
                "ArnEquals": {"ecs:cluster":
 "arn:aws:ecs:<region>:<aws_account_id>:cluster/<cluster_name>"}
            },
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:task-definition/
<task_family>:*"]
        }
    ]
}
```

Exemplo de tarefa de inicialização do Amazon ECS

Os recursos de `StartTask` são definições de tarefa. Para limitar em quais clusters e instâncias de contêiner um usuário pode iniciar definições de tarefa, você pode especificá-los no bloco

`Condition`. A vantagem é que você não precisa listar definições de tarefa e clusters nos recursos para permitir o acesso apropriado. É possível aplicar um, o outro ou ambos.

A política do IAM a seguir permite o início de qualquer revisão de uma definição de tarefa específica em um cluster específico e uma instância de contêiner específica.

> **Note**
>
> Para este exemplo, quando você chama a API `StartTask` com a AWS CLI ou outro AWS SDK, você deve especificar a revisão de definição de tarefa para que o mapeamento de `Resource` seja correspondente.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:StartTask"],
            "Condition": {
                "ArnEquals": {
                    "ecs:cluster": "arn:aws:ecs:<region>:<aws_account_id>:cluster/
<cluster_name>",
                    "ecs:container-instances":
 ["arn:aws:ecs:<region>:<aws_account_id>:container-instance/<cluster_name>/
<container_instance_UUID>"]
                }
            },
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:task-definition/
<task_family>:*"]
        }
    ]
}
```

Listar e descrever exemplos de tarefas do Amazon ECS

A política do IAM a seguir permite que um usuário liste tarefas para um cluster especificado:

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": ["ecs:ListTasks"],
            "Condition": {
                "ArnEquals": {"ecs:cluster":
 "arn:aws:ecs:<region>:<aws_account_id>:cluster/<cluster_name>"}
            },
            "Resource": ["*"]
        }
    ]
}
```

A política do IAM a seguir permite que um usuário descreva uma tarefa especificada em um cluster especificado:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:DescribeTasks"],
            "Condition": {
                "ArnEquals": {"ecs:cluster":
 "arn:aws:ecs:<region>:<aws_account_id>:cluster/<cluster_name>"}
            },
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:task/<cluster_name>/
<task_UUID>"]
        }
    ]
}
```

Exemplo de criação de serviço do Amazon ECS

A política do IAM a seguir permite que um usuário crie serviços do Amazon ECS no AWS Management Console:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "application-autoscaling:Describe*",
                "application-autoscaling:PutScalingPolicy",
                "application-autoscaling:RegisterScalableTarget",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "ecs:List*",
                "ecs:Describe*",
                "ecs:CreateService",
                "elasticloadbalancing:Describe*",
                "iam:GetPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:ListAttachedRolePolicies",
                "iam:ListRoles",
                "iam:ListGroups",
                "iam:ListUsers"
            ],
            "Resource": ["*"]
        }
    ]
}
```

Exemplo de serviço de atualização do Amazon ECS

A política do IAM a seguir permite que um usuário atualize serviços do Amazon ECS no AWS Management Console:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:Describe*",
                "application-autoscaling:PutScalingPolicy",
                "application-autoscaling:DeleteScalingPolicy",
                "application-autoscaling:RegisterScalableTarget",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "ecs:List*",
                "ecs:Describe*",
                "ecs:UpdateService",
                "iam:GetPolicy",
```

```
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:ListAttachedRolePolicies",
                "iam:ListRoles",
                "iam:ListGroups",
                "iam:ListUsers"
            ],
            "Resource": ["*"]
        }
    ]
}
```

Descrever serviços do Amazon ECS com base em etiquetas

É possível usar condições em sua política baseada em identidade para controlar o acesso aos recursos do Amazon ECS com base em etiquetas. Este exemplo mostra como você pode criar uma política que permite descrever seus serviços. No entanto, a permissão será concedida somente se a tag `Owner` tiver o valor do nome desse usuário. Essa política também concede as permissões necessárias concluir essa ação no console.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DescribeServices",
            "Effect": "Allow",
            "Action": "ecs;:DescribeServices",
            "Resource": "*"
        },
        {
            "Sid": "ViewServiceIfOwner",
            "Effect": "Allow",
            "Action": "ecs:DescribeServices",
            "Resource": "arn:aws:ecs:*:*:service/*",
            "Condition": {
                "StringEquals": {"ecs:ResourceTag/Owner": "${aws:username}"}
            }
        }
    ]
```

```
}
```

É possível anexar essa política aos usuários do IAM na sua conta. Se um usuário denominado `richard-roe` tentar descrever um serviço do Amazon ECS, o serviço deverá ser marcado `0wner=richard-roe` ou `owner=richard-roe`. Caso contrário, ele terá o acesso negado. A chave da tag de condição `0wner` corresponde a `0wner` e a `owner` porque os nomes das chaves de condição não fazem distinção entre maiúsculas e minúsculas. Para obter mais informações, consulte IAM JSON Policy Elements: Condition (Elementos da política JSON do IAM: Condição) no Guia do usuário do IAM.

Exemplo de negação da substituição de namespace do Service Connect do Amazon ECS

A política do IAM a seguir nega que um usuário substitua o namespace padrão do Service Connect em uma configuração de serviço. O namespace padrão é definido no cluster. Entretanto, é possível substituí-lo em uma configuração de serviço. Para manter a consistência, você pode definir todos os seus novos serviços para usar o mesmo namespace. Use as seguintes chaves de contexto para exigir que os serviços usem um namespace específico. Substitua `<region>`, `<aws_account_id>`, `<cluster_name>` e `<namespace_id>` no seguinte exemplo pelos seus próprios.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:CreateService",
                "ecs:UpdateService"
            ],
            "Condition": {
                "ArnEquals": {
                    "ecs:cluster": "arn:aws:ecs:<region>:<aws_account_id>:cluster/
<cluster_name>",
                    "ecs:namespace":
 "arn:aws:servicediscovery:<region>:<aws_account_id>:namespace/<namespace_id>"
                }
            },
            "Resource": "*"
        }
    ]
```

```
}
```

Políticas gerenciadas pela AWS para o Amazon Elastic Container Service

Para adicionar permissões a usuários, grupos e perfis, é mais fácil usar políticas gerenciadas pela AWS do que gravar políticas por conta própria. É necessário tempo e experiência para criar políticas gerenciadas pelo cliente do IAM que fornecem à sua equipe apenas as permissões de que precisam. Para começar rapidamente, é possível usar nossas políticas gerenciadas pela AWS. Essas políticas abrangem casos de uso comuns e estão disponíveis na sua conta da AWS. Para obter mais informações sobre políticas gerenciadas pela AWS, consulte Políticas gerenciadas pela AWS no Guia do usuário do IAM.

Os serviços da AWS mantêm e atualizam políticas gerenciadas pela AWS. Não é possível alterar as permissões em políticas gerenciadas pela AWS. Os serviços ocasionalmente acrescentam permissões adicionais a uma política gerenciada pela AWS para oferecer suporte a novos recursos. Esse tipo de atualização afeta todas as identidades (usuários, grupos e funções) em que a política está anexada. É mais provável que os serviços atualizem uma política gerenciada pela AWS quando um novo recurso for iniciado ou novas operações se tornarem disponíveis. Os serviços não removem permissões de uma política gerenciada pela AWS, portanto, as atualizações de políticas não suspendem suas permissões existentes.

Além disso, a AWS é compatível com políticas gerenciadas por perfis de trabalho que abrangem vários serviços. Por exemplo, a política ReadOnlyAccess política gerenciada pela AWS fornece acesso somente leitura a todos os serviços e recursos da AWS. Quando um serviço executa um novo atributo, a AWS adiciona permissões somente leitura para novas operações e recursos. Para obter uma lista e descrições das políticas de perfis de trabalho, consulte Políticas gerenciadas pela AWS para perfis de trabalho no Guia do usuário do IAM.

O Amazon ECS e o Amazon ECR fornecem várias políticas gerenciadas e relacionamentos de confiança que você pode anexar aos usuários, aos grupos, aos perfis, às instâncias do Amazon EC2 e às tarefas do Amazon ECS que permitem diferentes níveis de controle sobre recursos e operações de API. É possível aplicar essas políticas diretamente ou usá-las como ponto de partida para criar suas próprias políticas. Para obter mais informações sobre políticas gerenciadas pelo Amazon ECR, consulte Políticas gerenciadas pelo Amazon ECR.

AmazonECS_FullAccess

É possível anexar a política `AmazonECS_FullAccess` às identidades do IAM. Essa política concede acesso administrativo a recursos do Amazon ECS e concede a uma identidade do IAM (como usuário, grupo ou função) acesso aos serviços da AWS aos quais o Amazon ECS está integrado para usar todos os recursos do Amazon ECS. O uso dessa política permite o acesso a todos os recursos do Amazon ECS que estão disponíveis no AWS Management Console.

Para visualizar as permissões dessa política, consulte AmazonECS_FullAccess na Referência de políticas gerenciadas da AWS.

AmazonECSInfrastructureRolePolicyForVolumes

É possível anexar a política gerenciada `AmazonECSInfrastructureRolePolicyForVolumes` às suas entidades do IAM.

A política concede as permissões necessárias para que o Amazon ECS faça chamadas de API da AWS em seu nome. Você pode anexar essa política ao perfil do IAM que você fornece com a configuração do volume ao iniciar tarefas e serviços do Amazon ECS. O perfil permite que o Amazon ECS gerencie volumes anexados às suas tarefas. Para obter mais informações, consulte Amazon ECS infrastructure IAM role.

Para visualizar as permissões dessa política, consulte AmazonECSInfrastructureRolePolicyForVolumes na Referência de políticas gerenciadas pela AWS.

AmazonEC2ContainerServiceforEC2Role

É possível anexar a política `AmazonEC2ContainerServiceforEC2Role` às identidades do IAM. Essa política concede permissões administrativas que permitem que as instâncias de contêineres do Amazon ECS façam chamadas para a AWS em seu nome. Para obter mais informações, consulte Função do IAM de instância de contêiner do Amazon ECS.

O Amazon ECS anexa essa política a uma função de serviço que permite que o Amazon ECS execute ações em seu nome contra instâncias do Amazon EC2 ou instâncias externas.

Para visualizar as permissões dessa política, consulte AmazonEC2ContainerServiceforEC2Role na Referência de políticas gerenciadas pela AWS.

Considerações

Você deve considerar as seguintes recomendações e considerações quando usar a política do IAM gerenciada `AmazonEC2ContainerServiceforEC2Role`.

- Ao seguir o aviso de segurança padrão de concessão de privilégio mínimo, você pode modificar a política gerenciada `AmazonEC2ContainerServiceforEC2Role` para atender às suas necessidades específicas. Se qualquer permissão concedida na política gerenciada não for necessária para seu caso de uso, crie uma política personalizada e adicione apenas as permissões necessárias. Por exemplo, a permissão `UpdateContainerInstancesState` é fornecida para drenagem de instâncias spot. Se essa permissão não for necessária para o seu caso de uso, exclua-a usando uma política personalizada.
- Os contêineres em execução nas instâncias de contêiner têm acesso a todas as permissões dadas à função da instância de contêiner por meio de instance metadata. Recomendamos que você limite as permissões na função de instância de contêiner à lista mínima de permissões fornecidas na política gerenciada `AmazonEC2ContainerServiceforEC2Role` mostrada. Caso os contêineres das tarefas precisem de permissões adicionais não listadas aqui, recomendamos que estas tarefas sejam fornecidas com as próprias funções do IAM. Para obter mais informações, consulte Perfil do IAM para tarefas do Amazon ECS.

  É possível impedir que contêineres na ponte `docker0` acessem as permissões fornecidas para o perfil de instância de contêiner. Isso pode ser feito enquanto ainda concede as permissões fornecidas pelo Perfil do IAM para tarefas do Amazon ECS ao executar o seguinte comando iptables nas instâncias de contêiner. Os contêineres não podem consultar metadados de instância com essa regra em vigor. Esse comando pressupõe a configuração de ponte do Docker padrão e não funciona para contêineres que usam o modo de rede `host`. Para obter mais informações, consulte Modo de rede.

  ```
  sudo yum install -y iptables-services; sudo iptables --insert DOCKER USER 1 --in-interface docker+ --destination 169.254.169.254/32 --jump DROP
  ```

  Você deve salvar essa regra iptables em sua instância de contêiner para ela sobreviver uma reinicialização. Para a AMI otimizada para Amazon ECS, use o seguinte comando. Para outros sistemas operacionais, consulte a documentação do SO em questão.

  - Para a AMI do Amazon Linux 2 otimizada para o Amazon ECS:

    ```
    sudo iptables-save | sudo tee /etc/sysconfig/iptables && sudo systemctl enable --now iptables
    ```

  - Para a AMI do Amazon Linux otimizada para o Amazon ECS:

    ```
    sudo service iptables save
    ```

AmazonEC2ContainerServiceEventsRole

É possível anexar a política `AmazonEC2ContainerServiceEventsRole` às identidades do IAM. Essa política concede permissões que possibilitam que o Amazon EventBridge (anteriormente conhecido como CloudWatch Events) execute tarefas em seu nome. Essa política pode ser anexada à função do IAM especificada quando você cria tarefas programadas. Para obter mais informações, consulte Perfil do IAM para EventBridge do Amazon ECS.

Para visualizar as permissões dessa política, consulte AmazonEC2ContainerServiceEventsRole na Referência de políticas gerenciadas pela AWS.

AmazonECSTaskExecutionRolePolicy

A política do IAM `AmazonECSTaskExecutionRolePolicy` gerenciada concede as permissões necessárias para que o agente de contêiner do Amazon ECS e os agentes de contêiner do AWS Fargate façam chamadas de API da AWS em seu nome. Essa política pode ser adicionada à função do IAM de execução de tarefas. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.

Para visualizar as permissões dessa política, consulte AmazonECSTaskExecutionRolePolicy na Referência de políticas gerenciadas pela AWS.

AmazonECSServiceRolePolicy

A política do IAM gerenciada `AmazonECSServiceRolePolicy` permite que o Amazon Elastic Container Service gerencie seu cluster. Essa política pode ser adicionada ao seu perfil vinculado ao serviço AWSServiceRoleForECS.

Para visualizar as permissões dessa política, consulte AmazonECSServiceRolePolicy na Referência de políticas gerenciadas pela AWS.

`AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecur`

Também é possível anexar a política `AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity` às suas entidades do IAM. Essa política fornece acesso administrativo à AWS Private Certificate Authority, ao Secrets Manager e a outros serviços da AWS necessários para gerenciar os recursos de TLS do Amazon ECS Service Connect em seu nome.

Para visualizar as permissões dessa política, consulte AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity na Referência de políticas gerenciadas pela AWS.

`AWSApplicationAutoscalingECSServicePolicy`

Não é possível anexar `AWSApplicationAutoscalingECSServicePolicy` às entidades do IAM. Essa política é anexada a uma função vinculada ao serviço que permite que o Application Auto Scaling execute ações em seu nome. Para obter mais informações, consulte Funções vinculadas ao serviço para o Application Auto Scaling.

Para exibir as permissões dessa política, consulte AWSApplicationAutoscalingECSServicePolicy na Referência de política gerenciada da AWS.

`AWSCodeDeployRoleForECS`

Não é possível anexar `AWSCodeDeployRoleForECS` às entidades do IAM. Essa política é anexada a uma função vinculada ao serviço que permite que o CodeDeploy execute ações em seu nome. Para obter mais informações, consulte Criar uma função de serviço para o CodeDeploy no Guia do usuário do AWS CodeDeploy.

Para visualizar as permissões dessa política, consulte AWSCodeDeployRoleForECS na Referência de políticas gerenciadas pela AWS.

`AWSCodeDeployRoleForECSLimited`

Não é possível anexar `AWSCodeDeployRoleForECSLimited` às entidades do IAM. Essa política é anexada a uma função vinculada ao serviço que permite que o CodeDeploy execute ações em seu nome. Para obter mais informações, consulte Criar uma função de serviço para o CodeDeploy no Guia do usuário do AWS CodeDeploy.

Para visualizar as permissões dessa política, consulte AWSCodeDeployRoleForECSLimited na Referência de políticas gerenciadas pela AWS.

`AmazonECSInfrastructureRolePolicyForLoadBalancers`

Também é possível anexar a política `AmazonECSInfrastructureRolePolicyForLoadBalancers` às suas entidades do IAM. Esta política concede permissões que possibilitam que o Amazon ECS gerencie recursos do Elastic Load Balancing em seu nome. A política inclui:

- Permissões somente leitura para descrever receptores, regras, grupos de destino e integridade do destino
- Permissões para registrar e cancelar o registro de destinos com grupos de destino
- Permissões para modificar receptores de Application Load Balancers e Network Load Balancers
- Permissões para modificar regras de Application Load Balancers

Essas permissões permitem que o Amazon ECS gerencie automaticamente as configurações do balanceador de carga quando os serviços são criados ou atualizados, garantindo o roteamento adequado do tráfego para seus contêineres.

Para visualizar as permissões dessa política, consulte AmazonECSInfrastructureRolePolicyForLoadBalancers na Referência de políticas gerenciadas pela AWS.

`AmazonECSInfrastructureRolePolicyForVpcLattice`

Também é possível anexar a política `AmazonECSInfrastructureRolePolicyForVpcLattice` às suas entidades do IAM. Essa política fornece acesso a outros recursos do serviço da AWS necessários para gerenciar volumes associados a workloads do Amazon ECS em seu nome.

Para visualizar as permissões dessa política, consulte AmazonECSInfrastructureRolePolicyForVpcLattice na Referência de políticas gerenciadas pela AWS.

Fornece acesso a outros recursos do serviço da AWS necessários para gerenciar volumes associados a workloads do Amazon ECS em seu nome.

Atualizações do Amazon ECS para políticas gerenciadas pela AWS

Visualize detalhes sobre atualizações de políticas gerenciadas pela AWS para o Amazon ECS desde que este serviço passou a rastrear essas mudanças. Para obter alertas automáticos sobre alterações feitas nesta página, inscreva-se no feed RSS na página Histórico de documentos do Amazon ECS.

Alteração	Descrição	Data
Adicionar permissão a the section called “AmazonEC	A política `AmazonEC2 ContainerServicefo rEC2Role` gerenciada do	4 de agosto de 2025

Alteração	Descrição	Data
2ContainerServiceforEC2Role ” -	IAM foi atualizada para incluir a permissão `ecs:ListTagsForResource` . Essa permissão permite que o agente do Amazon ECS recupere tags de instâncias de tarefas e contêineres por meio do endpoint de metadados da tarefa (`${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`).	
Adicione permissões à política AmazonECSInfrastructureRolePolicyForLoadBalancers.	A política `AmazonECSInfrastructureRolePolicyForLoadBalancers` gerenciada do IAM foi atualizada com novas permissões para descrever, cancelar o registro e registrar grupos de destino.	25 de julho de 2025
Adicionar a nova política the section called “`AmazonECSInfrastructureRolePolicyForLoadBalancers` ”	Foi adicionada a nova política AmazonECSInfrastructureRolePolicyForLoadBalancers que fornece acesso a outros recursos dos serviços da AWS necessários para gerenciar balanceadores de carga associados às workloads do Amazon ECS.	15 de julho de 2025

Alteração	Descrição	Data
Adicione permissões à política AmazonECSServiceRolePolicy.	A política gerenciada do IAM `AmazonECSServiceRolePolicy` foi atualizada com novas permissões AWS Cloud Map, permitindo que o Amazon ECS atualize atributos de serviço AWS Cloud Map para serviços gerenciados pelo Amazon ECS.	15 de julho de 2025
Adicionar permissões ao AmazonECSServiceRolePolicy	A política gerenciada do IAM `AmazonECSServiceRolePolicy` foi atualizada com novas permissões AWS Cloud Map, permitindo que o Amazon ECS atualize atributos de serviço AWS Cloud Map para serviços gerenciados pelo Amazon ECS.	24 de junho de 2025
Adicionar permissões a AmazonECSInfrastructureRolePolicyForVolumes	A política `AmazonECSInfrastructureRolePolicyForVolumes` foi atualizada para adicionar a permissão `ec2:DescribeInstances`. Essa permissão ajuda a impedir a colisão de nome de dispositivo para volumes do Amazon EBS anexados a tarefas do Amazon ECS em execução na mesma instância de contêiner.	2 de junho de 2025

Alteração	Descrição	Data
Adicionar nova AmazonECS InfrastructureRolePolicyFor VpcLattice	Fornece acesso a outros recursos do serviço da AWS necessários para gerenciar volumes associados a workloads do Amazon ECS em seu nome.	18 de novembro de 2024
Adicionar permissões a AmazonECSInfrastructureRole PolicyForVolumes	A política `AmazonECS InfrastructureRole PolicyForVolumes` foi atualizada para permitir que os clientes criem um volume do Amazon EBS com base em um snapshot.	10 de outubro de 2024
Adicionou permissões a the section called “AmazonEC S_FullAccess”	A política `AmazonECS _FullAccess` foi atualizad a para adicionar permissõe s `iam:PassRole` para perfis do IAM para um perfil chamado `ecsInfras tructureRole` . Este é o perfil padrão do IAM criado pelo AWS Managemen t Console, destinado a ser usado como um perfil de infraestrutura do ECS que permite ao Amazon ECS gerenciar volumes do Amazon EBS anexados às tarefas do ECS.	13 de agosto de 2024

Alteração	Descrição	Data
Adicionar nova política AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity	Foi adicionada uma nova política AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity que fornece acesso administrativo ao AWS KMS, à AWS Private Certificate Authority e ao Secrets Manager e permite que os recursos de TLS do Amazon ECS Service Connect funcionem adequadamente.	22 de janeiro de 2024
Adicionar nova política AmazonECSInfrastructureRolePolicyForVolumes	A política `AmazonECSInfrastructureRolePolicyForVolumes` foi adicionada. A política concede as permissões necessárias ao Amazon ECS para fazer chamadas de API da AWS a fim de gerenciar volumes do Amazon EBS associados às workloads do Amazon ECS.	11 de janeiro de 2024
Adicionar permissões ao AmazonECSServiceRolePolicy	A política do IAM gerenciada `AmazonECSServiceRolePolicy` foi atualizada com novas permissões de `events` e permissões adicionais de `autoscaling` e `autoscaling-plans` .	4 de dezembro de 2023

Alteração	Descrição	Data
Adicionar permissões a AmazonEC2ContainerServiceEventsRole	A política do IAM gerenciada `AmazonECSServiceRolePolicy` foi atualizada para permitir o acesso à API `DiscoverInstancesRevision` do AWS Cloud Map.	4 de outubro de 2023
Adicione permissões ao AmazonEC2ContainerServiceforEC2Role	A política `AmazonEC2ContainerServiceforEC2Role` foi modificada para adicionar a permissão `ecs:TagResource`, que inclui uma condição que limita a permissão somente aos clusters recém-criados e às instâncias de contêiner registradas.	6 de março de 2023
Adicionar permissão a the section called “AmazonECS_FullAccess”	A política `AmazonECS_FullAccess` foi modificada para adicionar a permissão `elasticloadbalancing:AddTags`, que inclui uma condição que limita a permissão somente aos balanceadores de carga recém-criados, aos grupos de destino, às regras e aos receptores criados. Essa permissão não permite que sejam adicionadas tags a qualquer recurso já criado do Elastic Load Balancing.	4 de janeiro de 2023

Alteração	Descrição	Data
O Amazon ECS passou a rastrear as alterações	O Amazon ECS passou a rastrear as alterações para as políticas gerenciadas da AWS.	8 de junho de 2021

Políticas do IAM gerenciadas pela AWS desativadas para o Amazon Elastic Container Service

Os seguintes exemplos de políticas do IAM gerenciadas pela AWS estão desativadas. Essas políticas agora estão substituídas pelas políticas atualizadas. Recomendamos que você atualize seus usuários ou perfis para usar as políticas atualizadas.

AmazonEC2ContainerServiceFullAccess

> **Important**
>
> A política do IAM gerenciada `AmazonEC2ContainerServiceFullAccess` foi desativada a partir de 29 de janeiro de 2021, em resposta a uma descoberta relativa à segurança com a permissão `iam:passRole`. Essa permissão concede acesso a todos os recursos, incluindo credenciais para funções da conta. Agora que a política está desativada, você não pode anexá-la a quaisquer novos usuários ou perfis. Quaisquer usuários ou funções que já tenham a política anexada poderão continuar a usá-la. Contudo, recomendamos que você atualize seus usuários ou perfis para usar a política gerenciada `AmazonECS_FullAccess`. Para obter mais informações, consulte Migrar para a política gerenciada `AmazonECS_FullAccess`.

AmazonEC2ContainerServiceRole

> **Important**
>
> A política do IAM gerenciada `AmazonEC2ContainerServiceRole` está desativada. Ela foi substituída pela função vinculada ao serviço do Amazon ECS. Para obter mais informações, consulte Uso de perfis vinculados ao serviço para o Amazon ECS.

AmazonEC2ContainerServiceAutoscaleRole

> **Important**
>
> A política do IAM gerenciada `AmazonEC2ContainerServiceAutoscaleRole` está desativada. Ela foi substituída pela função vinculada ao serviço do Application Auto Scaling para o Amazon ECS. Para ter mais informações, consulte Funções vinculadas a serviço do Application Auto Scaling, no Guia do usuário do Application Auto Scaling.

Migrar para a política gerenciada **`AmazonECS_FullAccess`**

A política do IAM gerenciada `AmazonEC2ContainerServiceFullAccess` foi desativada em 29 de janeiro de 2021, em resposta a uma descoberta relativa à segurança com a permissão `iam:passRole`. Essa permissão concede acesso a todos os recursos, incluindo credenciais para funções da conta. Agora que a política está desativada, você não pode anexá-la a quaisquer novos grupos, usuários ou perfis. Quaisquer grupos, usuários ou funções que já tenham a política anexada poderão continuar a usá-la. Contudo, recomendamos que você atualize seus grupos, usuários ou perfis para usar a política gerenciada `AmazonECS_FullAccess`.

As permissões que são concedidas pela política `AmazonECS_FullAccess` incluem a lista completa de permissões necessárias para usar o ECS como administrador. Se você atualmente usa permissões concedidas pela política `AmazonEC2ContainerServiceFullAccess` que não estão na política `AmazonECS_FullAccess`, pode adicioná-las a uma declaração de política em linha. Para obter mais informações, consulte Políticas gerenciadas pela AWS para o Amazon Elastic Container Service.

Use as etapas a seguir para determinar se você tem grupos, usuários ou perfis que estão usando, no momento, a política gerenciada do IAM `AmazonEC2ContainerServiceFullAccess`. Em seguida, atualize-os para desvincular a política anterior e anexar a política `AmazonECS_FullAccess`.

Para atualizar um grupo, usuário ou perfil para usar a política AmazonECS_FullAccess policy (AWS Management Console)

1. Abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Policies (Políticas) e pesquise e selecione a política `AmazonEC2ContainerServiceFullAccess`.

3. Escolha a guia Policy usage (Uso da política), que exibe qualquer função do IAM que está usando essa política, no momento.
4. Para cada perfil do IAM que está usando a política `AmazonEC2ContainerServiceFullAccess` no momento, selecione o perfil e use as etapas a seguir para desanexar a política desativada e anexar a política `AmazonECS_FullAccess`.
   a. Na guia Permissions (Permissões), escolha o X ao lado da política AmazonEC2ContainerServiceFullAccess.
   b. Escolha Adicionar permissões.
   c. Escolha Attach existing policies directly (Anexar políticas existentes diretamente), pesquise e selecione a política AmazonECS_FullAccess e escolha Next: Review (Próximo: Análise).
   d. Analise as alterações e escolha Add permissions (Adicionar permissões).
   e. Repita essas etapas para cada grupo, usuário ou perfil que esteja usando a política `AmazonEC2ContainerServiceFullAccess`.

Para atualizar um grupo, usuário ou perfil para usar a política **`AmazonECS_FullAccess`** (AWS CLI)

1. Use o comando generate-service-last-accessed-details para gerar um relatório que inclua detalhes sobre quando a política desativada foi usada pela última vez.

```
aws iam generate-service-last-accessed-details \
     --arn arn:aws:iam::aws:policy/AmazonEC2ContainerServiceFullAccess
```

   Resultado do exemplo:

```
{
    "JobId": "32bb1fb0-1ee0-b08e-3626-ae83EXAMPLE"
}
```

2. Use o ID do trabalho da saída anterior com o comando get-service-last-accessed-details para recuperar o último relatório acessado do serviço. Esse relatório exibe o nome do recurso da Amazon (ARN) das entidades do IAM que usaram a política desativada pela última vez.

```
aws iam get-service-last-accessed-details \
      --job-id 32bb1fb0-1ee0-b08e-3626-ae83EXAMPLE
```

3. Use um dos seguintes comandos para desvincular a política `AmazonEC2ContainerServiceFullAccess` de um grupo, usuário ou perfil.

   - detach-group-policy
   - detach-role-policy
   - detach-user-policy

4. Use um dos comandos a seguir para anexar a política `AmazonECS_FullAccess` a um grupo, usuário ou perfil.

   - attach-group-policy
   - attach-role-policy
   - attach-user-policy

Uso de perfis vinculados ao serviço para o Amazon ECS

O Amazon Elastic Container Service usa perfis vinculados ao serviço do AWS Identity and Access Management (IAM). O perfil vinculado ao serviço é um tipo especial de perfil do IAM vinculado diretamente ao Amazon ECS. O perfil vinculado ao serviço é predefinido pelo Amazon ECS Service e inclui todas as permissões que o serviço requer para chamar outros produtos da AWS em seu nome.

Um perfil vinculado ao serviço facilita a configuração do Amazon ECS porque você não precisa adicionar as permissões necessárias manualmente. O Amazon ECS define as permissões dos perfis vinculados ao serviço e, a não ser que esteja definido de outra forma, somente o Amazon ECS poderá assumir os perfis. As permissões definidas incluem a política de confiança e a política de permissões, que não pode ser anexada a nenhuma outra entidade do IAM.

Para obter informações sobre outros produtos que oferecem suporte às funções vinculadas a serviços, consulte Serviços da AWS que funcionam com o IAM e procure os serviços que apresentam Yes (Sim) na coluna Funções vinculadas ao serviço. Escolha um Sim com um link para visualizar a documentação do perfil vinculado a esse serviço.

Permissões de perfil vinculado ao serviço para o Amazon ECS

O Amazon ECS usa o perfil vinculado ao serviço chamado AWSServiceRoleForECS.

O perfil vinculado ao serviço AWSServiceRoleForECS confia nos seguintes serviços para assumir o perfil:

- `ecs.amazonaws.com`

A política de permissões do perfil chamada AmazonECSServiceRolePolicy permite que o Amazon ECS realize as seguintes ações nos recursos especificados:

- Ação: ao usar o modo de rede `awsvpc` em suas tarefas do Amazon ECS, o Amazon ECS gerenciará o ciclo de vida das interfaces de rede elásticas associadas à tarefa. Isso também inclui etiquetas que o Amazon ECS adiciona às interfaces de rede elásticas.
- Ação: ao usar um balanceador de carga com o serviço do Amazon ECS, o Amazon ECS gerenciará o registro e o cancelamento do registro de recursos com o balanceador de carga.
- Ação: ao usar a descoberta de serviços do Amazon ECS, o Amazon ECS gerenciará os recursos do Route 53 e do AWS Cloud Map necessários para que a descoberta de serviços funcione.
- Ação: ao usar o serviço de escalabilidade automática do Amazon ECS, o Amazon ECS gerenciará os recursos do Auto Scaling necessários.
- Ação: o Amazon ECS cria e gerencia alarmes e fluxos de logs do CloudWatch que auxiliam no monitoramento de recursos do Amazon ECS.
- Ação: ao usar o Amazon ECS Exec, o Amazon ECS gerenciará as permissões necessárias para iniciar sessões do Amazon ECS Exec para as tarefas.
- Ação: ao usar o Amazon ECS Service Connect, o Amazon ECS gerencia os recursos necessários do AWS Cloud Map para usar o recurso.
- Ação: ao usar provedores de capacidade do Amazon ECS, o Amazon ECS gerenciará as permissões necessárias para modificar o grupo do Auto Scaling e suas instâncias do Amazon EC2.
- Ação: o Amazon ECS pode atualizar atributos de serviço AWS Cloud Map para serviços gerenciados pelo Amazon ECS.

Você deve configurar permissões para que uma entidade do IAM (por exemplo, um usuário, grupo ou função) crie, edite ou exclua um perfil vinculado a serviço. Para obter mais informações, consulte Permissões de perfil vinculado ao serviço no Guia do usuário do IAM.

Criar um perfil vinculado ao serviço para o Amazon ECS

Na maioria dos casos, você não precisa criar manualmente um perfil vinculado ao serviço. Quando você cria um cluster ou cria ou atualiza um serviço no AWS Management Console, na AWS CLI ou

na API da AWS, o Amazon ECS cria o perfil vinculado ao serviço para você. Se você não vê o perfil AWSServiceRoleForECS depois de criar um cluster, faça o seguinte para corrigir o problema:

- Verifique e configure as permissões para que o Amazon ECS crie, edite ou exclua um perfil vinculado ao serviço em seu nome. Para obter mais informações, consulte Service-linked role permissions (Permissões de nível vinculado a serviços) no Guia do usuário do IAM.
- Tente novamente a operação de criação do cluster ou crie manualmente o perfil vinculado ao serviço.

  É possível usar o console do IAM para criar o perfil vinculado ao serviço AWSServiceRoleForECS. Na AWS CLI ou na API do AWS, crie um perfil vinculado a serviço com o nome de serviço `ecs.amazonaws.com`. Para obter mais informações, consulte Criar uma função vinculada ao serviço no Manual do usuário do IAM.

> **Important**
>
> Esse perfil vinculado ao serviço pode aparecer em sua conta se você concluiu uma ação em outro serviço que usa os atributos compatíveis com esse perfil.

Se você excluir esse perfil vinculado ao serviço e precisar criá-lo novamente, poderá usar esse mesmo processo para recriar o perfil em sua conta. Quando você cria um cluster ou cria ou atualiza um serviço, o Amazon ECS cria o perfil vinculado ao serviço para você novamente.

Se você excluir esse perfil vinculado ao serviço, será possível usar esse mesmo processo do IAM para criar o perfil novamente.

Editar um perfil vinculado ao serviço do Amazon ECS

O Amazon ECS não permite editar o perfil vinculado ao serviço AWSServiceRoleForECS. Depois que criar um perfil vinculado ao serviço, você não poderá alterar o nome do perfil, pois várias entidades podem fazer referência a ele. No entanto, será possível editar a descrição do perfil usando o IAM. Para obter mais informações, consulte Atualizar uma função vinculada a serviço no Guia do usuário do IAM.

Excluir um perfil vinculado ao serviço do Amazon ECS

Se você não precisar mais usar um recurso ou serviço que requer um perfil vinculado ao serviço, é recomendável excluí-lo. Dessa forma, você não tem uma entidade não utilizada que não seja

monitorada ativamente ou mantida. No entanto, você deve limpar os recursos de seu perfil vinculado ao serviço antes de excluí-lo manualmente.

> **Note**
>
> Se o serviço do Amazon ECS estiver usando o perfil quando você tentar excluir os recursos, a exclusão poderá falhar. Se isso acontecer, espere alguns minutos e tente a operação novamente.

Para verificar se a função vinculada ao serviço tem uma sessão ativa

1. Abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Roles (Funções) e escolha o nome AWSServiceRoleForECS (e não a caixa de seleção).
3. Na página Resumo, escolha Consultor de Acesso e analise as atividades recentes para a função vinculada ao serviço.

   > **Note**
   >
   > Se você não tiver certeza se o Amazon ECS está usando a função AWSServiceRoleForECS, poderá tentar excluir a função. Se o serviço está usando a função, a exclusão falha e você pode visualizar as regiões em que a função está sendo usada. Se a função está sendo usada, você deve aguardar a sessão final antes de excluir a função. Não é possível revogar a sessão de uma função vinculada a um serviço.

Para remover os recursos do Amazon ECS usados pela função vinculada ao serviço AWSServiceRoleForECS

Você deve excluir todos os clusters do Amazon ECS em todas as regiões da AWS antes de poder excluir a função AWSServiceRoleForECS.

1. Dimensione todos os serviços do Amazon ECS até a contagem desejada de 0 em todas as regiões e, em seguida, exclua os serviços. Para obter mais informações, consulte Atualizar um serviço do Amazon ECS e Exclusão de um serviço do Amazon ECS usando o console.

2. Cancele à força o registro de todas as instâncias de contêiner de todos os clusters em todas as regiões. Para obter mais informações, consulte Cancelamento do registro de uma instância de contêiner do Amazon ECS.
3. Exclua todos os clusters do Amazon ECS em todas as regiões. Para obter mais informações, consulte Exclusão de um cluster do Amazon ECS.

Como excluir manualmente o perfil vinculado a serviço usando o IAM

Use o console do IAM, a AWS CLI ou a API da AWS para excluir o perfil vinculado ao serviço AWSServiceRoleForECS. Para obter mais informações, consulte Excluir um perfil vinculado ao serviço no Guia do usuário do IAM.

Regiões com suporte para os perfis vinculados ao serviço do Amazon ECS

O Amazon ECS é compatível com perfis vinculados ao serviço em todas as regiões em que o serviço está disponível. Para obter mais informações, consulte Regiões e endpoints da AWS.

Perfis do IAM para o Amazon ECS

Um perfil do IAM é uma identidade do IAM que você pode criar em sua conta que tem permissões específicas. No Amazon ECS, você pode criar perfis para conceder permissões aos recursos do Amazon ECS, como contêineres ou serviços.

Os perfis exigidos pelo Amazon ECS dependem do tipo de execução da definição da tarefa e dos recursos usados. Use a tabela a seguir para determinar quais perfis do IAM são necessários no Amazon ECS.

Função	Definição	Quando necessário	Mais informações
Função de execução de tarefas	Esse perfil permite que o Amazon ECS use outros serviços da AWS em seu nome.	Sua tarefa está hospedada no AWS Fargate ou em instâncias externas e: • extrai uma imagem de contêiner de um repositório privado do Amazon ECR.	Função do IAM de execução de tarefas do Amazon ECS

Função	Definição	Quando necessário	Mais informações
		• extrai uma imagem de contêiner de um repositório privado do Amazon ECR em uma conta diferente daquela que executa a tarefa. • envia logs de contêiner ao CloudWatch Logs usando o driver de log `awslogs`. Sua tarefa está hospedada no AWS Fargate ou em instâncias do Amazon EC2 e: • usa autenticação de registro privado. • usa monitoramento de runtime. • a definição de tarefa faz referênci a a dados confidenc iais usando segredos do Secrets Manager ou parâmetros do AWS Systems Manager Parameter Store.	

Função	Definição	Quando necessário	Mais informações
Função da tarefa	Esse perfil permite que o código da aplicação (no contêiner) use outros serviços da AWS.	Sua aplicação acessa outros serviços da AWS, como o Amazon S3.	Perfil do IAM para tarefas do Amazon ECS
Função de instância de contêiner	Esse perfil permite que as instâncias do EC2 ou instâncias externas se registrem no cluster.	Sua tarefa está hospedada em instâncias do Amazon EC2 ou em uma instância externa.	Função do IAM de instância de contêiner do Amazon ECS
Perfil do Amazon ECS Anywhere	Esse perfil permite que as instâncias externas acessem as APIs da AWS.	Sua tarefa está hospedada em instâncias externas.	Perfil do IAM para o Amazon ECS Anywhere
Infraestrutura do Amazon ECS para o perfil de balanceadores de carga	Esse perfil permite que o Amazon ECS gerencie recursos de balanceadores de carga nos clusters em seu nome para implantações azul/verde.	Você pode querer usar implantações azul/verde do Amazon ECS.	Perfil do IAM da infraestrutura do Amazon ECS para balanceadores de carga
Perfil para CodeDeploy do Amazon ECS	Esse perfil permite que o CodeDeploy faça atualizações nos serviços.	Use o tipo de implantação azul/verde do CodeDeDeploy para implantar serviços.	Função do IAM para CodeDeploy do Amazon ECS
Perfil para EventBridge do Amazon ECS	Esse perfil permite que o EventBridge faça atualizações nos serviços.	Use regras e destinos do EventBridge para agendar as tarefas.	Perfil do IAM para EventBridge do Amazon ECS

Função	Definição	Quando necessário	Mais informações
Perfil de infraestrutura do Amazon ECS	Esse perfil permite que o Amazon ECS gerencie recursos de infraestrutura nos clusters.	• Você deseja anexar volumes do Amazon EBS às tarefas do tipo de execução do Fargate ou EC2 do Amazon ECS. O perfil de infraestrutura permite que o Amazon ECS gerencie os volumes do Amazon EBS para as tarefas. • Você deseja usar o Transport Layer Security (TLS) para criptografar o tráfego entre os serviços do Amazon ECS Service Connect. • Você deseja criar grupos de destino do VPC Lattice.	Perfil do IAM de infraestrutura do Amazon ECS

Práticas recomendadas para perfis do IAM no Amazon ECS

Os perfis exigidos pelo Amazon ECS dependem do tipo de execução da definição da tarefa e dos recursos usados. Recomendamos criar perfis separados na tabela em vez de compartilhar funções.

Função	Definição	Quando necessário	Mais informações
Função de execução de tarefas	Esse perfil permite que o Amazon ECS use outros serviços da AWS em seu nome.	Sua tarefa está hospedada no AWS Fargate ou em instâncias externas e: • extrai uma imagem de contêiner de um repositório privado do Amazon ECR. • extrai uma imagem de contêiner de um repositório privado do Amazon ECR em uma conta diferente daquela que executa a tarefa. • envia logs de contêiner ao CloudWatch Logs usando o driver de log `awslogs`. Sua tarefa está hospedada no AWS Fargate ou em instâncias do Amazon EC2 e: • usa autenticação de registro privado. • usa monitoramento de runtime.	Função do IAM de execução de tarefas do Amazon ECS

Função	Definição	Quando necessário	Mais informações
		• a definição de tarefa faz referência a dados confidenciais usando segredos do Secrets Manager ou parâmetros do AWS Systems Manager Parameter Store.	
Função da tarefa	Esse perfil permite que o código da aplicação (no contêiner) use outros serviços da AWS.	Sua aplicação acessa outros serviços da AWS, como o Amazon S3.	Perfil do IAM para tarefas do Amazon ECS
Função de instância de contêiner	Esse perfil permite que as instâncias do EC2 ou instâncias externas se registrem no cluster.	Sua tarefa está hospedada em instâncias do Amazon EC2 ou em uma instância externa.	Função do IAM de instância de contêiner do Amazon ECS
Perfil do Amazon ECS Anywhere	Esse perfil permite que as instâncias externas acessem as APIs da AWS.	Sua tarefa está hospedada em instâncias externas.	Perfil do IAM para o Amazon ECS Anywhere
Perfil para CodeDeploy do Amazon ECS	Esse perfil permite que o CodeDeploy faça atualizações nos serviços.	Use o tipo de implantação azul/verde do CodeDeDeploy para implantar serviços.	Função do IAM para CodeDeploy do Amazon ECS

Função	Definição	Quando necessário	Mais informações
Perfil para EventBridge do Amazon ECS	Esse perfil permite que o EventBridge faça atualizações nos serviços.	Use regras e destinos do EventBridge para agendar as tarefas.	Perfil do IAM para EventBridge do Amazon ECS
Perfil de infraestrutura do Amazon ECS	Esse perfil permite que o Amazon ECS gerencie recursos de infraestrutura nos clusters.	• Você deseja anexar volumes do Amazon EBS às tarefas do tipo de execução do Fargate ou EC2 do Amazon ECS. O perfil de infraestrutura permite que o Amazon ECS gerencie os volumes do Amazon EBS para as tarefas. • Você deseja usar o Transport Layer Security (TLS) para criptografar o tráfego entre os serviços do Amazon ECS Service Connect. • Você deseja criar grupos de destino do VPC Lattice.	Perfil do IAM de infraestrutura do Amazon ECS

Função da tarefa

Recomendamos atribuir um perfil de tarefas. Seu perfil pode ser diferenciado do perfil da instância do Amazon EC2 que está sendo executada. A atribuição de um perfil a cada tarefa se alinha ao princípio do acesso de privilégio mínimo e permite um maior controle granular sobre ações e recursos.

Quando você adiciona um perfil de tarefa a uma definição de tarefa, o agente de contêiner do Amazon ECS cria automaticamente um token com um ID de credencial exclusivo (por exemplo, `12345678-90ab-cdef-1234-567890abcdef`) para a tarefa. Esse token e as credenciais do perfil são então adicionados ao cache interno do agente. O agente preenche a variável de ambiente `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI` no contêiner com o URI do ID da credencial (por exemplo, `/v2/credentials/12345678-90ab-cdef-1234-567890abcdef`).

É possível recuperar manualmente as credenciais do perfil temporário de dentro de um contêiner anexando a variável de ambiente ao endereço IP do agente de contêiner do Amazon ECS e executando o comando `curl` na string resultante.

```
curl 169.254.170.2$AWS_CONTAINER_CREDENTIALS_RELATIVE_URI
```

A saída esperada é a seguinte:

```
{
 "RoleArn": "arn:aws:iam::123456789012:role/SSMTaskRole-SSMFargateTaskIAMRole-
DASWWSF2WGD6",
 "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
 "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
 "Token": "IQoJb3JpZ2luX2VjEEM/Example==",
 "Expiration": "2021-01-16T00:51:53Z"
}
```

As versões mais recentes dos AWS SDKs buscam automaticamente essas credenciais da variável de ambiente `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI` ao fazer chamadas da API da AWS. Para obter informações sobre como renovar credenciais, consulte Renovação de credenciais da AWS no rePost.

A saída inclui um par de chaves de acesso que consiste em um ID de chave de acesso secreto e uma chave secreta que sua aplicação usa para acessar recursos da AWS. Também inclui um token AWS usado para verificar se as credenciais são válidas. Por padrão, as credenciais atribuídas às

tarefas usando funções de tarefa são válidas por seis horas. Depois disso, elas serão alternadas automaticamente pelo agente de contêiner do Amazon ECS.

Função de execução de tarefas

O perfil de execução de tarefas é usado para conceder ao agente de contêiner do Amazon ECS permissão para fazer chamar ações específicas da API da AWS em seu nome. Por exemplo, quando você usa AWS Fargate, o Fargate precisa de um perfil do IAM que permita extrair imagens do Amazon ECR e gravar registros no CloudWatch Logs. Um perfil do IAM também é necessária quando uma tarefa faz referência a um segredo armazenado no AWS Secrets Manager, como um segredo de extração de imagem.

> **Note**
>
> Se você estiver obtendo imagens como um usuário autenticado, é menos provável que seja afetado pelas mudanças que ocorreram nos limites da taxa de extração do Docker Hub. Para obter mais informações, consulte Autenticação de registro privado para instâncias de contêiner.
> Ao usar o Amazon ECR e o Amazon ECR Public, é possível evitar os limites impostos pelo Docker. Se você extrair imagens do Amazon ECR, isso também ajudará a reduzir os tempos de extração da rede e reduzirá as alterações na transferência de dados quando o tráfego sair da sua VPC.

> **Important**
>
> Ao usar o Fargate, você deve se autenticar em um registro de imagem privado usando `repositoryCredentials`. Não é possível definir as variáveis de ambiente `ECS_ENGINE_AUTH_TYPE` ou `ECS_ENGINE_AUTH_DATA` do agente de contêiner do Amazon ECS nem modificar o arquivo `ecs.config` para tarefas hospedadas no Fargate. Para obter mais informações, consulte Autenticação de registro privado para tarefas.

Função de instância de contêiner

A política do IAM gerenciada `AmazonEC2ContainerServiceforEC2Role` deve incluir as permissões a seguir. Ao seguir o aviso de segurança padrão de concessão de privilégio mínimo, a política gerenciada `AmazonEC2ContainerServiceforEC2Role` pode ser usada como um guia.

Se qualquer permissão concedida na política gerenciada não for necessária para seu caso de uso, crie uma política personalizada e adicione apenas as permissões necessárias.

- `ec2:DescribeTags`: (opcional) permite que uma entidade principal descreva as tags associadas a uma instância do Amazon EC2. Essa permissão é usada pelo agente de contêiner do Amazon ECS para oferecer suporte à propagação da etiqueta de recurso. Para obter mais informações, consulte Como os recursos são marcados.
- `ecs:CreateCluster`: (Opcional) permite que uma entidade principal crie um cluster do Amazon ECS. Essa permissão é usada pelo agente de contêiner do Amazon ECS para criar um cluster `default`, se nenhum já existir.
- `ecs:DeregisterContainerInstance`: (Opcional) permite que uma entidade principal cancele o registro de uma instância de contêiner do Amazon ECS de um cluster. O agente de contêiner do Amazon ECS não chama essa operação de API, mas essa permissão permanece para garantir a compatibilidade com versões anteriores.
- `ecs:DiscoverPollEndpoint`: (Obrigatório) essa ação retorna endpoints que o agente de contêiner do Amazon ECS usa para pesquisar atualizações.
- `ecs:Poll`: (Obrigatório) permite que o agente de contêiner do Amazon ECS se comunique com o ambiente de gerenciamento do Amazon ECS para relatar alterações no estado da tarefa.
- `ecs:RegisterContainerInstance`: (Obrigatório) permite que uma entidade principal cancele o registro de uma instância de contêiner com um cluster. Essa permissão é usada pelo agente de contêiner do Amazon ECS para registrar a instância do Amazon EC2 em um cluster e para oferecer suporte à propagação de tags de recurso.
- `ecs:StartTelemetrySession`: (Opcional) permite que o agente de contêiner do Amazon ECS se comunique com o ambiente de gerenciamento do Amazon ECS para relatar informações e métricas de integridade para cada contêiner e tarefa.

  Embora essa permissão não seja necessária, recomendamos que você a adicione para permitir que as métricas da instância do contêiner iniciem ações de escala e também recebam relatórios relacionados aos comandos de verificação de integridade.
- `ecs:TagResource`: (Opcional) permite que o agente de contêiner do Amazon ECS marque o cluster na criação e marque as instâncias de contêiner quando elas são registradas em um cluster.
- `ecs:UpdateContainerInstancesState`: permite que uma entidade principal modifique o status de uma instância de contêiner do Amazon ECS. Essa permissão é usada pelo agente de contêiner do Amazon ECS para drenagem de instâncias spot.

- `ecs:Submit*`: (Obrigatório) isso inclui as ações da API `SubmitAttachmentStateChanges`, `SubmitContainerStateChange` e `SubmitTaskStateChange`. Elas são usadas pelo agente de contêiner do Amazon ECS para relatar alterações de estado de cada recurso ao ambiente de gerenciamento do Amazon ECS. A permissão `SubmitContainerStateChange` não é mais usada pelo agente de contêiner do Amazon ECS, mas permanece para garantir a compatibilidade com versões anteriores.
- `ecr:GetAuthorizationToken`: (Opcional) permite que uma entidade principal recupere um token de autorização. O token de autorização representa as credenciais de autenticação do IAM e pode ser usado para acessar qualquer registro do Amazon ECR ao qual a entidade principal do IAM tem acesso. O token de autorização recebido é válido por 12 horas.
- `ecr:BatchCheckLayerAvailability`: (Opcional) quando uma imagem de contêiner é enviada para um repositório privado do Amazon ECR, cada camada de imagem é conferida para verificar se ela já foi enviada. Se já tiver sido, a camada da imagem será ignorada.
- `ecr:GetDownloadUrlForLayer`: (Opcional) quando uma imagem de contêiner é extraída de um repositório privado do Amazon ECR, essa API é chamada uma vez para cada camada de imagem que ainda não está armazenada em cache.
- `ecr:BatchGetImage`: (Opcional) quando uma imagem de contêiner é extraída de um repositório privado do Amazon ECR, essa API é chamada uma vez para recuperar o manifesto da imagem.
- `logs:CreateLogStream`: (Opcional) permite que uma entidade principal crie um fluxo de log do CloudWatch Logs para um grupo de logs especificado.
- `logs:PutLogEvents`: (Opcional) permite que uma entidade principal carregue um lote de eventos de log em um fluxo de log especificado.

Perfis vinculados a serviço

É possível usar o perfil vinculado ao serviço para o Amazon ECS conceder a permissão de serviço do Amazon ECS para chamar outras APIs de serviço em seu nome. O Amazon ECS precisa das permissões para criar e excluir interfaces de rede, registrar e cancelar o registro de destino com um grupo de destino. Ele também precisa das permissões necessárias para criar e excluir políticas de escalabilidade. Essas permissões são concedidas por meio do perfil vinculado ao serviço. Esse perfil é criado em seu nome na primeira vez que você usa o serviço.

> **Note**
>
> Se o perfil vinculado ao serviço for excluído inadvertidamente, será possível recriá-lo. Para obter instruções, consulte Criar o perfil vinculado ao serviço.

Recomendações de perfis

Recomendamos fazer o seguinte ao configurar suas políticas e perfis do IAM de tarefas.

Configurar acesso aos metadados do Amazon EC2

Ao executar suas tarefas em instâncias do Amazon EC2, recomendamos fortemente que você bloqueie o acesso aos metadados do Amazon EC2 para evitar que seus contêineres herdem o perfil atribuído a essas instâncias. Se a suas aplicações precisarem chamar uma ação da API da AWS, use perfis do IAM para tarefas em vez disso.

Para evitar que tarefas executadas no modo bridge acessem os metadados do Amazon EC2, execute o comando a seguir ou atualize os dados do usuário da instância. Para obter mais instruções sobre como atualizar os dados do usuário de uma instância, consulte este Artigo de suporte da AWS. Para obter mais informações sobre o bridge de definição de tarefas, consulte modo de rede de definição de tarefas.

```
sudo yum install -y iptables-services; sudo iptables --insert FORWARD 1 --in-interface
 docker+ --destination 169.254.169.254/32 --jump DROP
```

Para que essa alteração persista após uma reinicialização, execute o comando específico a seguir da sua imagem de máquina da Amazon (AMI):

- Amazon Linux 2

  ```
  sudo iptables-save | sudo tee /etc/sysconfig/iptables && sudo systemctl enable --now
   iptables
  ```

- Amazon Linux

  ```
  sudo service iptables save
  ```

Para tarefas que usem o modo de rede `awsvpc`, defina a variável de ambiente `ECS_AWSVPC_BLOCK_IMDS` como `true` no arquivo `/etc/ecs/ecs.config`.

Você deve definir a variável `ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST` como `false` no arquivo `ecs-agent config` para evitar que os contêineres em execução na rede do `host` acessem os metadados do Amazon EC2.

Uso do modo de rede **`awsvpc`**

Use o modo de rede `awsvpc` para restringir o fluxo de tráfego entre tarefas diferentes ou entre suas tarefas e outros serviços executados em sua Amazon VPC. Isso acrescenta um nível adicional de segurança. O modo de rede `awsvpc` fornece isolamento de rede em nível de tarefa para tarefas executadas no Amazon EC2. É o modo padrão ativado no AWS Fargate. É o único modo de rede que pode ser usado para atribuir um grupo de segurança às tarefas.

Use as informações do último acesso para refinar perfis

Recomendamos que você remova todas as ações que nunca tiverem sido usadas ou que não tiverem sido usadas por algum tempo. Isso evita que acessos indesejados ocorram. Para fazer isso, revise as informações do último acesso fornecidas pelo IAM e remova as ações que nunca foram usadas ou que não foram usadas recentemente. Faça isso seguindo as etapas a seguir.

Use o comando a seguir para gerar um relatório mostrando as informações do último acesso à política referenciada:

```
aws iam generate-service-last-accessed-details --arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

use o `JobId` que estava na saída para executar o comando a seguir. Após você fazer isso, será possível visualizar os resultados do relatório.

```
aws iam get-service-last-accessed-details --job-id 98a765b4-3cde-2101-2345-example678f9
```

Para obter mais informações, consulte Refinar permissões na AWS usando as informações do último acesso.

Monitorar o AWS CloudTrail para atividades suspeitas

É possível monitorar o AWS CloudTrail para quaisquer atividades suspeitas. A maioria das chamadas da API da AWS são registradas em log em AWS CloudTrail como eventos. Elas são

analisadas pelo AWS CloudTrail Insights, e você será alertado sobre qualquer comportamento suspeito associado às chamadas da API da `write`. Isso pode incluir um aumento no volume das chamadas. Esses alertas incluem informações como a hora em que a atividade incomum ocorreu e o principal ARN de identidade que contribuiu para as APIs.

É possível identificar ações que sejam executadas por tarefas com um perfil do IAM no AWS CloudTrail examinando a propriedade `userIdentity` do evento. No exemplo a seguir, o `arn` inclui o nome do perfil assumido, `s3-write-go-bucket-role`, seguido pelo nome da tarefa, `7e9894e088ad416eb5cab92afExample`.

```
"userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROA36C6WWEJ2YEXAMPLE:7e9894e088ad416eb5cab92afExample",
    "arn": "arn:aws:sts::123456789012:assumed-role/s3-write-go-bucket-
role/7e9894e088ad416eb5cab92afExample",
    ...
}
```

> **Note**
>
> Quando tarefas que assumem um perfil são executadas em instâncias de contêiner do Amazon EC2, uma solicitação é registrada pelo agente de contêiner do Amazon ECS no log de auditoria do agente que está localizado em um endereço no formato `/var/log/ecs/audit.log.YYYY-MM-DD-HH`. Para obter mais informações, consulte Log de perfis do IAM de tarefas e Registro em logs de eventos do Insights para trilhas.

Função do IAM de execução de tarefas do Amazon ECS

A função de execução de tarefas concede ao contêiner do Amazon ECS e aos agentes do Fargate permissão para fazer chamadas da API da AWS em seu nome. A função do IAM para execução da tarefa é necessária dependendo dos requisitos da sua tarefa. É possível ter várias funções de execução de tarefas para diferentes finalidades e serviços associados à sua conta.

> **Note**
>
> Essas permissões são disponibilizadas para o agente em execução na sua instância pelo Amazon ECS enviando periodicamente as credenciais temporárias do perfil, mas elas não são diretamente acessíveis pelos contêineres na tarefa. Para obter as permissões do IAM de

que o código da aplicação dentro do contêiner precisa para ser executado, consulte Perfil do IAM para tarefas do Amazon ECS.

Veja a seguir os casos de uso comuns para uma função do IAM de execução de tarefas:

- Sua tarefa está hospedada no AWS Fargate ou em uma instância externa e:
  - extrai uma imagem de contêiner de um repositório privado do Amazon ECR.
  - extrai uma imagem de contêiner de um repositório privado do Amazon ECR em uma conta diferente daquela que executa a tarefa.
  - envia logs de contêiner ao CloudWatch Logs usando o driver de log `awslogs`. Para obter mais informações, consulte Envio de logs do Amazon ECS para o CloudWatch .
- Suas tarefas são hospedadas no AWS Fargate ou em instâncias do Amazon EC2 e:
  - usa autenticação de registro privado. Para obter mais informações, consulte Permissões para autenticação no registro privado.
  - usa monitoramento de runtime.
  - a definição de tarefa faz referência a dados confidenciais usando segredos do Secrets Manager ou parâmetros do AWS Systems Manager Parameter Store. Para obter mais informações, consulte Permissões do Secrets Manager ou do Systems Manager.

> **Note**
>
> A função de execução de tarefas é compatível com a versão 1.16.0 e posterior do agente de contêiner do Amazon ECS.

O Amazon ECS fornece a política gerenciada denominada `AmazonECSTaskExecutionRolePolicy`, que contém as permissões que os casos de uso comuns descritos acima exigem. Para ter mais informações, consulte AmazonECSTaskExecutionRolePolicy no AWS Managed Policy Reference Guide. Pode ser necessário adicionar políticas em linha ao perfil de execução de tarefa para casos de uso especiais

O console do Amazon ECS cria um perfil de execução de tarefa. É possível anexar manualmente a política gerenciada do IAM para tarefas de modo a possibilitar que o Amazon ECS adicione permissões para futuros recursos e aprimoramentos à medida que eles forem introduzidos. É possível usar a pesquisa no console do IAM para buscar `ecsTaskExecutionRole` e ver se a conta

já tem o perfil de execução de tarefas. Para obter mais informações, consulte Pesquisa no console do IAM no Guia do usuário do IAM.

Se você extrair imagens como um usuário autenticado, haverá menos probabilidade sofrer impactos com as alterações que ocorreram no uso e limites do Docker Hub. Para obter mais informações, consulte Autenticação de registro privado para instâncias de contêiner.

Ao usar o Amazon ECR e o Amazon ECR Public, é possível evitar os limites impostos pelo Docker. Se você extrair imagens do Amazon ECR, isso também ajudará a reduzir os tempos de extração da rede e reduzirá as alterações na transferência de dados quando o tráfego sair da sua VPC.

Ao usar o Fargate, você deve se autenticar em um registro de imagem privado usando `repositoryCredentials`. Não é possível definir as variáveis de ambiente `ECS_ENGINE_AUTH_TYPE` ou `ECS_ENGINE_AUTH_DATA` do agente de contêiner do Amazon ECS nem modificar o arquivo `ecs.config` para tarefas hospedadas no Fargate. Para obter mais informações, consulte Autenticação de registro privado para tarefas.

Criar a função de execução de tarefa do

Se sua conta ainda não tiver um perfil de execução de tarefas, siga as etapas a seguir para criá-lo.

AWS Management Console

Para criar um perfil de serviço do Elastic Container Service (console do IAM)

1. Faça login no AWS Management Console e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Perfis e, em seguida, Criar perfil.
3. Em Tipo de Entidade Confiável, escolha AWS service (Serviço da AWS).
4. Em Serviço ou caso de uso, escolha Elastic Container Service e selecione o caso de uso da Tarefa do Elastic Container Service.
5. Escolha Próximo.
6. Na seção Adicionar permissões, pesquise AmazonECSTaskExecutionRolePolicy e selecione a política.
7. Escolha Próximo.
8. Em Nome do perfil, insira ecsTaskExecutionRole.
9. Reveja a função e escolha Criar função.

AWS CLI

Substitua cada *entrada do usuário* por suas próprias informações.

1. Crie um arquivo denominado `ecs-tasks-trust-policy.json` que contenha a política de confiança a ser usada para a função do IAM. O arquivo deve conter o seguinte:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs-tasks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Crie uma função do IAM denominada `ecsTaskExecutionRole` usando a política de confiança criada na etapa anterior.

```
aws iam create-role \
      --role-name ecsTaskExecutionRole \
      --assume-role-policy-document file://ecs-tasks-trust-policy.json
```

3. Anexe a política `AmazonECSTaskExecutionRolePolicy` gerenciada pela AWS à função`ecsTaskExecutionRole`.

```
aws iam attach-role-policy \
      --role-name ecsTaskExecutionRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy
```

Depois de criar o perfil, adicione a ele outras permissões para os atributos a seguir.

Recurso	Permissões adicionais
Extrair imagens de contêiner de registros privados fora da AWS (como Docker Hub, Quay.io ou seu próprio registro privado) usando as credenciais do Secrets Manager	Permissões para autenticação no registro privado
Transmitir dados confidenciais com o Systems Manager ou o Secrets Manager	Permissões do Secrets Manager ou do Systems Manager
Fazer com que as tarefas do Fargate extraiam imagens do Amazon ECR pelos endpoints da interface	Tarefas do Fargate que extraem imagens do Amazon ECR pelos endpoints da interface
Hospedar arquivos de configuração no bucket do Amazon S3	Permissões para o armazenamento de arquivos do Amazon S3
Configurar o Container Insights para visualizar os eventos de ciclo de vida do Amazon ECS	Permissões necessárias para habilitar os eventos de ciclo de vida do Amazon ECS no Container Insights
Visualizar eventos do ciclo de vida do Amazon ECS no Container Insights	Permissões necessárias para visualizar os eventos de ciclo de vida do Amazon ECS no Container Insights

Permissões para autenticação no registro privado

A autenticação de registro privado permite que suas tarefas do Amazon ECS extraiam imagens de contêiner de registros privados fora da AWS (como Docker Hub, Quay.io ou seu próprio registro privado) que exijam credenciais de autenticação. Esse recurso usa o Secrets Manager para armazenar com segurança suas credenciais de registro, que serão mencionadas na definição da tarefa usando o parâmetro `repositoryCredentials`.

Para obter mais informações sobre como configurar a autenticação de registro privado, consulte Uso de imagens de contêiner que não são da AWS no Amazon ECS.

Para fornecer acesso aos segredos que contêm suas credenciais de registro privado, adicione as permissões a seguir como uma política em linha à função de execução da tarefa. Para obter mais informações, consulte Adicionar e remover políticas do IAM.

- `secretsmanager:GetSecretValue`: obrigatório para recuperar as credenciais do registro privado do Secrets Manager.
- `kms:Decrypt`–-obrigatório somente se o segredo usar uma chave personalizada KMS e não a chave padrão. O nome do recurso da Amazon (ARN) da chave personalizada deve ser adicionado como um recurso.

Veja a seguir um exemplo de política em linha que adiciona as permissões.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:secret_name",
                "arn:aws:kms:<region>:<aws_account_id>:key/key_id"
            ]
        }
    ]
}
```

Permissões do Secrets Manager ou do Systems Manager

A permissão para o agente de contêiner extrair os recursos necessários do AWS Systems Manager ou do Secrets Manager. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.

Uso do Secrets Manager

Para fornecer acesso aos segredos do Secrets Manager criados por você, adicione manualmente as permissões a seguir ao perfil de execução da tarefa. Para obter informações sobre como gerenciar

permissões, consulte Adicionar e remover permissões de identidade do IAM no Guia do usuário do IAM.

- `secretsmanager:GetSecretValue`: obrigatório se você estiver fazendo referência a um segredo do Secrets Manager. Adiciona a permissão para recuperar o segredo do Secrets Manager.

O exemplo de política a seguir adiciona as permissões necessárias.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue"
      ],
      "Resource": [
        "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
      ]
    }
  ]
}
```

Uso do Systems Manager

> **Important**
>
> Para tarefas que usam o tipo de inicialização do EC2, você deve usar a variável de configuração do agente do ECS `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE=true` para usar esse recurso. É possível adicioná-lo ao arquivo `./etc/ecs/ecs.config` durante a criação da instância de contêiner ou adicioná-lo a uma instância existente e reiniciar o agente do ECS. Para obter mais informações, consulte Configuração do agente de contêiner do Amazon ECS.

Para dar acesso aos parâmetros do Systems Manager Parameter Store que você cria, adicione manualmente as permissões a seguir como uma política ao perfil de execução da tarefa. Para

obter informações sobre como gerenciar permissões, consulte Adicionar e remover permissões de identidade do IAM no Guia do usuário do IAM.

- `ssm:GetParameters`: obrigatório se você estiver fazendo referência a um parâmetro do Systems Manager Parameter Store em uma definição de tarefa. Adiciona a permissão para recuperar os parâmetros do Systems Manager.
- `secretsmanager:GetSecretValue`: obrigatório se você estiver fazendo referência direta a um segredo do Secrets Manager ou se o parâmetro do Systems Manager Parameter Store estiver fazendo referência a um segredo do Secrets Manager em uma definição de tarefa. Adiciona a permissão para recuperar o segredo do Secrets Manager.
- `kms:Decrypt`: obrigatório somente se o segredo usar uma chave gerenciada pelo cliente e não a chave padrão. O ARN da sua chave personalizada deve ser adicionado como recurso. Adiciona a permissão para descriptografar a chave gerenciada pelo cliente.

O exemplo de política a seguir adiciona as permissões necessárias:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:GetParameters",
        "secretsmanager:GetSecretValue",
        "kms:Decrypt"
      ],
      "Resource": [
        "arn:aws:ssm:region:aws_account_id:parameter/parameter_name",
        "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name",
        "arn:aws:kms:region:aws_account_id:key/key_id"
      ]
    }
  ]
}
```

Tarefas do Fargate que extraem imagens do Amazon ECR pelos endpoints da interface

Ao iniciar tarefas que usam o tipo de inicialização do Fargate que extraem imagens do Amazon ECR quando o Amazon ECR está configurado para usar um endpoint da VPC de interface, você pode

restringir o acesso das tarefas a uma VPC ou a um endpoint da VPC específico. Faça isso criando uma função de execução de tarefas que use chaves de condição do IAM.

Use as seguintes chaves de condição globais do IAM para restringir o acesso a uma VPC ou a um endpoint da VPC específico. Para obter mais informações, consulte Chaves de contexto de condição globais da AWS.

- `aws:SourceVpc`: restringe o acesso a uma VPC específica. Você pode restringir a VPC à VPC que hospeda a tarefa e o endpoint.
- `aws:SourceVpce`: restringe o acesso a um endpoint da VPC específico.

A política da função de execução de tarefas a seguir fornece um exemplo para adicionar chaves de condição.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:GetAuthorizationToken",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "arn:aws:ec2:arn:aws::111122223333:vpc-
endpoint/vpce-0123456789abcdef0",
                    "arn:aws:ec2:arn:aws::111122223333:vpc/vpc-0123456789abcdef0"
```

```
                }
            }
        }
    ]
}
```

Permissões do Amazon ECR

As seguintes permissões serão necessárias quando você precisar extrair imagens de contêiner de repositórios privados do Amazon ECR. O perfil de execução da tarefa deve ter essas permissões para permitir que os agentes do Amazon ECS e do Fargate extraiam imagens de contêiner em seu nome. Para implementações básicas do ECS, essas permissões deverão ser adicionadas ao perfil de execução da tarefa em vez do perfil do IAM da tarefa.

A política gerenciada do perfil de execução de tarefas do Amazon ECS (`AmazonECSTaskExecutionRolePolicy`) inclui as permissões necessárias para extrair imagens do Amazon ECR. Se você estiver usando a política gerenciada, não será necessário adicionar essas permissões separadamente.

Se você estiver criando uma política personalizada, inclua as seguintes permissões para permitir a extração de imagens do Amazon ECR:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchGetImage",
                "ecr:GetDownloadUrlForLayer",
                "ecr:GetAuthorizationToken"
            ],
            "Resource": "*"
        }
    ]
}
```

Observe que essas permissões são diferentes das permissões que podem ser exigidas o perfil do IAM da tarefa se o código da sua aplicação precisar interagir diretamente com as APIs do Amazon ECR. Para obter informações sobre permissões para o perfil do IAM para o Amazon ECR, consulte Permissões do Amazon ECR.

Permissões para o armazenamento de arquivos do Amazon S3

Quando você especifica um arquivo de configuração hospedado no Amazon S3, o perfil do IAM de execução de tarefa deve incluir a permissão `s3:GetObject` para o arquivo de configuração e a permissão `s3:GetBucketLocation` no bucket do Amazon S3 em que o arquivo está. Para obter mais informações, consulte Ações de políticas no Amazon S3 no Guia do usuário do Amazon Simple Storage Service.

O exemplo de política a seguir adiciona as permissões necessárias para recuperar um arquivo do Amazon S3. Especifique o nome do bucket do Amazon S3 e o nome do arquivo de configuração.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/folder_name/config_file_name"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket"
      ]
    }
  ]
}
```

Perfil do IAM para tarefas do Amazon ECS

Suas tarefas do Amazon ECS podem ter uma função do IAM associada a elas. As permissões concedidas no perfil do IAM são concedidas a contêineres em execução na tarefa. Esse perfil permite que o código da aplicação (em execução no contêiner) use outros serviços da AWS. O perfil de tarefas é necessário quando a aplicação acessa outros serviços da AWS, como o Amazon S3.

> **Note**
>
> Essas permissões não são acessadas pelo contêiner do Amazon ECS e por agentes do Fargate. Para obter as permissões do IAM que o Amazon ECS precisa para extrair imagens de contêineres e executar a tarefa, consulte Função do IAM de execução de tarefas do Amazon ECS.

Estes são os benefícios de usar perfis de tarefa:

- Separação de preocupações: se você estiver usando o tipo de execução do EC2, os perfis do IAM para tarefa permitem que você especifique as permissões do IAM para seus contêineres sem exigir que essas permissões sejam especificadas usando perfis de instância do EC2 (para mais informações, consulte Usar perfis de instância no Guia do usuário do AWS Identity and Access Management). Portanto, você pode implantar suas aplicações de maneira independente e uniforme nas instâncias de contêiner do ECS sem precisar modificar as permissões do IAM associadas às instâncias do EC2.
- Auditabilidade: o registro de acessos e de eventos está disponível por meio do CloudTrail para garantir auditoria retrospectiva. As credenciais de tarefa têm um contexto de "`taskArn`" anexado à sessão. Portanto assim os logs do CloudTrail mostram para qual tarefa as credenciais do perfil foram concedidas.
- Entrega uniforme de credenciais: o ECS entrega credenciais de perfil do IAM aos seus contêineres e os torna acessíveis por meio de uma interface bem definida, independentemente do tipo de execução associado às suas tarefas. No ECS Fargate, os perfis de instância do EC2 não estão disponíveis para contêineres em suas tarefas. Os perfis do IAM para tarefas permitem que você associe permissões do IAM aos seus contêineres, independentemente do tipo de execução, ao usar o AWS SDK ou AWS CLI em seus contêineres. Para obter mais informações sobre como o AWS SDK acessa essas credenciais, consulte Provedor de credenciais de contêiner.

> **Important**
>
> Os contêineres não são um limite de segurança e o uso dos perfis do IAM para tarefas não muda isso. Cada tarefa em execução no Fargate tem seu próprio limite de isolamento e não compartilha o kernel subjacente, os recursos de CPU, os recursos de memória nem a interface de rede elástica com outra tarefa. Para EC2 e instâncias de contêiner externas no ECS, não há isolamento de tarefas (ao contrário do Fargate) e os contêineres podem potencialmente acessar credenciais para outras tarefas na mesma instância de contêiner. Eles também podem acessar as permissões atribuídas ao perfil de instância de contêiner do ECS. Siga as recomendações em Recomendações de perfil para bloquear o acesso ao Amazon EC2 Instance Metadata Service para contêineres (para mais informações, consulte Use o serviço de metadados de instância para acessar metadados de instância no Guia do Usuário do Amazon EC2).
> Observe que ao especificar um perfil do IAM para uma tarefa, a AWS CLI ou outros SDKs nos contêineres dessa tarefa usam exclusivamente as credenciais da AWS fornecidas pelo perfil da tarefa e não herdam nenhuma permissão do IAM do Amazon EC2 ou da instância externa na qual estão em execução.

Criar o perfil do IAM de tarefa

Ao ser criada para suas tarefas, uma política do IAM deve incluir as permissões que você gostaria que fossem assumidas pelos contêineres nas tarefas. É possível usar uma política gerenciada existente da AWS ou criar uma política personalizada do zero que atenda às suas necessidades específicas. Para obter mais informações, consulte Criar políticas do IAM no Guia do usuário do IAM.

> **Important**
>
> Em tarefas do Amazon ECS (para todos os tipos de inicialização), recomendamos usar a política e o perfil do IAM em suas tarefas. Essas credenciais permitem que a tarefa faça solicitações da AWS API sem chamar `sts:AssumeRole` para assumir o mesmo perfil que já está associado à tarefa. Se a tarefa requer que o perfil assuma a si mesmo, é necessário criar uma política de confiança que permita explicitamente que o perfil se assuma. Para obter mais informações, consulte Atualizar uma política de confiança de perfil no Guia do usuário do IAM.

Após criar a política do IAM, será possível criar um perfil do IAM que inclua a política à qual você faz referência na definição de tarefa do Amazon ECS. É possível criar o perfil usando o caso de uso Tarefa do Elastic Container Service no console do IAM. Em seguida, você pode anexar a política do IAM específica ao perfil que concede as permissões desejadas aos contêineres da tarefa. Os procedimentos a seguir descrevem como fazer isso.

Se você tiver várias definições ou serviços de tarefas que exigem permissões do IAM, deverá considerar a criação de uma função para cada definição ou serviço de tarefa específico com as permissões mínimas necessárias para que as tarefas operem, de modo que você possa minimizar o acesso que fornece para cada tarefa.

Para obter informações sobre o endpoint de serviço da sua região, consulte Service endpoints no Guia de Referência geral da Amazon Web Services.

A função de tarefa do IAM deve ter uma política de confiança que especifique o serviço `ecs-tasks.amazonaws.com`. A permissão `sts:AssumeRole` permite que suas tarefas assumam uma função do IAM diferente daquela que a instância do Amazon EC2 usa. Dessa forma, sua tarefa não herda a função associada à instância do Amazon EC2. Veja a seguir um exemplo de política de confiança. Substitua o identificador de região e especifique o número da conta da AWS usada ao executar tarefas.

> **Important**
>
> Ao criar o perfil do IAM da tarefa, é recomendável usar as chaves de condição `aws:SourceAccount` ou `aws:SourceArn` na relação de confiança ou na política associada ao perfil para definir ainda mais o escopo das permissões e evitar o problema de segurança de confused deputy. Atualmente, não há suporte para o uso da chave de condição `aws:SourceArn` para especificar um cluster específico. É necessário usar o curinga para especificar todos os clusters. Para saber mais sobre o problema “confused deputy” e como proteger sua conta da AWS, consulte O problema “confused deputy” no Guia do usuário do IAM.

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
```

```
            "Effect":"Allow",
            "Principal":{
               "Service":[
                  "ecs-tasks.amazonaws.com"
               ]
            },
            "Action":"sts:AssumeRole",
            "Condition":{
               "ArnLike":{
               "aws:SourceArn":"arn:aws:ecs:us-west-2:111122223333:*"
               },
               "StringEquals":{
                  "aws:SourceAccount":"111122223333"
               }
            }
         }
      ]
}
```

Use o procedimentos a seguir para criar uma política para recuperar objetos do Amazon S3 usando um exemplo de política. Substitua cada *entrada do usuário* por seus próprios valores.

AWS Management Console

Para usar o editor de políticas JSON para criar uma política

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Políticas.

   Se essa for a primeira vez que você escolhe Políticas, a página Bem-vindo às políticas gerenciadas será exibida. Escolha Começar.
3. Na parte superior da página, escolha Criar política.
4. Na seção Editor de políticas, escolha a opção JSON.
5. Insira o seguinte documento de política JSON:

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
```

```
            "Effect":"Allow",
            "Action":[
               "s3:GetObject"
            ],
            "Resource":[
               "arn:aws:s3:::my-task-secrets-bucket/*"
            ]
         }
      ]
}
```

6. Escolha Próximo.

> **Note**
>
> É possível alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Próximo no editor Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte Restruturação de política no Guia do usuário do IAM.

7. Na página Revisar e criar, insira um Nome de política e uma Descrição (opcional) para a política que você está criando. Revise Permissões definidas nessa política para ver as permissões que são concedidas pela política.
8. Escolha Criar política para salvar sua nova política.

AWS CLI

Substitua cada *entrada do usuário* por seus próprios valores.

1. Crie um arquivo denominado `s3-policy.json` com o seguinte conteúdo.

   JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObject"
```

```
            ],
            "Resource":[
               "arn:aws:s3:::my-task-secrets-bucket/*"
            ]
         }
      ]
}
```

2. Use o comando da a seguir para criar a política do IAM usando o arquivo de documento da política JSON. Substitua cada *entrada do usuário* por seus próprios valores.

```
aws iam create-policy \
      --policy-name taskRolePolicy \
      --policy-document file://s3-policy.json
```

Use o procedimento a seguir para criar o perfil de serviço.

AWS Management Console

Para criar um perfil de serviço do Elastic Container Service (console do IAM)

1. Faça login no AWS Management Console e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Perfis e, em seguida, Criar perfil.
3. Em Tipo de Entidade Confiável, escolha AWS service (Serviço da AWS).
4. Em Serviço ou caso de uso, escolha Elastic Container Service e selecione o caso de uso da Tarefa do Elastic Container Service.
5. Escolha Próximo.
6. Em Adicionar permissões, pesquise e selecione a política que você criou.
7. Escolha Próximo.
8. Em Role name (Nome da função), digite um nome para sua função. Para este exemplo, insira `AmazonECSTaskS3BucketRole` para o nome da função.
9. Reveja a função e escolha Criar função.

AWS CLI

1. Crie um arquivo denominado `ecs-tasks-trust-policy.json` que contenha a política de confiança a ser usada no perfil do IAM. O arquivo deve ter o conteúdo a seguir. Substitua o identificador de região e especifique o número da conta da AWS usada ao executar tarefas.

   JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":[
               "ecs-tasks.amazonaws.com"
            ]
         },
         "Action":"sts:AssumeRole",
         "Condition":{
            "ArnLike":{
            "aws:SourceArn":"arn:aws:ecs:us-west-2:111122223333:*"
            },
            "StringEquals":{
               "aws:SourceAccount":"111122223333"
            }
         }
      }
   ]
}
```

2. Crie uma função do IAM denominada `ecsTaskRole` usando a política de confiança criada na etapa anterior.

```
aws iam create-role \
      --role-name ecsTaskRole \
      --assume-role-policy-document file://ecs-tasks-trust-policy.json
```

3. Recupere o ARN da política do IAM que você criou usando o comando a seguir. Substitua *`taskRolePolicy`* pelo nome da política que você criou.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`taskRolePolicy`].Arn'
```

4. Anexe a política do IAM criada ao perfil `ecsTaskRole`. Substitua `policy-arn` pelo ARN da política que você criou.

```
aws iam attach-role-policy \
      --role-name ecsTaskRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/taskRolePolicy
```

Depois de criar o perfil, adicione a ele outras permissões para os atributos a seguir.

Recurso	Permissões adicionais
Usar o ECS Exec	Permissões do ECS Exec
Usar uma imagem de um repositório privado do Amazon ECR	Permissões do Amazon ECR
Usar instâncias do EC2 (Windows e Linux)	Configuração adicional de instâncias do Amazon EC2
Usar instâncias externas	Configuração adicional para instância externa
Usar instâncias Windows no EC2	Configuração adicional de instância do Windows no Amazon EC2

Permissões do Amazon ECR

As permissões a seguir são necessárias quando o código do seu aplicativo precisa interagir diretamente com os repositórios do Amazon ECR. Observe que essas permissões não são necessárias no nível do perfil do IAM da tarefa para uma implementação básica na qual você só precise extrair imagens do Amazon ECR. Em vez disso, o perfil de execução de tarefas do Amazon ECS deverá ter essas permissões. Para obter mais informações sobre a função de execução de tarefas, consulte Função do IAM de execução de tarefas do Amazon ECS.

Se o código da sua aplicação em execução no contêiner precisar interagir diretamente com as APIs do Amazon ECR, você deverá adicionar as seguintes permissões a um perfil do IAM de tarefa e incluir o perfil do IAM de tarefa na sua definição de tarefa. Para mais informações, consulte Adicionando e Removendo Políticas do IAM no Guia de Usuário do IAM.

Use a seguinte política para seu perfil do IAM de tarefa para adicionar as permissões necessárias do Amazon ECR para aplicações em contêiner que precisem interagir diretamente com o Amazon ECR:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchGetImage",
                "ecr:GetDownloadUrlForLayer",
                "ecr:GetAuthorizationToken"
            ],
            "Resource": "*"
        }
    ]
}
```

Permissões do ECS Exec

O recurso ECS Exec requer um perfil do IAM de tarefas para conceder aos contêineres as permissões necessárias à comunicação entre o SSM Agent gerenciado (agente do `execute-command`) e o serviço do SSM. Você deve adicionar as seguintes permissões a uma função do IAM de tarefa e incluir a função do IAM de tarefa na definição de tarefa. Para mais informações, consulte Adicionando e Removendo Políticas do IAM no Guia de Usuário do IAM.

Use a política a seguir para sua função do IAM de tarefa para adicionar as permissões necessárias do SSM.

JSON

```
{
```

```
   "Version": "2012-10-17",
   "Statement": [
       {
       "Effect": "Allow",
       "Action": [
            "ssmmessages:CreateControlChannel",
            "ssmmessages:CreateDataChannel",
            "ssmmessages:OpenControlChannel",
            "ssmmessages:OpenDataChannel"
       ],
      "Resource": "*"
      }
   ]
}
```

Configuração adicional de instâncias do Amazon EC2

Recomendamos que você limite as permissões na função de instância de contêiner à lista mínima de permissões usadas na política gerenciada do IAM `AmazonEC2ContainerServiceforEC2Role`.

As instâncias do Amazon EC2 exigem pelo menos a versão `1.11.0` do agente de contêiner para usar os perfis de tarefas. Contudo, recomendamos usar a versão mais recente do agente de contêiner. Para obter informações sobre como verificar a versão do agente e atualizar para a versão mais recente, consulte Atualizar o agente de contêiner do Amazon ECS. Se você usa uma AMI otimizada para o Amazon ECS, sua instância precisa, pelo menos, da versão `1.11.0-1` do pacote `ecs-init`. Se as instâncias estiverem usando a AMI otimizada para o Amazon ECS mais recente, elas conterão as versões necessárias do agente de contêiner e do `ecs-init`. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.

Se não estiver usando a AMI otimizada para o Amazon ECS nas instâncias de contêiner, adicione a opção `--net=host` ao comando docker run que inicia o agente e as seguintes variáveis de configuração de agente na configuração desejada (para obter mais informações, consulte Configuração do agente de contêiner do Amazon ECS):

`ECS_ENABLE_TASK_IAM_ROLE=true`

Usa os perfis do IAM para tarefas em contêineres com os modos de rede `bridge` e `default`.

`ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST=true`

Usa os perfis do IAM para tarefas em contêineres com o modo de rede `host`. A variável é aceita somente em nas versões 1.12.0 e posteriores do agente.

Para um exemplo do comando run, consulte Atualizar manualmente o agente de contêiner do Amazon ECS (para AMIs não otimizadas para Amazon ECS). Você também precisará definir os seguintes comandos de rede na sua instância de contêiner para que os contêineres das suas tarefas possam recuperar as credenciais da AWS:

```
sudo sysctl -w net.ipv4.conf.all.route_localnet=1
sudo iptables -t nat -A PREROUTING -p tcp -d 169.254.170.2 --dport 80 -j DNAT --to-
destination 127.0.0.1:51679
sudo iptables -t nat -A OUTPUT -d 169.254.170.2 -p tcp -m tcp --dport 80 -j REDIRECT --
to-ports 51679
```

Você deve salvar essas regras iptables em sua instância de contêiner para elas sobreviverem a uma reinicialização. É possível usar os comandos iptables-save e iptables-restore para salvar as regras iptables e restaurá-las na inicialização. Para obter mais informações, consulte a documentação específica do seu sistema operacional.

Para evitar que contêineres executados por tarefas que usam o modo de rede `awsvpc` acessem as informações de credenciais fornecidas ao perfil de instância do Amazon EC2, e ainda permitindo que as permissões fornecidas pela função da tarefa, defina a variável de configuração de agente `ECS_AWSVPC_BLOCK_IMDS` como `true` no arquivo de configuração do agente e reinicie o agente. Para obter mais informações, consulte Configuração do agente de contêiner do Amazon ECS.

Para evitar que contêineres executados por tarefas que usam o modo de rede `bridge` acessem as informações de credenciais fornecidas ao perfil de instância do Amazon EC2, e ainda permitindo que as permissões fornecidas pela função da tarefa, executando o seguinte comando iptables nas instâncias do Amazon EC2. Esse comando não afeta contêineres nas tarefas que usam os modos de rede `host` ou `awsvpc`. Para obter mais informações, consulte Modo de rede.

- ```
  sudo yum install -y iptables-services; sudo iptables --insert DOCKER-USER 1 --in-
  interface docker+ --destination 169.254.169.254/32 --jump DROP
  ```

  Você deve salvar essa regra de iptables em sua instância do Amazon EC2 para ela sobreviver a uma reinicialização. Ao usar a AMI otimizada para Amazon ECS, você pode usar o comando a
  ```

seguir. Para outros sistemas operacionais, consulte a documentação do sistema operacional em questão.

```
sudo iptables-save | sudo tee /etc/sysconfig/iptables && sudo systemctl enable --now
 iptables
```

Configuração adicional para instância externa

Suas instâncias externas exigem pelo menos uma versão `1.11.0` do agente de contêiner para usar perfis do IAM de tarefa; no entanto, recomendamos usar a versão mais recente do agente de contêiner. Para obter informações sobre como verificar a versão do agente e atualizar para a versão mais recente, consulte Atualizar o agente de contêiner do Amazon ECS. Se você estiver usando a AMI do Linux otimizada para o Amazon ECS, sua instância também precisará, pelo menos, da versão `1.11.0-1` do pacote `ecs-init`. Se as instâncias estiverem usando a AMI otimizada para o Amazon ECS mais recente, elas conterão as versões necessárias do agente de contêiner e do `ecs-init`. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.

Se não estiver usando a AMI otimizada para o Amazon ECS nas instâncias de contêiner, adicione a opção `--net=host` ao comando docker run que inicia o agente e as seguintes variáveis de configuração de agente na configuração desejada (para obter mais informações, consulte Configuração do agente de contêiner do Amazon ECS):

`ECS_ENABLE_TASK_IAM_ROLE=true`

Usa os perfis do IAM para tarefas em contêineres com os modos de rede `bridge` e `default`.

`ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST=true`

Usa os perfis do IAM para tarefas em contêineres com o modo de rede `host`. A variável é aceita somente em nas versões 1.12.0 e posteriores do agente.

Para um exemplo do comando run, consulte Atualizar manualmente o agente de contêiner do Amazon ECS (para AMIs não otimizadas para Amazon ECS). Você também precisará definir os seguintes comandos de rede na sua instância de contêiner para que os contêineres das suas tarefas possam recuperar as credenciais da AWS:

```
sudo sysctl -w net.ipv4.conf.all.route_localnet=1
sudo iptables -t nat -A PREROUTING -p tcp -d 169.254.170.2 --dport 80 -j DNAT --to-
destination 127.0.0.1:51679
```

```
sudo iptables -t nat -A OUTPUT -d 169.254.170.2 -p tcp -m tcp --dport 80 -j REDIRECT --
to-ports 51679
```

Você deve salvar essas regras iptables em sua instância de contêiner para elas sobreviverem a uma reinicialização. É possível usar os comandos iptables-save e iptables-restore para salvar as regras iptables e restaurá-las na inicialização. Para obter mais informações, consulte a documentação específica do seu sistema operacional.

Configuração adicional de instância do Windows no Amazon EC2

> **Important**
>
> Isso se aplica somente a contêineres do Windows no EC2 que usem perfil de tarefas.

O perfil de tarefas com recursos do Windows requer configuração adicional no EC2.

- Ao iniciar as instâncias de contêiner, você deve definir a opção `-EnableTaskIAMRole` no script de dados de usuário das instâncias de contêiner. O `EnableTaskIAMRole` ativa o recurso de funções de tarefas do IAM para as tarefas. Por exemplo:

  ```
  <powershell>
  Import-Module ECSTools
  Initialize-ECSAgent -Cluster 'windows' -EnableTaskIAMRole
  </powershell>
  ```

- Você deve executar bootstrap no contêiner com os comandos de rede fornecidos em Script de inicialização do contêiner para o Amazon ECS.
- Você deve criar uma função e uma política do IAM para as tarefas. Para obter mais informações, consulte Criar o perfil do IAM de tarefa.
- As funções do IAM do provedor de credencial da tarefa usam a porta 80 da instância de contêiner. Portanto, se você configurar funções do IAM para tarefas na instância de contêiner, os contêineres não poderão usar a porta 80 como porta do host em qualquer mapeamento de porta. Para expor os contêineres na porta 80, recomendamos configurar um serviço para eles que use o balanceamento de carga. É possível usar a porta 80 no balanceador de carga. Ao fazer isso, o tráfego pode ser roteado para outra porta do host nas instâncias de contêiner. Para obter mais informações, consulte Uso do balanceamento de carga para distribuir o tráfego de serviço do Amazon ECS.

- Se a instância do Windows for reiniciada, você deverá excluir a interface proxy e reinicializar o agente de contêiner do Amazon ECS para trazer de volta o proxy de credenciais.

Script de inicialização do contêiner para o Amazon ECS

Para os contêineres acessarem o proxy de credencial na instância de contêiner para obter credenciais, o contêiner deve receber bootstrap com os comandos de rede obrigatórios. O script de exemplo de código a seguir deve ser executado nos contêineres quando eles iniciam.

> **Note**
>
> Não é necessário executar esse script quando é usado o modo de rede `awsvpc` no Windows.

Se você executar contêineres do Windows que incluem o Powershell, use o seguinte script:

```
# Copyright Amazon.com Inc. or its affiliates. All Rights Reserved.
#
# Licensed under the Apache License, Version 2.0 (the "License"). You may
# not use this file except in compliance with the License. A copy of the
# License is located at
#
# http://aws.amazon.com/apache2.0/
#
# or in the "license" file accompanying this file. This file is distributed
# on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
# express or implied. See the License for the specific language governing
# permissions and limitations under the License.

$gateway = (Get-NetRoute | Where { $_.DestinationPrefix -eq '0.0.0.0/0' } | Sort-Object
 RouteMetric | Select NextHop).NextHop
$ifIndex = (Get-NetAdapter -InterfaceDescription "Hyper-V Virtual Ethernet*" | Sort-
Object | Select ifIndex).ifIndex
New-NetRoute -DestinationPrefix 169.254.170.2/32 -InterfaceIndex $ifIndex -NextHop
 $gateway -PolicyStore ActiveStore # credentials API
New-NetRoute -DestinationPrefix 169.254.169.254/32 -InterfaceIndex $ifIndex -NextHop
 $gateway -PolicyStore ActiveStore # metadata API
```

Se você executar contêineres do Windows que tenham apenas o shell de comandos, use o seguinte script:

```
# Copyright Amazon.com Inc. or its affiliates. All Rights Reserved.
#
# Licensed under the Apache License, Version 2.0 (the "License"). You may
# not use this file except in compliance with the License. A copy of the
# License is located at
#
# http://aws.amazon.com/apache2.0/
#
# or in the "license" file accompanying this file. This file is distributed
# on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
# express or implied. See the License for the specific language governing
# permissions and limitations under the License.

for /f "tokens=1" %i  in ('netsh interface ipv4 show interfaces ^| findstr /x /r
 ".*vEthernet.*"') do set interface=%i
for /f "tokens=3" %i  in ('netsh interface ipv4 show addresses %interface% ^| findstr /
x /r ".*Default.Gateway.*"') do set gateway=%i
netsh interface ipv4 add route prefix=169.254.170.2/32 interface="%interface%"
 nexthop="%gateway%" store=active # credentials API
netsh interface ipv4 add route prefix=169.254.169.254/32 interface="%interface%"
 nexthop="%gateway%" store=active # metadata API
```

Função do IAM de instância de contêiner do Amazon ECS

As instâncias de contêiner do Amazon ECS, incluindo instâncias do Amazon EC2 e externas, executam o agente de contêiner do Amazon ECS e exigem uma função do IAM para o serviço saber que o agente pertence a você. Para poder iniciar instâncias de contêiner e registrá-las em um cluster, você deve criar uma função do IAM para as instâncias de contêiner usarem. A função é criada na conta que você usa para fazer login no console ou executar os comandos da AWS CLI

> **Important**
>
> Se você estiver registrando instâncias externas em seu cluster, a função do IAM usada também exigirá permissões do Systems Manager. Para obter mais informações, consulte Perfil do IAM para o Amazon ECS Anywhere.

O Amazon ECS fornece a política `AmazonEC2ContainerServiceforEC2Role` gerenciada do IAM que contém as permissões necessárias para usar o conjunto completo de recursos do Amazon ECS. Essa política gerenciada pode ser anexada a uma função do IAM e associada às instâncias

de contêiner. Como alternativa, você pode usar a política gerenciada como guia quando criar uma política personalizada para ser usada. A função de instância de contêiner fornece permissões necessárias para que o agente de contêiner do Amazon ECS e o daemon do Docker chamem APIs da AWS em seu nome. Para obter mais informações sobre a política gerenciada, consulte AmazonEC2ContainerServiceforEC2Role.

Criar o perfil da instância de contêiner

> **Important**
>
> Se você estiver registrando instâncias externas em seu cluster, consulte Perfil do IAM para o Amazon ECS Anywhere.

É possível criar manualmente o perfil e anexar a política do IAM gerenciada para instâncias de contêiner para permitir que o Amazon ECS adicione permissões para recursos e aprimoramentos futuros à medida que sejam introduzidos. Use o procedimento a seguir para anexar a política do IAM gerenciada, se necessário.

AWS Management Console

Para criar um perfil de serviço do Elastic Container Service (console do IAM)

1. Faça login no AWS Management Console e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Perfis e, em seguida, Criar perfil.
3. Em Tipo de Entidade Confiável, escolha AWS service (Serviço da AWS).
4. Em Serviço ou caso de uso, escolha Elastic Container Service e selecione o caso de uso Perfil do EC2 do Elastic Container Service.
5. Escolha Próximo.
6. Na seção Políticas de permissões, verifique se a política AmazonEC2ContainerServiceforEC2Role está selecionada.

> **Important**
>
> A política gerenciada AmazonEC2ContainerServiceforEC2Role deve ser anexada à função do IAM de instância de contêiner. Caso contrário, você receberá um erro ao usar o AWS Management Console para criar clusters.

7. Escolha Próximo.
8. Em Nome do perfil, insira ecsInstanceRole
9. Reveja a função e escolha Criar função.

AWS CLI

Substitua cada *entrada do usuário* por seus próprios valores.

1. Crie um arquivo chamado `instance-role-trust-policy.json` com o conteúdo a seguir.

   JSON

   ```
   {
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": { "Service": "ec2.amazonaws.com"},
         "Action": "sts:AssumeRole"
       }
     ]
   }
   ```

2. Use o comando a seguir para criar o perfil do IAM da instância utilizando o documento da política de confiança.

   ```
   aws iam create-role \
       --role-name ecsInstanceRole \
       --assume-role-policy-document file://instance-role-trust-policy.json
   ```

3. Crie um perfil de instância chamado `ecsInstanceRole-profile` usando o comando create-instance-profile.

```
aws iam create-instance-profile --instance-profile-name ecsInstanceRole-profile
```

Exemplo de resposta

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AIPAJTLBPJLEGREXAMPLE",
        "Roles": [],
        "CreateDate": "2022-04-12T23:53:34.093Z",
        "InstanceProfileName": "ecsInstanceRole-profile",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:instance-profile/ecsInstanceRole-
profile"
    }
}
```

4. Adicione a função *ecsInstanceRole* ao perfil de instância *ecsInstanceRole-profile*.

```
aws iam add-role-to-instance-profile \
    --instance-profile-name ecsInstanceRole-profile \
    --role-name ecsInstanceRole
```

5. Anexe a política gerenciada `AmazonEC2ContainerServiceRoleForEC2Role` ao perfil usando o comando a seguir.

```
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role \
    --role-name ecsInstanceRole
```

Depois de criar o perfil, adicione a ele outras permissões para os atributos a seguir.

Recurso	Permissões adicionais
O Amazon ECR tem a imagem do container	Permissões do Amazon ECR

Recurso	Permissões adicionais
Fazer com que o CloudWatch Logs monitore instâncias de contêiner	Monitorar permissões de instâncias de contêiner
Hospedar arquivos de configuração no bucket do Amazon S3	Acesso somente leitura ao Amazon S3

Permissões do Amazon ECR

O perfil de instância de contêiner do Amazon ECS que você usa com as instâncias de contêiner deve ter as seguintes permissões de política do IAM para o Amazon ECR.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchCheckLayerAvailability",
                "ecr:BatchGetImage",
                "ecr:GetDownloadUrlForLayer",
                "ecr:GetAuthorizationToken"
            ],
            "Resource": "*"
        }
    ]
}
```

Se você usa a política gerenciada `AmazonEC2ContainerServiceforEC2Role` para suas instâncias de contêiner, sua função tem as permissões apropriadas. Para verificar se a função oferece suporte ao Amazon ECR, consulte Função do IAM da instância de contêiner do Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

Permissões necessárias para definir a configuração da conta awsvpcTrunking

O Amazon ECS oferece suporte à inicialização de instâncias de contêiner com maior densidade de ENI usando tipos de instâncias do Amazon EC2 compatíveis. Ao usar esse recurso, recomendamos criar dois perfis da instância de contêiner. Habilite a configuração da conta `awsvpcTrunking` em um perfil e use esse perfil em tarefas que exigem truncamento de ENI. Para obter mais informações sobre a configuração da conta `awsvpcTrunking`, consulte Acesso aos recursos do Amazon ECS com as configurações de conta.

O perfil de instância de contêiner que você usa com as instâncias de contêiner precisa ter as seguintes permissões de política do IAM para definir a configuração de conta.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:ListAccountSettings",
                "ecs:ListAttributes",
                "ecs:PutAccountSetting"
            ],
            "Resource": "*"
        }
    ]
}
```

Para usar o perfil de instância de contêiner, adicione o seguinte aos dados do usuário da instância:

```
#!/bin/bash
aws ecs put-account-setting --name awsvpcTrunking --value enabled --region region
ECS_CLUSTER=MyCluster>> /etc/ecs/ecs.config
EOF
```

Para obter mais informações sobre como adicionar dados do usuário às suas instâncias do EC2, consulte Executar comandos na instância do Linux na inicialização no Guia do usuário do Amazon EC2 para instâncias do Linux.

Acesso somente leitura ao Amazon S3

Armazenar informações de configuração em um bucket privado no Amazon S3 e conceder acesso somente leitura à função do IAM de instância de contêiner é uma maneira segura e conveniente de permitir a configuração da instância de contêiner no momento da inicialização. É possível armazenar uma cópia do arquivo `ecs.config` em um bucket privado, usar os dados de usuário do Amazon EC2 para instalar a AWS CLI e copiar as informações de configuração em `/etc/ecs/ecs.config` quando a instância for inicializada.

Para obter mais informações sobre como criar um arquivo `ecs.config`, armazená-lo no Amazon S3 e inicializar instâncias com essa configuração, consulte Armazenamento da configuração da instância de contêiner do Amazon ECS no Amazon S3.

É possível usar o comando da AWS CLI a seguir para permitir o acesso somente leitura do Amazon S3 ao perfil da instância de contêiner. Substitua *ecsInstanceRole* pelo nome do perfil criado.

```
aws iam attach-role-policy \
      --role-name ecsInstanceRole \
      --policy-arn arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess
```

Também é possível usar o console do IAM para adicionar o acesso somente leitura do Amazon S3 (`AmazonS3ReadOnlyAccess`) ao perfil. Para obter mais informações, consulte Atualizar permissões para um perfil no Guia do usuário do AWS Identity and Access Management.

Permissões necessárias para definir a configuração da conta awsvpcTrunking

O perfil de instância de contêiner do Amazon ECS que você usa com suas instâncias de contêineres deve ter as permissões de política do IAM a seguir para definir a configuração de instância de contêiner

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:ListAccountSettings",
                "ecs:ListAttributes",
                "ecs:PutAccountSetting"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

Monitorar permissões de instâncias de contêiner

Para que suas instâncias de contêiner possam enviar dados de log para o CloudWatch Logs, é necessário criar uma política do IAM para permitir que o agente do Amazon ECS escreva os logs de aplicações do cliente no CloudWatch (normalmente feito via driver do `awslogs`). Após criar a política, anexe-a a `ecsInstanceRole`.

AWS Management Console

Para usar o editor de políticas JSON para criar uma política

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Políticas.

   Se essa for a primeira vez que você escolhe Políticas, a página Bem-vindo às políticas gerenciadas será exibida. Escolha Começar.
3. Na parte superior da página, escolha Criar política.
4. Na seção Editor de políticas, escolha a opção JSON.
5. Insira o seguinte documento de política JSON:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": ["arn:aws:logs:*:*:*"]
        }
```

```
    ]
}
```

6. Escolha Próximo.

> **Note**
>
> É possível alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Próximo no editor Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte Restruturação de política no Guia do usuário do IAM.

7. Na página Revisar e criar, insira um Nome de política e uma Descrição (opcional) para a política que você está criando. Revise Permissões definidas nessa política para ver as permissões que são concedidas pela política.
8. Escolha Criar política para salvar sua nova política.

Após criar a política, anexe-a ao perfil da instância de contêiner. Para obter informações sobre como anexar a política ao perfil, consulte Atualizar permissões para um perfil no Guia do usuário do AWS Identity and Access Management.

AWS CLI

1. Crie um arquivo denominado `instance-cw-logs.json` com o seguinte conteúdo.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": ["arn:aws:logs:*:*:*"]
        }
```

```
    ]
}
```

2. Use o comando da a seguir para criar a política do IAM usando o arquivo de documento da política JSON.

```
aws iam create-policy \
      --policy-name cwlogspolicy \
      --policy-document file://instance-cw-logs.json
```

3. Recupere o ARN da política do IAM que você criou usando o comando a seguir. Substitua *cwlogspolicy* pelo nome da política criada.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`cwlogspolicy`].Arn'
```

4. Use o comando a seguir para anexar a política ao perfil do IAM da instância de contêiner usando o ARN da política.

```
aws iam attach-role-policy \
      --role-name ecsInstanceRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/cwlogspolicy
```

Perfil do IAM para o Amazon ECS Anywhere

Quando você registra um servidor on-premises ou uma máquina virtual (VM) no seu cluster, o servidor ou a VM precisará de um perfil do IAM para se comunicar com as APIs da AWS. Você precisa criar essa função do IAM apenas uma vez para cada conta da AWS. No entanto, essa função do IAM deve ser associada a cada servidor ou VM que você registra em um cluster. Essa função é a função `ECSAnywhereRole`. É possível criar essa função manualmente. Como alternativa, o Amazon ECS pode criar a função em seu nome quando você registrar uma instância externa no AWS Management Console. Use a pesquisa no console do IAM para buscar `ecsAnywhereRole` e ver se a conta já tem o perfil. Para obter mais informações, consulte Pesquisa no console do IAM no Guia do usuário do IAM.

A AWS fornece duas políticas do IAM gerenciadas que podem ser usadas na criação da função do IAM do ECS Anywhere, as políticas `AmazonSSMManagedInstanceCore` e `AmazonEC2ContainerServiceforEC2Role`. A política `AmazonEC2ContainerServiceforEC2Role` inclui permissões que, provavelmente,

fornecem mais acesso do que você precisa. Portanto, dependendo do seu caso de uso específico, recomendamos que você crie uma política personalizada adicionando apenas as permissões desta política que você precisa que constem dela. Para obter mais informações, consulte Função do IAM de instância de contêiner do Amazon ECS.

A função do IAM de execução de tarefas concede ao agente de contêiner do Amazon ECS permissão para fazer chamadas da API da AWS em seu nome. Quando uma função do IAM de execução de tarefa é usada, ela deve ser especificada na definição da tarefa. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.

A função de execução da tarefa será necessária se alguma das seguintes condições se aplicar:

- Você está enviando logs de contêiner ao CloudWatch Logs usando o driver de log `awslogs`.
- Sua definição de tarefa especifica uma imagem de contêiner hospedada em um repositório privado do Amazon ECR. Porém, se o perfil do IAM `ECSAnywhereRole` associado à instância externa também incluir as permissões necessárias para extrair imagens do Amazon ECR, o perfil de execução de tarefa não precisará incluí-las.

Criar perfil do Amazon ECS Anywhere

Substitua cada *`entrada do usuário`* por suas próprias informações.

1. Crie um arquivo local denominado `ssm-trust-policy.json` com a política de confiança a seguir.

   JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"Service": [
      "ssm.amazonaws.com"
    ]},
    "Action": "sts:AssumeRole"
  }
}
```

2. Crie o perfil e anexe a política de confiança usando o comando da AWS CLI a seguir.

```
aws iam create-role --role-name ecsAnywhereRole --assume-role-policy-document
 file://ssm-trust-policy.json
```

3. Anexe as políticas gerenciadas da AWS usando o comando a seguir.

```
aws iam attach-role-policy --role-name ecsAnywhereRole --policy-arn
 arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
aws iam attach-role-policy --role-name ecsAnywhereRole --policy-arn
 arn:aws:iam::aws:policy/service-role/AmazonEC2ContainerServiceforEC2Role
```

Você também pode usar o fluxo de trabalho de política de confiança personalizada do IAM para criar o perfil. Para obter mais informações, consulte Criar um perfil usando políticas de confiança personalizadas (console) no Guia do usuário do IAM.

Perfil do IAM de infraestrutura do Amazon ECS

Um perfil do IAM de infraestrutura do Amazon ECS permite que o Amazon ECS gerencie recursos de infraestrutura nos clusters em seu nome, sendo usado quando:

- Você deseja anexar volumes do Amazon EBS às tarefas do tipo de execução do Fargate ou EC2 do Amazon ECS. O perfil de infraestrutura permite que o Amazon ECS gerencie os volumes do Amazon EBS para as tarefas.
- Você deseja usar o Transport Layer Security (TLS) para criptografar o tráfego entre os serviços do Amazon ECS Service Connect.
- Você deseja criar grupos de destino do Amazon VPC Lattice.

Quando o Amazon ECS assume esse perfil para realizar ações em seu nome, os eventos ficam visíveis no AWS CloudTrail. Se o Amazon ECS usar o perfil para gerenciar volumes do Amazon EBS anexados às tarefas, o log do CloudTrail `roleSessionName` será `ECSTaskVolumesForEBS`. Se o perfil for usado para criptografar o tráfego entre os serviços do Service Connect, o log do CloudTrail `roleSessionName` será `ECSServiceConnectForTLS`. Se a função for usada para criar grupos de destino para o VPC Lattice, o log do CloudTrail `roleSessionName` será `ECSNetworkingWithVPCLattice`. Você pode usar esse nome para pesquisar eventos no console do CloudTrail filtrando pelo Nome de usuário.

O Amazon ECS fornece as políticas gerenciadas que contêm as permissões necessárias para anexação de volume e TLS. Para obter mais

informações, consulte, AmazonECSInfrastructureRolePolicyForVolumes, AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity e AmazonECSInfrastructureRolePolicyForVpcLatticeno Guia de referência de políticas gerenciadas pela AWS.

Criar o perfil de infraestrutura do Amazon ECS

Substitua cada *entrada do usuário* por suas próprias informações.

1. Crie um arquivo denominado `ecs-infrastructure-trust-policy.json` que contenha a política de confiança a ser usada para a função do IAM. O arquivo deve conter o seguinte:

   JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessToECSForInfrastructureManagement",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Use o comando da AWS CLI a seguir para criar um perfil denominado `ecsInfrastructureRole` usando a política de confiança criada na etapa anterior.

```
aws iam create-role \
      --role-name ecsInfrastructureRole \
      --assume-role-policy-document file://ecs-infrastructure-trust-policy.json
```

3. Dependendo do seu caso de uso, anexe a política gerenciada da AWS, `AmazonECSInfrastructureRolePolicyForVolumes`, `AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity` ou `AmazonECSInfrastructureRolePolicyForVpcLattice` ao perfil `ecsInfrastructureRole`.

```
aws iam attach-role-policy \
      --role-name ecsInfrastructureRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSInfrastructureRolePolicyForVolumes
```

```
aws iam attach-role-policy \
      --role-name ecsInfrastructureRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity
```

Você também pode usar o fluxo de trabalho da Política de confiança personalizada do console do IAM para criar o perfil. Para obter mais informações, consulte Criar um perfil usando políticas de confiança personalizadas (console) no Guia do usuário do IAM.

Important

Se o perfil de infraestrutura do ECS estiver sendo usado pelo Amazon ECS para gerenciar volumes do Amazon EBS anexados às tarefas, verifique os pontos a seguir antes de interromper as tarefas que usam volumes do Amazon EBS.

- O perfil não foi excluído.
- A política de confiança do perfil não foi modificada para remover o acesso ao Amazon ECS (`ecs.amazonaws.com`).
- A política gerenciada `AmazonECSInfrastructureRolePolicyForVolumes` não foi removida. Caso precise modificar as permissões do perfil, retenha pelo menos `ec2:DetachVolume`, `ec2:DeleteVolume` e `ec2:DescribeVolumes` para excluir o volume.

Excluir ou modificar o perfil antes de interromper tarefas com volumes anexados do Amazon EBS resultará na paralisação das tarefas em `DEPROVISIONING` e na falha ao excluir os volumes associados do Amazon EBS. O Amazon ECS fará automaticamente uma nova tentativa, em intervalos regulares, de interromper a tarefa e excluir o volume até que as permissões necessárias sejam restauradas. Você pode ver o status do anexo do volume de uma tarefa e o motivo do status associado usando a API DescribeTasks.

Depois de criar o arquivo, você deverá conceder ao usuário permissão para passar o perfil para o Amazon ECS.

Permissão para passar o perfil de infraestrutura para o Amazon ECS

Para usar um perfil do IAM de infraestrutura do ECS, você deve conceder permissão ao usuário para passar o perfil ao Amazon ECS. Anexe a permissão `iam:PassRole` a seguir ao usuário. Substitua *`ecsInfrastructureRole`* pelo nome do perfil de infraestrutura que você criou.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [

        {
            "Action": "iam:PassRole",
            "Effect": "Allow",
            "Resource": ["arn:aws:iam::*:role/ecsInfrastructureRole"],
            "Condition": {
                "StringEquals": {"iam:PassedToService": "ecs.amazonaws.com"}
            }
        }
    ]
}
```

Para obter mais informações sobre `iam:Passrole` e as atualizações das permissões de usuário, consulte Conceder permissões a um usuário para passar um perfil para um serviço da AWS e Alteração de permissões de um usuário do IAM no Guia do usuário do AWS Identity and Access Management.

Perfil do IAM da infraestrutura do Amazon ECS para balanceadores de carga

Um perfil do IAM de infraestrutura do Amazon ECS para balanceadores de carga permite que o Amazon ECS gerencie recursos de balanceadores de carga nos clusters em seu nome, sendo usado quando:

- Você deseja usar implantações azul/verde com o Amazon ECS. O perfil de infraestrutura permite que o Amazon ECS gerencie recursos de balanceadores de carga das suas implantações.

- Você precisa do Amazon ECS para criar, modificar ou excluir recursos de balanceadores de carga, como grupos de destino e receptores, durante as operações de implantação.

Quando o Amazon ECS assume esse perfil para realizar ações em seu nome, os eventos ficam visíveis no AWS CloudTrail. Se o Amazon ECS usar o perfil para gerenciar recursos de balanceadores de carga para as implantações azul/verde, o log `roleSessionName` do CloudTrail será `ECSNetworkingWithELB` ou `ecs-service-scheduler`. Você pode usar esse nome para pesquisar eventos no console do CloudTrail filtrando pelo Nome de usuário.

O Amazon ECS fornece uma política gerenciada que contém as permissões necessárias para o gerenciamento de balanceadores de carga. Para obter mais informações, consulte AmazonECSInfrastructureRolePolicyForLoadBalancers no Guia de referência de políticas gerenciadas pela AWS.

Criação do perfil de infraestrutura do Amazon ECS para balanceadores de carga

Substitua cada *`entrada do usuário`* por suas próprias informações.

1. Crie um arquivo denominado `ecs-infrastructure-trust-policy.json` que contenha a política de confiança a ser usada para a função do IAM. O arquivo deve conter o seguinte:

   JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessToECSForInfrastructureManagement",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Use o comando da AWS CLI a seguir para criar um perfil denominado `ecsInfrastructureRoleForLoadBalancers` usando a política de confiança criada na etapa anterior.

```
aws iam create-role \
      --role-name ecsInfrastructureRoleForLoadBalancers \
      --assume-role-policy-document file://ecs-infrastructure-trust-policy.json
```

3. Anexe a política `AmazonECSInfrastructureRolePolicyForLoadBalancers` gerenciada pela AWS à função`ecsInfrastructureRoleForLoadBalancers`.

```
aws iam attach-role-policy \
      --role-name ecsInfrastructureRoleForLoadBalancers \
      --policy-arn arn:aws:iam::aws:policy/
AmazonECSInfrastructureRolePolicyForLoadBalancers
```

Você também pode usar o fluxo de trabalho da Política de confiança personalizada do console do IAM para criar o perfil. Para obter mais informações, consulte Criar um perfil usando políticas de confiança personalizadas (console) no Guia do usuário do IAM.

Important

Se o perfil de infraestrutura estiver sendo usada pelo Amazon ECS para gerenciar recursos de balanceadores de carga de suas implantações azul/verde, verifique o seguinte antes de excluir ou modificar o perfil:

- O perfil não é excluído enquanto as implantações ativas estão em andamento.
- A política de confiança do perfil não foi modificada para remover o acesso ao Amazon ECS (`ecs.amazonaws.com`).
- A política gerenciada `AmazonECSInfrastructureRolePolicyForLoadBalancers` não é removida enquanto as implantações ativas estão em andamento.

Excluir ou modificar o perfil durante as implantações azul/verde ativas pode resultar em falhas de implantação e deixar seus serviços em um estado inconsistente.

Depois de criar o arquivo, você deverá conceder ao usuário permissão para passar o perfil para o Amazon ECS.

Permissão para passar o perfil de infraestrutura para o Amazon ECS

Para usar um perfil do IAM de infraestrutura do ECS para balanceadores de carga, você deve conceder permissão ao usuário para passar o perfil para o Amazon ECS. Anexe a permissão `iam:PassRole` a seguir ao usuário. Substitua *`ecsInfrastructureRoleForLoadBalancers`* pelo nome do perfil de infraestrutura que você criou.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": "iam:PassRole",
            "Effect": "Allow",
            "Resource":
 ["arn:aws:iam::*:role/ecsInfrastructureRoleForLoadBalancers"],
            "Condition": {
                "StringEquals": {"iam:PassedToService": "ecs.amazonaws.com"}
            }
        }
    ]
}
```

Para obter mais informações sobre `iam:Passrole` e as atualizações das permissões de usuário, consulte Conceder permissões a um usuário para passar um perfil para um serviço da AWS e Alteração de permissões de um usuário do IAM no Guia do usuário do AWS Identity and Access Management.

Função do IAM para CodeDeploy do Amazon ECS

Antes de poder usar o tipo de implantação azul/verde do CodeDeploy com o Amazon ECS, o serviço do CodeDeploy precisa de permissões para atualizar o serviço do Amazon ECS em seu nome. Essas permissões são fornecidas pela função do IAM para CodeDeploy (`ecsCodeDeployRole`).

> **Note**
>
> Os usuários também precisam de permissões para usar o CodeDeploy; essas permissões são descritas em Permissões obrigatórias do IAM.

Existem duas políticas gerenciadas fornecidas. Para obter mais informações, consulte um dos tópicos a seguir no AWS Managed Policy Reference Guide:

- AWSCodeDeployRoleForECS: concede ao CodeDeploy permissão para atualizar qualquer recurso usando a ação associada.
- AWSCodeDeployRoleForECSLimited: concede ao CodeDeploy permissões mais limitadas.

Criação do perfil do CodeDeploy

Use os procedimentos a seguir para criar um perfil do CodeDeDeploy para o Amazon ECS

AWS Management Console

Para criar o perfil de serviço do CodeDeploy (console do IAM)

1. Faça login no AWS Management Console e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Perfis e, em seguida, Criar perfil.
3. Em Tipo de Entidade Confiável, escolha AWS service (Serviço da AWS).
4. Em Serviço ou caso de uso, escolha CodeDeploy e selecione o caso de uso CodeDeploy: ECS.
5. Escolha Próximo.
6. Na seção Anexar política de permissões, certifique-se de que a política AWSCodeDeployRoleForECS esteja selecionada.
7. Escolha Próximo.
8. Em Nome do perfil, insira ecsCodeDeployRole.
9. Reveja a função e escolha Criar função.

AWS CLI

Substitua cada *entrada do usuário* por suas próprias informações.

1. Crie um arquivo denominado `codedeploy-trust-policy.json` que contenha a política de confiança a ser usada no perfil do IAM para CodeDeploy.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": ["codedeploy.amazonaws.com"]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

2. Crie uma função do IAM denominada `ecsCodedeployRole` usando a política de confiança criada na etapa anterior.

```
aws iam create-role \
      --role-name ecsCodedeployRole \
      --assume-role-policy-document file://codedeploy-trust-policy.json
```

3. Anexe a política gerenciada `AWSCodeDeployRoleForECS` ou `AWSCodeDeployRoleForECSLimited` ao perfil `ecsTaskRole`.

```
aws iam attach-role-policy \
      --role-name ecsCodedeployRole \
      --policy-arn arn:aws:iam::aws:policy/AWSCodeDeployRoleForECS
```

```
aws iam attach-role-policy \
      --role-name ecsCodedeployRole \
      --policy-arn arn:aws:iam::aws:policy/AWSCodeDeployRoleForECSLimited
```

Quando as tarefas do serviço precisam de um perfil de execução de tarefa, você deve adicionar a permissão `iam:PassRole` para cada perfil de execução de tarefa ou de substituição de perfil de tarefa ao perfil do CodeDeploy como uma política.

Permissões do perfil de execução de tarefa

Quando as tarefas do serviço precisam de um perfil de execução de tarefa, você deve adicionar a permissão `iam:PassRole` para cada perfil de execução de tarefa ou de substituição de perfil de tarefa ao perfil do CodeDeploy como uma política. Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS e Perfil do IAM para tarefas do Amazon ECS. Depois, você anexa essa política ao perfil do CodeDeploy

Crie a política do

AWS Management Console

Para usar o editor de políticas JSON para criar uma política

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Políticas.

   Se essa for a primeira vez que você escolhe Políticas, a página Bem-vindo às políticas gerenciadas será exibida. Escolha Começar.
3. Na parte superior da página, escolha Criar política.
4. Na seção Editor de políticas, escolha a opção JSON.
5. Insira o seguinte documento de política JSON:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["arn:aws:iam::<aws_account_id>:role/
<ecsCodeDeployRole>"]
        }
    ]
}
```

6. Escolha Próximo.

> **Note**
>
> É possível alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Próximo no editor Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte Restruturação de política no Guia do usuário do IAM.

7. Na página Revisar e criar, insira um Nome de política e uma Descrição (opcional) para a política que você está criando. Revise Permissões definidas nessa política para ver as permissões que são concedidas pela política.
8. Escolha Criar política para salvar sua nova política.

Após criar a política, anexe-a ao perfil do CodeDeploy. Para obter informações sobre como anexar uma política a um perfil, consulte Atualizar permissões para um perfil no Guia do usuário do AWS Identity and Access Management.

AWS CLI

Substitua cada *entrada do usuário* por suas próprias informações.

1. Crie um arquivo denominado `blue-green-iam-passrole.json` com o seguinte conteúdo.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                   "arn:arn:aws:iam::111122223333:role/ecs-code-deploy-
role"
            ]
        }
    ]
}
```

2. Use o comando da a seguir para criar a política do IAM usando o arquivo de documento da política JSON.

```
aws iam create-policy \
      --policy-name cdTaskExecutionPolicy \
      --policy-document file://blue-green-iam-passrole.json
```

3. Recupere o ARN da política do IAM que você criou usando o comando a seguir.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`cdTaskExecutionPolicy`].Arn'
```

4. Use o comando a seguir para anexar a política ao perfil do IAM para CodeDeploy.

```
aws iam attach-role-policy \
      --role-name ecsCodedeployRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/cdTaskExecutionPolicy
```

Perfil do IAM para EventBridge do Amazon ECS

Antes de poder usar tarefas programadas do Amazon ECS com regras e destinos do EventBridge, o serviço do EventBridge precisa de permissões para executar tarefas do Amazon ECS em seu nome. Essas permissões são fornecidas pelo perfil do IAM do EventBridge (`ecsEventsRole`).

A política `AmazonEC2ContainerServiceEventsRole` é mostrada abaixo.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:RunTask"],
            "Resource": ["*"]
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["*"],
```

```
            "Condition": {
                "StringLike": {"iam:PassedToService": "ecs-tasks.amazonaws.com"}
            }
        },
        {
            "Effect": "Allow",
            "Action": "ecs:TagResource",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ecs:CreateAction": ["RunTask"]
                }
            }
        }
    ]
}
```

Se as tarefas programadas exigirem o uso do perfil de execução de tarefa, um perfil de tarefa ou uma substituição de perfil de tarefa, você deverá adicionar permissões `iam:PassRole` em cada perfil de execução de tarefa, perfil de tarefa ou substituição de perfil de tarefa ao perfil do IAM do EventBridge. Para obter mais informações sobre a função de execução de tarefas, consulte Função do IAM de execução de tarefas do Amazon ECS.

> **Note**
>
> Especifique o ARN completo da função de execução de tarefa ou da substituição da função de tarefa.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [

 "arn:aws:iam::111122223333:role/ecsTaskExecutionRole_or_TaskRole_name"
```

```
            ]
        }
    ]
}
```

Você pode permitir que o AWS Management Console crie o perfil do EventBridge ao configurar uma tarefa programada. Para obter mais informações, consulte Uso do Agendador do Amazon EventBridge para programar tarefas do Amazon ECS .

Criar o perfil do EventBridge

Substitua cada *entrada do usuário* por suas próprias informações.

1. Crie um arquivo denominado `eventbridge-trust-policy.json` que contenha a política de confiança a ser usada para a função do IAM. O arquivo deve conter o seguinte:

   JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "events.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Use o comando a seguir para criar um perfil do IAM denominado `ecsEventsRole` usando a política de confiança criada na etapa anterior.

```
aws iam create-role \
      --role-name ecsEventsRole \
      --assume-role-policy-document file://eventbridge-trust-policy.json
```

3. Anexe o `AmazonEC2ContainerServiceEventsRole` gerenciado pela AWS ao perfil `ecsEventsRole` usando o comando a seguir.

```
aws iam attach-role-policy \
      --role-name ecsEventsRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceEventsRole
```

Você também pode usar o fluxo de trabalho da Política de confiança personalizada do console do IAM (https://console.aws.amazon.com/iam/) para criar o perfil. Para obter mais informações, consulte Criar um perfil usando políticas de confiança personalizadas (console) no Guia do usuário do IAM.

Anexação de uma política ao perfil do **`ecsEventsRole`**

Use os procedimentos a seguir para adicionar permissões do perfil de execução de tarefas ao perfil do IAM para EventBridge.

AWS Management Console

Para usar o editor de políticas JSON para criar uma política

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Políticas.

   Se essa for a primeira vez que você escolhe Políticas, a página Bem-vindo às políticas gerenciadas será exibida. Escolha Começar.
3. Na parte superior da página, escolha Criar política.
4. Na seção Editor de políticas, escolha a opção JSON.
5. Insira o seguinte documento de política JSON:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["arn:aws:iam::<aws_account_id>:role/
<ecsTaskExecutionRole_or_TaskRole_name>"]
```

```
        }
    ]
}
```

6. Escolha Próximo.

> **Note**
>
> É possível alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Próximo no editor Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte Restruturação de política no Guia do usuário do IAM.

7. Na página Revisar e criar, insira um Nome de política e uma Descrição (opcional) para a política que você está criando. Revise Permissões definidas nessa política para ver as permissões que são concedidas pela política.
8. Escolha Criar política para salvar sua nova política.

Depois de criar a política, anexe-a ao perfil do EventBridge. Para obter informações sobre como anexar uma política a um perfil, consulte Atualizar permissões para um perfil no Guia do usuário do AWS Identity and Access Management.

AWS CLI

Substitua cada *entrada do usuário* por suas próprias informações.

1. Crie um arquivo denominado `ev-iam-passrole.json` com o seguinte conteúdo.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [

 "arn:aws:iam::111122223333:role/ecsTaskExecutionRole_or_TaskRole_name"
            ]
```

```
        }
    ]
}
```

2. Use o comando da AWS CLI a seguir para criar a política do IAM usando o arquivo de documento da política JSON.

```
aws iam create-policy \
      --policy-name eventsTaskExecutionPolicy \
      --policy-document file://ev-iam-passrole.json
```

3. Recupere o ARN da política do IAM que você criou usando o comando a seguir.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`eventsTaskExecutionPolicy`].Arn'
```

4. Use o comando a seguir para anexar a política ao perfil do IAM para EventBridge usando o ARN da política.

```
aws iam attach-role-policy \
      --role-name ecsEventsRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/eventsTaskExecutionPolicy
```

Permissões necessárias para usar o console do Amazon ECS

Ao seguir a prática recomendada de conceder privilégio mínimo, é possível usar a política gerenciada `AmazonECS_FullAccess` como um modelo para criar sua própria política personalizada. Dessa forma, você pode retirar ou adicionar permissões à política gerenciada com base nos seus requisitos específicos. Para obter mais informações, consulte AmazonECS_FullAccess na Referência de políticas gerenciadas pela AWS.

Permissões para criar perfis do IAM

As ações a seguir exigem permissões adicionais para concluir a operação:

- Registro de uma instância externa - para obter mais informações, consulte Perfil do IAM para o Amazon ECS Anywhere
- Registro de uma definição de tarefa - para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS

- Criação de uma regra do EventBridge para uso no agendamento de tarefas - para obter mais informações, consulte Perfil do IAM para EventBridge do Amazon ECS

É possível adicionar essas permissões criando um perfil no IAM antes de usá-las no console do Amazon ECS. Se você não criar as funções, o console do Amazon ECS as criará em seu nome.

Permissões necessárias para registrar uma instância externa em um cluster

Você precisa de permissões adicionais ao registrar uma instância externa em um cluster e quiser criar um novo perfil de instância externa (`ecsExternalInstanceRole`).

As permissões adicionais a seguir são necessárias:

- `iam`: permite que as entidades principais listem perfis do IAM e suas políticas anexadas.
- `ssm`: permite que os diretores registrem a instância externa no Systems Manager.

> **Note**
>
> Para escolher um `ecsExternalInstanceRole` existente, você deve ter as permissões `iam:GetRole` e `iam:PassRole`.

A política a seguir contém as permissões necessárias e limita as ações ao perfil `ecsExternalInstanceRole`.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:CreateInstanceProfile",
              "iam:AddRoleToInstanceProfile",
              "iam:ListInstanceProfilesForRole",
              "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/ecsExternalInstanceRole"
        },
```

```
        {
            "Effect": "Allow",
            "Action": ["iam:PassRole","ssm:CreateActivation"],
            "Resource": "arn:aws:iam::*:role/ecsExternalInstanceRole"
        }
    ]
}
```

Permissões necessárias para registrar uma definição de tarefa

Você precisa de permissões adicionais ao registrar uma definição de tarefa e quiser criar um novo perfil de execução de tarefas (`ecsTaskExecutionRole`).

As permissões adicionais a seguir são necessárias:

- `iam`: permite que as entidades principais listem perfis do IAM e suas políticas anexadas.

> **Note**
>
> Para escolher um `ecsTaskExecutionRole` existente, você deve ter a permissão `iam:GetRole`.

A política a seguir contém as permissões necessárias e limita as ações ao perfil `ecsTaskExecutionRole`.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/ecsTaskExecutionRole"
        }
    ]
}
```

Permissões necessárias para criar uma regra do EventBridge para tarefas agendadas

Você precisa de permissões adicionais ao programar uma tarefa e se quiser criar uma novo perfil do CloudWatch Events (`ecsEventsRole`).

As permissões adicionais a seguir são necessárias:

- `iam`: permite que as entidades principais criem e listem perfis do IAM e suas políticas associadas, além de permitir que o Amazon ECS passe o perfil para outros serviços assumirem o perfil.

> **Note**
>
> Para escolher um `ecsEventsRole` existente, você deve ter as permissões `iam:GetRole` e `iam:PassRole`.

A política a seguir contém as permissões necessárias e limita as ações ao perfil `ecsEventsRole`.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:GetRole",
              "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::*:role/ecsEventsRole"
        }
    ]
}
```

Permissões necessárias para visualizar implantações de serviço

Ao seguir a prática recomendada de conceder privilégio mínimo, é necessário adicionar permissões adicionais para visualizar implantações de serviço no console.

É necessário ter acesso às seguintes ações:

- ListServiceDeployments

- DescribeServiceDeployments
- DescribeServiceRevisions

É necessário ter acesso aos seguintes recursos:

- Serviço
- Implantação de serviços
- Revisão de serviços

O exemplo de política a seguir contém as permissões necessárias e limita as ações a um serviço especificado.

Substitua `account`, `cluster-name` e `service-name` por seus próprios valores.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "ecs:ListServiceDeployments",
              "ecs:DescribeServiceDeployments",
              "ecs:DescribeServiceRevisions"
            ],
            "Resource": [
             "arn:aws:ecs:us-east-1:123456789012:service/cluster-name/service-name",
             "arn:aws:ecs:us-east-1:123456789012:service-deployment/cluster-name/
service-name/*",
             "arn:aws:ecs:us-east-1:123456789012:service-revision/cluster-name/service-
name/*"
            ]
        }
    ]
}
```

Permissões necessárias para visualizar os eventos de ciclo de vida do Amazon ECS no Container Insights

As permissões a seguir são necessárias para visualizar os eventos de ciclo de vida. Adicione as permissões a seguir como uma política em linha ao perfil. Para obter mais informações, consulte Adicionar e remover políticas do IAM.

- events:DescribeRule
- events:ListTargetsByRule
- logs:DescribeLogGroups

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:DescribeRule",
        "events:ListTargetsByRule",
        "logs:DescribeLogGroups"
      ],
      "Resource": "*"
    }
  ]
}
```

Permissões necessárias para habilitar os eventos de ciclo de vida do Amazon ECS no Container Insights

As seguintes permissões são necessárias para configurar os eventos de ciclo de vida:

- events:PutRule
- events:PutTargets
- logs:CreateLogGroup

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:PutRule",
        "events:PutTargets",
        "logs:CreateLogGroup"
      ],
      "Resource": "*"
    }
  ]
}
```

Permissões necessárias para usar o console do Amazon ECS com o AWS CloudFormation

Antes de usar o AWS Management Console para criar seus recursos, você precisará ter as permissões corretas do IAM. Para obter informações sobre como configurar primeiro as permissões para o console do Amazon ECS em geral, consulte Permissões necessárias para o console do Amazon ECS.

O console do Amazon ECS é desenvolvido pelo AWS CloudFormation e exige permissões adicionais do IAM nos seguintes casos:

- Criar um cluster
- Criar um serviço
- Criação de um provedor de capacidade

É possível criar uma política para as permissões adicionais e, em seguida, anexá-las ao perfil do IAM que você usa para acessar o console. Para obter mais informações, consulte Criar políticas do IAM no Guia do usuário do IAM.

Permissões necessárias para criar um cluster

Ao criar um cluster no console, você precisa de permissões adicionais que concedam permissões para gerenciar pilhas do AWS CloudFormation.

As permissões adicionais a seguir são necessárias:

- `cloudformation`: permite que as entidades principais criem e gerenciem pilhas do AWS CloudFormation. Isso é necessário na criação de clusters do Amazon ECS usando o AWS Management Console e no gerenciamento subsequente destes clusters.
- `ssm`: permite que o AWS CloudFormation faça referência à AMI otimizada para o Amazon ECS mais recente. Isso é necessário ao criar clusters do Amazon ECS utilizando o AWS Management Console.

A política a seguir contém as permissões AWS CloudFormation necessárias e limita as ações aos recursos criados no console do Amazon ECS.

```
{
  "Statement": [
      {
          "Effect": "Allow",
          "Action": [
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:DescribeStack*",
                "cloudformation:UpdateStack"
             ],
            "Resource": [
                "arn:*:cloudformation:*:*:stack/Infra-ECS-Cluster-*"
            ]
      },
      {
          "Effect": "Allow",
          "Action": "ssm:GetParameters",
          "Resource": [
            "arn:aws:ssm:*:*:parameter/aws/service/ecs/optimized-ami/amazon-linux-2*/
*",
            "arn:aws:ssm:*:*:parameter/aws/service/ecs/optimized-ami/amazon-
linux-2023*/*"
          ]
      }
```

```
    ]
}
```

Se você não criou o perfil de instância de contêiner do Amazon ECS (`ecsInstanceRole`) e está criando um cluster que usa instâncias do Amazon EC2, o console criará o perfil em seu nome.

Além disso, caso use grupos do Auto Scaling, são necessárias permissões adicionais para que o console possa adicionar tags aos grupos do Auto Scaling ao usar o recurso de ajuste de escala automático de cluster.

As permissões adicionais a seguir são necessárias:

- `autoscaling`: permite que o console atribua tags ao grupo do Amazon EC2 Auto Scaling. Isso é necessário no gerenciamento de grupos do Amazon EC2 Auto Scaling quando for usado o recurso de escalabilidade automática do cluster. A tag é a gerenciada pelo ECS que o console adiciona automaticamente ao grupo para indicar que foi criada no console.
- `iam`: permite que as entidades principais listem funções do IAM e suas políticas anexadas. As entidades principais também podem listar perfis da instância disponíveis para as instâncias do Amazon EC2.

A política a seguir contém as permissões do IAM necessárias e limita as ações ao perfil `ecsInstanceRole`.

As permissões do Auto Scaling não são limitadas.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:CreateInstanceProfile",
              "iam:AddRoleToInstanceProfile",
              "iam:ListInstanceProfilesForRole",
              "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/ecsInstanceRole"
        },
        {
            "Effect": "Allow",
```

```
            "Action": "autoscaling:CreateOrUpdateTags",
            "Resource": "*"
        }
    ]
}
```

Permissões necessárias para criar um serviço

Ao criar um serviço no console, você precisa de permissões adicionais que concedam permissões para gerenciar pilhas do AWS CloudFormation. As permissões adicionais a seguir são necessárias:

- `cloudformation`: permite que as entidades principais criem e gerenciem pilhas do AWS CloudFormation. Isso é necessário na criação de serviços do Amazon ECS usando o AWS Management Console e no gerenciamento subsequente destes serviços.

A política a seguir contém as permissões necessárias e limita as ações aos recursos criados no console do Amazon ECS.

```
{
  "Statement": [
      {
          "Effect": "Allow",
          "Action": [
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:DescribeStack*",
                "cloudformation:UpdateStack"
            ],
            "Resource": [
                "arn:*:cloudformation:*:*:stack/ECS-Console-V2-Service-*"
            ]
      }
   ]
}
```

Permissões necessárias para funções do Lambda em implantações azul/verde do Amazon ECS

Ao usar funções do Lambda como ganchos do ciclo de vida de implantação em implantações azul/verde do Amazon ECS, você precisa criar um perfil do IAM com permissões específicas. Esse perfil

permite que o Amazon ECS invoque suas funções do Lambda em vários estágios do ciclo de vida das implantações.

As permissões adicionais a seguir são necessárias:

- `lambda:InvokeFunction`: permite que o Amazon ECS invoque funções do Lambda configuradas como ganchos do ciclo de vida durante o processo de implantação.

A seguinte política contém as permissões necessárias para os ganchos do ciclo de vida do Lambda:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction"
            ],
            "Resource": "arn:aws:lambda:*:*:function:*"
        }
    ]
}
```

Na política de confiança, você precisa permitir que o serviço assuma este perfil:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ecs.amazonaws.com"
                ]
            },
```

```
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Permissões obrigatórias do IAM para o ajuste de escala automático do serviço Amazon ECS

O Auto Scaling do serviço se torna possível por uma combinação das APIs do Amazon ECS, do CloudWatch e do Application Auto Scaling. Os serviços são criados e atualizados com o Amazon ECS, são criados alarmes com o CloudWatch e são criadas políticas de escalabilidade com o Application Auto Scaling.

Além das permissões padrão do IAM para criar e atualizar serviços, as permissões a seguir são necessárias para interagir com as configurações do ajuste de escala automático do serviço, conforme mostrado no exemplo de política a seguir.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:*",
                "ecs:DescribeServices",
                "ecs:UpdateService",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DeleteAlarms",
                "cloudwatch:DescribeAlarmHistory",
                "cloudwatch:DescribeAlarmsForMetric",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics",
                "cloudwatch:DisableAlarmActions",
                "cloudwatch:EnableAlarmActions",
                "iam:CreateServiceLinkedRole",
                "sns:CreateTopic",
                "sns:Subscribe",
                "sns:Get*",
                "sns:List*"
            ],
```

```
            "Resource": ["*"]
        }
    ]
}
```

Os exemplos Exemplo de criação de serviço do Amazon ECS e Exemplo de serviço de atualização do Amazon ECS de política do IAM mostram as permissões necessárias para o uso do Auto Scaling do serviço no AWS Management Console.

O serviço do Application Auto Scaling também precisa de permissão para descrever os serviços do Amazon ECS e os alarmes do CloudWatch, além de permissões para modificar a contagem desejada do serviço em seu nome. As permissões `sns:` são para as notificações que o CloudWatch envia para um tópico do Amazon SNS quando um limite for excedido. Se você usar a escalabilidade automática para os serviços do Amazon ECS, será criada uma função vinculada ao serviço denominada `AWSServiceRoleForApplicationAutoScaling_ECSService`. Essa função vinculada ao serviço concede permissão ao Application Auto Scaling para descrever os alarmes das políticas, monitorar a contagem atual de tarefas em execução do serviço e modificar a contagem desejada do serviço. A função gerenciada original do Amazon ECS para o Application Auto Scaling era `ecsAutoscaleRole`, mas ela não é mais necessária. Essa função vinculada ao serviço é a função padrão do Application Auto Scaling. Para ter mais informações, consulte Funções vinculadas a serviço do Application Auto Scaling, no Guia do usuário do Application Auto Scaling.

Se você tiver criado do perfil da instância de contêiner do Amazon ECS antes que as métricas do CloudWatch estejam disponíveis para o Amazon ECS, talvez seja necessário adicionar a permissão `ecs:StartTelemetrySession`. Para obter mais informações, consulte Considerações.

Conceder permissão para a atribuição de tags a recursos durante a criação

As ações de criação com tags de API do Amazon ECS a seguir permitem que você especifique tags ao criar o recurso. Se as tags forem especificadas na ação resource-creating, a AWS executará autorização adicional para verificar se as permissões corretas foram atribuídas para a criação de tags.

- `CreateCapacityProvider`
- `CreateCluster`
- `CreateService`
- `CreateTaskSet`
- `RegisterContainerInstance`

- `RegisterTaskDefinition`
- `RunTask`
- `StartTask`

É possível usar tags de recursos para implementar o controle baseado em atributo (ABAC). Para obter mais informações, consulte the section called “Controlar o acesso aos recursos do Amazon ECS usando tags de recursos” e Marcar recursos.

Para permitir a atribuição de tags na criação, crie ou modifique uma política para incluir as permissões para usar a ação que cria o recurso, como `ecs:CreateCluster` ou `ecs:RunTask` e a ação `ecs:TagResource`.

O exemplo a seguir demonstra uma política que permite aos usuários criar clusters e adicionar tags durante a criação do cluster. Os usuários não têm permissão para marcar recursos existentes (não podem chamar a ação `ecs:TagResource` diretamente).

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ecs:CreateCluster"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ecs:TagResource"
      ],
      "Resource": "*",
      "Condition": {
         "StringEquals": {
                  "ecs:CreateAction": [
                      "CreateCluster",
                      "CreateCapacityProvider",
                      "CreateService",
                      "CreateTaskSet",
                      "RegisterContainerInstance",
                      "RegisterTaskDefinition",
                      "RunTask",
```

```
                    "StartTask"
                ]
            }
        }
    }
  ]
}
```

A ação `ecs:TagResource` será avaliada somente se as tags forem aplicadas durante a ação resource-creating. Portanto, um usuário que tiver permissões para criar um recurso (pressupondo-se que não existam condições de marcação) não precisa de permissão para usar a ação `ecs:TagResource` se nenhuma tag for especificada na solicitação. Contudo, se o usuário tentar criar um recurso com tags, haverá falha na solicitação se o usuário não tiver permissão para usar a ação `ecs:TagResource`.

Controle do acesso a etiquetas específicas do Amazon ECS

É possível usar condições adicionais no elemento `Condition` de suas políticas do IAM para controlar as chaves de tag e os valores que podem ser aplicados aos recursos.

As chaves de condição a seguir podem ser usadas com os exemplos na seção anterior:

- `aws:RequestTag`: para indicar que uma chave de tag ou uma chave e um valor de tag específicos devem estar presentes em uma solicitação. Outras tags também podem ser especificadas na solicitação.
  - Use com o operador de condição `StringEquals` para impor uma combinação de chave e valor de tag específica, por exemplo, para impor a tag `cost-center=cc123`:

    ```
    "StringEquals": { "aws:RequestTag/cost-center": "cc123" }
    ```

  - Use com o operador de condição `StringLike` para impor uma chave de tag específica, por exemplo, para impor a chave de tag `purpose`:

    ```
    "StringLike": { "aws:RequestTag/purpose": "*" }
    ```

- `aws:TagKeys`: para aplicar as chaves de tags usadas na solicitação.
  - Use com o modificador `ForAllValues` para impor chaves de tags específicas se forem fornecidas na solicitação (se as tags forem especificadas na solicitação, somente chaves de tags específicas são permitidas; nenhuma outra tag é permitida). Por exemplo, as chaves de tags `environment` ou `cost-center` são permitidas:

```
"ForAllValues:StringEquals": { "aws:TagKeys": ["environment","cost-center"] }
```

- Use com o modificador `ForAnyValue` para impor a presença de pelo menos uma das chaves de tags especificadas na solicitação. Por exemplo, pelo menos uma das chaves de tags `environment` ou `webserver` deve estar presente na solicitação:

```
"ForAnyValue:StringEquals": { "aws:TagKeys": ["environment","webserver"] }
```

Essas chaves de condição podem ser aplicadas às ações resource-creating que são oferecem suporte à atribuição de tags, bem como a ação `ecs:TagResource`. Para saber se uma ação de API do Amazon ECS oferece suporte à atribuição de tags, consulte Ações, recursos e chaves de condição para o Amazon ECS.

Para forçar os usuários a especificarem tags quando criam um recurso, use a chave de condição `aws:RequestTag` ou a chave de condição `aws:TagKeys` com o modificador `ForAnyValue` na ação resource-creating. A ação `ecs:TagResource` não será avaliada se um usuário não especificar tags para a ação resource-creating.

Para condições, a chave de condição não diferencia maiúsculas de minúsculas, e o valor da condição diferencia maiúsculas de minúsculas. Portanto, para aplicar a diferenciação de maiúsculas de minúsculas de uma tag, use a chave de condição `aws:TagKeys`, onde a chave da tag é especificada como um valor na condição.

Para obter mais informações sobre condições de vários valores, consulte Condições com várias chaves de contexto ou valores no Guia do usuário do IAM.

Controlar o acesso aos recursos do Amazon ECS usando tags de recursos

Ao criar uma política do IAM que conceda permissão aos usuários para usar recursos do Amazon ECS, é possível incluir informações de tag no elemento `Condition` da política para controlar o acesso com base em tags. Isso é conhecido como controle de acesso baseado em atributo (ABAC). O ABAC oferece um controle melhor sobre quais recursos um usuário pode modificar, usar ou excluir. Para obter mais informações, consulte O que é ABAC para a AWS?

Por exemplo, é possível criar uma política que permite que os usuários excluam um cluster, mas nega a ação se o cluster tiver a tag `environment=production`. Para fazer isso, use a chave de condição `aws:ResourceTag` para permitir ou negar acesso ao recurso com base nas tags anexadas ao recurso.

```
"StringEquals": { "aws:ResourceTag/environment": "production" }
```

Para saber se uma ação de API do Amazon ECS oferece suporte ao controle de acesso usando a chave de condição `aws:ResourceTag`, consulte Ações, recursos e chaves de condição para Amazon ECS . Como as ações de `Describe` não oferecem suporte a permissões em nível de recurso, especifique-as em uma declaração separada sem condições.

Para obter exemplos de políticas do IAM, consulte Exemplos de políticas do Amazon ECS .

Se você permitir ou negar aos usuários o acesso a recursos com base em tags, considere negar explicitamente aos usuários a capacidade de adicionar essas tags ou removê-las dos mesmos recursos. Caso contrário, é possível que um usuário contorne suas restrições e obtenha acesso a um recurso modificando as tags.

Exemplos de políticas do Amazon ECS

É possível usar as políticas do IAM para conceder permissões aos usuários para visualizarem e trabalharem com recursos específicos no console do Amazon ECS. É possível usar os exemplos de políticas da seção anterior. No entanto, eles foram criados para solicitações feitas com a AWS CLI ou com um AWS SDK.

Exemplo: permitir que os usuários excluam um cluster do Amazon ECS com base em etiquetas

A política a seguir permite que os usuários excluam clusters quando a tag tiver um par chave/valor de “Propósito/Teste”.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ecs:DeleteCluster"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:ecs:us-east-1:111122223333:cluster/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Purpose": "Testing"
```

```
                }
            }
        }
    ]
}
```

Solução de problemas de identidade e acesso do Amazon Elastic Container Service

Use as informações a seguir para ajudar a diagnosticar e corrigir problemas comuns que você pode encontrar ao trabalhar com o Amazon ECS e o IAM.

Tópicos

- Não tenho autorização para executar uma ação no Amazon ECS
- Não estou autorizado a executar iam:PassRole
- Quero permitir que as pessoas fora da minha Conta da AWS acessem meus recursos do Amazon ECS
- Recursos adicionais para solução de problemas

Não tenho autorização para executar uma ação no Amazon ECS

Se você receber uma mensagem de erro informando que não tem autorização para executar uma ação, suas políticas deverão ser atualizadas para permitir que você realize a ação.

O erro do exemplo a seguir ocorre quando o usuário do IAM `mateojackson` tenta usar o console para visualizar detalhes sobre um atributo *`my-example-widget`* fictício, mas não tem as permissões `ecs:`*`GetWidget`* fictícias.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 ecs:GetWidget on resource: my-example-widget
```

Nesse caso, a política do usuário `mateojackson` deve ser atualizada para permitir o acesso ao recurso *`my-example-widget`* usando a ação `ecs:`*`GetWidget`*.

Se você precisar de ajuda, entre em contato com seu administrador AWS. Seu administrador é a pessoa que forneceu suas credenciais de login.

Não estou autorizado a executar iam:PassRole

Se receber uma mensagem de erro informando que você não tem autorização para executar a ação `iam:PassRole`, suas políticas devem ser atualizadas para permitir a transmissão de um perfil ao Amazon ECS.

Alguns Serviços da AWS permitem que você passe um perfil existente para o serviço, em vez de criar um perfil de serviço ou perfil vinculado ao serviço. Para fazer isso, é preciso ter permissões para passar o perfil para o serviço.

O exemplo de erro a seguir ocorre quando uma usuária do IAM chamada de `marymajor` tenta usar o console para executar uma ação no Amazon ECS. No entanto, a ação exige que o serviço tenha permissões concedidas por um perfil de serviço. Mary não tem permissões para passar o perfil para o serviço.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Nesse caso, as políticas de Mary devem ser atualizadas para permitir que ela realize a ação `iam:PassRole`.

Se você precisar de ajuda, entre em contato com seu administrador AWS. Seu administrador é a pessoa que forneceu suas credenciais de login.

Quero permitir que as pessoas fora da minha Conta da AWS acessem meus recursos do Amazon ECS

Você pode criar um perfil que os usuários de outras contas ou pessoas fora da organização podem usar para acessar seus recursos. É possível especificar quem é confiável para assumir o perfil. Para serviços que oferecem compatibilidade com políticas baseadas em recursos ou listas de controle de acesso (ACLs), você pode usar essas políticas para conceder às pessoas acesso aos seus recursos.

Para saber mais, consulte:

- Para saber se o Amazon ECS oferece suporte a esses recursos, consulte Como o Amazon Elastic Container Service funciona com o IAM.
- Para saber como conceder acesso a seus recursos em todas as Contas da AWS pertencentes a você, consulte Fornecimento de acesso a um usuário do IAM em outra Conta da AWS pertencente a você no Guia de usuário do IAM.

- Para saber como conceder acesso a seus recursos para Contas da AWS de terceiros, consulte Fornecimento de acesso a Contas da AWS pertencentes a terceiros no Guia do usuário do IAM.
- Para saber como conceder acesso por meio da federação de identidades, consulte Conceder acesso a usuários autenticados externamente (federação de identidades) no Guia do usuário do IAM.
- Para saber a diferença entre perfis e políticas baseadas em recurso para acesso entre contas, consulte Acesso a recursos entre contas no IAM no Guia do usuário do IAM.

Recursos adicionais para solução de problemas

As seguintes páginas fornecem informações sobre códigos de erro:

- Mensagens de erro de tarefa interrompida do Amazon ECS
- Visualizar mensagens de eventos de serviço do Amazon ECS

Práticas recomendadas do IAM para o Amazon ECS

É possível usar o AWS Identity and Access Management (IAM) para gerenciar e controlar o acesso aos seus serviços e recursos da AWS por meio de políticas baseadas em regras para fins de autenticação e autorização. Mais especificamente, por meio desse serviço, você controla o acesso aos seus recursos da AWS usando políticas aplicadas a usuários, grupos ou funções. Entre esses três, os usuários são contas que podem ter acesso aos seus recursos. Além disso, um perfil do IAM é um conjunto de permissões que podem ser assumidas por uma identidade autenticada, que não está associada a uma identidade específica fora do IAM. Para obter mais informações, consulte Amazon ECS overview of access management: Permissions and policies.

Siga a política de acesso de privilégio mínimo

Crie políticas com escopo para permitir que os usuários realizem seus trabalhos prescritos. Por exemplo, se um desenvolvedor precisar interromper periodicamente uma tarefa, crie uma política que permita apenas essa ação específica. O exemplo a seguir permite que um usuário interrompa uma tarefa que pertence a uma determinada `task_family` em um cluster com um nome do recurso da Amazon (ARN) específico. Referir-se a um ARN em uma condição também é um exemplo de uso de permissões em nível de recurso. É possível usar permissões em nível de recurso para especificar o recurso ao qual você deseja aplicar uma ação.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:StopTask"
            ],
            "Condition": {
                "ArnEquals": {
                       "ecs:cluster": "arn:aws:ecs:us-
east-1:111122223333:cluster/cluster_name"
                }
            },
            "Resource": [
                   "arn:aws:ecs:us-east-1:111122223333:task-
definition/task_family:*"
            ]
        }
    ]
}
```

Fazer com que os recursos de cluster atuem como limite administrativo

Políticas com escopo muito restrito podem causar uma proliferação de funções e aumentar a sobrecarga administrativa. Em vez de criar funções que tenham como escopo somente tarefas ou serviços específicos, crie funções que tenham como escopo os clusters e use o cluster como seu limite administrativo principal.

Criar pipelines automatizados para isolar os usuários finais da API

É possível limitar as ações que os usuários podem usar criando pipelines que empacotem e implantam automaticamente aplicações nos clusters do Amazon ECS. Isso delega efetivamente o trabalho de criar, atualizar e excluir tarefas ao pipeline. Para obter mais informações, consulte o Tutorial: implantação padrão do Amazon ECS com o CodePipeline no Guia do usuário do AWS CodePipeline.

Usar condições de política para um nível de segurança adicional

Quando precisar de uma camada adicional de segurança, adicione uma condição à sua política. Isso pode ser útil se você estiver executando uma operação privilegiada ou quando precisar restringir o conjunto de ações que podem ser executadas em determinados recursos. O exemplo de política a seguir exige autorização multifatorial ao excluir um cluster.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:DeleteCluster"
      ],
      "Condition": {
        "Bool": {
          "aws:MultiFactorAuthPresent": "true"
        }
      },
    "Resource": ["*"]
    }
  ]
}
```

As tags aplicadas aos serviços são propagadas para todas as tarefas que fazem parte desse serviço. Por isso, é possível criar funções que tenham como escopo os recursos do Amazon ECS com tags específicas. Na política a seguir, uma entidade principal do IAM inicia e interrompe todas as tarefas com uma chave de tag de `Department` e um valor de tag de `Accounting`.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:RunTask"
            ],
            "Resource": "arn:aws:ecs:arn:aws::111122223333:task/my-cluster/*",
            "Condition": {
                "StringEquals": {"ecs:ResourceTag/Department": "Accounting"}
            }
        }
    ]
}
```

Auditoria periódica do acesso às APIs

Um usuário pode mudar de perfil. Depois que eles mudarem de perfil, as permissões que lhes foram concedidas anteriormente podem não se aplicar mais. Certifique-se de auditar quem tem acesso às APIs do Amazon ECS e se esse acesso ainda é garantido. Considere integrar o IAM a uma solução de gerenciamento do ciclo de vida do usuário que revogue automaticamente o acesso quando um usuário deixar a organização. Para obter mais informações, consulte Diretrizes de auditoria de segurança da AWS no Guia do usuário do AWS Identity and Access Management.

Registrar e monitorar no Amazon Elastic Container Service

O monitoramento é uma parte importante para manter a confiabilidade, a disponibilidade e a performance do Amazon Elastic Container Service e das soluções da AWS. Você deve coletar dados de monitoramento de todas as partes da solução da AWS para depurar uma falha em vários pontos com mais facilidade, caso ocorra. A AWS fornece várias ferramentas para monitorar os recursos do Amazon ECS e responder a incidentes potenciais:

Alarmes do Amazon CloudWatch

> Observe uma única métrica ao longo de um período que você especificar e realize uma ou mais ações com base no valor da métrica em relação a um determinado limite ao longo de vários períodos. A ação é uma notificação enviada para um tópico do Amazon Simple Notification Service (Amazon SNS) ou uma política do Amazon EC2 Auto Scaling. Os alarmes do CloudWatch não invocam ações simplesmente por estarem em um estado específico. O estado deve ter sido alterado e mantido por um número específico de períodos. Para obter mais informações, consulte Monitoramento do Amazon ECS usando o CloudWatch.

Para serviços com tarefas que usam o tipo de inicialização do Fargate, você pode usar os alarmes do CloudWatch para expandir e reduzir as tarefas no serviço, com base em métricas do CloudWatch, como a utilização da CPU e da memória. Para obter mais informações, consulte Como escalar automaticamente o serviço do Amazon ECS.

Para clusters com tarefas ou serviços que usam o tipo de inicialização do EC2, você pode usar os alarmes do CloudWatch para expandir e reduzir as instâncias de contêiner com base em métricas do CloudWatch, como a reserva de memória do cluster.

Amazon CloudWatch Logs

Monitore, armazene e acesse os arquivos de log nos contêineres das tarefas do Amazon ECS especificando o driver de log `awslogs` nas definições de tarefa. Para obter mais informações, consulte Uso do driver awslogs.

Você também pode monitorar, armazenar e acessar os arquivos de log do sistema operacional e do agente de contêiner do Amazon ECS nas instâncias de contêiner do Amazon ECS. Este método para acessar logs pode ser usado para contêineres que usem o tipo de inicialização do EC2.

Amazon CloudWatch Events

Faça correspondência de eventos e direcione-os a uma ou mais funções ou fluxos de destino para fazer alterações, capturar informações de estado e realizar ações corretivas. Para obter mais informações, consulte Automatização de respostas a erros do Amazon ECS usando o EventBridge neste guia e O EventBridge é a evolução do Amazon CloudWatch Events no Guia do usuário Amazon EventBridge.

Logs do AWS CloudTrail

O CloudTrail fornece um registro de ações executadas por um usuário, uma função ou um serviço da AWS no Amazon ECS. Usando as informações coletadas pelo CloudTrail, é possível determinar a solicitação feita ap Amazon ECS, o endereço IP do qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita e detalhes adicionais. Para obter mais informações, consulte Registro em log das chamadas de API do Amazon ECS usando o AWS CloudTrail.

AWS Trusted Advisor

O Trusted Advisor conta com as práticas recomendadas aprendidas com o atendimento a centenas de milhões de clientes da AWS. O Trusted Advisor inspeciona seu ambiente da AWS e faz recomendações quando há oportunidades para economizar dinheiro, melhorar a performance e a disponibilidade do sistema e ajuda a corrigir falhas de segurança. Todos os clientes da

AWS têm acesso a cinco verificações do Trusted Advisor. Os clientes com um plano de suporte Business ou Enterprise podem ver todas as verificações do Trusted Advisor.

Consulte mais informações em AWS Trusted Advisor no Guia de Usuário Suporte.

AWS Compute Optimizer

O AWS Compute Optimizer é um serviço que analisa as métricas de configuração e utilização dos seus recursos da AWS. O serviço informa se seus recursos estão em condições ideais e gera recomendações de otimização para reduzir o custo e melhorar a performance de suas workloads.

Para obter mais informações, consulte Recomendações do AWS Compute Optimizer para o Amazon ECS.

Outra parte importante do monitoramento do Amazon ECS envolve o monitoramento manual dos itens que os alarmes do CloudWatch não abrangem. O CloudWatch, o Trusted Advisor e outros painéis de console da AWS apresentam uma visão rápida do estado do seu ambiente na AWS. Recomendamos que você também verifique os arquivos de log nas instâncias de contêiner e os contêineres nas tarefas.

Validação da conformidade do Amazon Elastic Container Service

Para saber se um AWS service (Serviço da AWS) está no escopo de programas de conformidade específicos, consulte Serviços da AWS no escopo por programa de conformidade e selecione o programa de conformidade em que você está interessado. Para obter informações gerais, consulte Programas de Conformidade da AWS.

É possível baixar relatórios de auditoria de terceiros usando o AWS Artifact. Para obter mais informações, consulte Baixar relatórios no AWS Artifact.

Sua responsabilidade de conformidade ao usar o Serviços da AWS é determinada pela confidencialidade dos seus dados, pelos objetivos de conformidade da sua empresa e pelos regulamentos e leis aplicáveis. A AWS fornece os recursos a seguir para ajudar com a conformidade:

- Governança e conformidade de segurança: esses guias de implementação de solução abordam considerações sobre a arquitetura e fornecem etapas para implantar recursos de segurança e conformidade.
- Referência de serviços qualificados para HIPAA: lista os serviços qualificados para HIPAA. Nem todos os Serviços da AWS estão qualificados pela HIPAA.

- Recursos de Conformidade da AWS: essa coleção de manuais e guias pode ser aplicada ao seu setor e local.
- Guias de conformidade do cliente da AWS: entenda o modelo de responsabilidade compartilhada sob a ótica da conformidade. Os guias resumem as práticas recomendadas para proteção de Serviços da AWS e mapeiam as diretrizes para controles de segurança em várias estruturas (incluindo o Instituto Nacional de Padrões e Tecnologia (NIST), o Conselho de Padrões de Segurança do Setor de Cartões de Pagamento (PCI) e a Organização Internacional de Padronização (ISO)).
- Avaliar recursos com regras no Guia do desenvolvedor do AWS Config: o serviço AWS Config avalia como as configurações de recursos estão em conformidade com práticas internas, diretrizes do setor e regulamentos.
- AWS Security Hub: este AWS service (Serviço da AWS) fornece uma visão abrangente do seu estado de segurança na AWS. O Security Hub usa controles de segurança para avaliar os recursos da AWS e verificar a conformidade com os padrões e as práticas recomendadas do setor de segurança. Para obter uma lista dos serviços e controles aceitos, consulte a Referência de controles do Security Hub.
- Amazon GuardDuty: este AWS service (Serviço da AWS) detecta possíveis ameaças às suas Contas da AWS, workloads, contêineres e dados ao monitorar o ambiente em busca de atividades suspeitas e maliciosas. O GuardDuty pode ajudar você a atender a diversos requisitos de conformidade, como o PCI DSS, com o cumprimento dos requisitos de detecção de intrusões requeridos por determinadas estruturas de conformidade.
- AWS Audit Manager: este AWS service (Serviço da AWS) ajuda você a auditar continuamente o seu uso da AWS para simplificar o modo como você gerencia os riscos e a conformidade com regulamentos e padrões do setor.

Práticas recomendadas de conformidade e de segurança para o Amazon ECS

Sua responsabilidade pela conformidade ao usar o Amazon ECS é determinada pela confidencialidade dos seus dados, pelos objetivos de conformidade da sua empresa e pelas leis e regulamentações aplicáveis.

Padrões de segurança de dados do setor de cartão de pagamento (PCI DSS)

É importante que você entenda o fluxo completo de dados do titular do cartão (CHD) no ambiente ao aderir ao PCI DSS. O fluxo de CHD determina a aplicabilidade do PCI DSS, define os limites

e os componentes de um ambiente de dados do titular do cartão (CDE) e, portanto, o escopo de uma avaliação do PCI DSS. A determinação precisa do escopo do PCI DSS é fundamental para definir a postura de segurança e, por fim, uma avaliação bem-sucedida. Os clientes devem ter um procedimento para determinação do escopo que garanta sua integridade e detecte alterações ou desvios do escopo.

A natureza temporária das aplicações em contêineres fornece complexidades adicionais à auditoria de configurações. Como resultado, os clientes precisam estar cientes de todos os parâmetros de configuração do contêiner para garantir que os requisitos de conformidade sejam atendidos em todas as fases do ciclo de vida do contêiner.

Para obter informações adicionais sobre como alcançar a compatibilidade com o PCI DSS no Amazon ECS, consulte os whitepapers a seguir.

- Arquitetura no Amazon ECS para conformidade com o PCI DSS

HIPAA (Lei de Portabilidade e Responsabilidade de Provedores de Saúde dos EUA)

Usar o Amazon ECS com workloads que processem informações de saúde protegidas (PHI) não requer configuração adicional. O Amazon ECS atua como um serviço de orquestração que coordena o início de contêineres no Amazon EC2. Ele não opera com ou sobre dados dentro da workload que está sendo orquestrada. Consistente com as regulamentações da HIPAA e com o Adendo de Associado Comercial da AWS, as PHI devem ser criptografadas em trânsito e em repouso quando acessadas por contêineres iniciados pelo Amazon ECS.

Vários mecanismos para criptografia em repouso estão disponíveis com cada opção de armazenamento da AWS, como o Amazon S3o , Amazon EBS e o AWS KMS. É possível implantar uma rede de sobreposição (como VNS3 ou Weave Net) para garantir a criptografia completa das PHI transferidas entre contêineres ou para fornecer uma camada redundante de criptografia. Recomenda-se também usar o log completo e direcionar todos os logs de contêiner para o Amazon CloudWatch. Para obter informações sobre o uso das práticas recomendadas de segurança da infraestrutura, consulte Proteção da infraestrutura em Pilar de segurança: AWS Well-Architected Framework.

AWS Security Hub

Use AWS Security Hub. Este AWS service (Serviço da AWS) fornece uma visão abrangente do seu estado de segurança na AWS. O Security Hub usa controles de segurança para avaliar os

recursos da AWS e verificar a conformidade com os padrões e as práticas recomendadas do setor de segurança. Para obter uma lista dos serviços e controles aceitos, consulte a Referência de controles do Security Hub.

Amazon GuardDuty com monitoramento de runtime para o Amazon ECS

O Amazon GuardDuty é um serviço de detecção de ameaças que ajuda a proteger contas, contêineres, workloads e dados no ambiente da AWS. Usando modelos de machine learning (ML) e recursos de detecção de anomalias e ameaças, o GuardDuty monitora continuamente diferentes fontes de log e atividades de runtime para identificar e priorizar possíveis riscos de segurança e atividades maliciosas no seu ambiente.

Use o monitoramento de runtime no GuardDuty para identificar comportamentos maliciosos ou não autorizados. O monitoramento de runtime protege as workloads em execução no Fargate e no EC2 monitorando continuamente as atividades de log e rede da AWS para identificar comportamentos maliciosos ou não autorizados. O monitoramento de runtime usa um agente de segurança do GuardDuty leve e totalmente gerenciado que analisa o comportamento no host, como acesso a arquivos, execução de processos e conexões de rede. Isso inclui problemas como escalação de privilégios, uso de credenciais expostas, comunicação com endereços IP ou domínios maliciosos e a presença de malware nas instâncias e workloads de contêiner do Amazon EC2. Para obter mais informações, consulte GuardDuty Runtime Monitoring no Guia do usuário do GuardDuty.

Recomendações sobre conformidade

Recomenda-se envolver os proprietários do programa de conformidade em sua empresa desde o início e usar o modelo de responsabilidade compartilhada da AWS para identificar a propriedade do controle de conformidade para obter sucesso com os programas de conformidade relevantes. Para obter mais informações, consulte Modelo de responsabilidade compartilhada da AWS para o Amazon ECS .

Padrão Federal de Processamento de Informações (FIPS-140) do AWS Fargate

Padrão Federal de Processamento de Informações (FIPS) O FIPS-140 é um padrão do governo americano e canadense que especifica os requisitos de segurança para módulos criptográficos que protegem informações sigilosas. O FIPS-140 define um conjunto de funções criptográficas validadas que podem ser usadas para criptografar dados em trânsito e dados em repouso.

Ao ativar a conformidade com o FIPS-140, é possível executar workloads no Fargate de forma compatível com o FIPS-140. Para obter mais informações sobre a conformidade com o FIPS-140, consulte Padrão Federal de Processamento de Informações (FIPS) 140-3.

Considerações sobre o FIPS-140 do AWS Fargate

Considere o seguinte ao usar a conformidade com FIPS-140 no Fargate:

- A conformidade com o FIPS-140 está disponível somente nas regiões AWS GovCloud (US).
- O Fargate oferece suporte ao FIPS-140 versão 140.3
- A conformidade com FIPS-140 está desativada por padrão. Você deve ativá-la.
- Suas tarefas devem usar a configuração a seguir para conformidade com o FIPS-140:
  - O `operatingSystemFamily` deve ser `LINUX`.
  - O `cpuArchitecture` deve ser `X86_64`.
  - A versão da plataforma Fargate deve ser `1.4.0` ou posterior.

Use FIPS no Fargate

Use o procedimento a seguir para usar a conformidade com FIPS-140 no Fargate.

1. Ative a conformidade com o FIPS-140. Para obter mais informações, consulte the section called “Conformidade do AWS Fargate com o Padrão Federal de Processamento de Informações (FIPS-140)”.
2. Opcionalmente, é possível usar o ECS Exec para executar o comando a seguir para verificar o status de conformidade com o FIPS-140 de um cluster.

   Substitua *`cluster-name`* pelo nome do cluster, *`task-id`* pelo ID ou o ARN da tarefa e *`container-name`* pelo nome do contêiner da tarefa para a qual você deseja executar o comando.

   Um valor de retorno de “1" indica que você está usando FIPS.

   ```
   aws ecs execute-command \
       --cluster cluster-name \
       --task task-id \
       --container container-name \
       --interactive \
   ```

```
    --command "cat /proc/sys/crypto/fips_enabled"
```

Uso do CloudTrail para auditoria do FIPS-140 do Fargate

O CloudTrail é ativado na conta da AWS quando ela é criada. Quando ocorre uma atividade de API e console no Amazon ECS, esta atividade é registrada em um evento do CloudTrail junto com outros eventos de serviço da AWS no Histórico de eventos. É possível visualizar, pesquisar e baixar eventos recentes em sua conta da AWS. Para obter mais informações, consulte Visualização de eventos com o histórico de eventos do CloudTrail.

Para obter um registro contínuo de eventos na sua conta da AWS, incluindo eventos do Amazon ECS, crie uma trilha que o CloudTrail use para entregar arquivos de log para um bucket do Amazon S3. Por padrão, ao criar uma trilha no console, ela é aplicada a todas as regiões da . A trilha registra logs de eventos de todas as Regiões na AWS divisória e entrega os arquivos do log para o bucket Amazon S3 especificado. Além disso, é possível configurar outros AWS serviços para melhor analisar e agir de acordo com dados coletados do evento nos logs CloudTrail. Para obter mais informações, consulte the section called “Registro em log das chamadas de API do Amazon ECS usando o AWS CloudTrail”.

O exemplo a seguir mostra uma entrada de log do CloudTrail que demonstra a ação de API `PutAccountSettingDefault`:

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAIV5AJI5LXF5EXAMPLE",
         "arn": "arn:aws:iam::123456789012:user/jdoe",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIPWIOFC3EXAMPLE",
    },
    "eventTime": "2023-03-01T21:45:18Z",
    "eventSource": "ecs.amazonaws.com",
    "eventName": "PutAccountSettingDefault",
    "awsRegion": "us-gov-east-1",
    "sourceIPAddress": "52.94.133.131",
    "userAgent": "aws-cli/2.9.8 Python/3.9.11 Windows/10 exe/AMD64 prompt/off command/
ecs.put-account-setting",
    "requestParameters": {
        "name": "fargateFIPSMode",
```

```
        "value": "enabled"
    },
    "responseElements": {
        "setting": {
            "name": "fargateFIPSMode",
            "value": "enabled",
            "principalArn": "arn:aws:iam::123456789012:user/jdoe"
        }
    },
    "requestID": "acdc731e-e506-447c-965d-f5f75EXAMPLE",
    "eventID": "6afced68-75cd-4d44-8076-0beEXAMPLE",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "ecs-fips.us-gov-east-1.amazonaws.com"
    }
}
```


Segurança da infraestrutura no Amazon Elastic Container Service

Como um serviço gerenciado, o Amazon Elastic Conteiner Service é protegido pela segurança de rede global da AWS. Para obter informações sobre serviços de segurança da AWS e como a AWS protege a infraestrutura, consulte AWS Segurança na Nuvem. Para projetar seu ambiente da AWS usando as práticas recomendadas de segurança da infraestrutura, consulte Proteção de Infraestrutura em Pilar de Segurança: AWS Estrutura bem arquitetada.

Você usa chamadas de API publicadas pela AWS para acessar o Amazon ECS por meio da rede. Os clientes devem oferecer compatibilidade com:

- Transport Layer Security (TLS). Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Conjuntos de criptografia com perfect forward secrecy (PFS) como DHE (Ephemeral Diffie-Hellman) ou ECDHE (Ephemeral Elliptic Curve Diffie-Hellman). A maioria dos sistemas modernos, como Java 7 e versões posteriores, comporta esses modos.

Além disso, as solicitações devem ser assinadas usando um ID da chave de acesso e uma chave de acesso secreta associada a uma entidade principal do IAM. Ou é possível usar o AWS Security Token Service (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.

É possível chamar essas operações de API de qualquer local de rede. O Amazon ECS oferece suporte a políticas de acesso com base em recursos, que podem incluir restrições com base no endereço IP de origem. Portanto, certifique-se de que as políticas sejam responsáveis pelo endereço IP da localização da rede. Também é possível usar políticas do Amazon ECS para controlar o acesso de endpoints da Amazon Virtual Private Cloud ou de VPCs específicas. Realmente, isso isola o acesso à rede para um determinado recurso do Amazon ECS apenas da VPC específica dentro da rede da AWS. Para obter mais informações, consulte Endpoints da VPC de interface do Amazon ECS (AWS PrivateLink).

Endpoints da VPC de interface do Amazon ECS (AWS PrivateLink)

É possível melhorar o procedimento de segurança da VPC configurando o Amazon ECS para usar um endpoint da VPC de interface. Os endpoints de interface são desenvolvidos pelo AWS PrivateLink, uma tecnologia que permite que você acesse APIs do Amazon ECS de modo privado usando endereços IP privados. O AWS PrivateLink restringe todo o tráfego de rede entre a VPC e o Amazon ECS à rede da Amazon. Você não precisa de um gateway da Internet, de um dispositivo NAT ou de um gateway privado virtual.

Para obter mais informações sobre o AWS PrivateLink e endpoints da VPC, consulte Endpoints da VPC no Guia do usuário da Amazon VPC.

Considerações

Considerações sobre endpoints nas regiões introduzidas a partir de 23 de dezembro de 2023

Antes de configurar endpoints da VPC de interface para o Amazon ECS, leve em conta as seguintes considerações:

- Você deve ter os seguintes endpoints da VPC específicos da região:

> **Note**
>
> Se você não configurar todos os endpoints, seu tráfego passará pelos endpoints públicos, não pelo endpoint da VPC.

- `com.amazonaws.region.ecs-agent`
- `com.amazonaws.region.ecs-telemetry`
- `com.amazonaws.region.ecs`

Por exemplo, a região Oeste do Canadá (Calgary) (ca-west-1) precisa dos seguintes endpoints da VPC:

- `com.amazonaws.ca-west-1.ecs-agent`
- `com.amazonaws.ca-west-1.ecs-telemetry`
- `com.amazonaws.ca-west-1.ecs`

- Quando você usar um modelo para criar recursos da AWS na nova região e o modelo tiver sido copiado de uma região introduzida antes de 23 de dezembro de 2023, dependendo da região de origem da cópia, faça uma das operações a seguir.

  Por exemplo, a região de origem da cópia é Leste dos EUA (Norte da Virgínia) (us-east-1). A região de destino da cópia é Oeste do Canadá (Calgary) (ca-west-1).

Configuração	Ação
A região de origem da cópia não tem nenhum endpoint da VPC.	Crie todos os três endpoints da VPC para a nova região (por exemplo, `com.amazonaws.ca-west-1.ecs-agent`).
A região de origem da cópia contém endpoints da VPC específicos da região.	a. Crie todos os três endpoints da VPC para a nova região (por exemplo, `com.amazonaws.ca-west-1.ecs-agent`). b. Exclua todos os três endpoints da VPC para a região de origem da cópia (por exemplo, `com.amazo`

Configuração	Ação	
	`naws.us-east-1.ecs-agent`).	

Considerações sobre endpoints da VPC do Amazon ECS para o tipo de inicialização do Fargate

Quando há um endpoint da VPC para `ecr.dkr` e `ecr.api` na mesma VPC em que uma tarefa do Fargate é implantada, o endpoint da VPC é usado. Se não houver um endpoint da VPC, a interface do Fargate é usada.

Antes de configurar endpoints da VPC de interface para o Amazon ECS, leve em conta as seguintes considerações:

- As tarefas que usam o tipo de inicialização do Fargate não exigem endpoints da VPC de interface para o Amazon ECS, mas você pode precisar de endpoints da VPC de interface para o Amazon ECR, o Secrets Manager ou o Amazon CloudWatch Logs descritos nos pontos a seguir.
  - Para permitir que suas tarefas extraiam imagens privadas do Amazon ECR, você deve criar endpoints da VPC de interface para o Amazon ECR. Para obter mais informações, consulte Endpoints da VPC de interface (AWS PrivateLink) no Guia do usuário do Amazon Elastic Container Registry.

    > **Important**
    >
    > Se você configurar o Amazon ECR para usar um endpoint da VPC de interface, crie uma função de execução de tarefas que inclua chaves de condição para restringir o acesso a uma VPC ou a um endpoint da VPC específico. Para obter mais informações, consulte Tarefas do Fargate que extraem imagens do Amazon ECR pelos endpoints da interface.

  - Para permitir que suas tarefas extraiam dados sigilosos do Secrets Manager, é necessário criar endpoints da VPC de interface para o Secrets Manager. Para obter mais informações, consulte Usar o Secrets Manager com endpoints da VPC no Guia do usuário do AWS Secrets Manager.
  - Se a VPC não tiver um gateway da internet e as tarefas usarem o driver de log `awslogs` para enviar informações de log para o CloudWatch Logs, você deverá criar um endpoint da VPC de interface para o CloudWatch Logs. Para obter mais informações, consulte Usar o CloudWatch Logs com endpoints da VPC de interface no Guia do usuário do Amazon CloudWatch Logs.

- Atualmente, os endpoints da VPC não oferecem suporte a solicitações entre Regiões. Crie o endpoint na mesma região em que você planeja emitir as chamadas de API para o Amazon ECS. Por exemplo, suponha que você queira executar tarefas na região Leste dos EUA (Norte da Virgínia). Para isso, você deve criar o endpoint da VPC do Amazon ECS no Leste dos EUA (Norte da Virgínia). Um endpoint da VPC do Amazon ECS criado em qualquer outra região não poderá executar tarefas no Leste dos EUA (Norte da Virgínia).
- Os endpoints da VPC oferecem suporte somente a DNS fornecidos pela Amazon por meio do Amazon Route 53. Se quiser usar seu próprio DNS, você pode usar o encaminhamento de DNS condicional. Para obter mais informações, consulte Conjuntos de Opções de DHCP no Manual do Usuário da Amazon VPC.
- O grupo de segurança anexado ao endpoint da VPC deve permitir conexões de entrada na porta 443 na sub-rede privada da VPC.
- O gerenciamento do Service Connect do proxy Envoy usa o endpoint da VPC `com.amazonaws.region.ecs-agent`. Quando você não usa os endpoints da VPC, o gerenciamento do proxy Envoy pelo Service Connect usa o endpoint `ecs-sc` nessa região. Para obter uma lista dos endpoints do Amazon ECS em cada região, consulte Endpoints e cotas do Amazon ECS.

Considerações para endpoints da VPC do Amazon ECS para o tipo de inicialização do EC2.

Antes de configurar endpoints da VPC de interface para o Amazon ECS, leve em conta as seguintes considerações:

- As tarefas que usam o tipo de inicialização do EC2 exigem que as instâncias de contêiner nas quais são iniciadas executem a versão `1.25.1` ou posterior do agente de contêiner do Amazon ECS. Para obter mais informações, consulte Gerenciamento de instâncias de contêiner do Linux no Amazon ECS.
- Para permitir que suas tarefas extraiam dados sigilosos do Secrets Manager, é necessário criar endpoints da VPC de interface para o Secrets Manager. Para obter mais informações, consulte Usar o Secrets Manager com endpoints da VPC no Guia do usuário do AWS Secrets Manager.
- Se a VPC não tiver um gateway da internet e as tarefas usarem o driver de log `awslogs` para enviar informações de log para o CloudWatch Logs, você deverá criar um endpoint da VPC de interface para o CloudWatch Logs. Para obter mais informações, consulte Usar o CloudWatch Logs com endpoints da VPC de interface no Guia do usuário do Amazon CloudWatch Logs.
- Atualmente, os endpoints da VPC não oferecem suporte a solicitações entre Regiões. Crie o endpoint na mesma região em que você planeja emitir as chamadas de API para o Amazon ECS.

Por exemplo, suponha que você queira executar tarefas na região Leste dos EUA (Norte da Virgínia). Para isso, você deve criar o endpoint da VPC do Amazon ECS no Leste dos EUA (Norte da Virgínia). Um endpoint da VPC do Amazon ECS criado em qualquer outra região não pode executar tarefas no Leste dos EUA (Norte da Virgínia).

- Os endpoints da VPC oferecem suporte somente a DNS fornecidos pela Amazon por meio do Amazon Route 53. Se quiser usar seu próprio DNS, você pode usar o encaminhamento de DNS condicional. Para obter mais informações, consulte Conjuntos de Opções de DHCP no Manual do Usuário da Amazon VPC.
- O grupo de segurança anexado ao endpoint da VPC deve permitir conexões de entrada na porta 443 na sub-rede privada da VPC.

Entendendo os padrões de nomenclatura de endpoints do Amazon ECS

É importante entender que o agente do Amazon ECS pode fazer solicitações aos endpoints com sufixos numerados, como:

- `ecs-a-1.region.amazonaws.com`, `ecs-a-2.region.amazonaws.com` etc. para endpoints de agente
- `ecs-t-1.region.amazonaws.com`, `ecs-t-2.region.amazonaws.com` etc. para endpoints de telemetria

Esse comportamento ocorre porque o agente do Amazon ECS usa a API DiscoverPollEndpoint para determinar dinamicamente a qual endpoint específico se conectar. Se seus endpoints da VPC não processarem adequadamente essas variações de endpoints numerados, o agente voltará a usar endpoints públicos, mesmo que você tenha configurado endpoints de VPC para os nomes da base.

O perfil da API DiscoverPollEndpoint

A API DiscoverPollEndpoint é usada pelo agente do Amazon ECS para descobrir o endpoint apropriado para a sondagem de tarefas. Quando o agente chama essa API, ele recebe um URL específico do endpoint que pode incluir um sufixo numerado. Para garantir que seus endpoints da VPC funcionem corretamente, sua configuração de rede precisará permitir que o agente:

1. Acesse a API DiscoverPollEndpoint
2. Conecte-se aos URLs de endpoint retornados, incluindo aqueles com sufixos numerados

Se você estiver solucionando problemas de conectividade do endpoint da VPC, verifique se seu agente consegue atingir tanto os endpoints da base quanto quaisquer variações numeradas que possam ser retornadas pela API DiscoverPollEndpoint.

Criar os endpoints da VPC para o Amazon ECS

Para criar os endpoints da VPC para o serviço Amazon ECS, use o procedimento Acessar um serviço da AWS usando um endpoint da VPC de interface no Guia do usuário da Amazon VPC para criar os endpoints a seguir. Se tiver instâncias de contêiner existentes na VPC, você deverá criar os endpoints na ordem em que são listados. Se você planeja criar as instâncias de contêiner depois que o VPC endpoint for criado, a ordem não importa.

> **Note**
>
> Se você não configurar todos os endpoints, seu tráfego passará pelos endpoints públicos, não pelo endpoint da VPC.
> Quando você cria endpoints, o Amazon ECS também cria um nome DNS privado para o endpoint. Por exemplo, `ecs-a.region.amazonaws.com` para ecs-agent e `ecs-t.region.amazonaws.com` para ecs-telemetry.

- `com.amazonaws.`*`region`*`.ecs-agent`
- `com.amazonaws.`*`region`*`.ecs-telemetry`
- `com.amazonaws.`*`region`*`.ecs`

> **Note**
>
> A *`região`* representa o identificador da região para uma região da AWS compatível com o Amazon ECS, como `us-east-2` para a região Leste dos EUA (Ohio).

O endpoint `ecs-agent` usa a API `ecs:poll` e o endpoint `ecs-telemetry` usa a API `ecs:poll` e `ecs:StartTelemetrySession`.

Se você tiver tarefas existentes que estão usando o tipo de inicialização do EC2, depois de ter criado os endpoints da VPC, cada instância de contêiner deverá obter a nova configuração. Para que isso aconteça, você deve reinicializar cada instância de contêiner ou reiniciar o agente de contêiner do Amazon ECS em cada instância de contêiner. Para reiniciar o agente do contêiner, faça o seguinte.

Para reiniciar o agente de contêiner do Amazon ECS

1. Faça login em sua instância de contêiner via SSH.
2. Pare o agente de contêiner do .

```
sudo docker stop ecs-agent
```

3. Inicie o agente do contêiner

```
sudo docker start ecs-agent
```

Depois que os endpoints da VPC forem criados e o agente de contêiner do Amazon ECS for reiniciado em cada instância de contêiner, todas as tarefas recém-inicializadas obterão a nova configuração.

Criar uma política de endpoint da VPC para o Amazon ECS

É possível anexar uma política de endpoint ao endpoint da VPC que controla o acesso ao Amazon ECS. Essa política especifica as seguintes informações:

- A entidade principal que pode realizar ações.
- As ações que podem ser realizadas.
- Os recursos aos quais as ações podem ser aplicadas.

Para obter mais informações, consulte Controlar o acesso a serviços com VPC endpoints no Guia do usuário da Amazon VPC.

Exemplo: política de endpoint da VPC para ações do Amazon ECS

Veja a seguir um exemplo de política de endpoint para o Amazon ECS. Quando anexada a um endpoint, essa política concede acesso à permissão para criar e listar clusters. Como as ações `CreateCluster` e `ListClusters` não aceitam recursos, a definição de recurso é definida como * para todos os recursos.

```
{
   "Statement":[
    {
      "Principal":"*",
```

```
      "Effect": "Allow",
      "Action": [
        "ecs:CreateCluster",
        "ecs:ListClusters"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

Modelo de responsabilidade compartilhada da AWS para o Amazon ECS

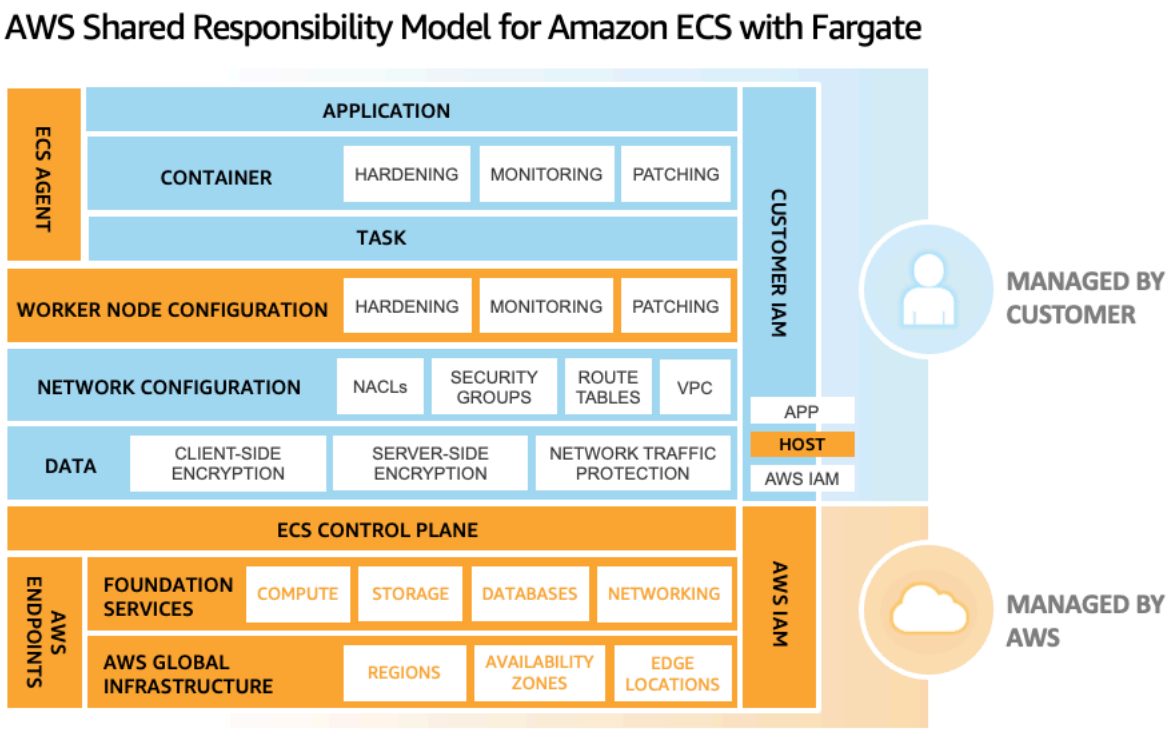
Segurança e conformidade são uma responsabilidade compartilhada entre a AWS e o cliente. Esse modelo compartilhado pode ajudar a reduzir os encargos operacionais do cliente porque a AWS opera, gerencia e controla os componentes do sistema operacional do host e a camada de virtualização, incluindo a segurança física das instalações em que o serviço opera. O cliente assume o gerenciamento e a responsabilidade pelo sistema operacional convidado (inclusive por atualizações e correções de segurança) e por outro software de aplicação associado, bem como pela configuração do firewall dos grupos de segurança fornecido pela AWS. Os clientes devem examinar cuidadosamente os serviços que escolherem, pois suas respectivas responsabilidades variam de acordo com os serviços utilizados, a integração desses serviços ao seu ambiente de TI e as leis e regulamentações aplicáveis. A natureza dessa responsabilidade compartilhada também oferece a flexibilidade e o controle do cliente que permitem a implantação.

Tipo de inicialização do Fargate

A ilustração abaixo mostra o modelo de responsabilidade compartilhada para o tipo de inicialização do Fargate. O Fargate executa cada workload em um ambiente virtualizado de hardware isolado. Como resultado, cada tarefa recebe uma capacidade de infraestrutura dedicada. As workloads em contêineres executadas no Fargate não compartilham um sistema operacional, o kernel do Linux, a interface de rede, o armazenamento efêmero, a CPU ou a memória com outras tarefas. Ao utilizar o Fargate, os clientes não são responsáveis por proteger a infraestrutura computacional que executa seus contêineres. O Fargate provisionará e corrigirá a infraestrutura na qual as workloads do cliente são executadas. Para obter mais informações, consulte Retirada e manutenção de tarefas para o AWS Fargate no Amazon ECS .

Você é responsável por gerenciar os seguintes recursos:

- Configuração de rede, incluindo VPC, NACLs, grupos de segurança e tabelas de rotas
- Criptografia de armazenamento de clientes e serviços. Para obter mais informações, consulte Opções de armazenamento para tarefas do Amazon ECS.
- Imagens de contêiner. Para obter mais informações, consulte Práticas recomendadas de segurança para tarefas e contêineres do Amazon ECS.
- Permissões do IAM para as aplicações usando o perfil de tarefa. Para obter mais informações, consulte Perfil do IAM para tarefas do Amazon ECS.

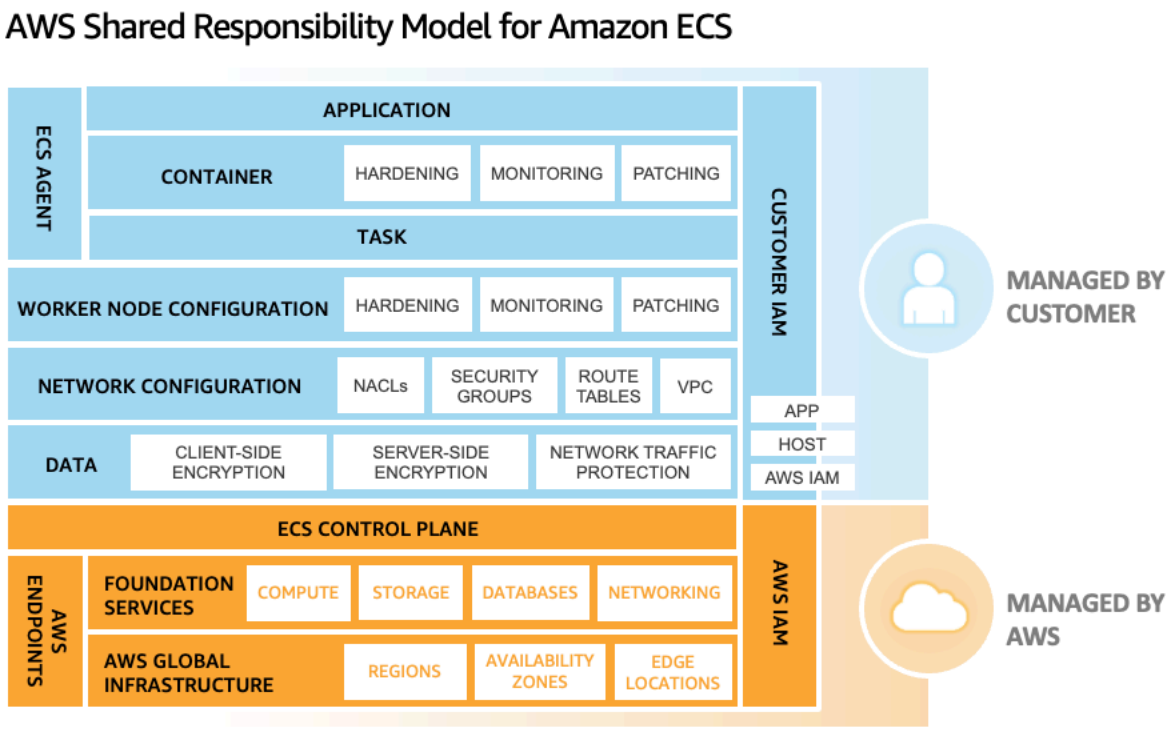


Tipo de inicialização do EC2

A ilustração a seguir mostra a responsabilidade compartilhada para o tipo de inicialização do EC2. Ao executar tarefas em instâncias do EC2, você é responsável por manter suas instâncias do EC2, além dos seguintes recursos:

- O agente do Amazon ECS.
- • A AMI da instância do EC2, incluindo patches e hardening.
- Configuração de rede, incluindo VPC, NACLs, grupos de segurança e tabelas de rotas

- Criptografia de armazenamento de clientes e serviços. Para obter mais informações, consulte Opções de armazenamento para tarefas do Amazon ECS.
- Imagens de contêiner. Para obter mais informações, consulte Práticas recomendadas de segurança para tarefas e contêineres do Amazon ECS.
- Permissões do IAM para as aplicações usando o perfil de tarefa. Para obter mais informações, consulte Perfil do IAM para tarefas do Amazon ECS.



Práticas recomendadas de segurança de rede para o Amazon ECS

A segurança de rede é um tópico amplo que engloba vários subtópicos. Isso inclui criptografia em trânsito, segmentação e isolamento de rede, firewall, roteamento de tráfego e observabilidade.

Criptografia em trânsito

A criptografia do tráfego de rede impede que usuários não autorizados interceptem e leiam dados quando esses dados são transmitidos pela rede. Com o Amazon ECS, a criptografia de rede pode ser implementada de qualquer uma das formas a seguir.

- Uso de instâncias do Nitro:

Por padrão, o tráfego é criptografado automaticamente entre os tipos de instância do Nitro a seguir: C5n, G4, I3en, M5dn, M5n, P3dn e R5n. O tráfego não é criptografado quando é roteado por meio de um gateway de trânsito, balanceador de carga ou intermediário similar.

- Criptografia em trânsito
- Anúncio de o que há de novo de 2019
- Esta palestra do Re:inForce 2019

- Uso de Server Name Indication (SNI) com um o Application Load Balancer:

  O Application Load Balancer (ALB) e o Network Load Balancer (NLB) são compatíveis com Server Name Indication (SNI). Ao usar o SNI, será possível colocar várias aplicações seguras em um único receptor. Para isso, cada uma tem seu próprio certificado TLS. Recomendamos que você provisione certificados para o balanceador de carga usando o AWS Certificate Manager (ACM) e, em seguida, adicione-os à lista de certificados do ouvinte. O balanceador de carga AWS usa um algoritmo inteligente de seleção de certificado com suporte para SNI. Se o nome de host fornecido por um cliente corresponder a um único certificado na lista, o balanceador de carga escolherá esse certificado. Se um nome de host fornecido por um cliente corresponder a vários certificados na lista, o balanceador de carga selecionará um certificado ao qual o cliente possa oferecer suporte. Os exemplos incluem o certificado autoassinado ou um certificado gerado por meio do ACM.

  - SNI com Application Load Balancer
  - SNI com Network Load Balancer

- Criptografia de ponta a ponta com certificados TLS:

  Isso envolve a implantação de um certificado TLS com a tarefa. Isso pode ser um certificado autoassinado ou um certificado de uma autoridade de certificação confiável. É possível obter o certificado referenciando um segredo para o certificado. Caso contrário, você poderá optar por executar um contêiner que emita uma solicitação de assinatura de certificado (CSR) para o ACM e, em seguida, monte o segredo resultante em um volume compartilhado.

  - Manutenção da segurança da camada de transporte por todo o caminho até seus contêineres com o uso do Network Load Balancer com o Amazon ECS, parte 1
  - Manutenção do Transport Layer Security (TLS) por todo o caminho até o seu contêiner, parte 2: uso do AWS Private Certificate Authority

Redes de tarefas

As recomendações a seguir são em consideração a como o Amazon ECS funciona. O Amazon ECS não usa uma rede de sobreposição. Em vez disso, as tarefas são configuradas para operar em diferentes modos de rede. Por exemplo, tarefas configuradas para usar o modo `bridge` adquirem um endereço IP não roteável de uma rede Docker que é executada em cada host. As tarefas configuradas para usar o modo de rede `awsvpc` adquirem um endereço IP da sub-rede do host. As tarefas configuradas com rede `host` usam a interface de rede do host. `awsvpc` é o modo de rede preferido. Isso ocorre porque é o único modo possível de ser usado para atribuir grupos de segurança às tarefas. Também é o único modo disponível para tarefas do AWS Fargate no Amazon ECS.

Grupos de segurança para tarefas

Recomendamos configurar suas tarefas para usar o modo de rede `awsvpc`. Depois de configurar sua tarefa para usar esse modo, o agente do Amazon ECS provisionará e anexará automaticamente uma interface de rede elástica (ENI) à tarefa. Quando a ENI for provisionada, a tarefa será registrada em um grupo de segurança da AWS. O grupo de segurança atua como um firewall virtual para controlar o tráfego de entrada e saída.

Se você usar um firewall personalizado com tarefas ou serviços, adicione uma regra de saída para permitir tráfego para os endpoints de gerenciamento de agente do Amazon ECS ("`ecs-a-*.`*`region`*`.amazonaws.com`"), os endpoints de telemetria ("`ecs-t-*.`*`region`*`.amazonaws.com`") e os endpoints de gerenciamento do Service Connect Envoy ("`ecs-sc.`*`region`*`.api.aws`").

AWS PrivateLink e Amazon ECS

AWS PrivateLink é uma tecnologia de rede que permite criar endpoints privados para diferentes serviços da AWS, incluindo o Amazon ECS. Os endpoints são necessários em ambientes de sandbox, onde não há nenhum gateway da Internet (IGW) conectado ao Amazon VPC e nenhuma rota alternativa para a Internet. O uso de AWS PrivateLink garante que as chamadas para o serviço Amazon ECS permaneçam dentro da Amazon VPC e não atravessem a Internet. Para obter instruções sobre como criar endpoints de AWS PrivateLink para o Amazon ECS e outros serviços relacionados, consulte Endpoints da Amazon VPC de interface do Amazon ECS.

> **Important**
>
> As tarefas do AWS Fargate não exigem um endpoint AWS PrivateLink para o Amazon ECS.

Tanto o Amazon ECR quanto o Amazon ECS oferecem suporte a políticas de endpoints. Essas políticas permitem refinar o acesso às APIs de um serviço. Por exemplo, é possível criar uma política de endpoint para o Amazon ECR que só permita que imagens sejam enviadas para registros em contas específicas da AWS. Uma política como essa pode ser usada para evitar que os dados sejam exfiltrados por meio de imagens de contêineres e, ao mesmo tempo, permitir que os usuários acessem registros autorizados do Amazon ECR. Para obter mais informações, consulte Uso de políticas de endpoint da VPC.

A política a seguir permite que todas as entidades principais da AWS da sua conta realizem todas as ações somente nos seus repositórios do Amazon ECR:

```
{
  "Statement": [
    {
      "Sid": "LimitECRAccess",
      "Principal": "*",
      "Action": "*",
      "Effect": "Allow",
      "Resource": "arn:aws:ecr:region:account_id:repository/*"
    },
  ]
}
```

É possível aprimorar isso ainda mais definindo uma condição que use a nova propriedade `PrincipalOrgID`. Isso evita o envio e a extração de imagens por uma entidade principal do IAM que não faça parte do seu AWS Organizations. Para obter mais informações, consulte aws:PrincipalOrgID.

Recomendamos aplicar a mesma política a ambos os endpoints `com.amazonaws.region.ecr.dkr` e `com.amazonaws.region.ecr.api`.

Configurações do agente de contêiner

O arquivo de configuração do agente de contêiner do Amazon ECS inclui várias variáveis de ambiente relacionadas à segurança da rede. `ECS_AWSVPC_BLOCK_IMDS` e `ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST` são usadas para bloquear o acesso de uma tarefa aos metadados do Amazon EC2. `HTTP_PROXY` é usada para configurar o agente para rotear por meio de um proxy HTTP para se conectar à Internet. Para obter instruções sobre como configurar o agente e o runtime do Docker para rotear por meio de um proxy, consulte Configuração do proxy HTTP.

> **Important**
>
> Essas configurações não estão disponíveis quando você usa o AWS Fargate.

Recomendações de segurança de rede

Recomendamos que você faça o seguinte ao configurar sua Amazon VPC, balanceadores de carga e rede.

Uso da criptografia de rede com o Amazon ECS, quando aplicável

Você deve usar criptografia de rede quando aplicável. Alguns programas de conformidade, como o PCI DSS, exigem que você criptografe os dados em trânsito se eles contiverem dados do titular do cartão. Se sua workload tiver requisitos semelhantes, configure a criptografia de rede.

Os navegadores modernos avisam quando os usuários se conectam a sites inseguros. Se o seu serviço for administrado por um balanceador de carga voltado para o público, use TLS/SSL para criptografar o tráfego do navegador do cliente para o balanceador de carga e recriptografar no backend, se necessário.

Uso do modo de rede **`awsvpc`** e de grupos de segurança para controlar o tráfego entre tarefas e outros recursos no Amazon ECS

Você deve usar o modo de rede `awsvpc` e grupos de segurança quando precisar controlar o tráfego entre tarefas e entre tarefas e outros recursos de rede. Se o serviço ficar atrás de um ALB, use grupos de segurança para permitir apenas a entrada de tráfego de outros recursos de rede que usem o mesmo grupo de segurança que o seu ALB. Se a sua aplicação estiver protegida por um NLB, configure o grupo de segurança da tarefa para permitir somente tráfego de entrada do intervalo CIDR do Amazon VPC e dos endereços IP estáticos atribuídos ao NLB.

Grupos de segurança também devem ser usados para controlar o tráfego entre tarefas e outros recursos dentro da Amazon VPC, como bancos de dados do Amazon RDS.

Criação de clusters do Amazon ECS em VPCs da Amazon separadas quando o tráfego de rede precisar ser estritamente isolado

Você deve criar clusters em Amazon VPCs separadas quando o tráfego de rede precisar ser estritamente isolado. Evite executar workloads que tenham requisitos rígidos de segurança em

clusters com workloads que não precisem cumprir esses requisitos. Quando o isolamento estrito da rede for obrigatório, crie clusters em Amazon VPCs separadas e exponha seletivamente serviços a outras Amazon VPCs usando endpoints da Amazon VPC. Para obter mais informações, consulte VPC endpoints.

Configuração de endpoints do AWS PrivateLink para o Amazon ECS, quando necessário

Você deve configurar endpoints do AWS PrivateLink quando houver garantia. Se sua política de segurança impedir que você conecte um gateway da Internet (IGW) às suas Amazon VPCs, configure endpoints de AWS PrivateLink para o Amazon ECS e outros serviços, como Amazon ECR, AWS Secrets Manager e Amazon CloudWatch.

Uso de logs de fluxo da Amazon VPC para analisar o tráfego de e para tarefas de execução prolongada no Amazon ECS

Você deve usar o Amazon VPC Flow Logs para analisar o tráfego de e para tarefas de longa execução. As tarefas que usam o modo de rede `awsvpc` obtêm seu próprio ENI. Ao fazer isso, é possível monitorar o tráfego que entra e sai de tarefas individuais usando o Amazon VPC Flow Logs. Uma atualização recente do Amazon VPC Flow Logs (v3) enriquece os logs com metadados de tráfego, incluindo o ID da VPC, o ID da sub-rede e o ID da instância. Esses metadados podem ser usados para ajudar a restringir uma investigação. Para obter mais informações, consulte Amazon VPC Flow Logs.

> **Note**
>
> Devido à natureza temporária dos contêineres, os logs de fluxo nem sempre são uma forma eficaz de analisar padrões de tráfego entre diferentes contêineres ou contêineres e outros recursos de rede.

Práticas recomendadas de segurança para tarefas e contêineres do Amazon ECS

Você deve considerar a imagem do contêiner como sua primeira linha de defesa contra um ataque. Uma imagem insegura e mal construída pode permitir que um invasor escape dos limites do contêiner e tenha acesso ao host. Você deve fazer o seguinte para reduzir o risco de isso acontecer.

Recomendamos fazer o seguinte ao configurar suas tarefas e contêineres.

Criar imagens mínimas ou usar imagens sem distribuição

Comece removendo todos os binários estranhos da imagem do contêiner. Se você estiver usando uma imagem desconhecida da Galeria Pública do Amazon ECR, inspecione a imagem para fazer referência ao conteúdo de cada uma das camadas do contêiner. É possível usar uma aplicação como o Dive para fazer isso.

Como alternativa, é possível usar imagens sem distribuição que incluam apenas sua aplicação e suas dependências de runtime. Elas não contêm gerenciadores de pacotes ou shells. Imagens sem distribuição melhoram a "relação sinal/ruído dos scanners e reduzem a carga de estabelecer a proveniência exatamente do que você precisa". Para obter mais informações, consulte a documentação do GitHub sobre sem distribuição.

O Docker tem um mecanismo para a criação de imagens a partir de uma imagem mínima reservada conhecida como rascunho. Para obter mais informações, consulte Criação de uma imagem principal simples com o uso de rascunho na documentação do Docker. Com linguagens como Go, é possível criar um binário vinculado estático e referenciá-lo em seu Dockerfile. O exemplo a seguir mostra como fazer isso.

```
############################
# STEP 1 build executable binary
############################
FROM golang:alpine AS builder
# Install git.
# Git is required for fetching the dependencies.
RUN apk update && apk add --no-cache git
WORKDIR $GOPATH/src/mypackage/myapp/
COPY . .
# Fetch dependencies.
# Using go get.
RUN go get -d -v
# Build the binary.
RUN go build -o /go/bin/hello
############################
# STEP 2 build a small image
############################
FROM scratch
# Copy our static executable.
COPY --from=builder /go/bin/hello /go/bin/hello
# Run the hello binary.
```

```
ENTRYPOINT ["/go/bin/hello"]
This creates a container image that consists of your application and nothing else,
 making it extremely secure.
```

O exemplo anterior também é um exemplo de uma compilação em vários estágios. Esses tipos de compilações são atraentes do ponto de vista da segurança, pois podem ser usadas para minimizar o tamanho da imagem final enviada ao seu registro de contêiner. Imagens de contêiner sem ferramentas de criação e outros binários estranhos melhoram sua postura de segurança ao reduzir a superfície de ataque da imagem. Para obter mais informações sobre compilações em vários estágios, consulte Compilações em vários estágios na documentação do Docker.

Verifique suas imagens em busca de vulnerabilidades

De forma semelhante às de máquinas virtuais, as imagens de contêiner podem conter binários e bibliotecas de aplicações com vulnerabilidades ou desenvolver vulnerabilidades ao longo do tempo. A melhor maneira de se proteger contra ataques é verificar regularmente suas imagens com um verificador de imagens.

O Amazon ECR oferece duas versões de verificação básica que usam o banco de dados de vulnerabilidades e exposições comuns (CVEs):

- Verificação básica nativa da AWS: usa tecnologia nativa da AWS que agora está em disponibilidade geral e é recomendada. Essa verificação básica aprimorada foi desenvolvida para fornecer aos clientes resultados de verificação e detecção de vulnerabilidades melhores em um amplo conjunto de sistemas operacionais populares. Isso permite que os clientes fortaleçam ainda mais a segurança de suas imagens de contêiner. Todos os novos registros de clientes são incluídos nesta versão aprimorada por padrão.
- Verificação básica do Clair: a versão de verificação básica anterior que usa o projeto de código aberto Clair e está obsoleta. Para obter mais informações sobre o Clair, consulte Clair no GitHub.

Tanto a verificação básica nativa da AWS quanto o Clair são compatíveis com todas as regiões listadas em Serviços da AWS por região, exceto aquelas que foram adicionadas após setembro de 2024. Como o suporte ao Clair está obsoleto, o Clair não receberá suporte em novas regiões à medida que forem adicionadas e não terá mais suporte em todas as regiões a partir de 1º de outubro de 2025.

O Amazon ECR usa a severidade de um CVE da fonte de distribuição upstream, se disponível. Do contrário, é usada a pontuação do Common Vulnerability Scoring System (CVSS). A pontuação do

CVSS pode ser usada para obter a classificação de gravidade de vulnerabilidade do NVD. Para obter mais informações, consulte Classificações de gravidade de vulnerabilidade do NVD.

Você também pode ver os resultados de uma verificação no console do Amazon ECR ou chamando a API DescribeImageScanFindings. Imagens com uma vulnerabilidade `HIGH` ou `CRITICAL` devem ser excluídas ou recompiladas. Se uma imagem que foi implantada desenvolver uma vulnerabilidade, ela deverá ser substituída assim que possível.

O Docker Desktop Edge versão 2.3.6.0 ou posterior pode varrer imagens locais. As varreduras são executadas pelo Snyk, um serviço de segurança de aplicações. Quando as vulnerabilidades são descobertas, o Snyk identifica as camadas e as dependências com a vulnerabilidade no Dockerfile. Ele também recomenda alternativas seguras, como o uso de uma imagem base mais fina com menos vulnerabilidades ou atualizar um pacote específico para uma versão mais recente. Ao usar a varredura do Docker, os desenvolvedores podem resolver possíveis problemas de segurança antes de enviar suas imagens para o registro.

- Automatização da conformidade de imagens usando o Amazon ECR e o AWS Security Hub explica como revelar informações de vulnerabilidade do Amazon ECR no AWS Security Hub e automatizar a remediação bloqueando o acesso a imagens vulneráveis.

Remover permissões especiais de suas imagens

Os sinalizadores de direitos de acesso `setuid` e `setgid` permitem a execução um executável com as permissões do proprietário ou do grupo do executável. Remova todos os binários com esses direitos de acesso da sua imagem, pois esses binários podem ser usados para aumentar os privilégios. Considere remover todos os shells e utilitários como o `nc` e o `curl` que possam ser usados para fins maliciosos. É possível encontrar os arquivos com direitos de acesso `setuid` e `setgid` usando o comando a seguir.

```
find / -perm /6000 -type f -exec ls -ld {} \;
```

Para remover essas permissões especiais desses arquivos, adicione a diretiva a seguir à imagem do contêiner.

```
RUN find / -xdev -perm /6000 -type f -exec chmod a-s {} \; || true
```

Criar um conjunto de imagens selecionadas

Em vez de permitir que os desenvolvedores criem suas próprias imagens, crie um conjunto de imagens verificadas para as diferentes pilhas de aplicações em sua organização. Ao fazer isso, os desenvolvedores podem prescindir de aprender a compor Dockerfiles e se concentrar em escrever código. À medida que as alterações são mescladas em sua base de código, um pipeline de CI/CD pode compilar automaticamente o ativo e depois armazená-lo em um repositório de artefatos. E, por último, copie o artefato na imagem apropriada antes de enviá-lo a um registro do Docker, como o Amazon ECR. No mínimo, você deve criar um conjunto de imagens básicas a partir das quais os desenvolvedores possam criar seus próprios Dockerfiles. Evite extrair imagens do Docker Hub. Você nem sempre sabe o que está na imagem, e cerca de um quinto das 1000 imagens principais apresentam vulnerabilidades. Uma lista dessas imagens e suas vulnerabilidades pode ser encontrada em https://vulnerablecontainers.org/.

Varrer pacotes de aplicações e bibliotecas em busca de vulnerabilidades

O uso de bibliotecas de código aberto agora é comum. Assim como acontece com sistemas operacionais e pacotes de sistema operacional, essas bibliotecas podem ter vulnerabilidades. Como parte do ciclo de vida de desenvolvimento, essas bibliotecas devem ser varridas e atualizadas quando vulnerabilidades críticas forem encontradas.

O Docker Desktop executa varreduras locais usando o Snyk. Ele também pode ser usado para encontrar vulnerabilidades e possíveis problemas de licenciamento em bibliotecas de código aberto. Ele pode ser integrado diretamente aos fluxos de trabalho do desenvolvedor, oferecendo a capacidade de mitigar os riscos apresentados pelas bibliotecas de código aberto. Para obter mais informações, consulte os tópicos a seguir.

- As Ferramentas de segurança de aplicações de código aberto incluem uma lista de ferramentas para detectar vulnerabilidades em aplicações.

Executar análise estática de código

Você deve realizar uma análise estática de código antes de criar uma imagem de contêiner. Ela é executada em seu código-fonte e é usada para identificar erros de codificação e códigos que possam ser explorados por um agente mal-intencionado, como injeções de falhas. Você pode usar o Amazon Inspector. Para obter mais informações, consulte Verificação de imagens de contêiner do Amazon ECR com o Amazon Inspector no Guia do usuário do Amazon Inspector.

Executar contêineres como um usuário não raiz

Você deve executar contêineres como um usuário não raiz. Por padrão, os contêineres são executados como o usuário `root`, a menos que a diretiva `USER` esteja incluída em seu Dockerfile. Os recursos padrão do Linux atribuídos pelo Docker restringem as ações que podem ser executadas como `root`, mas apenas marginalmente. Por exemplo, um contêiner em execução como `root` ainda não tem permissão para acessar dispositivos.

Como parte do seu pipeline de CI/CD, você deve executar lint nos Dockerfiles para procurar a diretiva `USER` e fazer falhar a compilação caso ela estiver ausente. Para obter mais informações, consulte os tópicos a seguir.

- O Dockerfile-lint é uma ferramenta de código aberto da RedHat que pode ser usada para verificar se o arquivo está em conformidade com as práticas recomendadas.
- O Hadolint é outra ferramenta para a compilação de imagens do Docker que está em conformidade com as práticas recomendadas.

Usar um sistema de arquivos raiz somente para leitura

Você deve usar um sistema de arquivos raiz somente para leitura. O sistema de arquivos raiz de um contêiner é gravável por padrão. Quando você configura um contêiner com um sistema de arquivos raiz `RO` (somente leitura), ele força você a definir explicitamente onde os dados podem ser persistidos. Isso reduz sua superfície de ataque, pois o sistema de arquivos do contêiner não pode ser gravado, a menos que as permissões sejam concedidas especificamente.

> **Note**
>
> Ter um sistema de arquivos raiz somente para leitura pode causar problemas com determinados pacotes de sistema operacional que esperam poder gravar no sistema de arquivos. Se você planeja usar sistemas de arquivos raiz somente para leitura, faça um teste minucioso com antecedência.

Configurar tarefas com limites de CPU e memória (Amazon EC2)

Você deve configurar tarefas com limites de CPU e memória para minimizar o risco a seguir. Os limites de recursos de uma tarefa definem um limite superior para a quantidade de CPU e memória que pode ser reservada por todos os contêineres em uma tarefa. Se nenhum limite for definido,

as tarefas terão acesso à CPU e à memória do host. Isso pode causar problemas em que tarefas implantadas em um host compartilhado podem privar outras tarefas de recursos do sistema.

> **Note**
>
> O Amazon ECS, em tarefas do AWS Fargate, exige que você especifique limites de CPU e memória, pois ele usa esses valores para fins de cobrança. Uma tarefa que consuma todos os recursos do sistema não é um problema para o Amazon ECS Fargate, pois cada tarefa é executada em sua própria instância dedicada. Se você não especificar um limite de memória, o Amazon ECS alocará um mínimo de 4 MB para cada contêiner. Da mesma forma, se nenhum limite de CPU for definido para a tarefa, o agente de contêiner do Amazon ECS atribuirá a ela um mínimo de 2 CPUs.

Usar tags imutáveis com o Amazon ECR

Com o Amazon ECR, é possível e deve usar imagens de configuração com tags imutáveis. Isso evita enviar uma versão alterada ou atualizada de uma imagem para o seu repositório de imagens com uma tag idêntica. Isso protege contra um invasor que faça push de uma versão comprometida de uma imagem sobre a sua imagem com a mesma tag. Ao usar tags imutáveis, você efetivamente se força a enviar uma nova imagem com uma tag diferente para cada alteração.

Evitar executar contêineres como privilegiados (Amazon EC2)

Você deve evitar executar contêineres como privilegiados. Para o segundo plano, contêineres executados como `privileged` são executados com privilégios estendidos no host. Isso significa que o contêiner herda todos os recursos do Linux atribuídos a `root` no host. Seu uso deve ser severamente restrito ou proibido. Recomendamos definir a variável de ambiente `ECS_DISABLE_PRIVILEGED` do agente de contêiner do Amazon ECS como `true`, para evitar que os contêineres sejam executados como `privileged` em determinados hosts, caso o `privileged` não seja necessário. Como alternativa, é possível usar AWS Lambda para varrer suas definições de tarefas para o uso do parâmetro `privileged`.

> **Note**
>
> Não há suporte para a execução de um contêiner como `privileged` no o Amazon ECS no AWS Fargate.

Remover recursos desnecessários do Linux do contêiner

A seguir há uma lista dos recursos padrão do Linux atribuídos aos contêineres do Docker. Para obter mais informações sobre cada recurso, consulte Visão geral dos recursos do Linux.

```
CAP_CHOWN, CAP_DAC_OVERRIDE, CAP_FOWNER, CAP_FSETID, CAP_KILL,
CAP_SETGID, CAP_SETUID, CAP_SETPCAP, CAP_NET_BIND_SERVICE,
CAP_NET_RAW, CAP_SYS_CHROOT, CAP_MKNOD, CAP_AUDIT_WRITE,
CAP_SETFCAP
```

Se um contêiner não exigir todos os recursos do kernel do Docker listados acima, considere retirá-los do contêiner. Para obter mais informações sobre cada recurso de kernel do Docker, consulte KernelCapabilities. É possível descobrir quais recursos estão em uso fazendo o seguinte:

- Instale o pacote do sistema operacional libcap-ng e execute o utilitário `pscap` para listar os recursos que cada processo está usando.
- Você também pode usar o capsh para decifrar quais recursos um processo está usando.

Usar uma chave gerenciada pelo cliente (CMK) para criptografar imagens enviadas ao Amazon ECR.

Você deve usar uma chave gerenciada pelo cliente (CMK) para criptografar imagens enviadas ao Amazon ECR. As imagens enviadas para o Amazon ECR são automaticamente criptografadas em repouso com uma chave gerenciada AWS Key Management Service (AWS KMS). Se você preferir usar sua própria chave, o Amazon ECR agora oferece suporte à criptografia do AWS KMS com chaves gerenciadas pelo cliente (CMK). Antes de habilitar a criptografia do lado do servidor com uma CMK, revise as considerações listadas na documentação sobre criptografia em repouso.

Tutoriais para o Amazon ECS

Os seguintes tutoriais mostram como executar tarefas comuns ao usar o Amazon ECS.

Você pode usar os tutoriais a seguir para saber mais sobre como começar a usar o Amazon ECS.

Visão geral do tutorial	Saiba mais
Conceitos básicos do Amazon ECS no Fargate.	Saiba como criar uma tarefa do Linux no Amazon ECS para o tipo de inicialização do Fargate
Conceitos básicos de contêineres do Windows no Fargate.	Saiba como criar uma tarefa do Windows no Amazon ECS para o tipo de inicialização do Fargate
Conceitos básicos de contêineres do Windows para o tipo de inicialização do EC2.	Saiba como criar uma tarefa do Windows no Amazon ECS para o tipo de inicialização do EC2

Você pode usar qualquer um dos seguintes tutoriais para implantar tarefas no Amazon ECS usando a AWS CLI

Visão geral do tutorial	Saiba mais
Criar uma tarefa do Linux para o tipo de inicialização do Fargate.	Criar uma tarefa do Linux no Amazon ECS para o tipo de inicialização do Fargate com a AWS CLI
Criar uma tarefa do Windows para o tipo de inicialização do Fargate.	Criar uma tarefa do Windows no Amazon ECS para o tipo

Visão geral do tutorial	Saiba mais
	de inicialização do Fargate com a AWS CLI
Criar uma tarefa do Linux para o tipo de inicialização do EC2.	Criar uma tarefa do Amazon ECS para o tipo de inicialização do EC2 com a AWS CLI

Você pode usar qualquer um dos tutoriais a seguir para saber mais sobre monitoramento e registro em log.

Visão geral do tutorial	Saiba mais
Configurar uma função do Lambda simples que atua como receptor de eventos de tarefa e os grava em um fluxo de logs do CloudWatch Logs.	Configurar o Amazon ECS para atuar como receptor de eventos do CloudWatch Events
Configurar uma regra de evento do Amazon EventBridge que só capture eventos de tarefa em que a execução da tarefa foi interrompida porque um de seus contêineres essenciais foi encerrado.	Enviar alertas do Amazon Simple Notification Service de eventos de tarefa interrompida do Amazon ECS
Concatenar mensagens de log que pertenciam originalmente a um contexto, mas foram divididas em vários registros ou linhas de log.	Concatenar mensagens de log de várias linhas ou de rastreamento de pilha do Amazon ECS
Implantar contêineres do Fluent Bit nas respectivas instâncias do Windows sendo	Implantar o Fluent Bit em contêineres do Windows no Amazon ECS

Visão geral do tutorial	Saiba mais	
executadas no Amazon ECS para transmitir logs gerados pelas tarefas do Windows para o Amazon CloudWatch para registro em um log centralizado.		

Você pode usar qualquer um dos tutoriais a seguir para saber mais sobre como usar a autenticação do Active Directory com uma conta de serviço gerenciado em grupo no Amazon ECS.

Visão geral do tutorial	Saiba mais	
Usar uma conta de serviço gerenciado em grupo com contêineres do Linux no EC2.	Usar gMSA para contêineres do Linux do EC2 no Amazon ECS	
Usar uma conta de serviço gerenciada em grupo com contêineres do Windows no EC2.	Saiba como usar gMSAs para contêineres do Windows do Amazon EC2 para o Amazon ECS	
Usar a conta de serviço gerenciado em grupo com contêineres do Linux no Fargate.	Uso de gMSA em contêineres do Linux no Fargate	
Criar uma tarefa que executa um contêiner do Windows que tem credenciais para acessar o Active Directory com uma conta de serviço gerenciado em grupo sem domínio.	Usar contêineres do Windows no Amazon ECS com gMSA sem domínio usando a AWS CLI	

Criar uma tarefa do Linux no Amazon ECS para o tipo de inicialização do Fargate com a AWS CLI

As etapas a seguir ajudarão você a configurar um cluster, registrar uma definição de tarefa, executar uma tarefa do Linux e realizar outros cenários comuns no Amazon ECS com a AWS CLI. Use a versão mais recente da AWS CLI. Para obter informações sobre como atualizar para a versão mais recente Consulte instalar ou atualizar para a versão mais recente da AWS CLI.

> **Note**
>
> É possível usar endpoints de serviço de pilha dupla para interagir com o Amazon ECS via AWS CLI, SDKs e API do Amazon ECS sobre IPv4 e IPv6. Para obter mais informações, consulte Usar endpoints de pilha dupla do Amazon ECS.

Tópicos

- Pré-requisitos
- Etapa 1: criar um cluster
- Etapa 2: registrar uma definição de tarefa do Linux
- Etapa 3: listar definições de tarefa
- Etapa 4: criar um serviço
- Etapa 5: listar serviços
- Etapa 6: descrever o serviço em execução
- Etapa 7: testar
- Etapa 8: limpar

Pré-requisitos

Este tutorial pressupõe que os pré-requisitos a seguir foram concluídos.

- A versão mais recente da AWS CLI está instalada e configurada. Para obter mais informações sobre como instalar ou atualizar sua AWS CLI, consulte Instalar ou atualizar para a versão mais recente da AWS CLI.
- As etapas em Configuração para usar o Amazon ECS foram concluídas.

- O usuário do IAM tem as permissões necessárias especificadas no exemplo de política do IAM de AmazonECS_FullAccess.
- Você tem uma VPC e um grupo de segurança criados para uso. Este tutorial usa uma imagem de contêiner hospedada no Amazon ECR Public, portanto, sua tarefa deve ter acesso à Internet. Para fornecer à sua tarefa uma rota para a Internet, use uma das opções a seguir.
  - Use uma sub-rede privada com um gateway NAT que tenha um endereço IP elástico.
  - Use uma sub-rede pública e atribua um endereço IP público à tarefa.

  Para obter mais informações, consulte the section called “Criar uma nuvem privada virtual”.

  Para obter informações sobre grupos de segurança e regras, consulte Grupos de segurança padrão para VPCs e Exemplo de regras no Guia do usuário da Amazon Virtual Private Cloud.
- Se você seguir este tutorial usando uma sub-rede privada, poderá usar o Amazon ECS Exec para interagir diretamente com seu contêiner e testar a implantação. Você precisará criar um perfil do IAM de tarefa para usar o ECS Exec. Para obter mais informações sobre o perfil do IAM da tarefa e outros pré-requisitos, consulte Monitorar contêineres do Amazon ECS com o Amazon ECS Exec.
- (Opcional) O AWS CloudShell é uma ferramenta que oferece aos clientes uma linha de comando sem precisar criar a própria instância do EC2. Para obter mais informações, consulte O que é o AWS CloudShell? no Guia do usuário do AWS CloudShell.

Etapa 1: criar um cluster

Por padrão, sua conta recebe um cluster `default`.

> **Note**
>
> O benefício de usar o cluster `default` fornecido é que você não precisa especificar a opção `--cluster` *`cluster_name`* nos comandos subsequentes. Caso você crie o próprio cluster, não padrão, é necessário especificar `--cluster` *`cluster_name`* para cada comando que deseja usar com esse cluster.

Crie o próprio cluster com um nome exclusivo usando o seguinte comando:

```
aws ecs create-cluster --cluster-name fargate-cluster
```

Saída:

```
{
    "cluster": {
        "status": "ACTIVE",
        "defaultCapacityProviderStrategy": [],
        "statistics": [],
        "capacityProviders": [],
        "tags": [],
        "clusterName": "fargate-cluster",
        "settings": [
            {
                "name": "containerInsights",
                "value": "disabled"
            }
        ],
        "registeredContainerInstancesCount": 0,
        "pendingTasksCount": 0,
        "runningTasksCount": 0,
        "activeServicesCount": 0,
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster"
    }
}
```

Etapa 2: registrar uma definição de tarefa do Linux

Para executar uma tarefa no cluster do ECS, você deve registrar uma definição de tarefa. As definições de tarefa são listas de contêineres agrupados. O exemplo a seguir é uma definição de tarefa simples que cria um aplicativo web em PHP usando a imagem de contêiner httpd hospedada no Docker Hub. Para obter mais informações sobre os parâmetros de definição de tarefa disponíveis, consulte Definições de tarefa do Amazon ECS. Para este tutorial, o `taskRoleArn` só será necessário se você estiver implantando a tarefa em uma sub-rede privada e quiser testar a implantação. Substitua o `taskRoleArn` pelo perfil da tarefa do IAM que você criou para usar o ECS Exec, conforme mencionado em Pré-requisitos.

```
 {
        "family": "sample-fargate",
        "networkMode": "awsvpc",
        "taskRoleArn": "arn:aws:iam::aws_account_id:role/execCommandRole",
        "containerDefinitions": [
            {
                "name": "fargate-app",
                "image": "public.ecr.aws/docker/library/httpd:latest",
```

```
                "portMappings": [
                    {
                        "containerPort": 80,
                        "hostPort": 80,
                        "protocol": "tcp"
                    }
                ],
                "essential": true,
                "entryPoint": [
                    "sh",
                    "-c"
                ],
                "command": [
                    "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in Amazon
 ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html && httpd-
foreground\""
                ]
            }
        ],
        "requiresCompatibilities": [
            "FARGATE"
        ],
        "cpu": "256",
        "memory": "512"
}
```

Salve a definição da tarefa JSON como um arquivo e passe-a com a opção `--cli-input-json file://`*`path_to_file.json`*.

Para usar um arquivo JSON para definições de contêiner:

```
aws ecs register-task-definition --cli-input-json file://$HOME/tasks/fargate-task.json
```

O comando register-task-definition retorna uma descrição da definição de tarefa depois de concluir o registro.

Etapa 3: listar definições de tarefa

É possível listar as definições de tarefa para a conta a qualquer momento com o comando list-task-definitions. A saída desse comando mostra os valores `family` e `revision` valores que você pode usar juntos ao chamar run-task ou start-task.

```
aws ecs list-task-definitions
```

Saída:

```
{
    "taskDefinitionArns": [
        "arn:aws:ecs:region:aws_account_id:task-definition/sample-fargate:1"
    ]
}
```

Etapa 4: criar um serviço

Depois de você registrar uma tarefa para sua conta, poderá criar um serviço para a tarefa registrada no seu cluster. Para este exemplo, crie um serviço com uma instância da definição de tarefa `sample-fargate:1` em execução no cluster. A tarefa requer uma rota para a internet, então há duas maneiras de conseguir isso. Uma maneira é usar uma sub-rede privada configurada com um gateway NAT com um endereço IP elástico em uma sub-rede pública. Outra maneira é usar uma sub-rede pública e atribuir um endereço IP público à tarefa. Fornecemos os dois exemplos abaixo.

Exemplo usando uma sub-rede privada. A `enable-execute-command` opção é necessária para usar o Amazon ECS Exec.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service --
task-definition sample-fargate:1 --desired-count 1 --launch-type "FARGATE" --network-
configuration "awsvpcConfiguration={subnets=[subnet-abcd1234],securityGroups=[sg-
abcd1234]}" --enable-execute-command
```

Exemplo usando uma sub-rede pública.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service --
task-definition sample-fargate:1 --desired-count 1 --launch-type "FARGATE" --network-
configuration "awsvpcConfiguration={subnets=[subnet-abcd1234],securityGroups=[sg-
abcd1234],assignPublicIp=ENABLED}"
```

O comando create-service retorna uma descrição da definição de tarefa depois de concluir o registro.

Etapa 5: listar serviços

Liste os serviços para o seu cluster. Você deve ver o serviço que criou na seção anterior. É possível utilizar o nome do serviço ou o ARN completo retornado por esse comando e usá-lo para descrever o serviço depois.

```
aws ecs list-services --cluster fargate-cluster
```

Saída:

```
{
    "serviceArns": [
        "arn:aws:ecs:region:aws_account_id:service/fargate-cluster/fargate-service"
    ]
}
```

Etapa 6: descrever o serviço em execução

Descreva o serviço usando o nome do serviço recuperado anteriormente para obter mais informações sobre a tarefa.

```
aws ecs describe-services --cluster fargate-cluster --services fargate-service
```

Se for bem-sucedido, isso retornará uma descrição das falhas de serviço e os serviços. Por exemplo, na seção `services`, você encontrará informações sobre implantações, como o status das tarefas como pendentes ou em execução. Também é possível encontrar informações sobre a definição da tarefa, a configuração da rede e os eventos com time stamp. Na seção de falhas, você encontrará informações sobre falhas, se houver, associadas à chamada. Para solução de problemas, consulte Mensagens de eventos de serviço. Para obter mais informações sobre a descrição do serviço, consulte Descrever serviços.

```
{
    "services": [
        {
            "networkConfiguration": {
                "awsvpcConfiguration": {
                    "subnets": [
                        "subnet-abcd1234"
```

```
                ],
                "securityGroups": [
                    "sg-abcd1234"
                ],
                "assignPublicIp": "ENABLED"
            }
        },
        "launchType": "FARGATE",
        "enableECSManagedTags": false,
        "loadBalancers": [],
        "deploymentController": {
            "type": "ECS"
        },
        "desiredCount": 1,
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster",
        "serviceArn": "arn:aws:ecs:region:aws_account_id:service/fargate-service",
        "deploymentConfiguration": {
            "maximumPercent": 200,
            "minimumHealthyPercent": 100
        },
        "createdAt": 1692283199.771,
        "schedulingStrategy": "REPLICA",
        "placementConstraints": [],
        "deployments": [
            {
                "status": "PRIMARY",
                "networkConfiguration": {
                    "awsvpcConfiguration": {
                        "subnets": [
                            "subnet-abcd1234"
                        ],
                        "securityGroups": [
                            "sg-abcd1234"
                        ],
                        "assignPublicIp": "ENABLED"
                    }
                },
                "pendingCount": 0,
                "launchType": "FARGATE",
                "createdAt": 1692283199.771,
                "desiredCount": 1,
                "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-
definition/sample-fargate:1",
                "updatedAt": 1692283199.771,
```

```
                    "platformVersion": "1.4.0",
                    "id": "ecs-svc/9223370526043414679",
                    "runningCount": 0
                }
            ],
            "serviceName": "fargate-service",
            "events": [
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 53c0de40-ea3b-489f-a352-623bf1235f08) (task d0aec985-901b-488f-9fb4-61b991b332a3).",
                    "id": "92b8443e-67fb-4886-880c-07e73383ea83",
                    "createdAt": 1510811841.408
                },
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 b4911bee-7203-4113-99d4-e89ba457c626) (task cc5853e3-6e2d-4678-8312-74f8a7d76474).",
                    "id": "d85c6ec6-a693-43b3-904a-a997e1fc844d",
                    "createdAt": 1510811601.938
                },
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 cba86182-52bf-42d7-9df8-b744699e6cfc) (task f4c1ad74-a5c6-4620-90cf-2aff118df5fc).",
                    "id": "095703e1-0ca3-4379-a7c8-c0f1b8b95ace",
                    "createdAt": 1510811364.691
                }
            ],
            "runningCount": 0,
            "status": "ACTIVE",
            "serviceRegistries": [],
            "pendingCount": 0,
            "createdBy": "arn:aws:iam::aws_account_id:user/user_name",
            "platformVersion": "LATEST",
            "placementStrategy": [],
            "propagateTags": "NONE",
            "roleArn": "arn:aws:iam::aws_account_id:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
            "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-definition/
sample-fargate:1"
        }
    ],
    "failures": []
}
```

Etapa 7: testar

Tarefa de teste implantada usando sub-rede pública

Descreva a tarefa no serviço para que você possa obter a Elastic Network Interface (ENI) para a tarefa.

Primeiramente, obtenha o ARN da tarefa.

```
aws ecs list-tasks --cluster fargate-cluster --service fargate-service
```

A saída contém o ARN da tarefa.

```
{
    "taskArns": [
        "arn:aws:ecs:us-east-1:123456789012:task/fargate-service/EXAMPLE
    ]
}
```

Descreva a tarefa e localize o ID da ENI. Use o ARN da tarefa para o parâmetro `tasks`.

```
aws ecs describe-tasks --cluster fargate-cluster --tasks arn:aws:ecs:us-
east-1:123456789012:task/service/EXAMPLE
```

As informações do anexo estão listadas na saída.

```
{
    "tasks": [
        {
            "attachments": [
                {
                    "id": "d9e7735a-16aa-4128-bc7a-b2d5115029e9",
                    "type": "ElasticNetworkInterface",
                    "status": "ATTACHED",
                    "details": [
                        {
                            "name": "subnetId",
                            "value": "subnetabcd1234"
                        },
                        {
                            "name": "networkInterfaceId",
```

```
                            "value": "eni-0fa40520aeEXAMPLE"
                        },
                    ]
                }
…
}
```

Descreva o ENI para obter o endereço IP público.

```
aws ec2 describe-network-interfaces --network-interface-id  eni-0fa40520aeEXAMPLE
```

O endereço IP público está na saída.

```
{
    "NetworkInterfaces": [
        {
            "Association": {
                "IpOwnerId": "amazon",
                "PublicDnsName": "ec2-34-229-42-222.compute-1.amazonaws.com",
                "PublicIp": "198.51.100.2"
            },
…
}
```

Insira o endereço IP público no seu navegador da Web, e você deverá ver uma página da Web que exibe a aplicação de exemplo Amazon ECS.

Tarefa de teste implantada usando sub-rede privada

Descreva a tarefa e localize `managedAgents` para verificar se o `ExecuteCommandAgent` está em execução. Anote o `privateIPv4Address` para uso posterior.

```
aws ecs describe-tasks --cluster fargate-cluster --tasks arn:aws:ecs:us-
east-1:123456789012:task/fargate-service/EXAMPLE
```

As informações do agente gerenciado estão listadas na saída.

```
{
     "tasks": [
        {
```

```
            "attachments": [
                {
                    "id": "d9e7735a-16aa-4128-bc7a-b2d5115029e9",
                    "type": "ElasticNetworkInterface",
                    "status": "ATTACHED",
                    "details": [
                        {
                            "name": "subnetId",
                            "value": "subnetabcd1234"
                        },
                        {
                            "name": "networkInterfaceId",
                            "value": "eni-0fa40520aeEXAMPLE"
                        },
                        {
                            "name": "privateIPv4Address",
                            "value": "10.0.143.156"
                        }
                    ]
                }
            ],
    ...
    "containers": [
        {
        ...
       "managedAgents": [
                        {
                            "lastStartedAt": "2023-08-01T16:10:13.002000+00:00",
                            "name": "ExecuteCommandAgent",
                            "lastStatus": "RUNNING"
                        }
                ],
        ...
  }
```

Depois de verificar se o `ExecuteCommandAgent` está em execução, execute o comando a seguir para executar um shell interativo no contêiner da tarefa.

```
aws ecs execute-command --cluster fargate-cluster \
    --task  arn:aws:ecs:us-east-1:123456789012:task/fargate-service/EXAMPLE  \
    --container  fargate-app \
    --interactive \
    --command "/bin/sh"
```

Depois que o shell interativo estiver em execução, execute os comandos a seguir para instalar o cURL.

```
apt update
```

```
apt install curl
```

Depois de instalar o cURL, execute o comando a seguir usando o endereço IP privado que você obteve anteriormente.

```
 curl 10.0.143.156
```

Você deve ver o equivalente em HTML da página da Web da aplicação de amostra do Amazon ECS.

```
<html>
    <head>
     <title>Amazon ECS Sample App</title>
     <style>body {margin-top: 40px; background-color: #333;} </style>
    </head>
      <body>
      <div style=color:white;text-align:center>
      <h1>Amazon ECS Sample App</h1>
      <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p>
      </div>
      </body>
</html>
```

Etapa 8: limpar

Ao concluir este tutorial, você deve limpar os recursos associados para evitar cobranças por recursos não utilizados.

Exclua o serviço.

```
aws ecs delete-service --cluster fargate-cluster --service fargate-service --force
```

Excluir o cluster.

```
aws ecs delete-cluster --cluster fargate-cluster
```

Criar uma tarefa do Windows no Amazon ECS para o tipo de inicialização do Fargate com a AWS CLI

As etapas a seguir ajudarão você a configurar um cluster, registrar uma definição de tarefa, executar uma tarefa do Windows e realizar outros cenários comuns no Amazon ECS com a AWS CLI. Você deve usar a versão mais recente da AWS CLI. Para obter informações sobre como atualizar para a versão mais recente Consulte instalar ou atualizar para a versão mais recente da AWS CLI.

> **Note**
>
> É possível usar endpoints de serviço de pilha dupla para interagir com o Amazon ECS via AWS CLI, SDKs e API do Amazon ECS sobre IPv4 e IPv6. Para obter mais informações, consulte Usar endpoints de pilha dupla do Amazon ECS.

Tópicos

- Pré-requisitos
- Etapa 1: criar um cluster
- Etapa 2: registrar uma definição de tarefa do Windows
- Etapa 3: listar definições de tarefa
- Etapa 4: criar um serviço
- Etapa 5: listar serviços
- Etapa 6: descrever o serviço em execução
- Etapa 7: limpar

Pré-requisitos

Este tutorial pressupõe que os pré-requisitos a seguir foram concluídos.

- A versão mais recente da AWS CLI está instalada e configurada. Para obter mais informações sobre como instalar ou atualizar sua AWS CLI, consulte Instalar ou atualizar para a versão mais recente da AWS CLI.

- As etapas em Configuração para usar o Amazon ECS foram concluídas.
- O usuário do IAM tem as permissões necessárias especificadas no exemplo de política do IAM de AmazonECS_FullAccess.
- Você tem uma VPC e um grupo de segurança criados para uso. Este tutorial usa uma imagem de contêiner hospedada no Docker Hub, portanto, sua tarefa deve ter acesso à Internet. Para fornecer à sua tarefa uma rota para a Internet, use uma das opções a seguir.
  - Use uma sub-rede privada com um gateway NAT que tenha um endereço IP elástico.
  - Use uma sub-rede pública e atribua um endereço IP público à tarefa.

  Para obter mais informações, consulte the section called “Criar uma nuvem privada virtual”.

  Para obter informações sobre grupos de segurança e regras, consulte Grupos de segurança padrão para VPCs e Exemplo de regras no Guia do usuário da Amazon Virtual Private Cloud.
- (Opcional) O AWS CloudShell é uma ferramenta que oferece aos clientes uma linha de comando sem precisar criar a própria instância do EC2. Para obter mais informações, consulte O que é o AWS CloudShell? no Guia do usuário do AWS CloudShell.

Etapa 1: criar um cluster

Por padrão, sua conta recebe um cluster `default`.

> **Note**
>
> O benefício de usar o cluster `default` fornecido é que você não precisa especificar a opção `--cluster` *`cluster_name`* nos comandos subsequentes. Caso você crie o próprio cluster, não padrão, é necessário especificar `--cluster` *`cluster_name`* para cada comando que deseja usar com esse cluster.

Crie o próprio cluster com um nome exclusivo usando o seguinte comando:

```
aws ecs create-cluster --cluster-name fargate-cluster
```

Saída:

```
{
    "cluster": {
```

```
        "status": "ACTIVE",
        "statistics": [],
        "clusterName": "fargate-cluster",
        "registeredContainerInstancesCount": 0,
        "pendingTasksCount": 0,
        "runningTasksCount": 0,
        "activeServicesCount": 0,
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster"
    }
}
```

Etapa 2: registrar uma definição de tarefa do Windows

Para executar uma tarefa do Windows no seu cluster do Amazon ECS, você deve registrar uma definição de tarefa. As definições de tarefa são listas de contêineres agrupados. O exemplo a seguir é uma definição de tarefa simples que cria uma aplicação da Web. Para obter mais informações sobre os parâmetros de definição de tarefa disponíveis, consulte Definições de tarefa do Amazon ECS.

```
{
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file
 -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>'; C:\\ServiceMonitor.exe
 w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
```

```
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "networkMode": "awsvpc",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["FARGATE"]
}
```

O exemplo de JSON acima pode ser passado para a AWS CLI de duas maneiras: é possível salvar o JSON de definição de tarefa como um arquivo e passá-lo com a opção `--cli-input-json file://path_to_file.json`.

Para usar um arquivo JSON para definições de contêiner:

```
aws ecs register-task-definition --cli-input-json file://$HOME/tasks/fargate-task.json
```

O comando register-task-definition retorna uma descrição da definição de tarefa depois de concluir o registro.

Etapa 3: listar definições de tarefa

É possível listar as definições de tarefa para a conta a qualquer momento com o comando list-task-definitions. A saída desse comando mostra os valores `family` e `revision` valores que você pode usar juntos ao chamar run-task ou start-task.

```
aws ecs list-task-definitions
```

Saída:

```
{
    "taskDefinitionArns": [
        "arn:aws:ecs:region:aws_account_id:task-definition/sample-fargate-windows:1"
    ]
}
```

Etapa 4: criar um serviço

Depois de você registrar uma tarefa para sua conta, poderá criar um serviço para a tarefa registrada no seu cluster. Para este exemplo, crie um serviço com uma instância da definição de tarefa `sample-fargate:1` em execução no cluster. A tarefa requer uma rota para a internet, então há duas maneiras de conseguir isso. Uma maneira é usar uma sub-rede privada configurada com um gateway NAT com um endereço IP elástico em uma sub-rede pública. Outra maneira é usar uma sub-rede pública e atribuir um endereço IP público à tarefa. Fornecemos os dois exemplos abaixo.

Exemplo usando uma sub-rede privada.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service
 --task-definition sample-fargate-windows:1 --desired-count 1 --launch-type
 "FARGATE" --network-configuration "awsvpcConfiguration={subnets=[subnet-
abcd1234],securityGroups=[sg-abcd1234]}"
```

Exemplo usando uma sub-rede pública.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service
 --task-definition sample-fargate-windows:1 --desired-count 1 --launch-type
 "FARGATE" --network-configuration "awsvpcConfiguration={subnets=[subnet-
abcd1234],securityGroups=[sg-abcd1234],assignPublicIp=ENABLED}"
```

O comando create-service retorna uma descrição da definição de tarefa depois de concluir o registro.

Etapa 5: listar serviços

Liste os serviços para o seu cluster. Você deve ver o serviço que criou na seção anterior. É possível utilizar o nome do serviço ou o ARN completo retornado por esse comando e usá-lo para descrever o serviço depois.

```
aws ecs list-services --cluster fargate-cluster
```

Saída:

```
{
    "serviceArns": [
        "arn:aws:ecs:region:aws_account_id:service/fargate-service"
    ]
}
```

Etapa 6: descrever o serviço em execução

Descreva o serviço usando o nome do serviço recuperado anteriormente para obter mais informações sobre a tarefa.

```
aws ecs describe-services --cluster fargate-cluster --services fargate-service
```

Se for bem-sucedido, isso retornará uma descrição das falhas de serviço e os serviços. Por exemplo, na seção de serviços, você encontrará informações sobre implantações, como o status das tarefas como pendentes ou em execução. Também é possível encontrar informações sobre a definição da tarefa, a configuração da rede e os eventos com time stamp. Na seção de falhas, você encontrará informações sobre falhas, se houver, associadas à chamada. Para solução de problemas, consulte Mensagens de eventos de serviço. Para obter mais informações sobre a descrição do serviço, consulte Descrever serviços.

```
{
    "services": [
        {
            "status": "ACTIVE",
            "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-definition/
sample-fargate-windows:1",
            "pendingCount": 2,
            "launchType": "FARGATE",
            "loadBalancers": [],
            "roleArn": "arn:aws:iam::aws_account_id:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
            "placementConstraints": [],
            "createdAt": 1510811361.128,
            "desiredCount": 2,
            "networkConfiguration": {
                "awsvpcConfiguration": {
                    "subnets": [
                        "subnet-abcd1234"
                    ],
                    "securityGroups": [
                        "sg-abcd1234"
                    ],
                    "assignPublicIp": "DISABLED"
                }
            },
            "platformVersion": "LATEST",
```

```
            "serviceName": "fargate-service",
            "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster",
            "serviceArn": "arn:aws:ecs:region:aws_account_id:service/fargate-service",
            "deploymentConfiguration": {
                "maximumPercent": 200,
                "minimumHealthyPercent": 100
            },
            "deployments": [
                {
                    "status": "PRIMARY",
                    "networkConfiguration": {
                        "awsvpcConfiguration": {
                            "subnets": [
                                "subnet-abcd1234"
                            ],
                            "securityGroups": [
                                "sg-abcd1234"
                            ],
                            "assignPublicIp": "DISABLED"
                        }
                    },
                    "pendingCount": 2,
                    "launchType": "FARGATE",
                    "createdAt": 1510811361.128,
                    "desiredCount": 2,
                    "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-
definition/sample-fargate-windows:1",
                    "updatedAt": 1510811361.128,
                    "platformVersion": "0.0.1",
                    "id": "ecs-svc/9223370526043414679",
                    "runningCount": 0
                }
            ],
            "events": [
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 53c0de40-ea3b-489f-a352-623bf1235f08) (task d0aec985-901b-488f-9fb4-61b991b332a3).",
                    "id": "92b8443e-67fb-4886-880c-07e73383ea83",
                    "createdAt": 1510811841.408
                },
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 b4911bee-7203-4113-99d4-e89ba457c626) (task cc5853e3-6e2d-4678-8312-74f8a7d76474).",
                    "id": "d85c6ec6-a693-43b3-904a-a997e1fc844d",
```

```
                    "createdAt": 1510811601.938
                },
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 cba86182-52bf-42d7-9df8-b744699e6cfc) (task f4c1ad74-a5c6-4620-90cf-2aff118df5fc).",
                    "id": "095703e1-0ca3-4379-a7c8-c0f1b8b95ace",
                    "createdAt": 1510811364.691
                }
            ],
            "runningCount": 0,
            "placementStrategy": []
        }
    ],
    "failures": []
}
```

Etapa 7: limpar

Ao concluir este tutorial, você deve limpar os recursos associados para evitar cobranças por recursos não utilizados.

Exclua o serviço.

```
aws ecs delete-service --cluster fargate-cluster --service fargate-service --force
```

Excluir o cluster.

```
aws ecs delete-cluster --cluster fargate-cluster
```

Criar uma tarefa do Amazon ECS para o tipo de inicialização do EC2 com a AWS CLI

As etapas a seguir ajudarão você a configurar um cluster, registrar uma definição de tarefa, executar uma tarefa e realizar outros cenários comuns no Amazon ECS com a AWS CLI. Use a versão mais recente da AWS CLI. Para obter informações sobre como atualizar para a versão mais recente Consulte instalar ou atualizar para a versão mais recente da AWS CLI.

> **Note**
>
> É possível usar endpoints de serviço de pilha dupla para interagir com o Amazon ECS via AWS CLI, SDKs e API do Amazon ECS sobre IPv4 e IPv6. Para obter mais informações, consulte Usar endpoints de pilha dupla do Amazon ECS.

Tópicos

- Pré-requisitos
- Criar um cluster
- Executar uma instância de contêiner com a AMI do Amazon ECS
- Listar instâncias de contêiner
- Descrever sua instância de contêiner
- Registrar uma definição de tarefa
- Listar definições de tarefa
- Criar um serviço
- Listar serviços
- Descrever o serviço
- Descrever a tarefa em execução
- Testar o servidor Web
- Limpar recursos

Pré-requisitos

Este tutorial pressupõe que os seguintes pré-requisitos foram concluídos:

- A versão mais recente da AWS CLI está instalada e configurada. Para obter mais informações sobre como instalar ou atualizar sua AWS CLI, consulte Instalar ou atualizar para a versão mais recente da AWS CLI.
- As etapas em Configuração para usar o Amazon ECS foram concluídas.
- O usuário do IAM tem as permissões necessárias especificadas no exemplo de política do IAM de AmazonECS_FullAccess.
- Você tem um perfil do IAM de instância de contêiner criado para uso. Para obter informações, consulte Função do IAM de instância de contêiner do Amazon ECS.

- Você tem uma VPC criada para uso. Para obter mais informações, consulte the section called "Criar uma nuvem privada virtual".
- (Opcional) O AWS CloudShell é uma ferramenta que oferece aos clientes uma linha de comando sem precisar criar a própria instância do EC2. Para obter mais informações, consulte O que é o AWS CloudShell? no Guia do usuário do AWS CloudShell.

Criar um cluster

Por padrão, a conta recebe um cluster `default` quando você ativa a primeira instância de contêiner.

> **Note**
>
> O benefício de usar o cluster `default` fornecido é que você não precisa especificar a opção `--cluster` *`cluster_name`* nos comandos subsequentes. Caso você crie o próprio cluster, não padrão, é necessário especificar `--cluster` *`cluster_name`* para cada comando que deseja usar com esse cluster.

Crie o próprio cluster com um nome exclusivo usando o seguinte comando:

```
aws ecs create-cluster --cluster-name MyCluster
```

Saída:

```
{
    "cluster": {
        "clusterName": "MyCluster",
        "status": "ACTIVE",
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/MyCluster"
    }
}
```

Executar uma instância de contêiner com a AMI do Amazon ECS

Instâncias de contêiner são instâncias do EC2 que executam o agente de contêiner do Amazon ECS e foram registradas em um cluster. Nesta seção, você executará uma instância do EC2 usando a AMI otimizada para ECS.

Para executar uma instância de contêiner com o AWS CLI

1. Recupere o ID da mais recente AMI do Amazon Linux 2 otimizada para ECS para sua Região da AWS usando o seguinte comando. Esse comando usa o AWS Systems Manager Parameter Store para obter o ID da mais recente AMI otimizada para ECS. A AMI inclui o agente de contêiner do Amazon ECS e o runtime do Docker pré-instalados.

   ```
   aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
   recommended --query 'Parameters[0].Value' --output text | jq -r '.image_id'
   ```

   Saída:

   ```
   ami-abcd1234
   ```

2. Crie um grupo de segurança que permita acesso SSH para gerenciar sua instância de contêiner e acesso HTTP para o servidor Web.

   ```
   aws ec2 create-security-group --group-name ecs-tutorial-sg --description "ECS
    tutorial security group"
   ```

   Saída:

   ```
   {
       "GroupId": "sg-abcd1234"
   }
   ```

3. Adicione uma regra de entrada ao grupo de segurança executando o seguinte comando.

   ```
   aws ec2 authorize-security-group-ingress --group-id sg-abcd1234 --protocol tcp --
   port 80 --cidr 0.0.0.0/0
   ```

   Saída:

   ```
   {
       "Return": true,
       "SecurityGroupRules": [
           {
               "SecurityGroupRuleId": "sgr-efgh5678",
               "GroupId": "sg-abcd1234",
               "GroupOwnerId": "123456789012",
   ```

```
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIpv4": "0.0.0.0/0"
        }
    ]
}
```

Agora, o grupo de segurança permite o acesso SSH do intervalo de IP especificado e acesso HTTP de qualquer lugar. Em um ambiente de produção, você deve restringir o acesso SSH ao seu endereço IP específico e considerar limitar o acesso HTTP conforme necessário.

4. Crie um par de chaves do EC2 para acesso SSH à sua instância de contêiner.

```
aws ec2 create-key-pair --key-name ecs-tutorial-key --query 'KeyMaterial' --output
 text > ecs-tutorial-key.pem
chmod 400 ecs-tutorial-key.pem
```

A chave privada fica salva em sua máquina local com as permissões apropriadas para acesso SSH.

5. Inicie uma instância do EC2 usando a AMI otimizada para ECS e configure-a para se juntar ao seu cluster.

```
aws ec2 run-instances --image-id ami-abcd1234 --instance-type t3.micro --key-
name ecs-tutorial-key --security-group-ids sg-abcd1234 --iam-instance-profile
 Name=ecsInstanceRole --user-data '#!/bin/bash
echo ECS_CLUSTER=MyCluster >> /etc/ecs/ecs.config'
{
    "Instances": [
        {
            "InstanceId": "i-abcd1234",
            "ImageId": "ami-abcd1234",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "PrivateDnsName": "",
            "PublicDnsName": "",
            "StateReason": {
                "Code": "pending",
                "Message": "pending"
```

```
            },
            "InstanceType": "t3.micro",
            "KeyName": "ecs-tutorial-key",
            "LaunchTime": "2025-01-13T10:30:00.000Z"
        }
    ]
}
```

O script de dados do usuário configura o agente do Amazon ECS para registrar a instância com o seu `MyCluster`. A instância usa a o perfil do IAM `ecsInstanceRole`, que fornece as permissões necessárias para o agente.

Listar instâncias de contêiner

Depois de alguns minutos do início de sua instância de contêiner, o agente do Amazon ECS registra a instância junto ao cluster padrão. É possível listar as instâncias de contêiner em um cluster executando o comando a seguir:

```
aws ecs list-container-instances --cluster default
```

Saída:

```
{
    "containerInstanceArns": [
        "arn:aws:ecs:us-east-1:aws_account_id:container-instance/container_instance_ID"
    ]
}
```

Descrever sua instância de contêiner

Depois que tiver o ARN ou o ID de uma instância de contêiner, será possível usar o comando describe-container-instances para obter informações úteis sobre a instância, como recursos de memória e CPU restantes e registrados.

```
aws ecs describe-container-instances --cluster MyCluster --container-
instances container_instance_ID
```

Saída:

```
{
    "failures": [],
    "containerInstances": [
        {
            "status": "ACTIVE",
            "registeredResources": [
                {
                    "integerValue": 1024,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "CPU",
                    "doubleValue": 0.0
                },
                {
                    "integerValue": 995,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "MEMORY",
                    "doubleValue": 0.0
                },
                {
                    "name": "PORTS",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [
                        "22",
                        "2376",
                        "2375",
                        "51678"
                    ],
                    "type": "STRINGSET",
                    "integerValue": 0
                },
                {
                    "name": "PORTS_UDP",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [],
                    "type": "STRINGSET",
                    "integerValue": 0
                }
            ],
            "ec2InstanceId": "instance_id",
```

```
            "agentConnected": true,
            "containerInstanceArn": "arn:aws:ecs:us-west-2:aws_account_id:container-
instance/container_instance_ID",
            "pendingTasksCount": 0,
            "remainingResources": [
                {
                    "integerValue": 1024,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "CPU",
                    "doubleValue": 0.0
                },
                {
                    "integerValue": 995,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "MEMORY",
                    "doubleValue": 0.0
                },
                {
                    "name": "PORTS",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [
                        "22",
                        "2376",
                        "2375",
                        "51678"
                    ],
                    "type": "STRINGSET",
                    "integerValue": 0
                },
                {
                    "name": "PORTS_UDP",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [],
                    "type": "STRINGSET",
                    "integerValue": 0
                }
            ],
            "runningTasksCount": 0,
            "attributes": [
                {
```

```
                    "name": "com.amazonaws.ecs.capability.privileged-container"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.17"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.18"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.19"
                },
                {
                    "name": "com.amazonaws.ecs.capability.logging-driver.json-file"
                },
                {
                    "name": "com.amazonaws.ecs.capability.logging-driver.syslog"
                }
            ],
            "versionInfo": {
                "agentVersion": "1.5.0",
                "agentHash": "b197edd",
                "dockerVersion": "DockerVersion: 1.7.1"
            }
        }
    ]
}
```

Também é possível encontrar o ID da instância do Amazon EC2 que é possível usar para monitorar a instância no console do Amazon ECS ou com o comando aws ec2 describe-instances --instance-id ***instance_id***.

Registrar uma definição de tarefa

Para executar uma tarefa em seu cluster do Amazon ECS, você precisa registrar uma definição de tarefa. As definições de tarefa são listas de contêineres agrupados. O exemplo a seguir é uma definição de tarefa simples que usa uma imagem do `nginx`. Para obter mais informações sobre os parâmetros de definição de tarefa disponíveis, consulte Definições de tarefa do Amazon ECS.

```
{
    "family": "nginx-task",
    "containerDefinitions": [
        {
```

```
            "name": "nginx",
            "image": "public.ecr.aws/ecs-sample-image/amazon-ecs-sample:latest",
            "cpu": 256,
            "memory": 512,
            "essential": true,
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "requiresCompatibilities": ["EC2"],
    "networkMode": "bridge"
}
```

O exemplo de JSON acima pode ser passado para a AWS CLI de duas maneiras: é possível salvar o JSON de definição de tarefa como um arquivo e passá-lo com a opção `--cli-input-json file://path_to_file.json`. Como alternativa, você pode ignorar as aspas no JSON e passar as definições do contêiner do JSON na linha de comando. Caso você opte por passar as definições de contêiner na linha de comando, o comando ainda exige um parâmetro `--family` usado para manter várias versões da definição de tarefa associadas entre si.

Para usar um arquivo JSON para definições de contêiner:

```
aws ecs register-task-definition --cli-input-json file://$HOME/tasks/nginx.json
```

O register-task-definition retornará uma descrição da definição de tarefa depois de concluir o registro.

```
{
    "taskDefinition": {
        "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/nginx-
task:1",
        "family": "nginx-task",
        "revision": 1,
        "status": "ACTIVE",
        "containerDefinitions": [
            {
                "name": "nginx",
                "image": "public.ecr.aws/docker/library/nginx:latest",
```

```
                "cpu": 256,
                "memory": 512,
                "essential": true,
                "portMappings": [
                    {
                        "containerPort": 80,
                        "hostPort": 80,
                        "protocol": "tcp"
                    }
                ],
                "environment": [],
                "mountPoints": [],
                "volumesFrom": []
            }
        ],
        "volumes": [],
        "networkMode": "bridge",
        "compatibilities": [
            "EC2"
        ],
        "requiresCompatibilities": [
            "EC2"
        ]
    }
}
```

Listar definições de tarefa

É possível listar as definições de tarefa para a conta a qualquer momento com o comando list-task-definitions. A saída desse comando mostra os valores `family` e `revision` que você pode usar juntos ao chamar create-service.

```
aws ecs list-task-definitions
```

Saída:

```
{
    "taskDefinitionArns": [
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/sleep360:1",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/sleep360:2",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/nginx-task:1",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:3",
```

```
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:4",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:5",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:6"
    ]
}
```

Criar um serviço

Depois de ter registrado uma tarefa para sua conta e iniciado uma instância de contêiner que está registrada no seu cluster, você pode criar um serviço do Amazon ECS que executa e mantém um número desejado de tarefas simultaneamente usando a definição de tarefa que você registrou. Neste exemplo, você coloca uma única instância da definição de tarefa `nginx:1` no cluster padrão.

```
aws ecs create-service --cluster default --service-name nginx-service --task-
definition nginx-task:1 --desired-count 1
```

Saída:

```
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-east-1:aws_account_id:service/MyCluster/nginx-
service",
        "serviceName": "nginx-service",
        "clusterArn": "arn:aws:ecs:us-east-1:aws_account_id:cluster/MyCluster",
        "taskDefinition": "arn:aws:ecs:us-east-1:aws_account_id:task-definition/nginx-
task:1",
        "desiredCount": 1,
        "runningCount": 0,
        "pendingCount": 0,
        "launchType": "EC2",
        "status": "ACTIVE",
        "createdAt": "2025-01-13T10:45:00.000Z"
    }
}
```

Listar serviços

Liste os serviços para o seu cluster. Você deve ver o serviço que criou na seção anterior. É possível anotar o ID do serviço ou o ARN completo retornado por esse comando e usá-lo para descrever o serviço depois.

```
aws ecs list-services --cluster MyCluster
```

Saída:

```
{
    "taskArns": [
        "arn:aws:ecs:us-east-1:aws_account_id:task/task_ID"
    ]
}
```

Descrever o serviço

Descreva o serviço usando o seguinte comando para obter mais informações sobre o serviço.

```
aws ecs describe-services --cluster MyCluster --services nginx-service
```

Saída:

```
{
    "services": [
        {
            "serviceArn": "arn:aws:ecs:us-east-1:aws_account_id:service/MyCluster/
nginx-service",
            "serviceName": "nginx-service",
            "clusterArn": "arn:aws:ecs:us-east-1:aws_account_id:cluster/MyCluster",
            "taskDefinition": "arn:aws:ecs:us-east-1:aws_account_id:task-definition/
nginx-task:1",
            "desiredCount": 1,
            "runningCount": 1,
            "pendingCount": 0,
            "launchType": "EC2",
            "status": "ACTIVE",
            "createdAt": "2025-01-13T10:45:00.000Z",
            "events": [
                {
                    "id": "abcd1234-5678-90ab-cdef-1234567890ab",
                    "createdAt": "2025-01-13T10:45:30.000Z",
                    "message": "(service nginx-service) has started 1 tasks: (task
 abcd1234-5678-90ab-cdef-1234567890ab)."
                }
            ]
        }
```

```
    ]
}
```

Descrever a tarefa em execução

Após descrever o serviço, execute o seguinte comando para obter mais informações sobre a tarefa que está sendo executada como parte do seu serviço.

```
aws ecs list-tasks --cluster MyCluster --service-name nginx-service
```

Saída:

```
{
    "tasks": [
        {
            "taskArn": "arn:aws:ecs:us-east-1:aws_account_id:task/MyCluster/
abcd1234-5678-90ab-cdef-1234567890ab",
            "clusterArn": "arn:aws:ecs:us-east-1:aws_account_id:cluster/MyCluster",
            "taskDefinitionArn": "arn:aws:ecs:us-east-1:aws_account_id:task-definition/
nginx-task:1",
            "containerInstanceArn": "arn:aws:ecs:us-east-1:aws_account_id:container-
instance/MyCluster/abcd1234-5678-90ab-cdef-1234567890ab",
            "lastStatus": "RUNNING",
            "desiredStatus": "RUNNING",
            "containers": [
                {
                    "containerArn": "arn:aws:ecs:us-east-1:aws_account_id:container/
MyCluster/abcd1234-5678-90ab-cdef-1234567890ab/abcd1234-5678-90ab-cdef-1234567890ab",
                    "taskArn": "arn:aws:ecs:us-east-1:aws_account_id:task/MyCluster/
abcd1234-5678-90ab-cdef-1234567890ab",
                    "name": "nginx",
                    "lastStatus": "RUNNING",
                    "networkBindings": [
                        {
                            "bindIP": "0.0.0.0",
                            "containerPort": 80,
                            "hostPort": 80,
                            "protocol": "tcp"
                        }
                    ]
                }
            ],
```

```
            "createdAt": "2025-01-13T10:45:00.000Z",
            "startedAt": "2025-01-13T10:45:30.000Z"
        }
    ]
}
```

Testar o servidor Web

Para testar o servidor Web

1. Recupere o endereço IP público da sua instância de contêiner executando o seguinte comando.

   ```
   aws ec2 describe-instances --instance-ids i-abcd1234 --query
    'Reservations[0].Instances[0].PublicIpAddress' --output text
   ```

   Saída:

   ```
   203.0.113.25
   ```

2. Após recuperar o endereço IP, execute o seguinte comando `curl` com o endereço IP.

   ```
   curl http://203.0.113.25
   ```

   Saída:

   ```
   <!DOCTYPE html>
   <html>
   <head>
   <title>Welcome to nginx!</title>
   ...
   </head>
   <body>
   <h1>Welcome to nginx!</h1>
   <p>If you can see this page, the nginx web server is successfully installed and
    working.</p>
   ...
   </body>
   </html>
   ```

A página de boas-vindas do nginx confirmará que seu serviço está funcionando com sucesso e pode ser acessado pela Internet.

Limpar recursos

Para evitar a geração de cobranças, limpe os recursos que foram criados neste tutorial.

Como limpar recursos

1. Atualize o serviço para ter zero tarefas desejadas e, em seguida, exclua o serviço.

```
aws ecs update-service --cluster MyCluster --service nginx-service --desired-count
 0
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-east-1:123456789012:service/MyCluster/nginx-
service",
        "serviceName": "nginx-service",
        "desiredCount": 0,
        "runningCount": 1,
        "pendingCount": 0,
        "status": "ACTIVE"
    }
}
```

2. Aguarde até que as tarefas em execução parem e, em seguida, exclua o serviço.

```
aws ecs delete-service --cluster MyCluster --service nginx-service
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-east-1:123456789012:service/MyCluster/nginx-
service",
        "serviceName": "nginx-service",
        "status": "DRAINING"
    }
}
```

3. Encerre a instância de contêiner que você criou.

```
aws ec2 terminate-instances --instance-ids i-abcd1234
{
```

```
    "TerminatingInstances": [
        {
            "InstanceId": "i-abcd1234",
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

4. Limpe o grupo de segurança e o par de chaves que você criou.

```
aws ec2 delete-security-group --group-id sg-abcd1234
aws ec2 delete-key-pair --key-name ecs-tutorial-key
rm ecs-tutorial-key.pem
```

5. Exclua o cluster do Amazon ECS.

```
aws ecs delete-cluster --cluster MyCluster
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/MyCluster",
        "clusterName": "MyCluster",
        "status": "INACTIVE"
    }
}
```

Configurar o Amazon ECS para atuar como receptor de eventos do CloudWatch Events

Saiba como configurar uma função do Lambda simples que atua como receptor de eventos de tarefa e os grava em um fluxo de logs do CloudWatch Logs.

Pré-requisito: configurar um cluster de teste

Caso você não tenha um cluster em execução para capturar eventos, siga as etapas em the section called “Criar um cluster para o tipo de inicialização do Fargate” para criar um. Ao final deste tutorial, você executará uma tarefa nesse cluster para testar se configurou a função Lambda corretamente.

Etapa 1: criar a função do Lambda

Neste procedimento, você criará uma função simples do Lambda para funcionar como um destino para mensagens da sequência de eventos do Amazon ECS.

1. Abra o console do AWS Lambda em https://console.aws.amazon.com/lambda/.
2. Escolha Create function.
3. Na tela Author from scratch, faça o seguinte:
   a. Em Name (Nome), insira um valor.
   b. Para Runtime, escolha sua versão do Python, por exemplo, Python 3.9.
   c. Em Role (Função), escolha Create a new role with basic Lambda permissions (Criar uma nova função com permissões básicas do Lambda).
4. Escolha a opção Criar função.
5. Na seção Function code, edite o código de exemplo de acordo com o exemplo a seguir.

```
import json

def lambda_handler(event, context):
    if event["source"] != "aws.ecs":
       raise ValueError("Function only supports input from events with a source
 type of: aws.ecs")

    print('Here is the event:')
    print(json.dumps(event))
```

   Essa é uma função simples do Python 3.9 que imprime o evento enviado pelo Amazon ECS. Se tudo estiver configurado corretamente, no final deste tutorial, você verá que os detalhes do evento aparecerão no fluxo de log do CloudWatch Logs associado a essa função Lambda.
6. Escolha Salvar.

Etapa 2: registrar uma regra de evento

Em seguida, você cria uma regra de evento do CloudWatch Events que captura eventos de tarefa vindos dos clusters do Amazon ECS. Essa regra captura todos os eventos vindos de todos os clusters dentro da conta em que ela está definida. As próprias mensagens de tarefa contêm informações sobre a origem do evento, inclusive o cluster no qual reside, que você pode usar para filtrar e classificar eventos programaticamente.

> **Note**
>
> Quando você usa o AWS Management Console para criar uma regra de evento, o console adiciona automaticamente as permissões do IAM necessárias para conceder ao CloudWatch Events permissão para chamar a função Lambda. Caso esteja criando uma regra de evento usando a AWS CLI, você precisa conceder essa permissão explicitamente. Para obter mais informações, consulte Eventos no Amazon EventBridge e Padrões de eventos do Amazon EventBridge no Guia do usuário do Amazon EventBridge.

Para encaminhar eventos para a função Lambda

1. Abra o console do CloudWatch, em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Events (Eventos), Rules (Regras), Create rule (Criar regra).
3. Em Event Source, escolha ECS como origem do evento. Por padrão, a regra se aplica a todos os eventos do Amazon ECS para todos os grupos do Amazon ECS. Como alternativa, você pode selecionar eventos específicos ou um grupo específico do Amazon ECS.
4. Em Targets (Destinos), escolha Add target (Adicionar destino), em Target type (Tipo de destino), escolha Lambda function (Função Lambda) e, em seguida, selecione a função Lambda.
5. Escolha Configure details (Configurar detalhes).
6. Em Rule definition, digite um nome e uma descrição para a regra e escolha Create rule.

Etapa 3: criar uma definição de tarefa

Crie uma definição de tarefa.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, selecione Definições de tarefas.

3. Escolha Create new Task Definition (Criar nova definição de tarefa), Create new revision with JSON (Criar nova revisão com JSON).
4. Copie e cole o exemplo de definição de tarefa a seguir na caixa e escolha Save (Salvar).

```
{
   "containerDefinitions": [
      {
         "command": [
            "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</
title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
         ],
         "entryPoint": [
            "sh",
            "-c"
         ],
         "essential": true,
         "image": "public.ecr.aws/docker/library/httpd:2.4",
         "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
               "awslogs-group" : "/ecs/fargate-task-definition",
               "awslogs-region": "us-east-1",
               "awslogs-stream-prefix": "ecs"
            }
         },
         "name": "sample-fargate-app",
         "portMappings": [
            {
               "containerPort": 80,
               "hostPort": 80,
               "protocol": "tcp"
            }
         ]
      }
   ],
   "cpu": "256",
   "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
   "family": "fargate-task-definition",
   "memory": "512",
```

```
    "networkMode": "awsvpc",
    "runtimePlatform": {
         "operatingSystemFamily": "LINUX"
     },
    "requiresCompatibilities": [
        "FARGATE"
     ]
}
```

5. Escolha Criar.

Etapa 4: testar a regra

Por fim, você cria uma regra de evento do CloudWatch Events que captura eventos de tarefa vindos dos clusters do Amazon ECS. Essa regra captura todos os eventos vindos de todos os clusters dentro da conta em que ela está definida. As próprias mensagens de tarefa contêm informações sobre a origem do evento, inclusive o cluster no qual reside, que você pode usar para filtrar e classificar eventos programaticamente.

Como testar sua regra

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Escolha Task definitions (Definições de tarefa).
3. Escolha console-sample-app-static e, em seguida, escolha Deploy (Implantar), Run new task (Executar nova tarefa).
4. Em Cluster, escolha default (padrão) e, em seguida, escolha Deploy (Implantar).
5. Abra o console CloudWatch em https://console.aws.amazon.com/cloudwatch/.
6. No painel de navegação, escolha Logs e selecione o grupo de logs para sua função do Lambda (por exemplo, /aws/lambda/*my-function*).
7. Selecione um fluxo de log para visualizar os dados do evento.

Enviar alertas do Amazon Simple Notification Service de eventos de tarefa interrompida do Amazon ECS

Configurar uma regra de evento do Amazon EventBridge que só capture eventos de tarefa em que a execução da tarefa foi interrompida porque um de seus contêineres essenciais foi encerrado. O

evento só envia eventos de tarefa com uma propriedade `stoppedReason` específica para o tópico do Amazon SNS designado.

Pré-requisito: configurar um cluster de teste

Caso você não tenha um cluster em execução para capturar eventos, siga as etapas em Conceitos básicos do console usando contêineres de Linux no AWS Fargate para criar um. Ao final deste tutorial, você executará uma tarefa nesse cluster para testar se configurou o tópico do Amazon SNS e a regra do EventBridge corretamente.

Pré-requisito: configurar permissões para o Amazon SNS

Para permitir que o EventBridge publique em um tópico do Amazon SNS, use os comandos aws sns get-topic-attributes e aws sns set-topic-attributes.

Para obter informações sobre como adicionar a permissão, consulte Permissões do Amazon SNS no Guia do desenvolvedor do Amazon Simple Notification Service.

Adicione as seguintes permissões:

```
{
  "Sid": "PublishEventsToMyTopic",
  "Effect": "Allow",
  "Principal": {
     "Service": "events.amazonaws.com"
  },
  "Action": "sns: Publish",
  "Resource": "arn:aws:sns:region:account-id:TaskStoppedAlert",
}
```

Etapa 1: criar e se inscrever em um tópico do Amazon SNS

Neste tutorial, você configura um tópico do Amazon SNS para funcionar como um destino de evento para a nova regra de evento.

Para obter informações sobre como criar e assinar um tópico do Amazon SNS, consulte Conceitos básicos do Amazon SNS no Guia do desenvolvedor do Amazon Simple Notification Service e use a tabela a seguir para determinar quais opções selecionar.

Opção	Valor
Tipo	Padrão
Name	TaskStoppedAlert
Protocolo	E-mail
Endpoint	Um endereço de email ao qual você atualmente tem acesso

Etapa 2: registrar uma regra de evento

Em seguida, você registra uma regra de evento que captura apenas eventos de tarefa parada para tarefas com contêineres parados.

Para obter informações sobre como criar um tópico do Amazon SNS e assiná-lo, consulte Criar uma regra no Amazon EventBridge no Guia do usuário do Amazon EventBridge e use a tabela a seguir para determinar quais opções selecionar.

Opção	Valor
Tipo de regra	Regra com um padrão de evento
Origem do evento.	Eventos da AWS ou eventos de parceiros do EventBridge
Padrão de evento	Padrão personalizado (editor JSON)

Padrão de evento	<pre>{
	"source":[
	"aws.ecs"
	],
	"detail-type":[
	"ECS Task State Change"
	],</pre>

Opção	Valor

	<pre>"detail":{
	"lastStatus":[
	"STOPPED"
	],
	"stoppedReason":[
	"Essentia
	l container in task
	exited"
	]
	}
	}</pre>

Target type	Serviço da AWS
Alvo	Tópico do SNS
Tópico	TaskStoppedAlert (O tópico que você criou na Etapa 1)

Etapa 3: testar a regra

Verifique se a regra está funcionando ao executar uma tarefa que sai pouco depois que é iniciada. Caso a regra de evento esteja configurada corretamente, você recebe uma mensagem de e-mail em alguns minutos com o texto do evento. Se você tiver uma definição de tarefa existente que possa atender aos requisitos da regra, execute uma tarefa ao usá-la. Se você não fizer isso, as etapas a seguir vão orientá-lo no registro de uma definição de tarefa Fargate e na sua execução.

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Task definitions (Definições de tarefa).
3. Escolha Create new task definition (Criar nova definição de tarefa), Create new task definition with JSON (Criar nova definição de tarefa com JSON).
4. Na caixa do editor JSON, edite o arquivo JSON e copie o seguinte no editor.

```
{
   "containerDefinitions":[
      {
         "command":[
```

```
            "sh",
            "-c",
            "sleep 5"
         ],
         "essential":true,
         "image":"public.ecr.aws/amazonlinux/amazonlinux:latest",
         "name":"test-sleep"
      }
   ],
   "cpu":"256",
   "executionRoleArn":"arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
   "family":"fargate-task-definition",
   "memory":"512",
   "networkMode":"awsvpc",
   "requiresCompatibilities":[
      "FARGATE"
   ]
}
```

5. Escolha Criar.

Para executar uma tarefa no console

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Na página Clusters, escolha o cluster que você criou nos pré-requisitos.
3. Na guia Tasks (Tarefas), escolha Run new task (Executar nova tarefa).
4. Em Application type (Tipo de aplicação), escolha Task (Tarefa).
5. Em Definição de tarefa, selecione fargate-task-definition.
6. Em Desired tasks (Tarefas desejadas), insira o número de tarefas que serão iniciadas.
7. Escolha Criar.

Concatenar mensagens de log de várias linhas ou de rastreamento de pilha do Amazon ECS

Começando com AWS para o Fluent Bit versão 2.22.0, um filtro de várias linhas está incluído. O filtro de várias linhas ajuda a concatenar mensagens de log que pertencem originalmente a um contexto, mas foram divididas em vários registros ou linhas de log. Para obter mais informações sobre o filtro multilinha, consulte a documentação do Fluent Bit.

Exemplos comuns de mensagens de log dividido incluem:

- Rastreamentos de pilhas.
- Aplicações que imprimem logs em várias linhas.
- Mensagens de log que foram divididas porque eram maiores que o tamanho máximo do buffer do runtime especificado. É possível concatenar mensagens de log divididas pelo runtime do contêiner seguindo o exemplo no GitHub: Exemplo do FireLens: Concatenar logs de contêiners parciais/divididos.

Permissões obrigatórias do IAM

Você deve primeiro ter as permissões do IAM necessárias para que o agente de contêiner extraia as imagens de contêiner do Amazon ECR e para que o contêiner encaminhe logs ao CloudWatch Logs.

Para essas permissões, você precisa ter as seguintes funções:

- Uma função do IAM de tarefa.
- Um perfil do IAM de execução de tarefa.

Para usar o editor de políticas JSON para criar uma política

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Políticas.

   Se essa for a primeira vez que você escolhe Políticas, a página Bem-vindo às políticas gerenciadas será exibida. Escolha Começar.
3. Na parte superior da página, escolha Criar política.
4. Na seção Editor de políticas, escolha a opção JSON.
5. Insira o seguinte documento de política JSON:

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action": [
   "logs:CreateLogStream",
```

```
    "logs:CreateLogGroup",
    "logs:PutLogEvents"
   ],
   "Resource": "*"
  }]
}
```

6. Escolha Próximo.

> **Note**
>
> É possível alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Próximo no editor Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte Restruturação de política no Guia do usuário do IAM.

7. Na página Revisar e criar, insira um Nome de política e uma Descrição (opcional) para a política que você está criando. Revise Permissões definidas nessa política para ver as permissões que são concedidas pela política.
8. Escolha Criar política para salvar sua nova política.

Determinar quando usar configuração de logs multilinha

Veja a seguir um exemplo de trechos de log exibidos no console do CloudWatch Logs com a configuração de log padrão. É possível ver a linha que começa com `log` para determinar se precisa do filtro multilinha. Quando o contexto é o mesmo, você pode usar a configuração de log de várias linhas. Neste exemplo, o contexto é "com.myproject.model.MyProject".

```
2022-09-20T15:47:56:595-05-00                              {"container_id":
 "82ba37cada1d44d389b03e78caf74faa-EXAMPLE", "container_name": "example-
app", "source=": "stdout", "log": ": "     at com.myproject.modele.
(MyProject.badMethod.java:22)",
    {
      "container_id":  "82ba37cada1d44d389b03e78caf74faa-EXAMPLE",
      "container_name: ": "example-app",
      "source": "stdout",
      "log": ": "     at com.myproject.model.MyProject.badMethod(MyProject.java:22)",
      "ecs_cluster": "default",
      "ecs_task_arn": "arn:aws:region:123456789012:task/default/
b23c940d29ed4714971cba72cEXAMPLE",
```

```
      "ecs_task_definition": "firelense-example-multiline:3"
     }
```

```
2022-09-20T15:47:56:595-05-00                              {"container_id":
 "82ba37cada1d44d389b03e78caf74faa-EXAMPLE", "container_name": "example-app", "stdout",
 "log": ": "     at com.myproject.modele.(MyProject.oneMoreMethod.java:18)",
    {
      "container_id":  "82ba37cada1d44d389b03e78caf74faa-EXAMPLE",
      "container_name: ": "example-app",
      "source": "stdout",
      "log": ": "     at
 com.myproject.model.MyProject.oneMoreMethod(MyProject.java:18)",
      "ecs_cluster": "default",
      "ecs_task_arn": "arn:aws:region:123456789012:task/default/
b23c940d29ed4714971cba72cEXAMPLE,
      "ecs_task_definition": "firelense-example-multiline:3"
     }
```

Depois de usar a configuração de log multilinha, a saída será semelhante ao exemplo abaixo.

```
2022-09-20T15:47:56:595-05-00                              {"container_id":
 "82ba37cada1d44d389b03e78caf74faa-EXAMPLE", "container_name": "example-app",
 "stdout",...
    {
      "container_id":  "82ba37cada1d44d389b03e78caf74faa-EXAMPLE",
      "container_name: ": "example-app",
      "source": "stdout",
      "log:    "September 20, 2022 06:41:48 Exception in thread \"main\"
 java.lang.RuntimeException: Something has gone wrong, aborting!\n
    at com.myproject.module.MyProject.badMethod(MyProject.java:22)\n    at
    at com.myproject.model.MyProject.oneMoreMethod(MyProject.java:18)
 com.myproject.module.MyProject.main(MyProject.java:6)",
      "ecs_cluster": "default",
      "ecs_task_arn": "arn:aws:region:123456789012:task/default/
b23c940d29ed4714971cba72cEXAMPLE",
      "ecs_task_definition": "firelense-example-multiline:2"
     }
```

Opções de análise e concatenação

Para analisar logs e concatenar linhas divididas devido a novas linhas, use qualquer uma destas duas opções.

- Use seu próprio arquivo do analisador que contém as regras para analisar e concatenar linhas que pertencem à mesma mensagem.
- Usar um analisador integrado do Fluent Bit. Para obter uma lista de idiomas compatíveis com os analisadores integrados Fluent Bit, consulte a documentação do Fluent Bit.

O tutorial a seguir orientará você sobre as etapas de cada caso de uso. As etapas mostram como concatenar várias linhas e enviar os logs para o Amazon CloudWatch. É possível especificar um destino diferente para os logs.

Exemplo: use um analisador criado por você

Neste exemplo, você concluirá as seguintes etapas:

1. Criar e carregar a imagem para um contêiner Fluent Bit.
2. Criar e carregar a imagem para uma aplicação de várias linhas de demonstração que executa, falha e gera um rastreamento de pilha de várias linhas.
3. Criar a definição de tarefa e executar a tarefa.
4. Exibir os logs para verificar se as mensagens que abrangem várias linhas aparecem concatenadas.

Criar e carregar a imagem para um contêiner Fluent Bit

Essa imagem incluirá o arquivo do analisador no qual você especifica a expressão regular e um arquivo de configuração que faz referência ao arquivo do analisador.

1. Criar uma pasta com o nome `FluentBitDockerImage`.
2. Na pasta, crie um arquivo do analisador que contém as regras para analisar o log e concatenar linhas que pertencem à mesma mensagem.

   a. Cole o seguinte conteúdo no arquivo do analisador:

```
[MULTILINE_PARSER]
```

```
    name          multiline-regex-test
    type          regex
    flush_timeout 1000
    #
    # Regex rules for multiline parsing
    # ---------------------------------
    #
    # configuration hints:
    #
    #  - first state always has the name: start_state
    #  - every field in the rule must be inside double quotes
    #
    # rules |   state name  | regex pattern                  | next state
    # ------|---------------|--------------------------------------------
    rule      "start_state"   "/(Dec \d+ \d+\:\d+\:\d+)(.*)/"  "cont"
    rule      "cont"          "/^\s+at.*/"                     "cont"
```

À medida que você personaliza seu padrão de regex, recomendamos que você use um editor de expressões regulares para testar a expressão.

b. Salve o arquivo como `parsers_multiline.conf`.

3. Dentro da pasta `FluentBitDockerImage`, crie um arquivo de configuração personalizado que faça referência ao arquivo do analisador que você criou na etapa anterior.

   Para obter mais informações sobre o arquivo de configuração personalizado, consulte Especificar um arquivo de configuração personalizado no Guia do desenvolvedor do Amazon Elastic Container Service

   a. Cole o seguinte conteúdo no arquivo:

```
[SERVICE]
    flush                1
    log_level            info
    parsers_file         /parsers_multiline.conf

[FILTER]
    name                 multiline
    match                *
    multiline.key_content log
    multiline.parser     multiline-regex-test
```

> **Note**
>
> Você deve usar o caminho absoluto do analisador.

b. Salve o arquivo como `extra.conf`.

4. Dentro da pasta `FluentBitDockerImage`, crie o Dockerfile com a imagem Fluent Bit e os arquivos do analisador e de configuração que você criou.

   a. Cole o seguinte conteúdo no arquivo:

```
FROM public.ecr.aws/aws-observability/aws-for-fluent-bit:latest

ADD parsers_multiline.conf /parsers_multiline.conf
ADD extra.conf /extra.conf
```

   b. Salve o arquivo como `Dockerfile`.

5. Usando o Dockerfile, crie uma imagem Fluent Bit personalizada com os arquivos de configuração personalizados e do analisador incluídos.

> **Note**
>
> É possível colocar o arquivo do analisador e o arquivo de configuração em qualquer lugar da imagem do Docker, exceto em `/fluent-bit/etc/fluent-bit.conf`, pois esse caminho de arquivo é usado pelo FireLens.

   a. Crie a imagem: `docker build -t fluent-bit-multiline-image.`

      Onde: `fluent-bit-multiline-image` é o nome da imagem neste exemplo.

   b. Verifique se a imagem foi criada corretamente: `docker images –filter reference=fluent-bit-multiline-image`

      Se tiver êxito, a saída mostrará a imagem e a etiqueta `latest`.

6. Carregue a imagem personalizada do Fluent Bit no Amazon Elastic Container Registry.

   a. Crie um repositório do Amazon ECR para armazenar a imagem: `aws ecr create-repository --repository-name fluent-bit-multiline-repo --region us-east-1`

Onde: `fluent-bit-multiline-repo` é o nome do repositório e `us-east-1` é a região neste exemplo.

A saída fornece os detalhes do novo repositório.

b. Marque sua imagem com o valor `repositoryUri` da saída anterior: `docker tag` *`fluent-bit-multiline-image repositoryUri`*

Exemplo: `docker tag fluent-bit-multiline-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

c. Execute a imagem do Docker para verificar se ela é executada corretamente: `docker images –filter reference=`*`repositoryUri`*

Na saída, o nome do repositório muda de fluent-bit-multiline-repo para o `repositoryUri`.

d. Autentique-se no Amazon ECR executando o comando `aws ecr get-login-password` e especificando o ID de registro para o qual deseja fazer a autenticação: `aws ecr get-login-password | docker login --username AWS --password-stdin` *`registry ID`*`.dkr.ecr.`*`region`*`.amazonaws.com`

Exemplo: `ecr get-login-password | docker login --username AWS --password-stdin xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com`

Uma mensagem de login com êxito é exibida.

e. Envie a imagem para o Amazon ECR: `docker push` *`registry ID`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`repository name`*

Exemplo: `docker push xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

Crie e carregue a imagem para uma aplicação de várias linhas de demonstração

Essa imagem incluirá um arquivo de script Python que executa a aplicação e um arquivo de log de amostra.

Quando você executa a tarefa, a aplicação simula execuções, depois falha e cria um rastreamento de pilha.

1. Crie uma pasta denominada `multiline-app`: `mkdir multiline-app`

2. Crie um arquivo de script do Python.

   a. Na pasta `multiline-app`, crie um arquivo de texto e denomine-o `main.py`.

   b. Cole o seguinte conteúdo no arquivo:

```
import os
import time
file1 = open('/test.log', 'r')
Lines = file1.readlines()

count = 0

for i in range(10):
    print("app running normally...")
    time.sleep(1)

# Strips the newline character
for line in Lines:
    count += 1
    print(line.rstrip())
print(count)
print("app terminated.")
```

   c. Salve o arquivo `main.py`.

3. Crie um arquivo de log de exemplo.

   a. Na pasta `multiline-app`, crie um arquivo de texto e denomine-o `test.log`.

   b. Cole o seguinte conteúdo no arquivo:

```
single line...
Dec 14 06:41:08 Exception in thread "main" java.lang.RuntimeException:
 Something has gone wrong, aborting!
    at com.myproject.module.MyProject.badMethod(MyProject.java:22)
    at com.myproject.module.MyProject.oneMoreMethod(MyProject.java:18)
    at com.myproject.module.MyProject.anotherMethod(MyProject.java:14)
    at com.myproject.module.MyProject.someMethod(MyProject.java:10)
    at com.myproject.module.MyProject.main(MyProject.java:6)
another line...
```

   c. Salve o arquivo `test.log`.

4. Dentro da pasta `multiline-app`, crie o Dockerfile.

   a. Cole o seguinte conteúdo no arquivo:

   ```
   FROM public.ecr.aws/amazonlinux/amazonlinux:latest
   ADD test.log /test.log

   RUN yum upgrade -y && yum install -y python3

   WORKDIR /usr/local/bin

   COPY main.py .

   CMD ["python3", "main.py"]
   ```

   b. Salve o arquivo `Dockerfile`.

5. Usando o Dockerfile, crie uma imagem.

   a. Crie a imagem: `docker build -t multiline-app-image`

   Onde: `multiline-app-image` é o nome da imagem neste exemplo.

   b. Verifique se a imagem foi criada corretamente: `docker images –filter reference=multiline-app-image`

   Se tiver êxito, a saída mostrará a imagem e a etiqueta `latest`.

6. Carregue a imagem no Amazon Elastic Container Registry.

   a. Crie um repositório do Amazon ECR para armazenar a imagem: `aws ecr create-repository --repository-name multiline-app-repo --region us-east-1`

   Onde: `multiline-app-repo` é o nome do repositório e `us-east-1` é a região neste exemplo.

   A saída fornece os detalhes do novo repositório. Anote o valor do `repositoryUri`, pois ele será necessário na próxima etapa.

   b. Marque sua imagem com o valor `repositoryUri` da saída anterior: `docker tag` *`multiline-app-image repositoryUri`*

   Exemplo: `docker tag multiline-app-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/multiline-app-repo`

c. Execute a imagem do Docker para verificar se ela é executada corretamente: `docker images –filter reference=`*`repositoryUri`*

   Na saída, o nome do repositório muda de `multiline-app-repo` para o valor `repositoryUri`.

d. Envie a imagem para o Amazon ECR: `docker push` *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`repository name`*

   Exemplo: `docker push` *`xxxxxxxxxxxx`*`.dkr.ecr.`*`us-east-1`*`.amazonaws.com/multiline-app-repo`

Crie a definição da tarefa e execute a tarefa

1. Crie um arquivo de definição de tarefa com o nome de arquivo `multiline-task-definition.json`.
2. Cole o seguinte conteúdo no arquivo `multiline-task-definition.json`:

```
{
    "family": "firelens-example-multiline",
    "taskRoleArn": "task role ARN,
    "executionRoleArn": "execution role ARN",
    "containerDefinitions": [
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-
multiline-image:latest",
            "name": "log_router",
            "firelensConfiguration": {
                "type": "fluentbit",
                "options": {
                    "config-file-type": "file",
                    "config-file-value": "/extra.conf"
                }
            },
            "memoryReservation": 50
        },
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/multiline-app-
image:latest",
```

```
            "name": "app",
            "logConfiguration": {
                "logDriver": "awsfirelens",
                "options": {
                    "Name": "cloudwatch_logs",
                    "region": "us-east-1",
                    "log_group_name": "multiline-test/application",
                    "auto_create_group": "true",
                    "log_stream_prefix": "multiline-"
                }
            },
            "memoryReservation": 100
        }
    ],
    "requiresCompatibilities": ["FARGATE"],
    "networkMode": "awsvpc",
    "cpu": "256",
    "memory": "512"
}
```

Substitua o seguinte na definição de tarefa `multiline-task-definition.json`:

a. *task role ARN*

   Para encontrar o ARN da função de tarefa, acesse o console do IAM. Escolha Roles (Funções) e encontre a função de tarefa `ecs-task-role-for-firelens` que você criou. Escolha a função e copie o ARN que consta na seção Summary (Resumo).

b. *execution role ARN*

   Para encontrar o ARN da função de execução, acesse o console do IAM. Selecione Roles (Funções) e localize a função `ecsTaskExecutionRole`. Escolha a função e copie o ARN que consta na seção Summary (Resumo).

c. *aws_account_id*

   Para encontrar seu `aws_account_id`, faça login no AWS Management Console. Escolha seu nome de usuário no canto superior direito e copie o ID da conta.

d. *us-east-1*

   Substitua a região, se necessário.

3. Registre o arquivo de definição de tarefa: `aws ecs register-task-definition --cli-input-json file://multiline-task-definition.json --region region`
4. Abra o console em https://console.aws.amazon.com/ecs/v2.
5. No painel de navegação, selecione Task Definitions (Definições de tarefa) e, em seguida, escolha a família `firelens-example-multiline`, pois registramos a definição de tarefa para essa família na primeira linha da definição de tarefa acima.
6. Escolha a versão mais recente.
7. Escolha Implantar, Executar tarefa.
8. Na página Executar tarefa, para Cluster, escolha o cluster e, em seguida, em Rede, para Sub-redes, escolha as sub-redes disponíveis para sua tarefa.
9. Escolha Criar.

Verifique se as mensagens de log de várias linhas no Amazon CloudWatch parecem concatenadas

1. Abra o console do CloudWatch, em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, expanda Logs e escolha Log groups (Grupos de log).
3. Escolha o grupo de logs `multiline-test/applicatio`.
4. Escolha o log. Vejas as mensagens. As linhas que correspondem às regras no arquivo do analisador são concatenadas e aparecem como uma única mensagem.

   O trecho de log a seguir mostra linhas concatenadas em um único evento de rastreamento de pilha Java:

```
{
    "container_id": "xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "log": "Dec 14 06:41:08 Exception in thread \"main\"
 java.lang.RuntimeException: Something has gone wrong, aborting!\n
 at com.myproject.module.MyProject.badMethod(MyProject.java:22)\n    at
 com.myproject.module.MyProject.oneMoreMethod(MyProject.java:18)\n
 at com.myproject.module.MyProject.anotherMethod(MyProject.java:14)\n
 at com.myproject.module.MyProject.someMethod(MyProject.java:10)\n    at
 com.myproject.module.MyProject.main(MyProject.java:6)",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:2"
```

```
}
```

O trecho de log a seguir mostra como a mesma mensagem é exibida com apenas uma única linha se você executar um contêiner do Amazon ECS que não esteja configurado para concatenar mensagens de log multinha.

```
{
    "log": "Dec 14 06:41:08 Exception in thread \"main\"
 java.lang.RuntimeException: Something has gone wrong, aborting!",
    "container_id": "xxxxxx-xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:3"
}
```

Exemplo: use um analisador integrado Fluent Bit

Neste exemplo, você concluirá as seguintes etapas:

1. Criar e carregar a imagem para um contêiner Fluent Bit.
2. Criar e carregar a imagem para uma aplicação de várias linhas de demonstração que executa, falha e gera um rastreamento de pilha de várias linhas.
3. Criar a definição de tarefa e executar a tarefa.
4. Exibir os logs para verificar se as mensagens que abrangem várias linhas aparecem concatenadas.

Criar e carregar a imagem para um contêiner Fluent Bit

Essa imagem incluirá um arquivo de configuração que faz referência ao analisador Fluent Bit.

1. Criar uma pasta com o nome `FluentBitDockerImage`.
2. Dentro da pasta `FluentBitDockerImage`, crie um arquivo de configuração personalizado que faça referência ao arquivo do analisador integrado Fluent Bit.

Para obter mais informações sobre o arquivo de configuração personalizado, consulte Especificar um arquivo de configuração personalizado no Guia do desenvolvedor do Amazon Elastic Container Service

a. Cole o seguinte conteúdo no arquivo:

```
[FILTER]
    name                  multiline
    match                 *
    multiline.key_content log
    multiline.parser      go
```

b. Salve o arquivo como `extra.conf`.

3. Dentro da pasta `FluentBitDockerImage`, crie o Dockerfile com a imagem Fluent Bit e os arquivos do analisador e de configuração que você criou.

a. Cole o seguinte conteúdo no arquivo:

```
FROM public.ecr.aws/aws-observability/aws-for-fluent-bit:latest
ADD extra.conf /extra.conf
```

b. Salve o arquivo como `Dockerfile`.

4. Usando o Dockerfile, crie uma imagem Fluent Bit personalizada com o arquivo de configuração personalizado incluído.

> **Note**
>
> É possível colocar o arquivo de configuração em qualquer lugar da imagem do Docker, exceto em `/fluent-bit/etc/fluent-bit.conf`, pois esse caminho de arquivo é usado pelo FireLens.

a. Crie a imagem: `docker build -t fluent-bit-multiline-image.`

Onde: `fluent-bit-multiline-image` é o nome da imagem neste exemplo.

b. Verifique se a imagem foi criada corretamente: `docker images –filter reference=fluent-bit-multiline-image`

Se tiver êxito, a saída mostrará a imagem e a etiqueta `latest`.

5. Carregue a imagem personalizada do Fluent Bit no Amazon Elastic Container Registry.

   a. Crie um repositório do Amazon ECR para armazenar a imagem: `aws ecr create-repository --repository-name fluent-bit-multiline-repo --region us-east-1`

      Onde: `fluent-bit-multiline-repo` é o nome do repositório e `us-east-1` é a região neste exemplo.

      A saída fornece os detalhes do novo repositório.

   b. Marque sua imagem com o valor `repositoryUri` da saída anterior: `docker tag` *`fluent-bit-multiline-image repositoryUri`*

      Exemplo: `docker tag fluent-bit-multiline-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

   c. Execute a imagem do Docker para verificar se ela é executada corretamente: `docker images –filter reference=`*`repositoryUri`*

      Na saída, o nome do repositório muda de fluent-bit-multiline-repo para o `repositoryUri`.

   d. Autentique-se no Amazon ECR executando o comando `aws ecr get-login-password` e especificando o ID de registro para o qual deseja fazer a autenticação: `aws ecr get-login-password | docker login --username AWS --password-stdin` *`registry ID`*`.dkr.ecr.`*`region`*`.amazonaws.com`

      Exemplo: `ecr get-login-password | docker login --username AWS --password-stdin xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com`

      Uma mensagem de login com êxito é exibida.

   e. Envie a imagem para o Amazon ECR: `docker push` *`registry ID`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`repository name`*

      Exemplo: `docker push xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

Crie e carregue a imagem para uma aplicação de várias linhas de demonstração

Essa imagem incluirá um arquivo de script Python que executa a aplicação e um arquivo de log de amostra.

1. Crie uma pasta denominada `multiline-app`: `mkdir multiline-app`
2. Crie um arquivo de script do Python.
   a. Na pasta `multiline-app`, crie um arquivo de texto e denomine-o `main.py`.
   b. Cole o seguinte conteúdo no arquivo:

```
import os
import time
file1 = open('/test.log', 'r')
Lines = file1.readlines()

count = 0

for i in range(10):
    print("app running normally...")
    time.sleep(1)

# Strips the newline character
for line in Lines:
    count += 1
    print(line.rstrip())
print(count)
print("app terminated.")
```

   c. Salve o arquivo `main.py`.
3. Crie um arquivo de log de exemplo.
   a. Na pasta `multiline-app`, crie um arquivo de texto e denomine-o `test.log`.
   b. Cole o seguinte conteúdo no arquivo:

```
panic: my panic

goroutine 4 [running]:
panic(0x45cb40, 0x47ad70)
```

```
  /usr/local/go/src/runtime/panic.go:542 +0x46c fp=0xc42003f7b8 sp=0xc42003f710
 pc=0x422f7c
main.main.func1(0xc420024120)
  foo.go:6 +0x39 fp=0xc42003f7d8 sp=0xc42003f7b8 pc=0x451339
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003f7e0
 sp=0xc42003f7d8 pc=0x44b4d1
created by main.main
  foo.go:5 +0x58

goroutine 1 [chan receive]:
runtime.gopark(0x4739b8, 0xc420024178, 0x46fcd7, 0xc, 0xc420028e17, 0x3)
  /usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc420053e30 sp=0xc420053e00
 pc=0x42503c
runtime.goparkunlock(0xc420024178, 0x46fcd7, 0xc, 0x1000f010040c217, 0x3)
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc420053e70 sp=0xc420053e30
 pc=0x42512e
runtime.chanrecv(0xc420024120, 0x0, 0xc420053f01, 0x4512d8)
  /usr/local/go/src/runtime/chan.go:506 +0x304 fp=0xc420053f20 sp=0xc420053e70
 pc=0x4046b4
runtime.chanrecv1(0xc420024120, 0x0)
  /usr/local/go/src/runtime/chan.go:388 +0x2b fp=0xc420053f50 sp=0xc420053f20
 pc=0x40439b
main.main()
  foo.go:9 +0x6f fp=0xc420053f80 sp=0xc420053f50 pc=0x4512ef
runtime.main()
  /usr/local/go/src/runtime/proc.go:185 +0x20d fp=0xc420053fe0 sp=0xc420053f80
 pc=0x424bad
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc420053fe8
 sp=0xc420053fe0 pc=0x44b4d1

goroutine 2 [force gc (idle)]:
runtime.gopark(0x4739b8, 0x4ad720, 0x47001e, 0xf, 0x14, 0x1)
  /usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc42003e768 sp=0xc42003e738
 pc=0x42503c
runtime.goparkunlock(0x4ad720, 0x47001e, 0xf, 0xc420000114, 0x1)
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc42003e7a8 sp=0xc42003e768
 pc=0x42512e
runtime.forcegchelper()
  /usr/local/go/src/runtime/proc.go:238 +0xcc fp=0xc42003e7e0 sp=0xc42003e7a8
 pc=0x424e5c
runtime.goexit()
```

```
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003e7e8
 sp=0xc42003e7e0 pc=0x44b4d1
created by runtime.init.4
  /usr/local/go/src/runtime/proc.go:227 +0x35

goroutine 3 [GC sweep wait]:
runtime.gopark(0x4739b8, 0x4ad7e0, 0x46fdd2, 0xd, 0x419914, 0x1)
  /usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc42003ef60 sp=0xc42003ef30
 pc=0x42503c
runtime.goparkunlock(0x4ad7e0, 0x46fdd2, 0xd, 0x14, 0x1)
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc42003efa0 sp=0xc42003ef60
 pc=0x42512e
runtime.bgsweep(0xc42001e150)
  /usr/local/go/src/runtime/mgcsweep.go:52 +0xa3 fp=0xc42003efd8
 sp=0xc42003efa0 pc=0x419973
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003efe0
 sp=0xc42003efd8 pc=0x44b4d1
created by runtime.gcenable
  /usr/local/go/src/runtime/mgc.go:216 +0x58
one more line, no multiline
```

c. Salve o arquivo `test.log`.

4. Dentro da pasta `multiline-app`, crie o Dockerfile.

   a. Cole o seguinte conteúdo no arquivo:

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
ADD test.log /test.log

RUN yum upgrade -y && yum install -y python3

WORKDIR /usr/local/bin

COPY main.py .

CMD ["python3", "main.py"]
```

   b. Salve o arquivo `Dockerfile`.

5. Usando o Dockerfile, crie uma imagem.

   a. Crie a imagem: `docker build -t multiline-app-image`

Onde: `multiline-app-image` é o nome da imagem neste exemplo.

b. Verifique se a imagem foi criada corretamente: `docker images –filter reference=multiline-app-image`

Se tiver êxito, a saída mostrará a imagem e a etiqueta `latest`.

6. Carregue a imagem no Amazon Elastic Container Registry.

a. Crie um repositório do Amazon ECR para armazenar a imagem: `aws ecr create-repository --repository-name multiline-app-repo --region us-east-1`

Onde: `multiline-app-repo` é o nome do repositório e `us-east-1` é a região neste exemplo.

A saída fornece os detalhes do novo repositório. Anote o valor do `repositoryUri`, pois ele será necessário na próxima etapa.

b. Marque sua imagem com o valor `repositoryUri` da saída anterior: `docker tag multiline-app-image repositoryUri`

Exemplo: `docker tag multiline-app-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/multiline-app-repo`

c. Execute a imagem do Docker para verificar se ela é executada corretamente: `docker images –filter reference=repositoryUri`

Na saída, o nome do repositório muda de `multiline-app-repo` para o valor `repositoryUri`.

d. Envie a imagem para o Amazon ECR: `docker push aws_account_id.dkr.ecr.region.amazonaws.com/repository name`

Exemplo: `docker push xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/multiline-app-repo`

Crie a definição da tarefa e execute a tarefa

1. Crie um arquivo de definição de tarefa com o nome de arquivo `multiline-task-definition.json`.
2. Cole o seguinte conteúdo no arquivo `multiline-task-definition.json`:

```
{
    "family": "firelens-example-multiline",
    "taskRoleArn": "task role ARN,
    "executionRoleArn": "execution role ARN",
    "containerDefinitions": [
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-
multiline-image:latest",
            "name": "log_router",
            "firelensConfiguration": {
                "type": "fluentbit",
                "options": {
                    "config-file-type": "file",
                    "config-file-value": "/extra.conf"
                }
            },
            "memoryReservation": 50
        },
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/multiline-app-
image:latest",
            "name": "app",
            "logConfiguration": {
                "logDriver": "awsfirelens",
                "options": {
                    "Name": "cloudwatch_logs",
                    "region": "us-east-1",
                    "log_group_name": "multiline-test/application",
                    "auto_create_group": "true",
                    "log_stream_prefix": "multiline-"
                }
            },
            "memoryReservation": 100
        }
    ],
    "requiresCompatibilities": ["FARGATE"],
    "networkMode": "awsvpc",
    "cpu": "256",
    "memory": "512"
}
```

Substitua o seguinte na definição de tarefa `multiline-task-definition.json`:

a. *`task role ARN`*

   Para encontrar o ARN da função de tarefa, acesse o console do IAM. Escolha Roles (Funções) e encontre a função de tarefa `ecs-task-role-for-firelens` que você criou. Escolha a função e copie o ARN que consta na seção Summary (Resumo).

b. *`execution role ARN`*

   Para encontrar o ARN da função de execução, acesse o console do IAM. Selecione Roles (Funções) e localize a função `ecsTaskExecutionRole`. Escolha a função e copie o ARN que consta na seção Summary (Resumo).

c. *`aws_account_id`*

   Para encontrar seu `aws_account_id`, faça login no AWS Management Console. Escolha seu nome de usuário no canto superior direito e copie o ID da conta.

d. *`us-east-1`*

   Substitua a região, se necessário.

3. Registre o arquivo de definição de tarefa: `aws ecs register-task-definition --cli-input-json file://multiline-task-definition.json --region us-east-1`
4. Abra o console em https://console.aws.amazon.com/ecs/v2.
5. No painel de navegação, selecione Task Definitions (Definições de tarefa) e, em seguida, escolha a família `firelens-example-multiline`, pois registramos a definição de tarefa para essa família na primeira linha da definição de tarefa acima.
6. Escolha a versão mais recente.
7. Escolha Implantar, Executar tarefa.
8. Na página Executar tarefa, para Cluster, escolha o cluster e, em seguida, em Rede, para Sub-redes, escolha as sub-redes disponíveis para sua tarefa.
9. Escolha Criar.

Verifique se as mensagens de log de várias linhas no Amazon CloudWatch parecem concatenadas

1. Abra o console do CloudWatch, em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, expanda Logs e escolha Log groups (Grupos de log).

3. Escolha o grupo de logs `multiline-test/applicatio`.
4. Escolha o log e veja as mensagens. As linhas que correspondem às regras no arquivo do analisador são concatenadas e aparecem como uma única mensagem.

   O trecho de log a seguir mostra um rastreamento de pilha Go concatenado em um único evento:

```
{
    "log": "panic: my panic\n\ngoroutine 4 [running]:\npanic(0x45cb40,
 0x47ad70)\n  /usr/local/go/src/runtime/panic.go:542 +0x46c fp=0xc42003f7b8
 sp=0xc42003f710 pc=0x422f7c\nmain.main.func1(0xc420024120)\n  foo.go:6
 +0x39 fp=0xc42003f7d8 sp=0xc42003f7b8 pc=0x451339\nruntime.goexit()\n  /usr/
local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003f7e0 sp=0xc42003f7d8
 pc=0x44b4d1\ncreated by main.main\n  foo.go:5 +0x58\n\ngoroutine 1 [chan receive]:
\nruntime.gopark(0x4739b8, 0xc420024178, 0x46fcd7, 0xc, 0xc420028e17, 0x3)\n  /usr/
local/go/src/runtime/proc.go:280 +0x12c fp=0xc420053e30 sp=0xc420053e00 pc=0x42503c
\nruntime.goparkunlock(0xc420024178, 0x46fcd7, 0xc, 0x1000f010040c217, 0x3)\n
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc420053e70 sp=0xc420053e30
 pc=0x42512e\nruntime.chanrecv(0xc420024120, 0x0, 0xc420053f01, 0x4512d8)\n
  /usr/local/go/src/runtime/chan.go:506 +0x304 fp=0xc420053f20 sp=0xc420053e70
 pc=0x4046b4\nruntime.chanrecv1(0xc420024120, 0x0)\n  /usr/local/go/src/runtime/
chan.go:388 +0x2b fp=0xc420053f50 sp=0xc420053f20 pc=0x40439b\nmain.main()\n
 foo.go:9 +0x6f fp=0xc420053f80 sp=0xc420053f50 pc=0x4512ef\nruntime.main()\n
  /usr/local/go/src/runtime/proc.go:185 +0x20d fp=0xc420053fe0 sp=0xc420053f80
 pc=0x424bad\nruntime.goexit()\n  /usr/local/go/src/runtime/asm_amd64.s:2337
 +0x1 fp=0xc420053fe8 sp=0xc420053fe0 pc=0x44b4d1\n\ngoroutine 2 [force gc
 (idle)]:\nruntime.gopark(0x4739b8, 0x4ad720, 0x47001e, 0xf, 0x14, 0x1)\n  /
usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc42003e768 sp=0xc42003e738
 pc=0x42503c\nruntime.goparkunlock(0x4ad720, 0x47001e, 0xf, 0xc420000114, 0x1)\n
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc42003e7a8 sp=0xc42003e768
 pc=0x42512e\nruntime.forcegchelper()\n  /usr/local/go/src/runtime/proc.go:238
 +0xcc fp=0xc42003e7e0 sp=0xc42003e7a8 pc=0x424e5c\nruntime.goexit()\n  /usr/
local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003e7e8 sp=0xc42003e7e0
 pc=0x44b4d1\ncreated by runtime.init.4\n  /usr/local/go/src/runtime/proc.go:227
 +0x35\n\ngoroutine 3 [GC sweep wait]:\nruntime.gopark(0x4739b8, 0x4ad7e0,
 0x46fdd2, 0xd, 0x419914, 0x1)\n  /usr/local/go/src/runtime/proc.go:280 +0x12c
 fp=0xc42003ef60 sp=0xc42003ef30 pc=0x42503c\nruntime.goparkunlock(0x4ad7e0,
 0x46fdd2, 0xd, 0x14, 0x1)\n  /usr/local/go/src/runtime/proc.go:286 +0x5e
 fp=0xc42003efa0 sp=0xc42003ef60 pc=0x42512e\nruntime.bgsweep(0xc42001e150)\n
  /usr/local/go/src/runtime/mgcsweep.go:52 +0xa3 fp=0xc42003efd8 sp=0xc42003efa0
 pc=0x419973\nruntime.goexit()\n  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1
 fp=0xc42003efe0 sp=0xc42003efd8 pc=0x44b4d1\ncreated by runtime.gcenable\n  /usr/
local/go/src/runtime/mgc.go:216 +0x58",
    "container_id": "xxxxxx-xxxxxx",
```

```
    "container_name": "app",
    "source": "stdout",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:2"
}
```

O trecho de log a seguir mostra como o mesmo evento aparece se você executar um contêiner ECS que não esteja configurado para concatenar mensagens de log de várias linhas. O campo de log contém uma única linha.

```
{
    "log": "panic: my panic",
    "container_id": "xxxxxx-xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:3"
```

> **Note**
>
> Se seus registros forem para arquivos de log em vez da saída padrão, recomendamos especificar os parâmetros de configuração `multiline.parser` e `multiline.key_content` no Plug-in de entrada final em vez do Filtro.

Implantar o Fluent Bit em contêineres do Windows no Amazon ECS

O Fluent Bit é um processador e roteador de logs rápido e flexível compatível com vários sistemas operacionais. Ele pode ser usado para encaminhar logs a vários destinos da AWS, como o Amazon CloudWatch Logs, Firehose, Amazon S3 e Amazon OpenSearch Service. O Fluent Bit é compatível com soluções comuns de parceiros, como Datadog, Splunk e servidores HTTP personalizados. Para obter mais informações sobre o Fluent Bit, consulte o site do Fluent Bit .

O AWS para Fluent Bit está disponível no Amazon ECR na Galeria Pública do Amazon ECR e em um repositório do Amazon ECR na maioria das regiões para alta disponibilidade. Para obter mais informações, consulte aws-for-fluent-bit no site do GitHub.

Este tutorial explica como implantar contêineres do Fluent Bit em instâncias do Windows em execução no Amazon ECS para transmitir logs gerados pelas tarefas do Windows para o Amazon CloudWatch para logs centralizados.

Este tutorial usa a seguinte abordagem:

- O Fluent Bit é executado como um serviço com a estratégia de programação do Daemon. Essa estratégia garante que uma única instância do Fluent Bit sempre seja executada nas instâncias de contêiner no cluster.
  - Recebe na porta 24224 usando o plug-in de entrada de encaminhamento.
  - Exponha a porta 24224 ao host para que o runtime do docker possa enviar logs para o Fluent Bit usando essa porta exposta.
  - Tem uma configuração que permite que o Fluent Bit envie os registros de logs para destinos especificados.
- Inicie todos os outros contêineres de tarefas do Amazon ECS usando o driver de log fluentd. Para obter mais informações, consulte Fluentd logging driver (Driver de log do Fluentd) no site de documentação do Docker.
  - O Docker se conecta ao soquete TCP 24224 no localhost dentro do namespace do host.
  - O agente do Amazon ECS adiciona rótulos aos contêineres, incluindo o nome do cluster, o nome da família da definição de tarefa, o número da revisão da definição de tarefa, o ARN da tarefa e o nome do contêiner. As mesmas informações são adicionadas ao registro de logs usando a opção labels (rótulos) do driver de logs do docker fluentd. Para obter mais informações, consulte labels, labels-regex, env e env-regex no site de documentação do Docker.
  - Como a opção `async` do driver de logs fluentd está definida como `true`, quando o contêiner do Fluent Bit é reiniciado, o docker armazena os logs em buffer até que o contêiner do Fluent Bit seja reiniciado. É possível aumentar o limite de armazenamento em buffer definindo a opção fluentd-buffer-limit. Para obter mais informações, consulte fluentd-buffer-limit no site da documentação do Docker.

O fluxo de trabalho é o seguinte:

- O contêiner do Fluent Bit inicia e recebe na porta 24224, que está exposta ao host.
- O Fluent Bit usa as credenciais do perfil do IAM da tarefa especificadas na definição da tarefa.
- Outras tarefas iniciadas na mesma instância usam o driver de logs do docker do fluentd para se conectar ao contêiner do Fluent Bit na porta 24224.

- Quando os contêineres da aplicação geram logs, o runtime do docker marca esses registros, adiciona metadados especificados nos rótulos e os encaminha pela porta 24224 no namespace do host.
- O Fluent Bit recebe o registro de log na porta 24224 porque ela está exposta ao namespace do host.
- O Fluent Bit executa seu processamento interno e roteia os logs, conforme especificado.

Este tutorial usa a configuração padrão do Fluent Bit do CloudWatch, que faz o seguinte:

- Cria um novo grupo de logs para cada cluster e família de definição de tarefa.
- Cria um novo fluxo de logs para cada contêiner de tarefas no grupo de logs gerado acima sempre que uma nova tarefa é iniciada. Cada fluxo será marcado com o ID da tarefa à qual o contêiner pertence.
- Adiciona metadados, incluindo o nome do cluster, o ARN da tarefa, o nome do contêiner da tarefa, a família de definição de tarefa e o número da revisão da definição de tarefa em cada entrada de log.

  Por exemplo, se você tiver `task_1` com `container_1` e `container_2` e `task_2` com `container_3`, os fluxos de logs do CloudWatch serão os seguintes:

  - `/aws/ecs/windows.ecs_task_1`

    `task-out.`*`TASK_ID`*`.container_1`

    `task-out.`*`TASK_ID`*`.container_2`
  - `/aws/ecs/windows.ecs_task_2`

    `task-out.`*`TASK_ID`*`.container_3`

> **Note**
>
> É possível usar endpoints de serviço de pilha dupla para interagir com o Amazon ECS via AWS CLI, SDKs e API do Amazon ECS sobre IPv4 e IPv6. Para obter mais informações, consulte Usar endpoints de pilha dupla do Amazon ECS.

Etapas

- Pré-requisitos
- Etapa 1: criar os perfis de acesso do IAM
- Etapa 2: criar uma instância de contêiner do Windows do Amazon ECS
- Etapa 3: configurar o Fluent Bit
- Etapa 4: registrar uma definição de tarefa do Fluent Bit no Windows que roteia os logs para o CloudWatch
- Etapa 5: executar a definição de tarefa ecs-windows-fluent-bit como um serviço do Amazon ECS usando a estratégia de programação do daemon
- Etapa 6: registrar uma definição de tarefa do Windows que gere os logs
- Etapa 7: executar a definição de tarefa windows-app-task
- Etapa 8: verificar os logs no CloudWatch
- Etapa 9: limpar

Pré-requisitos

Este tutorial pressupõe que os seguintes pré-requisitos foram concluídos:

- A versão mais recente da AWS CLI está instalada e configurada. Para obter mais informações, consulte Instalar ou atualizar para a versão mais recente da AWS CLI.
- A imagem do contêiner `aws-for-fluent-bit` está disponível para os seguintes sistemas operacionais Windows:
  - Windows Server 2019 Core
  - Windows Server 2019 Full
  - Windows Server 2022 Core
  - Windows Server 2022 Full
- As etapas em Configuração para usar o Amazon ECS foram concluídas.
- Você tem um cluster. Neste tutorial, o nome do cluster é FluentBit-cluster.
- Você tem uma VPC com uma sub-rede pública em que a instância do EC2 será iniciada. É possível usar a VPC padrão. Você também pode usar uma sub-rede privada que permita que os endpoints do Amazon CloudWatch alcancem a sub-rede. Para obter mais informações sobre endpoints do Amazon CloudWatch, consulte Endpoints e cotas do Amazon CloudWatch na Referência geral da AWS. Para obter mais informações sobre como usar o assistente da Amazon VPC para criar uma VPC, consulte the section called “Criar uma nuvem privada virtual”.

Etapa 1: criar os perfis de acesso do IAM

Crie os perfis do IAM do Amazon ECS.

1. Crie o perfil da instância de contêiner do Amazon ECS de nome "ecsInstanceRole". Para obter mais informações, consulte Perfil do IAM de instância de contêiner do Amazon ECS.
2. Crie um perfil do IAM para a tarefa do Fluent Bit denominada `fluentTaskRole`. Para obter mais informações, consulte the section called “Função do IAM de tarefa”.

   As permissões do IAM concedidas no perfil do IAM são assumidas pelos contêineres das tarefas. Para permitir que o Fluent Bit envie logs para o CloudWatch, é preciso anexar as seguintes permissões ao perfil do IAM da tarefa.

   JSON

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
       {
           "Effect": "Allow",
           "Action": [
               "logs:CreateLogStream",
               "logs:CreateLogGroup",
               "logs:DescribeLogStreams",
               "logs:PutLogEvents"
           ],
           "Resource": "*"
       }
       ]
   }
   ```

3. Anexe a política ao perfil.

   a. Salve o conteúdo acima em um arquivo denominado `fluent-bit-policy.json`.

   b. Execute o comando a seguir para anexar a política em linha ao perfil `fluentTaskRole` do IAM.

   ```
   aws iam put-role-policy --role-name fluentTaskRole --policy-name
    fluentTaskPolicy --policy-document file://fluent-bit-policy.json
   ```

Etapa 2: criar uma instância de contêiner do Windows do Amazon ECS

Crie uma instância de contêiner do Windows do Amazon ECS.

Para criar uma instância do Amazon ECS

1. Use o comando `aws ssm get-parameters` para recuperar o ID da AMI para a região que hospeda sua VPC. Para obter mais informações, consulte Recuperar os metadados da AMI otimizada para Amazon ECS.
2. Use o console do Amazon EC2 para iniciar a instância.
   a. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
   b. Na barra de navegação, selecione a Região a ser usada.
   c. No Painel do EC2, escolha Launch Instance (Iniciar instância).
   d. Em Name (Nome), insira um nome exclusivo.
   e. Em Application and OS Images (Amazon Machine Image) (Imagens de aplicações e SO [Amazon Machine Image]), escolha a AMI que você recuperou na primeira etapa.
   f. Em tipo de instância, escolha `t3.xlarge`.
   g. Em Key pair (login) (Par de chaves [login]), escolha um par de chaves.
   h. Em Network settings (Configurações de rede), em Security group (Grupo de segurança), selecione um grupo de segurança existente ou crie outro.
   i. Em Network settings (Configurações de rede), em Auto-assign Public IPC (Atribuir IP público automaticamente), selecione Enable (Habilitar).
   j. Em Advanced details (Detalhes avançados), em IAM instance profile (Perfil de instância do IAM), escolha ecsInstanceRole.
   k. Configure a instância de contêiner do Amazon ECS com os dados de usuário a seguir. Em Advanced Details (Detalhes avançados), cole o seguinte script no campo User data (Dados do usuário), substituindo *`cluster_name`* pelo nome do cluster.

```
<powershell>
Import-Module ECSTools
Initialize-ECSAgent -Cluster cluster-name -EnableTaskENI -EnableTaskIAMRole -
LoggingDrivers '["awslogs","fluentd"]'
</powershell>
```

   l. Quando estiver pronto, selecione o campo de confirmação e, então, escolha Launch Instances.

m. Uma página de confirmação informa que sua instância está sendo executada. Selecione Visualizar instâncias para fechar a página de confirmação e voltar ao console.

Etapa 3: configurar o Fluent Bit

É possível usar a configuração padrão a seguir fornecida pela AWS para começar rapidamente:

- Amazon CloudWatch, que é baseado no plug-in do Fluent Bit para Amazon CloudWatch no Manual oficial do Fluent Bit.

Como alternativa, é possível usar outras configurações padrão fornecidas pela AWS. Para obter mais informações, consulte Overriding the entrypoint for the Windows image (Substituir o ponto de entrada da imagem do Windows) no site `aws-for-fluent-bit` do Github.

A configuração padrão do Fluent Bit do Amazon CloudWatch é mostrada abaixo.

Substitua as seguintes variáveis:

- *`região`* pela região para a qual você deseja enviar logs do Amazon CloudWatch.

```
[SERVICE]
    Flush               5
    Log_Level           info
    Daemon              off

[INPUT]
    Name                forward
    Listen              0.0.0.0
    Port                24224
    Buffer_Chunk_Size   1M
    Buffer_Max_Size     6M
    Tag_Prefix          ecs.

# Amazon ECS agent adds the following log keys as labels to the docker container.
# We would use fluentd logging driver to add these to log record while sending it to
 Fluent Bit.
[FILTER]
    Name                modify
    Match               ecs.*
    Rename              com.amazonaws.ecs.cluster ecs_cluster
```

```
    Rename              com.amazonaws.ecs.container-name ecs_container_name
    Rename              com.amazonaws.ecs.task-arn ecs_task_arn
    Rename              com.amazonaws.ecs.task-definition-family
 ecs_task_definition_family
    Rename              com.amazonaws.ecs.task-definition-version
 ecs_task_definition_version

[FILTER]
    Name                rewrite_tag
    Match               ecs.*
    Rule                $ecs_task_arn ^([a-z-:0-9]+)/([a-zA-Z0-9-_]+)/([a-z0-9]+)$
 out.$3.$ecs_container_name false
    Emitter_Name        re_emitted

[OUTPUT]
    Name                cloudwatch_logs
    Match               out.*
    region              region
    log_group_name      fallback-group
    log_group_template  /aws/ecs/$ecs_cluster.$ecs_task_definition_family
    log_stream_prefix   task-
    auto_create_group   On
```

Cada log que entra no Fluent Bit tem uma tag que você especifica ou é gerada automaticamente quando você não fornece uma. As tags podem ser usadas para rotear diferentes logs para destinos diferentes. Para obter informações adicionais, consulte Tag no Manual oficial do Fluent Bit.

A configuração do Fluent Bit descrita acima tem as seguintes propriedades:

- O plug-in de entrada de encaminhamento recebe o tráfego de entrada na porta TCP 24224.
- Cada entrada de log recebida nessa porta tem uma tag que o plug-in de entrada de encaminhamento modifica para prefixar o registro com a string `ecs.`.
- O pipeline interno do Fluent Bit roteia a entrada de log para modificar o filtro usando o regex Match (Corresponder). Esse filtro substitui as chaves no registro de log JSON pelo formato que o Fluent Bit pode consumir.
- A entrada de log modificada é consumida pelo filtro rewrite_tag. Esse filtro altera a tag do registro de log para o formato out.*TASK_ID*.*CONTAINER_NAME*.
- A nova tag será roteada para o plug-in cloudwatch_logs de saída, que cria os grupos e fluxos de logs, conforme descrito anteriormente, usando as opções `log_group_template` e

`log_stream_prefix` do plug-in de saída do CloudWatch. Para obter informações adicionais, consulte Configuration parameters (Parâmetros de configuração) no Manual oficial do Fluent Bit.

Etapa 4: registrar uma definição de tarefa do Fluent Bit no Windows que roteia os logs para o CloudWatch

Registre uma definição de tarefa do Fluent Bit no Windows que roteia os logs para o CloudWatch.

> **Note**
>
> Essa definição de tarefa expõe a porta 24224 do contêiner do Fluent Bit para a porta 24224 do host. Verifique se essa porta não está aberta em seu grupo de segurança de instâncias do EC2 para impedir o acesso externo.

Para registrar uma definição de tarefa

1. Crie um arquivo denominado `fluent-bit.json` com o seguinte conteúdo:

   Substitua as seguintes variáveis:

   - *`task-iam-role`* pelo nome do recurso da Amazon (ARN) do perfil do IAM da tarefa
   - *`region`* (região) pela região em que a tarefa é executada

```
{
  "family": "ecs-windows-fluent-bit",
  "taskRoleArn": "task-iam-role",
  "containerDefinitions": [
    {
      "name": "fluent-bit",
      "image": "public.ecr.aws/aws-observability/aws-for-fluent-
bit:windowsservercore-latest",
      "cpu": 512,
      "portMappings": [
        {
          "hostPort": 24224,
          "containerPort": 24224,
          "protocol": "tcp"
        }
```

```
      ],
      "entryPoint": [
        "Powershell",
        "-Command"
      ],
      "command": [
        "C:\\entrypoint.ps1 -ConfigFile C:\\ecs_windows_forward_daemon\
\cloudwatch.conf"
      ],
      "environment": [
        {
          "name": "AWS_REGION",
          "value": "region"
        }
      ],
      "memory": 512,
      "essential": true,
      "logConfiguration": {
        "logDriver": "awslogs",
        "options": {
          "awslogs-group": "/ecs/fluent-bit-logs",
          "awslogs-region": "region",
          "awslogs-stream-prefix": "flb",
          "awslogs-create-group": "true"
        }
      }
    }
  ],
  "memory": "512",
  "cpu": "512"
}
```

2. Use o comando a seguir para registrar a definição de tarefa.

```
aws ecs register-task-definition --cli-input-json file://fluent-bit.json --
region region
```

É possível listar as definições de tarefa para a conta executando o comando `list-task-definitions`. A saída exibe os valores da família e da revisão que você pode usar com `run-task` ou `start-task`.

Etapa 5: executar a definição de tarefa **`ecs-windows-fluent-bit`** como um serviço do Amazon ECS usando a estratégia de programação do daemon

Depois de registrar uma definição de tarefa para sua conta, será possível executar uma tarefa no cluster. Para este tutorial, você executa uma única instância da definição de tarefa `ecs-windows-fluent-bit:1` no cluster `FluentBit-cluster`. Execute a tarefa em um serviço que use a estratégia de programação do daemon, o que garante que uma única instância do Fluent Bit sempre seja executada em cada instância de contêiner.

Para executar uma tarefa

1. Execute o comando a seguir para iniciar a definição de tarefa `ecs-windows-fluent-bit:1`(registrada na etapa anterior) como um serviço.

   > **Note**
   >
   > Essa definição de tarefa usa o driver de log `awslogs`. Sua instância de contêiner precisa ter as permissões necessárias.

   Substitua as seguintes variáveis:

   - *`region`* (região) pela região em que o serviço é executado

   ```
   aws ecs create-service \
       --cluster FluentBit-cluster \
       --service-name FluentBitForwardDaemonService \
       --task-definition ecs-windows-fluent-bit:1 \
       --launch-type EC2 \
       --scheduling-strategy DAEMON \
       --region region
   ```

2. Execute o comando a seguir para listar as tarefas.

   Substitua as seguintes variáveis:

   - *`region`* (região) pela região em que as tarefas de serviço são executadas

```
aws ecs list-tasks --cluster FluentBit-cluster --region region
```

Etapa 6: registrar uma definição de tarefa do Windows que gere os logs

Registre uma definição de tarefa do Windows que gere os logs. Essa definição de tarefa implanta uma imagem de contêiner do Windows que gravará um número incremental no `stdout` a cada segundo.

A definição de tarefa usa o driver de log fluentd que se conecta à porta 24224 que o plug-in do Fluent Bit recebe. O agente do Amazon ECS rotula cada contêiner do Amazon ECS com tags, incluindo o nome do cluster, o ARN da tarefa, o nome da família da definição de tarefa, o número da revisão da definição de tarefa e o nome do contêiner da tarefa. Esses rótulos de valores-chave são passados para o Fluent Bit.

> **Note**
>
> Essa tarefa usa o modo de rede `default`. Entretanto, também é possível usar o modo de rede `awsvpc` com a tarefa.

Para registrar uma definição de tarefa

1. Crie um arquivo denominado `windows-app-task.json` com o seguinte conteúdo:

```
{
  "family": "windows-app-task",
  "containerDefinitions": [
    {
      "name": "sample-container",
      "image": "mcr.microsoft.com/windows/servercore:ltsc2019",
      "cpu": 512,
      "memory": 512,
      "essential": true,
      "entryPoint": [
        "Powershell",
        "-Command"
      ],
      "command": [
```

```
        "$count=1;while(1) { Write-Host $count; sleep 1; $count=$count+1;}"
      ],
      "logConfiguration": {
        "logDriver": "fluentd",
        "options": {
          "fluentd-address": "localhost:24224",
          "tag": "{{ index .ContainerLabels \"com.amazonaws.ecs.task-definition-
family\" }}",
          "fluentd-async": "true",
          "labels": "com.amazonaws.ecs.cluster,com.amazonaws.ecs.container-
name,com.amazonaws.ecs.task-arn,com.amazonaws.ecs.task-definition-
family,com.amazonaws.ecs.task-definition-version"
        }
      }
    }
  ],
  "memory": "512",
  "cpu": "512"
}
```

2. Use o comando a seguir para registrar a definição de tarefa.

   Substitua as seguintes variáveis:

   - *`region`* (região) pela região em que a tarefa é executada

   ```
   aws ecs register-task-definition --cli-input-json file://windows-app-task.json --
   region region
   ```

   É possível listar as definições de tarefa para a conta executando o comando `list-task-definitions`. A saída exibe os valores da família e da revisão que você pode usar com `run-task` ou `start-task`.

Etapa 7: executar a definição de tarefa **`windows-app-task`**

Depois de registrar a definição de tarefa `windows-app-task`, execute-a no cluster `FluentBit-cluster`.

Para executar uma tarefa

1. Execute a definição de tarefa `windows-app-task:1` que você registrou na etapa anterior.

   Substitua as seguintes variáveis:

   - *`region`* (região) pela região em que a tarefa é executada

   ```
   aws ecs run-task --cluster FluentBit-cluster --task-definition windows-app-task:1
    --count 2 --region region
   ```

2. Execute o comando a seguir para listar as tarefas.

   ```
   aws ecs list-tasks --cluster FluentBit-cluster
   ```

Etapa 8: verificar os logs no CloudWatch

Para verificar a configuração do Fluent Bit, verifique os seguintes grupos de log no console do CloudWatch:

- `/ecs/fluent-bit-logs`: este é o grupo de logs que corresponde ao contêiner do daemon do Fluent Bit que está sendo executado na instância de contêiner.
- `/aws/ecs/FluentBit-cluster.windows-app-task`: este é o grupo de logs que corresponde a todas as tarefas iniciadas para a família de definição de tarefa `windows-app-task` dentro do cluster `FluentBit-cluster`.

  `task-out.`*`FIRST_TASK_ID`*`.sample-container`: este fluxo de logs contém todos os logs gerados pela primeira instância da tarefa no contêiner de tarefas do contêiner de amostra.

  `task-out.`*`SECOND_TASK_ID`*`.sample-container`: este fluxo de logs contém todos os logs gerados pela segunda instância da tarefa no contêiner de tarefas do contêiner de amostra.

O fluxo de logs `task-out.`*`TASK_ID`*`.sample-container` tem campos semelhantes aos seguintes:

```
{
    "source": "stdout",
```

```
    "ecs_task_arn": "arn:aws:ecs:region:0123456789012:task/FluentBit-
cluster/13EXAMPLE",
    "container_name": "/ecs-windows-app-task-1-sample-container-cEXAMPLE",
    "ecs_cluster": "FluentBit-cluster",
    "ecs_container_name": "sample-container",
    "ecs_task_definition_version": "1",
    "container_id": "61f5e6EXAMPLE",
    "log": "10",
    "ecs_task_definition_family": "windows-app-task"
}
```

Para verificar a configuração do Fluent Bit

1. Abra o console do CloudWatch, em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Grupos de logs. Certifique-se de que você esteja na região em que implantou o Fluent Bit para seus contêineres.

   Na lista de grupos de log na Região da AWS, você deve ver o seguinte:

   - `/ecs/fluent-bit-logs`
   - `/aws/ecs/FluentBit-cluster.windows-app-task`

   Caso você veja esses grupos de logs, a configuração do Fluent Bit estará verificada.

Etapa 9: limpar

Ao concluir este tutorial, limpe os recursos associados a ele para evitar cobranças por recursos que você não está usando.

Para limpar os recursos do tutorial

1. Interrompa as tarefas `windows-simple-task` e `ecs-fluent-bit`. Para obter mais informações, consulte the section called “Interrupção de uma tarefa”.
2. Execute o comando a seguir para excluir o grupo de logs `/ecs/fluent-bit-logs`. Para obter mais informações sobre a exclusão de grupos de logs, consulte delete-log-group na Referência do AWS Command Line Interface.

   ```
   aws logs delete-log-group --log-group-name /ecs/fluent-bit-logs
   ```

```
aws logs delete-log-group --log-group-name /aws/ecs/FluentBit-cluster.windows-app-task
```

3. Execute o comando a seguir para encerrar a instância.

```
aws ec2 terminate-instances --instance-ids instance-id
```

4. Execute o comando a seguir para excluir os perfis do IAM.

```
aws iam delete-role --role-name ecsInstanceRole
aws iam delete-role --role-name fluentTaskRole
```

5. Execute o comando a seguir para excluir o cluster do Amazon ECS.

```
aws ecs delete-cluster --cluster FluentBit-cluster
```

Usar gMSA para contêineres do Linux do EC2 no Amazon ECS

O Amazon ECS oferece suporte à autenticação do Active Directory para contêineres do Linux no EC2 por meio de um tipo especial de conta de serviço denominada Conta de serviço gerenciada por grupo (gMSA).

Aplicações de rede baseadas no Linux, como as aplicações .NET Core, podem usar o Active Directory para facilitar o gerenciamento da autenticação e da autorização entre usuários e serviços. É possível usar esse recurso criando aplicações que se integram ao Active Directory e são executadas em servidores associados ao domínio. Mas, como os contêineres do Linux não podem ser associados a um domínio, você precisa configurar um contêiner do Linux para ser executado com o gMSA.

Um contêiner do Linux executado com gMSA depende do daemon `credentials-fetcher` executado na instância do Amazon EC2 hospedeira do contêiner. Ou seja, o daemon recupera as credenciais do gMSA do controlador de domínio do Active Directory e, em seguida, transfere essas credenciais para a instância de contêiner. Para obter mais informações sobre contas de serviço, consulte Criação de gMSAs para contêineres do Windows no site do Microsoft Learn.

Considerações

Considere o seguinte antes de usar gMSA para contêineres do Linux:

- Se seus contêineres são executados no EC2, é possível usar gMSA para contêineres do Windows e contêineres do Linux. Para obter informações sobre como usar gMSA no contêiner do Linux no Fargate, consulte Uso de gMSA em contêineres do Linux no Fargate.
- Talvez você precise de um computador Windows associado ao domínio para concluir os pré-requisitos. Por exemplo, é possível que seja preciso um computador Windows associado ao domínio para criar o gMSA no Active Directory com o PowerShell. As ferramentas PowerShell do Active Director RSAT estão disponíveis somente para Windows. Para obter mais informações, consulte Instalação das ferramentas de administração do Active Directory.
- Você escolhe entre gMSA sem domínio e associar cada instância a um único domínio. Ao usar gMSA sem domínio, a instância de contêiner não será associada ao domínio, outras aplicações na instância não podem usar as credenciais para acessar o domínio e as tarefas que associam domínios diferentes podem ser executadas na mesma instância.

  Em seguida, escolha o armazenamento de dados para o CredSpec e, opcionalmente, para as credenciais de usuário do Active Directory para o gMSA sem domínio.

  O Amazon ECS usa um arquivo de especificação de credenciais do Active Directory (CredSpec). Esse arquivo contém os metadados de gMSA usados para propagar o contexto da conta de gMSA para o contêiner. Você gera o arquivo CredSpec e o armazena em uma das opções de armazenamento do CredSpec na tabela a seguir, específica para o sistema operacional das instâncias de contêiner. Para usar o método sem domínio, uma seção opcional no arquivo CredSpec pode especificar credenciais em uma das opções de armazenamento de domainless user credentials na tabela a seguir, específica para o sistema operacional das instâncias de contêiner.

Local de armazenamento	Linux	Windows
Amazon Simple Storage Service	CredSpec	CredSpec
AWS Secrets Manager	credenciais do usuário sem domínio	credenciais do usuário sem domínio
Armazenamento de parâmetros do Amazon EC2 Systems Manager	CredSpec	CredSpec, credenciais do usuário sem domínio

Local de armazenamento	Linux	Windows
Arquivo local	N/D	CredSpec

Pré-requisitos

Antes de usar o recurso gMSA para contêineres do Linux com o Amazon ECS, verifique se você concluiu o seguinte:

- Você configurou um domínio do Active Directory com os recursos que deseja que seus contêineres acessem. O Amazon ECS oferece suporte às configurações a seguir:
  - Um Active Directory AWS Directory Service. O AWS Directory Service é um Active Directory gerenciado pela AWS e hospedado no Amazon EC2. Para obter mais informações, consulte Conceitos básicos do Microsoft AD gerenciado pela AWS no Guia de administração do AWS Directory Service.
  - Um Active Directory on-premises. Você deve garantir que a instância de contêiner Linux do Amazon ECS possa se associar ao domínio. Para obter mais informações, consulte AWS Direct Connect.
- Você tem uma conta de gMSA existente no Active Directory. Para obter mais informações, consulte Usar gMSA para contêineres do Linux do EC2 no Amazon ECS.
- Você instalou e está executando o daemon `credentials-fetcher` em uma instância de contêiner do Linux do Amazon ECS. Você também adicionou um conjunto inicial de credenciais ao daemon `credentials-fetcher` para se autenticar no Active Directory.

  > **Note**
  >
  > O daemon `credentials-fetcher` só está disponível para Amazon Linux 2023 e Fedora 37 e versões posteriores. O daemon não está disponível para o Amazon Linux 2. Para obter mais informações, consulte aws/credentials-fetcher no GitHub.

- Você configurou as credenciais do daemon `credentials-fetcher` para se autenticar no Active Directory. As credenciais devem ser um membro do grupo de segurança do Active Directory que tenha acesso à conta gMSA. Existem várias opções em Decida se você quer associar as instâncias ao domínio ou usar gMSA sem domínio..
- Você adicionou as permissões necessárias do IAM. As permissões necessárias dependem dos métodos escolhidos para as credenciais iniciais e para armazenar a especificação da credencial:

- Se você usa o gMSA sem domínio para as credenciais iniciais, as permissões do IAM para o AWS Secrets Manager são necessárias no perfil de execução da tarefa.
- Se você armazenar a especificação da credencial no SSM Parameter Store, as permissões do IAM para o Amazon EC2 Systems Manager Parameter Store serão necessárias no perfil de execução da tarefa.
- Se você armazenar a especificação da credencial no Amazon S3, as permissões do IAM para o Amazon Simple Storage Service serão necessárias no perfil de execução da tarefa.

Configuração de contêineres do Linux compatíveis com o gMSA no Amazon ECS

Preparar a infraestrutura

As etapas a seguir são considerações e configurações que são executadas uma vez. Depois de concluir essas etapas, será possível automatizar a criação de instâncias de contêiner para reutilizar essa configuração.

Decida como as credenciais iniciais serão fornecidas e configure os dados do usuário do EC2 em um modelo de execução do EC2 reutilizável para instalar o daemon `credentials-fetcher`.

1. Decida se você quer associar as instâncias ao domínio ou usar gMSA sem domínio.
   - Associar instâncias do EC2 ao domínio do Active Directory

     - Associar as instâncias por dados do usuário

       Adicione as etapas de associação ao domínio do Active Directory aos seus dados de usuário do EC2 em um modelo de execução do EC2. Vários grupos do Amazon EC2 Auto Scaling podem usar o mesmo modelo de execução.

       É possível usar as etapas em Associar a um domínio do Active Directory ou FreeIPA no Fedora Docs.
   - Crie um usuário do Active Directory para gMSA sem domínio

     O daemon `credentials-fetcher` tem um recurso chamado gMSA sem domínio. Esse recurso requer um domínio, mas a instância do EC2 não precisa estar associada ao domínio. Ao usar gMSA sem domínio, a instância de contêiner não será associada ao domínio, outras aplicações na instância não podem usar as credenciais para acessar o

domínio e as tarefas que associam domínios diferentes podem ser executadas na mesma instância. Em vez disso, você fornece o nome de um segredo no AWS Secrets Manager no arquivo CredSpec. O segredo deve conter um nome de usuário, uma senha e o domínio no qual fazer login.

Há suporte para esse recurso e ele pode ser usado com contêineres Linux e Windows.

Esse recurso é semelhante ao recurso gMSA support for non-domain-joined container hosts. Para obter mais informações sobre o recurso do Windows, consulte Arquitetura e melhorias do gMSA no site do Microsoft Learn.

a. Crie um usuário no seu domínio do Active Directory. O usuário no Active Directory deve ter permissão para acessar as contas de serviço do gMSA que você usa nas tarefas.

b. Crie um segredo no AWS Secrets Manager, depois de criar o usuário no Active Directory. Para obter mais informações, consulte Criar um segredo no AWS Secrets Manager.

c. Insira o nome de usuário, a senha e o domínio do usuário nos pares de valores-chave JSON denominados `username`, `password` e `domainName`, respectivamente.

```
{"username":"username","password":"passw0rd", "domainName":"example.com"}
```

d. Adicione a configuração ao arquivo CredSpec da conta de serviço. A `HostAccountConfig` adicional contém o nome do recurso da Amazon (ARN) do segredo no Secrets Manager

   No Windows, o `PluginGUID` deve corresponder ao GUID no trecho de exemplo a seguir. No Linux, o `PluginGUID` é ignorado. Substitua `MySecret` pelo exemplo com o nome do recurso da Amazon (ARN) do seu segredo.

```
    "ActiveDirectoryConfig": {
        "HostAccountConfig": {
            "PortableCcgVersion": "1",
            "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
            "PluginInput": {
                "CredentialArn": "arn:aws:secretsmanager:aws-
region:111122223333:secret:MySecret"
            }
        }
```

e. O recurso gMSA sem domínio precisa de permissões adicionais no perfil de execução da tarefa. Siga a etapa (Opcional) Segredo de gMSA sem domínio.

2. Configurar instâncias e instalar o daemon **`credentials-fetcher`**

É possível instalar o daemon `credentials-fetcher` com um script de dados do usuário em seu modelo de inicialização do EC2. Os exemplos a seguir demonstram dois tipos de dados do usuário, `cloud-config` YAML ou script de bash. Esses exemplos são para o Amazon Linux 2023 (AL2023). Substitua `MyCluster` pelo nome do cluster do Amazon ECS ao qual você deseja que essas instâncias se associem.

- **`cloud-config`** YAML

```
Content-Type: text/cloud-config
package_reboot_if_required: true
packages:
  # prerequisites
  - dotnet
  - realmd
  - oddjob
  - oddjob-mkhomedir
  - sssd
  - adcli
  - krb5-workstation
  - samba-common-tools
  # https://github.com/aws/credentials-fetcher gMSA credentials management for
 containers
  - credentials-fetcher
write_files:
# configure the ECS Agent to join your cluster.
# replace MyCluster with the name of your cluster.
- path: /etc/ecs/ecs.config
  owner: root:root
  permissions: '0644'
  content: |
    ECS_CLUSTER=MyCluster
    ECS_GMSA_SUPPORTED=true
runcmd:
# start the credentials-fetcher daemon and if it succeeded, make it start after
 every reboot
- "systemctl start credentials-fetcher"
- "systemctl is-active credentials-fetcher && systemctl enable credentials-
fetcher"
```

- Script bash

  Se você se sentir mais confortável com scripts de bash e tiver várias variáveis nas quais escrever em `/etc/ecs/ecs.config`, use o formato `heredoc` a seguir. Esse formato grava tudo entre as linhas que começam com cat e `EOF` no arquivo de configuração.

```
#!/usr/bin/env bash
set -euxo pipefail

# prerequisites
timeout 30 dnf install -y dotnet realmd oddjob oddjob-mkhomedir sssd adcli
 krb5-workstation samba-common-tools
# install https://github.com/aws/credentials-fetcher gMSA credentials
 management for containers
timeout 30 dnf install -y credentials-fetcher

# start credentials-fetcher
systemctl start credentials-fetcher
systemctl is-active credentials-fetcher && systemctl enable credentials-fetcher

cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_GMSA_SUPPORTED=true
EOF
```

Há variáveis de configuração opcionais para o daemon `credentials-fetcher` que pode ser configurado em `/etc/ecs/ecs.config`. Recomendamos que você defina as variáveis nos dados do usuário no bloco YAML ou `heredoc` forma semelhante aos exemplos anteriores. Isso evitará problemas com a configuração parcial que podem ocorrer com a edição de um arquivo várias vezes. Para obter mais informações sobre a configuração do agente de ECS, consulte Agente de contêiner do Amazon ECS no GitHub.

- Opcionalmente, é possível usar a variável `CREDENTIALS_FETCHER_HOST` se alterar a configuração do daemon `credentials-fetcher` para mover o soquete para outro local.

Configuração de permissões e segredos

Execute as etapas a seguir uma vez para cada aplicação e cada definição de tarefa. Use as práticas recomendadas de concessão de privilégio mínimo e restrinja as permissões usadas na política. Dessa forma, cada tarefa só pode ler os segredos de que precisa.

1. (Opcional) Segredo de gMSA sem domínio

   Se você usar o método sem domínio em que a instância não está associada ao domínio, siga esta etapa.

   É preciso adicionar as permissões a seguir como uma política em linha ao perfil do IAM de execução da tarefa. Isso dá ao daemon `credentials-fetcher` acesso ao segredo do Secrets Manager. Substitua o exemplo `MySecret` pelo nome do recurso da Amazon (ARN) do seu segredo na lista `Resource`.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                   "arn:aws:ssm:arn:aws: 111122223333:secret:MySecret"
            ]
        }
    ]
}
```

> **Note**
>
> Se você usar sua própria chave KMS para criptografar seu segredo, deverá adicionar as permissões necessárias a esse perfil e adicioná-lo à política de chaves AWS KMS.

2. Decida se você está usando o SSM Parameter Store ou o S3 para armazenar o CredSpec

   O Amazon ECS é compatível com as seguintes maneiras de referenciar o caminho do arquivo no campo `credentialSpecs` de definição da tarefa.

   Se você associar as instâncias em um domínio único, use o prefixo `credentialspec:` no início do ARN na string. Se você usa gMSA sem domínio, use `credentialspecdomainless:`.

   Para obter mais informações sobre o CredSpec, consulte Arquivo de especificação de credenciais.

   - Bucket do Amazon S3

     Adicione a especificação da credencial a um bucket do Amazon S3. Em seguida, referencie o nome do recurso da Amazon (ARN) do bucket do Amazon S3 no campo `credentialSpecs` da definição de tarefa.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::${BucketName}/
${ObjectName}"
            ],
            ...
        }
    ],
    ...
}
```

     Para fornecer às suas tarefas acesso ao bucket do S3, adicione as permissões a seguir como uma política em linha ao perfil do IAM de execução da tarefa do Amazon ECS.

     JSON

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "VisualEditor",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/{object}"
            ]
        }
    ]
}
```

- Parâmetro do Parameter Store do SSM

  Adicione a especificação da credencial a um parâmetro do SSM Parameter Store. Em seguida, referencie o nome do recurso da Amazon (ARN) do parâmetro do SSM Parameter Store no campo `credentialSpecs` da definição da tarefa.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:ssm:aws-
region:111122223333:parameter/parameter_name"
            ],
            ...
        }
    ],
    ...
}
```

  Para fornecer às suas tarefas acesso ao parâmetro do SSM Parameter Store, adicione as permissões a seguir como uma política em linha ao perfil do IAM de execução da tarefa do Amazon ECS.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameters"
            ],
            "Resource": [
            "arn:aws:ssm:arn:aws::111122223333:parameter/parameter_name"
            ]
        }
    ]
}
```

Arquivo de especificação de credenciais

O Amazon ECS usa um arquivo de especificação de credenciais do Active Directory (CredSpec). Esse arquivo contém os metadados do gMSA usados para propagar o contexto da conta gMSA para o contêiner de Linux. Você gera o CredSpec e o referencia no campo `credentialSpecs` em sua definição de tarefa. O arquivo CredSpec não contém nenhum segredo.

Veja a seguir um exemplo de arquivo CredSpec.

```
{
    "CmsPlugins": [
        "ActiveDirectory"
    ],
    "DomainJoinConfig": {
        "Sid": "S-1-5-21-2554468230-2647958158-2204241789",
        "MachineAccountName": "WebApp01",
        "Guid": "8665abd4-e947-4dd0-9a51-f8254943c90b",
        "DnsTreeName": "example.com",
        "DnsName": "example.com",
        "NetBiosName": "example"
    },
    "ActiveDirectoryConfig": {
        "GroupManagedServiceAccounts": [
```

```
            {
                "Name": "WebApp01",
                "Scope": "example.com"
            }
        ],
        "HostAccountConfig": {
            "PortableCcgVersion": "1",
            "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
            "PluginInput": {
                "CredentialArn": "arn:aws:secretsmanager:aws-
region:111122223333:secret:MySecret"
            }
        }
    }
}
```

Criação de um CredSpec

Você cria um CredSpec usando o módulo CredSpec PowerShell em um computador Windows que esteja associado ao domínio. Siga as etapas em Criar uma especificação de credencial no site do Microsoft Learn.

Uso de gMSA em contêineres do Linux no Fargate

O Amazon ECS oferece suporte à autenticação do Active Directory para contêineres do Linux no Fargate por meio de um tipo especial de conta de serviço denominada Conta de serviço gerenciada por grupo (gMSA).

Aplicações de rede baseadas no Linux, como as aplicações .NET Core, podem usar o Active Directory para facilitar o gerenciamento da autenticação e da autorização entre usuários e serviços. É possível usar esse recurso criando aplicações que se integram ao Active Directory e são executadas em servidores associados ao domínio. Mas, como os contêineres do Linux não podem ser associados a um domínio, você precisa configurar um contêiner do Linux para ser executado com o gMSA.

Considerações

Considere o seguinte antes de usar o gMSA em contêineres do Linux no Fargate:

- Você deve estar executando a versão da plataforma 1.4 ou posterior.

- Talvez você precise de um computador Windows associado ao domínio para concluir os pré-requisitos. Por exemplo, é possível que seja preciso um computador Windows associado ao domínio para criar o gMSA no Active Directory com o PowerShell. As ferramentas PowerShell do Active Director RSAT estão disponíveis somente para Windows. Para obter mais informações, consulte Instalação das ferramentas de administração do Active Directory.
- Você deve usar o gMSA sem domínio.

  O Amazon ECS usa um arquivo de especificação de credenciais do Active Directory (CredSpec). Esse arquivo contém os metadados de gMSA usados para propagar o contexto da conta de gMSA para o contêiner. Você gera o arquivo CredSpec e o armazena em um bucket do Amazon S3.
- Uma tarefa só é compatível com um Active Directory.

Pré-requisitos

Antes de usar o recurso gMSA para contêineres do Linux com o Amazon ECS, verifique se você concluiu o seguinte:

- Você configurou um domínio do Active Directory com os recursos que deseja que seus contêineres acessem. O Amazon ECS oferece suporte às configurações a seguir:
  - Um Active Directory AWS Directory Service. O AWS Directory Service é um Active Directory gerenciado pela AWS e hospedado no Amazon EC2. Para obter mais informações, consulte Conceitos básicos do Microsoft AD gerenciado pela AWS no Guia de administração do AWS Directory Service.
  - Um Active Directory on-premises. Você deve garantir que a instância de contêiner Linux do Amazon ECS possa se associar ao domínio. Para obter mais informações, consulte AWS Direct Connect.
- Você tem uma conta existente do gMSA no Active Directory e um usuário com permissão para acessar a conta de serviço do gMSA. Para obter mais informações, consulte Crie um usuário do Active Directory para gMSA sem domínio.
- Você tem um bucket do Amazon S3. Para obter mais informações, consulte Como criar um bucket no Guia do usuário do Amazon S3.

Configuração de contêineres do Linux compatíveis com o gMSA no Amazon ECS

Preparar a infraestrutura

As etapas a seguir são considerações e configurações que são executadas uma vez.

- Crie um usuário do Active Directory para gMSA sem domínio

  Ao usar gMSA sem domínio, o contêiner não é associado ao domínio. Outras aplicações executadas no contêiner não podem usar as credenciais para acessar o domínio. Tarefas que usam um domínio diferente podem ser executadas no mesmo contêiner. Forneça o nome de um segredo no AWS Secrets Manager no arquivo CredSpec. O segredo deve conter um nome de usuário, uma senha e o domínio no qual fazer login.

  Esse recurso é semelhante ao recurso gMSA support for non-domain-joined container hosts. Para obter mais informações sobre o recurso do Windows, consulte Arquitetura e melhorias do gMSA no site do Microsoft Learn.

  a. Configure um usuário em seu domínio do Active Directory. O usuário no Active Directory deve ter permissão para acessar a conta de serviço do gMSA que você usa nas tarefas.

  b. Você tem uma VPC e sub-redes que podem resolver o nome de domínio do Active Directory. Configure a VPC com opções de DHCP com o nome de domínio que direciona ao nome de serviço do Active Directory. Para obter informações sobre como configurar opções de DHCP em uma VPC, consulte Trabalhar com conjuntos de opções DHCP no Guia do usuário da Amazon Virtual Private Cloud.

  c. Crie um segredo no AWS Secrets Manager.

  d. Crie o arquivo de especificação da credencial.

Configuração de permissões e segredos

Execute as etapas a seguir uma vez para cada aplicação e cada definição de tarefa. Use as práticas recomendadas de concessão de privilégio mínimo e restrinja as permissões usadas na política. Dessa forma, cada tarefa só pode ler os segredos de que precisa.

1. Crie um usuário no seu domínio do Active Directory. O usuário no Active Directory deve ter permissão para acessar as contas de serviço do gMSA que você usa nas tarefas.

2. Crie um segredo no AWS Secrets Manager, depois de criar o usuário no Active Directory. Para obter mais informações, consulte Criar um segredo no AWS Secrets Manager.
3. Insira o nome de usuário, a senha e o domínio do usuário nos pares de valores-chave JSON denominados `username`, `password` e `domainName`, respectivamente.

```
{"username":"username","password":"passw0rd", "domainName":"example.com"}
```

4. É preciso adicionar as permissões a seguir como uma política em linha ao perfil do IAM de execução da tarefa. Isso dá ao daemon `credentials-fetcher` acesso ao segredo do Secrets Manager. Substitua o exemplo `MySecret` pelo nome do recurso da Amazon (ARN) do seu segredo na lista `Resource`.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
            "arn:aws:secretsmanager:arn:aws::111122223333:secret:MySecret"
            ]
        }
    ]
}
```

> **Note**
>
> Se você usar sua própria chave KMS para criptografar seu segredo, deverá adicionar as permissões necessárias a esse perfil e adicioná-lo à política de chaves AWS KMS.

5. Adicione a especificação da credencial a um bucket do Amazon S3. Em seguida, referencie o nome do recurso da Amazon (ARN) do bucket do Amazon S3 no campo `credentialSpecs` da definição de tarefa.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::${BucketName}/${ObjectName}"
            ],
            ...
        }
    ],
    ...
}
```

Para fornecer às suas tarefas acesso ao bucket do S3, adicione as permissões a seguir como uma política em linha ao perfil do IAM de execução da tarefa do Amazon ECS.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListObjects"
            ],
            "Resource": [
                "arn:aws:s3:::{bucket_name}",
                "arn:aws:s3:::{bucket_name}/{object}"
            ]
        }
    ]
}
```

Arquivo de especificação de credenciais

O Amazon ECS usa um arquivo de especificação de credenciais do Active Directory (CredSpec). Esse arquivo contém os metadados do gMSA usados para propagar o contexto da conta gMSA para o contêiner de Linux. Você gera o CredSpec e o referencia no campo `credentialSpecs` em sua definição de tarefa. O arquivo CredSpec não contém nenhum segredo.

Veja a seguir um exemplo de arquivo CredSpec.

```
{
    "CmsPlugins": [
        "ActiveDirectory"
    ],
    "DomainJoinConfig": {
        "Sid": "S-1-5-21-2554468230-2647958158-2204241789",
        "MachineAccountName": "WebApp01",
        "Guid": "8665abd4-e947-4dd0-9a51-f8254943c90b",
        "DnsTreeName": "example.com",
        "DnsName": "example.com",
        "NetBiosName": "example"
    },
    "ActiveDirectoryConfig": {
        "GroupManagedServiceAccounts": [
            {
                "Name": "WebApp01",
                "Scope": "example.com"
            }
        ],
        "HostAccountConfig": {
            "PortableCcgVersion": "1",
            "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
            "PluginInput": {
                "CredentialArn": "arn:aws:secretsmanager:aws-
region:111122223333:secret:MySecret"
            }
        }
    }
}
```

Como criar um CredSpec e fazer upload em um Amazon S3

Você cria um CredSpec usando o módulo CredSpec PowerShell em um computador Windows que esteja associado ao domínio. Siga as etapas em Criar uma especificação de credencial no site do Microsoft Learn.

Depois de criar o arquivo de especificação da credencial, faça upload dele em um bucket do Amazon S3. Copie o arquivo CredSpec para o computador ou ambiente em que você está executando os comandos da AWS CLI.

Execute o comando AWS CLI a seguir para fazer upload do CredSpec no Amazon S3. Substitua *amzn-s3-demo-bucket* pelo nome do bucket do Amazon S3. É possível armazenar o arquivo como um objeto em qualquer bucket e local, mas deve permitir o acesso a esse bucket e ao local na política que você anexa ao perfil de execução da tarefa.

No PowerShell, use o seguinte comando:

```
$ Write-S3Object -BucketName "amzn-s3-demo-bucket" -Key "ecs-domainless-gmsa-credspec"
 -File "gmsa-cred-spec.json"
```

O comando da AWS CLI a seguir usa caracteres de continuação de barra invertida que são usados pelo `sh` e por shells compatíveis.

```
$ aws s3 cp gmsa-cred-spec.json \
s3://amzn-s3-demo-bucket/ecs-domainless-gmsa-credspec
```

Usar contêineres do Windows no Amazon ECS com gMSA sem domínio usando a AWS CLI

O tutorial a seguir mostra como criar uma tarefa do Amazon ECS que execute um contêiner do Windows com credenciais para acessar o Active Directory com a AWS CLI. Ao usar gMSA sem domínio, a instância de contêiner não será associada ao domínio, outras aplicações na instância não podem usar as credenciais para acessar o domínio e as tarefas que associam domínios diferentes podem ser executadas na mesma instância.

Tópicos

- Pré-requisitos
- Etapa 1: criar e configurar a conta gMSA nos Serviços de Domínio Active Directory (AD DS)
- Etapa 2: fazer upload de credenciais no Secrets Manager

- Etapa 3: modificar seu JSON do CredSpec para incluir informações do gMSA sem domínio
- Etapa 4: fazer upload do CredSpec no Amazon S3
- Etapa 5: (opcional) criar um cluster do Amazon ECS
- Etapa 6: criar um perfil do IAM para instâncias de contêiner
- Etapa 7: criar um perfil de execução de tarefa personalizado
- Etapa 8: criar um perfil de tarefa para o Amazon ECS Exec
- Etapa 9: registrar uma definição de tarefa que usa gMSA sem domínio
- Etapa 10: registrar uma instância de contêiner do Windows no cluster
- Etapa 11: verificar a instância de contêiner
- Etapa 12: executar uma tarefa do Windows
- Etapa 13: verificar se o contêiner tem credenciais gMSA
- Etapa 14: limpar
- Depuração do gMSA sem domínio do Amazon ECS para contêineres Windows

Pré-requisitos

Este tutorial pressupõe que os seguintes pré-requisitos foram concluídos:

- As etapas em Configuração para usar o Amazon ECS foram concluídas.
- O usuário do IAM tem as permissões necessárias especificadas no exemplo de política do IAM de AmazonECS_FullAccess.
- A versão mais recente da AWS CLI está instalada e configurada. Para obter mais informações sobre como instalar ou fazer upgrade da AWS CLI, consulte Instalar a AWS Command Line Interface.

  > **Note**
  >
  > É possível usar endpoints de serviço de pilha dupla para interagir com o Amazon ECS via AWS CLI, SDKs e API do Amazon ECS sobre IPv4 e IPv6. Para obter mais informações, consulte Usar endpoints de pilha dupla do Amazon ECS.

- Você configurou um domínio do Active Directory com os recursos que deseja que seus contêineres acessem. O Amazon ECS oferece suporte às configurações a seguir:

  - Um Active Directory AWS Directory Service. O AWS Directory Service é um Active Directory gerenciado pela AWS e hospedado no Amazon EC2. Para obter mais informações, consulte Conceitos básicos do Microsoft AD gerenciado pela AWS no Guia de administração do AWS Directory Service.
  - Um Active Directory on-premises. Você deve garantir que a instância de contêiner Linux do Amazon ECS possa se associar ao domínio. Para obter mais informações, consulte AWS Direct Connect.
- Você tem uma VPC e sub-redes que podem resolver o nome de domínio do Active Directory.
- Você escolhe entre gMSA sem domínio e associar cada instância a um único domínio. Ao usar gMSA sem domínio, a instância de contêiner não será associada ao domínio, outras aplicações na instância não podem usar as credenciais para acessar o domínio e as tarefas que associam domínios diferentes podem ser executadas na mesma instância.

  Em seguida, escolha o armazenamento de dados para o CredSpec e, opcionalmente, para as credenciais de usuário do Active Directory para o gMSA sem domínio.

  O Amazon ECS usa um arquivo de especificação de credenciais do Active Directory (CredSpec). Esse arquivo contém os metadados de gMSA usados para propagar o contexto da conta de gMSA para o contêiner. Você gera o arquivo CredSpec e o armazena em uma das opções de armazenamento do CredSpec na tabela a seguir, específica para o sistema operacional das instâncias de contêiner. Para usar o método sem domínio, uma seção opcional no arquivo CredSpec pode especificar credenciais em uma das opções de armazenamento de domainless user credentials na tabela a seguir, específica para o sistema operacional das instâncias de contêiner.

Local de armazenamento	Linux	Windows
Amazon Simple Storage Service	CredSpec	CredSpec
AWS Secrets Manager	credenciais do usuário sem domínio	credenciais do usuário sem domínio
Armazenamento de parâmetros do Amazon EC2 Systems Manager	CredSpec	CredSpec, credenciais do usuário sem domínio

Local de armazenamento	Linux	Windows
Arquivo local	N/D	CredSpec

- (Opcional) O AWS CloudShell é uma ferramenta que oferece aos clientes uma linha de comando sem precisar criar a própria instância do EC2. Para obter mais informações, consulte O que é o AWS CloudShell? no Guia do usuário do AWS CloudShell.

Etapa 1: criar e configurar a conta gMSA nos Serviços de Domínio Active Directory (AD DS)

Crie e configure uma conta gMSA no domínio do Active Directory.

> **Note**
>
> Esta etapa cria duas contas separadas: uma conta de serviço gerenciada pelo grupo (gMSA) que fornece a identidade para seus contêineres e uma conta de usuário comum que é usada para autenticação de domínio. Essas contas têm propósitos diferentes e devem ter nomes diferentes.

1. Gerar uma chave raiz do Serviço de Distribuição de Chaves

   > **Note**
   >
   > Se estiver usando o AWS Directory Service, será possível ignorar esta etapa.

   A chave raiz do KDS e as permissões do gMSA são configuradas com seu Microsoft AD gerenciado pela AWS.

   Se você ainda não criou uma conta de serviço do gMSA em seu domínio, primeiro precisará gerar uma chave raiz do Serviço de Distribuição de Chaves (KDS). O KDS é responsável por criar, alternar e liberar a senha do gMSA para os hosts autorizados. Quando o `ccg.exe` precisa recuperar as credenciais do gMSA, ele entra em contato com o KDS para recuperar a senha atual.

Para verificar se a chave raiz do KDS já foi criada, execute o cmdlet do PowerShell a seguir com privilégios de administrador de domínio em um controlador de domínio usando o módulo do `ActiveDirectory` do PowerShell. Para obter mais informações sobre o módulo, consulte Módulo do ActiveDirectory no site do Microsoft Learn.

```
PS C:\> Get-KdsRootKey
```

Se o comando retornar uma ID de chave, será possível ignorar o restante desta etapa. Caso contrário, crie a chave raiz do KDS executando o comando a seguir:

```
PS C:\> Add-KdsRootKey -EffectiveImmediately
```

Embora o argumento `EffectiveImmediately` para o comando implique que a chave entra em vigor imediatamente, é preciso esperar 10 horas antes que a chave raiz do KDS seja replicada e esteja disponível para uso em todos os controladores de domínio.

2. Criar a conta gMSA

   Para criar a conta gMSA e permitir que o `ccg.exe` recupere a senha do gMSA, execute os comandos a seguir do PowerShell em um Windows Server ou cliente com acesso ao domínio. Substitua `ExampleAccount` pelo nome que você deseja para sua conta gMSA e `example-domain` pelo nome de domínio do Active Directory (por exemplo, se seu domínio for `contoso.com`, use `contoso`).

   a.
   ```
   PS C:\> Install-WindowsFeature RSAT-AD-PowerShell
   ```

   b.
   ```
   PS C:\> New-ADGroup -Name "ExampleAccount Authorized Hosts" -SamAccountName
    "ExampleAccountHosts" -GroupScope DomainLocal
   ```

   c.
   ```
   PS C:\> New-ADServiceAccount -Name "ExampleAccount" -DnsHostName "example-
   domain" -ServicePrincipalNames "host/ExampleAccount", "host/example-domain" -
   PrincipalsAllowedToRetrieveManagedPassword "ExampleAccountHosts"
   ```

   d. Crie um usuário com uma senha permanente que não expire. Essas credenciais são armazenadas no AWS Secrets Manager e usadas por cada tarefa para se associar ao domínio. Esta é uma conta de usuário separada da conta gMSA criada acima. Substitua `ExampleServiceUser` pelo nome que você deseja para essa conta de usuário do serviço.

```
PS C:\> New-ADUser -Name "ExampleServiceUser" -AccountPassword (ConvertTo-
SecureString -AsPlainText "Test123" -Force) -Enabled 1 -PasswordNeverExpires 1
```

e.
```
PS C:\> Add-ADGroupMember -Identity "ExampleAccountHosts" -Members
 "ExampleServiceUser"
```

f. Instale o módulo PowerShell para criar objetos do CredSpec no Active Directory e gere a saída de JSON do CredSpec.

```
PS C:\> Install-PackageProvider -Name NuGet -Force
```

```
PS C:\> Install-Module CredentialSpec
```

g.
```
PS C:\> New-CredentialSpec -AccountName ExampleAccount
```

3. Copie a saída de JSON do comando anterior em um arquivo chamado `gmsa-cred-spec.json`. Esse é o arquivo CredSpec. Ele será usado na Etapa 3, Etapa 3: modificar seu JSON do CredSpec para incluir informações do gMSA sem domínio.

Etapa 2: fazer upload de credenciais no Secrets Manager

Copie as credenciais do Active Directory em um sistema seguro de armazenamento de credenciais, para que cada tarefa as recupere. Esse é o método do gMSA sem domínio. Ao usar gMSA sem domínio, a instância de contêiner não será associada ao domínio, outras aplicações na instância não podem usar as credenciais para acessar o domínio e as tarefas que associam domínios diferentes podem ser executadas na mesma instância.

Esta etapa usa a AWS CLI. É possível executar esses comandos no AWS CloudShell no shell padrão, que é o `bash`.

- Execute o comando AWS CLI a seguir e substitua o nome de usuário, a senha e o nome do domínio de acordo com o seu ambiente. Use o nome da conta do usuário do serviço (não o nome da conta gMSA) como o nome de usuário. Guarde o ARN do segredo para usar na próxima etapa, Etapa 3: modificar seu JSON do CredSpec para incluir informações do gMSA sem domínio

O comando a seguir usa caracteres de continuação de barra invertida que são usados pelo `sh` e por shells compatíveis. Esse comando não é compatível com o PowerShell. Você deve modificar o comando para usá-lo com o PowerShell.

```
$ aws secretsmanager create-secret \
--name gmsa-plugin-input \
--description "Amazon ECS - gMSA Portable Identity." \
--secret-string "{\"username\":\"ExampleServiceUser\",\"password\":\"Test123\",
\"domainName\":\"contoso.com\"}"
```

Etapa 3: modificar seu JSON do CredSpec para incluir informações do gMSA sem domínio

Antes de fazer o upload de CredSpec para uma das opções de armazenamento, adicione informações ao CredSpec com o ARN do segredo no Secrets Manager da etapa anterior. Para obter mais informações, consulte Configuração adicional de especificações de credenciais para o caso de uso de host de contêiner não associado a domínio no site do Microsoft Learn.

1. Adicione as informações a seguir ao arquivo CredSpec dentro do `ActiveDirectoryConfig`. Substitua o ARN pelo segredo no Secrets Manager da etapa anterior.

   Observe que o valor `PluginGUID` deve corresponder ao GUID no trecho de exemplo a seguir, e é obrigatório.

   ```
   "HostAccountConfig": {
         "PortableCcgVersion": "1",
         "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
         "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
   region:111122223333:secret:gmsa-plugin-input\"}"
       }
   ```

   Você também pode usar um segredo no SSM Parameter Store usando o ARN neste formato: `\"arn:aws:ssm:aws-region:111122223333:parameter/gmsa-plugin-input\"`.

2. Depois de modificar o arquivo CredSpec, ele será semelhante à saída do exemplo a seguir:

   ```
   {
     "CmsPlugins": [
   ```

```
    "ActiveDirectory"
  ],
  "DomainJoinConfig": {
    "Sid": "S-1-5-21-4066351383-705263209-1606769140",
    "MachineAccountName": "ExampleAccount",
    "Guid": "ac822f13-583e-49f7-aa7b-284f9a8c97b6",
    "DnsTreeName": "example-domain",
    "DnsName": "example-domain",
    "NetBiosName": "example-domain"
  },
  "ActiveDirectoryConfig": {
    "GroupManagedServiceAccounts": [
      {
        "Name": "ExampleAccount",
        "Scope": "example-domain"
      },
      {
        "Name": "ExampleAccount",
        "Scope": "example-domain"
      }
    ],
    "HostAccountConfig": {
      "PortableCcgVersion": "1",
      "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
      "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
region:111122223333:secret:gmsa-plugin-input\"}"
    }
  }
}
```

Etapa 4: fazer upload do CredSpec no Amazon S3

Esta etapa usa a AWS CLI. É possível executar esses comandos no AWS CloudShell no shell padrão, que é o `bash`.

1. Copie o arquivo CredSpec para o computador ou ambiente em que você está executando os comandos da AWS CLI.
2. Execute o comando AWS CLI a seguir para fazer upload do CredSpec no Amazon S3. Substitua `MyBucket` pelo nome do bucket do Amazon S3. É possível armazenar o arquivo como um

objeto em qualquer bucket e local, mas deve permitir o acesso a esse bucket e ao local na política que você anexa ao perfil de execução da tarefa.

O comando a seguir usa caracteres de continuação de barra invertida que são usados pelo `sh` e por shells compatíveis. Esse comando não é compatível com o PowerShell. Você deve modificar o comando para usá-lo com o PowerShell.

```
$ aws s3 cp gmsa-cred-spec.json \
s3://MyBucket/ecs-domainless-gmsa-credspec
```

Etapa 5: (opcional) criar um cluster do Amazon ECS

Por padrão, sua conta tem um cluster Amazon ECS chamado `default`. Esse cluster é usado por padrão na AWS CLI, nos SDKs e no AWS CloudFormation. É possível usar clusters adicionais para agrupar e organizar tarefas e infraestrutura e atribuir padrões para algumas configurações.

É possível criar um cluster a partir do AWS Management Console, da AWS CLI, de SDKs ou do AWS CloudFormation. As definições e a configuração no cluster não afetam o gMSA.

Esta etapa usa a AWS CLI. É possível executar esses comandos no AWS CloudShell no shell padrão, que é o `bash`.

```
$ aws ecs create-cluster --cluster-name windows-domainless-gmsa-cluster
```

> **Important**
>
> Caso você escolha criar seu o próprio cluster, você deverá especificar `--cluster clusterName` para cada comando que pretender usar com esse cluster.

Etapa 6: criar um perfil do IAM para instâncias de contêiner

Uma instância de contêiner é um computador host para executar contêineres em tarefas do ECS, por exemplo, instâncias do Amazon EC2. Cada instância de contêiner se registra em um cluster do Amazon ECS. Para poder iniciar instâncias do Amazon EC2 e registrá-las em um cluster, você deve criar um perfil do IAM para suas instâncias de contêiner usarem.

Para criar o perfil da instância de contêiner, consulte Função do IAM de instância de contêiner do Amazon ECS. O `ecsInstanceRole` padrão tem permissões suficientes para concluir este tutorial.

Etapa 7: criar um perfil de execução de tarefa personalizado

O Amazon ECS pode usar um perfil do IAM diferente para as permissões necessárias para iniciar cada tarefa, em vez do perfil da instância de contêiner. Esse perfil é chamado de perfil de execução de tarefa. Recomendamos criar um perfil de execução de tarefa apenas com as permissões necessárias para que o ECS execute a tarefa, também conhecidas como permissões de privilégio mínimo. Para obter mais informações sobre o princípio do privilégio mínimo, consulte SEC03-BP02 Conceder acesso com privilégio mínimo no AWS Well-Architected Framework.

1. Para criar um perfil de execução de tarefa, consulte Criar a função de execução de tarefa do. As permissões padrão permitem que a instância de contêiner extraia imagens de contêiner do Amazon Elastic Container Registry e `stdout` e `stderr` de suas aplicações para serem registradas em log no Amazon CloudWatch Logs.

   Como o perfil precisa de permissões personalizadas para este tutorial, é possível atribuir ao perfil um nome diferente de `ecsTaskExecutionRole`. Este tutorial usa `ecsTaskExecutionRole` em etapas adicionais.
2. Adicione as permissões a seguir criando uma política personalizada, uma política em linha que só exista para esse perfil ou uma política que você possa reutilizar. Substitua o ARN para o `Resource` na primeira declaração pelo bucket do Amazon S3 e a localização, e o segundo `Resource` pelo ARN do segredo no Secrets Manager.

   Se você criptografar o segredo no Secrets Manager com uma chave personalizada, também deverá permitir `kms:Decrypt` para a chave.

   Se você usar o SSM Parameter Store em vez do Secrets Manager, deverá permitir `ssm:GetParameter` para o parâmetro, em vez de `secretsmanager:GetSecretValue`.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::MyBucket/ecs-domainless-gmsa-credspec/
gmsa-cred-spec.json"
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource":
 "arn:aws:secretsmanager:arn:aws:: 111122223333:secret:gmsa-plugin-AbCdEf"
        }
    ]
}
```

Etapa 8: criar um perfil de tarefa para o Amazon ECS Exec

Este tutorial usa o Amazon ECS Exec para verificar a funcionalidade executando um comando dentro de uma tarefa em execução. Para usar o ECS Exec, o serviço ou a tarefa deve ativar o ECS Exec, e o perfil da tarefa (mas não o perfil de execução da tarefa) deve ter permissões `ssmmessages`. Para saber mais sobre a política do IAM necessária, consulte Permissões do ECS Exec.

Esta etapa usa a AWS CLI. É possível executar esses comandos no AWS CloudShell no shell padrão, que é o `bash`.

Para criar um perfil de tarefa usando a AWS CLI, siga estas etapas.

1. Crie um arquivo chamado `ecs-tasks-trust-policy.json` com o conteúdo a seguir:

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
```

```
        }
    ]
}
```

2. Criar um perfil do IAM. É possível substituir o nome `ecs-exec-demo-task-role`, mas mantenha-o nas etapas a seguir.

   O comando a seguir usa caracteres de continuação de barra invertida que são usados pelo `sh` e por shells compatíveis. Esse comando não é compatível com o PowerShell. Você deve modificar o comando para usá-lo com o PowerShell.

   ```
   $ aws iam create-role --role-name ecs-exec-demo-task-role \
   --assume-role-policy-document file://ecs-tasks-trust-policy.json
   ```

   É possível excluir o arquivo `ecs-tasks-trust-policy.json`.

3. Crie um arquivo chamado `ecs-exec-demo-task-role-policy.json` com o conteúdo a seguir:

   JSON

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": [
                   "ssmmessages:CreateControlChannel",
                   "ssmmessages:CreateDataChannel",
                   "ssmmessages:OpenControlChannel",
                   "ssmmessages:OpenDataChannel"
               ],
               "Resource": "*"
           }
       ]
   }
   ```

4. Crie uma política do IAM e anexe-a ao perfil da etapa anterior.

O comando a seguir usa caracteres de continuação de barra invertida que são usados pelo sh e por shells compatíveis. Esse comando não é compatível com o PowerShell. Você deve modificar o comando para usá-lo com o PowerShell.

```
$ aws iam put-role-policy \
    --role-name ecs-exec-demo-task-role \
    --policy-name ecs-exec-demo-task-role-policy \
    --policy-document file://ecs-exec-demo-task-role-policy.json
```

É possível excluir o arquivo `ecs-exec-demo-task-role-policy.json`.

Etapa 9: registrar uma definição de tarefa que usa gMSA sem domínio

Esta etapa usa a AWS CLI. É possível executar esses comandos no AWS CloudShell no shell padrão, que é o `bash`.

1. Crie um arquivo chamado `windows-gmsa-domainless-task-def.json` com o conteúdo a seguir:

```
{
  "family": "windows-gmsa-domainless-task",
  "containerDefinitions": [
    {
      "name": "windows_sample_app",
      "image": "mcr.microsoft.com/windows/servercore/iis",
      "cpu": 1024,
      "memory": 1024,
      "essential": true,
      "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::ecs-domainless-gmsa-
credspec/gmsa-cred-spec.json"
      ],
      "entryPoint": [
        "powershell",
        "-Command"
      ],
      "command": [
        "New-Item -Path C:\\inetpub\\wwwroot\\index.html -ItemType file -Value
 '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
```

```
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>' -Force ; C:\
\ServiceMonitor.exe w3svc"
      ],
      "portMappings": [
        {
          "protocol": "tcp",
          "containerPort": 80,
          "hostPort": 8080
        }
      ]
    }
  ],
  "taskRoleArn": "arn:aws:iam::111122223333:role/ecs-exec-demo-task-role",
  "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole"
}
```

2. Registre a definição de tarefa executando o comando a seguir.

   O comando a seguir usa caracteres de continuação de barra invertida que são usados pelo sh e por shells compatíveis. Esse comando não é compatível com o PowerShell. Você deve modificar o comando para usá-lo com o PowerShell.

```
$ aws ecs register-task-definition \
--cli-input-json file://windows-gmsa-domainless-task-def.json
```

Etapa 10: registrar uma instância de contêiner do Windows no cluster

Inicie uma instância de Windows do Amazon EC2 e execute o agente de contêiner do ECS para registrá-la como uma instância de contêiner no cluster. O ECS executa tarefas nas instâncias de contêiner registradas no cluster em que as tarefas são iniciadas.

1. Para iniciar uma instância de Windows do Amazon EC2 que esteja configurada para o Amazon ECS no AWS Management Console, consulte Iniciar uma instância de contêiner do Windows do Amazon ECS. Pare na etapa de dados do usuário.
2. Em gMSA, os dados do usuário devem definir a variável de ambiente `ECS_GMSA_SUPPORTED` antes de iniciar o agente de contêiner do ECS.

   Para o ECS Exec, o agente deve começar com o argumento `-EnableTaskIAMRole`.

Para proteger o perfil do IAM da instância impedindo que as tarefas cheguem ao serviço da Web IMDS do EC2 para recuperar as credenciais do perfil, adicione o argumento `-AwsvpcBlockIMDS`. Isso se aplica somente às tarefas que usem o modo de rede `awsvpc`.

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_GMSA_SUPPORTED", $TRUE, "Machine")
Import-Module ECSTools
Initialize-ECSAgent -Cluster windows-domainless-gmsa-cluster -EnableTaskIAMRole -
AwsvpcBlockIMDS
</powershell>
```

3. Revise um resumo da configuração da instância no painel Summary (Resumo) e, quando você estiver pronto, escolha Launch instance (Iniciar instância).

Etapa 11: verificar a instância de contêiner

É possível verificar se há uma instância de contêiner no cluster usando o AWS Management Console. No entanto, o gMSA precisa de recursos adicionais indicados como atributos. Esses atributos não são visíveis no AWS Management Console, portanto, este tutorial usa o AWS CLI.

Esta etapa usa a AWS CLI. É possível executar esses comandos no AWS CloudShell no shell padrão, que é o `bash`.

1. Listar as instâncias de contêiner no cluster. As instâncias de contêiner têm um ID diferente do ID da instância do EC2.

```
$ aws ecs list-container-instances
```

Saída:

```
{
    "containerInstanceArns": [
        "arn:aws:ecs:aws-region:111122223333:container-
instance/default/MyContainerInstanceID"
    ]
}
```

Por exemplo, `526bd5d0ced448a788768334e79010fd` é um ID de instância de contêiner válido.

2. Use o ID da instância de contêiner da etapa anterior para obter os detalhes da instância de contêiner. Substitua `MyContainerInstanceID` pelo ID.

   O comando a seguir usa caracteres de continuação de barra invertida que são usados pelo `sh` e por shells compatíveis. Esse comando não é compatível com o PowerShell. Você deve modificar o comando para usá-lo com o PowerShell.

   ```
   $ aws ecs describe-container-instances \
        ----container-instances MyContainerInstanceID
   ```

   Observe que a saída é muito longa.

3. Verifique se a lista `attributes` tem um objeto com a chave chamada `name` e um valor de `ecs.capability.gmsa-domainless`. Veja a seguir um exemplo do objeto.

   Saída:

   ```
   {
       "name": "ecs.capability.gmsa-domainless"
   }
   ```

Etapa 12: executar uma tarefa do Windows

Execute uma tarefa do Amazon ECS. Se houver apenas uma instância de contêiner no cluster, será possível usar `run-task`. Se houver muitas instâncias de contêiner diferentes, talvez seja mais fácil usar `start-task` e especificar o ID da instância de contêiner para executar a tarefa do que adicionar restrições de posicionamento à definição da tarefa para controlar em que tipo de instância de contêiner executar essa tarefa.

Esta etapa usa a AWS CLI. É possível executar esses comandos no AWS CloudShell no shell padrão, que é o `bash`.

1. O comando a seguir usa caracteres de continuação de barra invertida que são usados pelo `sh` e por shells compatíveis. Esse comando não é compatível com o PowerShell. Você deve modificar o comando para usá-lo com o PowerShell.

```
aws ecs run-task --task-definition windows-gmsa-domainless-task \
    --enable-execute-command --cluster windows-domainless-gmsa-cluster
```

Anote o ID da tarefa retornado pelo comando.

2. Execute o comando a seguir para verificar se a tarefa foi iniciada. Esse comando espera e não retorna o prompt do shell até que a tarefa seja iniciada. Substitua `MyTaskID` pelo ID da tarefa da etapa anterior.

```
$ aws ecs wait tasks-running --task MyTaskID
```

Etapa 13: verificar se o contêiner tem credenciais gMSA

Verifique se o contêiner na tarefa tem um token Kerberos. gMSA

Esta etapa usa a AWS CLI. É possível executar esses comandos no AWS CloudShell no shell padrão, que é o `bash`.

1. O comando a seguir usa caracteres de continuação de barra invertida que são usados pelo `sh` e por shells compatíveis. Esse comando não é compatível com o PowerShell. Você deve modificar o comando para usá-lo com o PowerShell.

```
$ aws ecs execute-command \
--task MyTaskID \
--container windows_sample_app \
--interactive \
--command powershell.exe
```

A saída será um prompt do PowerShell.

2. Execute o comando a seguir no terminal do PowerShell dentro do contêiner.

```
PS C:\> klist get ExampleAccount$
```

Na saída, observe que `Principal` é o que você criou anteriormente.

Etapa 14: limpar

Ao concluir este tutorial, você deve limpar os recursos associados para evitar cobranças por recursos não utilizados.

Esta etapa usa a AWS CLI. É possível executar esses comandos no AWS CloudShell no shell padrão, que é o `bash`.

1. Interrompa a tarefa. Substitua `MyTaskID` pelo ID da tarefa da etapa 12, Etapa 12: executar uma tarefa do Windows.

   ```
   $ aws ecs stop-task --task MyTaskID
   ```

2. Encerre a instância do Amazon EC2. Depois disso, a instância de contêiner no cluster será excluída automaticamente após uma hora.

   É possível encontrar e encerrar a instância usando o console do Amazon EC2. É possível executar o comando a seguir: Para executar o comando, encontre o ID da instância do EC2 na saída do comando `aws ecs describe-container-instances` da etapa 1, Etapa 11: verificar a instância de contêiner. i-10a64379 é um exemplo de ID de instância do EC2.

   ```
   $ aws ec2 terminate-instances --instance-ids MyInstanceID
   ```

3. Exclua o arquivo CredSpec no Amazon S3. Substitua `MyBucket` pelo nome do bucket do Amazon S3.

   ```
   $ aws s3api delete-object --bucket MyBucket --key ecs-domainless-gmsa-credspec/
   gmsa-cred-spec.json
   ```

4. Exclua o segredo no Secrets Manager. Se você usou o SSM Parameter Store em vez disso, exclua o parâmetro.

   O comando a seguir usa caracteres de continuação de barra invertida que são usados pelo `sh` e por shells compatíveis. Esse comando não é compatível com o PowerShell. Você deve modificar o comando para usá-lo com o PowerShell.

   ```
   $ aws secretsmanager delete-secret --secret-id gmsa-plugin-input \
       --force-delete-without-recovery
   ```

5. Cancele o registro e exclua a definição de tarefa. Ao cancelar o registro da definição da tarefa, você a marca como inativa para que não possa ser usada para iniciar novas tarefas. Em seguida, é possível excluir a definição da tarefa.

   a. Cancele o registro da definição da tarefa especificando a versão. O ECS cria automaticamente versões das definições de tarefas, que são numeradas a partir de 1. Você faz referência às versões no mesmo formato dos rótulos nas imagens do contêiner, como `:1`.

   ```
   $ aws ecs deregister-task-definition --task-definition windows-gmsa-domainless-task:1
   ```

   b. Exclua a definição da tarefa.

   ```
   $ aws ecs delete-task-definitions --task-definition windows-gmsa-domainless-task:1
   ```

6. (Opcional) Exclua o cluster do ECS se você tiver criado um cluster.

   ```
   $ aws ecs delete-cluster --cluster windows-domainless-gmsa-cluster
   ```

Depuração do gMSA sem domínio do Amazon ECS para contêineres Windows

Status da tarefa do Amazon ECS

O ECS tenta iniciar uma tarefa exatamente uma vez. Qualquer tarefa que tenha um problema é interrompida e definida para o status `STOPPED`. Há dois tipos comuns de problemas com tarefas. Primeiro, tarefas que não puderam ser iniciadas. Segundo, tarefas em que a aplicação foi interrompida dentro de um dos contêineres. No AWS Management Console, observe o campo Motivo da parada para obter o motivo pelo qual a tarefa foi interrompida. Na AWS CLI, descreva a tarefa e veja o `stoppedReason`. Para ver as etapas em AWS Management Console e AWS CLI, consulte Visualizar erros de tarefa interrompida do Amazon ECS.

Eventos do Windows

Os eventos do Windows para gMSA em contêineres são registrados em log no arquivo de `Microsoft-Windows-Containers-CCG` log e podem ser encontrados no Visualizador de

Eventos na seção Aplicações e Serviços em `Logs\Microsoft\Windows\Containers-CCG\Admin`. Para obter mais dicas de depuração, consulte Solucionar problemas de GMSAs para contêineres Windows no site do Microsoft Learn.

Plug-in gMSA do agente do ECS

O registro em log do plug-in gMSA para o agente do ECS na instância do contêiner do Windows está no diretório a seguir, `C:/ProgramData/Amazon/gmsa-plugin/`. Examine esse registro para ver se as credenciais do usuário sem domínio foram baixadas do local de armazenamento, como o Secrets Manager, e se o formato da credencial foi lido corretamente.

Saiba como usar gMSAs para contêineres do Windows do Amazon EC2 para o Amazon ECS

O Amazon ECS é compatível com a autenticação do Active Directory para contêineres do Windows por meio de um tipo especial de conta de serviço denominado Conta de serviço gerenciada pelo grupo (gMSA).

Aplicativos de rede baseados no Windows, como os aplicativos.NET, geralmente usam o Active Directory para facilitar o gerenciamento de autenticação e autorização entre usuários e serviços. Os desenvolvedores geralmente projetam seus aplicativos para se integrarem ao Active Directory e serem executados em servidores associados ao domínio para essa finalidade. Como os contêineres do Windows não podem ser associados ao domínio, os contêineres do Windows devem ser configurados para ser executado com o gMSA.

Um contêiner do Windows em execução com a gMSA depende da instância do Amazon EC2 do host para recuperar as credenciais da gMSA do controlador de domínio do Active Directory e fornecê-las à instância de contêiner. Para obter mais informações, consulte Criar gMSAs para contêineres do Windows.

> **Note**
>
> Não há suporte para esse recurso nos contêineres do Windows no Fargate.

Tópicos

- Considerações

- Pré-requisitos
- Configuração do gMSA para contêineres do Windows no Amazon ECS

Considerações

Ao usar gMSAs para contêineres do Windows, deve-se levar em consideração o seguinte:

- Ao usar a AMI do Windows Server 2016 Full otimizada para Amazon ECS para as instâncias de contêiner, o nome do host do contêiner deve ser o mesmo que o nome da conta gMSA definido no arquivo de especificação de credenciais. Para especificar um nome de host para um contêiner, use o parâmetro de definição de contêiner `hostname`. Para obter mais informações, consulte Configurações de rede.
- Você escolhe entre gMSA sem domínio e associar cada instância a um único domínio. Ao usar gMSA sem domínio, a instância de contêiner não será associada ao domínio, outras aplicações na instância não podem usar as credenciais para acessar o domínio e as tarefas que associam domínios diferentes podem ser executadas na mesma instância.

  Em seguida, escolha o armazenamento de dados para o CredSpec e, opcionalmente, para as credenciais de usuário do Active Directory para o gMSA sem domínio.

  O Amazon ECS usa um arquivo de especificação de credenciais do Active Directory (CredSpec). Esse arquivo contém os metadados de gMSA usados para propagar o contexto da conta de gMSA para o contêiner. Você gera o arquivo CredSpec e o armazena em uma das opções de armazenamento do CredSpec na tabela a seguir, específica para o sistema operacional das instâncias de contêiner. Para usar o método sem domínio, uma seção opcional no arquivo CredSpec pode especificar credenciais em uma das opções de armazenamento de domainless user credentials na tabela a seguir, específica para o sistema operacional das instâncias de contêiner.

Local de armazenamento	Linux	Windows
Amazon Simple Storage Service	CredSpec	CredSpec
AWS Secrets Manager	credenciais do usuário sem domínio	credenciais do usuário sem domínio

Local de armazenamento	Linux	Windows
Armazenamento de parâmetros do Amazon EC2 Systems Manager	CredSpec	CredSpec, credenciais do usuário sem domínio
Arquivo local	N/D	CredSpec

Pré-requisitos

Antes de usar o recurso gMSA para contêineres do Windows com o Amazon ECS, verifique se você concluiu o seguinte:

- Você configurou um domínio do Active Directory com os recursos que deseja que seus contêineres acessem. O Amazon ECS oferece suporte às configurações a seguir:
  - Um Active Directory AWS Directory Service. O AWS Directory Service é um Active Directory gerenciado pela AWS e hospedado no Amazon EC2. Para obter mais informações, consulte Conceitos básicos do Microsoft AD gerenciado pela AWS no Guia de administração do AWS Directory Service.
  - Um Active Directory on-premises. Você deve garantir que a instância de contêiner Linux do Amazon ECS possa se associar ao domínio. Para obter mais informações, consulte AWS Direct Connect.
- Você tem uma conta de gMSA existente no Active Directory. Para obter mais informações, consulte Criar gMSAs para contêineres do Windows.
- Você escolheu usar o gMSA sem domínio ou a instância de contêiner do Windows do Amazon ECS que hospeda a tarefa do Amazon ECS deve ser um domínio associado ao Active Directory e fazer parte do grupo de segurança do Active Directory que tem acesso à conta gMSA.

  Ao usar gMSA sem domínio, a instância de contêiner não será associada ao domínio, outras aplicações na instância não podem usar as credenciais para acessar o domínio e as tarefas que associam domínios diferentes podem ser executadas na mesma instância.
- Você adicionou as permissões necessárias do IAM. As permissões necessárias dependem dos métodos escolhidos para as credenciais iniciais e para armazenar a especificação da credencial:
  - Se você usa o gMSA sem domínio para as credenciais iniciais, as permissões do IAM para o AWS Secrets Manager são necessárias no perfil da instância do Amazon EC2.

- Se você armazenar a especificação da credencial no SSM Parameter Store, as permissões do IAM para o Amazon EC2 Systems Manager Parameter Store serão necessárias no perfil de execução da tarefa.
- Se você armazenar a especificação da credencial no Amazon S3, as permissões do IAM para o Amazon Simple Storage Service serão necessárias no perfil de execução da tarefa.

Configuração do gMSA para contêineres do Windows no Amazon ECS

Para configurar o gMSA para contêineres de Windows no Amazon ECS, siga o tutorial completo que inclui a configuração dos pré-requisitos Usar contêineres do Windows no Amazon ECS com gMSA sem domínio usando a AWS CLI.

As seções a seguir abordam a configuração do CredSpec em detalhes.

Tópicos

- Exemplo CredSpec
- Configuração de gMSA sem domínio
- Referência a um arquivo de especificações de credenciais em uma definição de tarefa

Exemplo CredSpec

O Amazon ECS usa um arquivo de especificação de credenciais que contém os metadados de gMSA usados para propagar o contexto da conta gMSA para o contêiner do Windows. É possível gerar o arquivo de especificação de credenciais e mencioná-lo no campo `credentialSpec` em sua definição de tarefa. O arquivo de especificação de credenciais não contém nenhum segredo.

Veja a seguir um exemplo de arquivo de especificação de credenciais:

```
{
    "CmsPlugins": [
        "ActiveDirectory"
    ],
    "DomainJoinConfig": {
        "Sid": "S-1-5-21-2554468230-2647958158-2204241789",
        "MachineAccountName": "WebApp01",
        "Guid": "8665abd4-e947-4dd0-9a51-f8254943c90b",
        "DnsTreeName": "contoso.com",
        "DnsName": "contoso.com",
```

```
        "NetBiosName": "contoso"
    },
    "ActiveDirectoryConfig": {
        "GroupManagedServiceAccounts": [
            {
                "Name": "WebApp01",
                "Scope": "contoso.com"
            }
        ]
    }
}
```

Configuração de gMSA sem domínio

Recomendamos a utilização de gMSA sem domínio em vez de associar as instâncias do contêiner a um único domínio. Ao usar gMSA sem domínio, a instância de contêiner não será associada ao domínio, outras aplicações na instância não podem usar as credenciais para acessar o domínio e as tarefas que associam domínios diferentes podem ser executadas na mesma instância.

1. Antes de fazer o upload de CredSpec para uma das opções de armazenamento, adicione informações ao CredSpec com o ARN do segredo no Secrets Manager ou no SSM Parameter Store. Para obter mais informações, consulte Configuração adicional de especificações de credenciais para o caso de uso de host de contêiner não associado a domínio no site do Microsoft Learn.

   Formato de credencial de gMSA sem domínio

   A seguir está o formato JSON para as credenciais de gMSA sem domínio do seu Active Directory. Armazene as credenciais no Secrets Manager ou no SSM Parameter Store.

   ```
   {
       "username":"WebApp01",
       "password":"Test123!",
       "domainName":"contoso.com"
   }
   ```

2. Adicione as informações a seguir ao arquivo CredSpec dentro do `ActiveDirectoryConfig`. Substitua o ARN pelo segredo no Secrets Manager ou no SSM Parameter Store.

   Observe que o valor `PluginGUID` deve corresponder ao GUID no trecho de exemplo a seguir, e é obrigatório.

```
    "HostAccountConfig": {
        "PortableCcgVersion": "1",
        "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
        "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
region:111122223333:secret:gmsa-plugin-input\"}"
    }
```

Você também pode usar um segredo no SSM Parameter Store usando o ARN neste formato: \"arn:aws:ssm:*aws-region*:111122223333:parameter/*gmsa-plugin-input*\".

3. Depois de modificar o arquivo CredSpec, ele será semelhante à saída do exemplo a seguir:

```
{
  "CmsPlugins": [
    "ActiveDirectory"
  ],
  "DomainJoinConfig": {
    "Sid": "S-1-5-21-4066351383-705263209-1606769140",
    "MachineAccountName": "WebApp01",
    "Guid": "ac822f13-583e-49f7-aa7b-284f9a8c97b6",
    "DnsTreeName": "contoso",
    "DnsName": "contoso",
    "NetBiosName": "contoso"
  },
  "ActiveDirectoryConfig": {
    "GroupManagedServiceAccounts": [
      {
        "Name": "WebApp01",
        "Scope": "contoso"
      },
      {
        "Name": "WebApp01",
        "Scope": "contoso"
      }
    ],
    "HostAccountConfig": {
      "PortableCcgVersion": "1",
      "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
      "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
region:111122223333:secret:gmsa-plugin-input\"}"
    }
```

```
  }
}
```

Referência a um arquivo de especificações de credenciais em uma definição de tarefa

O Amazon ECS é compatível com as seguintes maneiras de referenciar o caminho do arquivo no campo `credentialSpecs` de definição da tarefa. Para cada uma dessas opções, é possível fornecer `credentialspec:` ou `domainlesscredentialspec:`, dependendo se você estiver associando as instâncias de contêiner a um domínio único ou usando gMSA sem domínio, respectivamente.

Bucket do Amazon S3

Adicione a especificação de credencial a um bucket do Amazon S3 e, em seguida, referencie o nome do recurso da Amazon (ARN) do bucket do Amazon S3 no campo `credentialSpecs` da definição de tarefa.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::${BucketName}/${ObjectName}"
            ],
            ...
        }
    ],
    ...
}
```

Você também deve adicionar as seguintes permissões como uma política alinhada com a função do IAM de execução de tarefas do Amazon ECS para que suas tarefas tenham acesso ao bucket do Amazon S3.

JSON

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::{bucket_name}",
                "arn:aws:s3:::{bucket_name}/{object}"
            ]
        }
    ]
}
```

Parâmetro do Parameter Store do SSM

Adicione a especificação de credencial a um parâmetro do Parameter Store do SSM e, em seguida, referencie o nome do recurso da Amazon (ARN) do parâmetro do Parameter Store do SSM no campo `credentialSpecs` da definição de tarefa.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [

 "credentialspecdomainless:arn:aws:ssm:region:111122223333:parameter/parameter_name"
            ],
            ...
        }
    ],
    ...
}
```

Você também deve adicionar as seguintes permissões como uma política alinhada com a função do IAM de execução de tarefas do Amazon ECS para que suas tarefas tenham acesso ao parâmetro do Parameter Store do SSM.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameters"
            ],
            "Resource": [
            "arn:aws:ssm:arn:aws::111122223333:parameter/parameter_name"
            ]
        }
    ]
}
```

Arquivo local

Com os detalhes da especificação de credenciais em um arquivo local, faça referência ao caminho do arquivo no campo `credentialSpecs` da definição da tarefa. O caminho do arquivo referenciado deve ser relativo ao diretório `C:\ProgramData\Docker\CredentialSpecs` e usar a barra invertida ('\') como separador do caminho do arquivo.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspec:file://CredentialSpecDir\CredentialSpecFile.json"
            ],
            ...
        }
```

```
    ],
    ...
}
```

Uso do EC2 Image Builder para criar AMIs personalizadas otimizadas para Amazon ECS

A AWS recomenda usar as AMIs otimizadas para Amazon ECS por serem pré-configuradas com os requisitos e recomendações para executar as workloads de contêiner. Pode haver momentos em que você precise personalizar a AMI para adicionar software. Você pode usar o EC2 Image Builder para criar, gerenciar e implantar suas imagens de servidor. Você retém a propriedade das imagens personalizadas criadas na sua conta. Você pode usar os pipelines do EC2 Image Builder para automatizar atualizações e patches do sistema das imagens ou utilizar um comando autônomo para criar uma imagem com seus recursos de configuração definidos.

Você cria uma fórmula para a imagem. A fórmula inclui uma imagem pai e quaisquer componentes adicionais. Você também cria um pipeline que distribui sua AMI personalizada.

Você cria uma fórmula para a imagem. Uma fórmula de imagem do Image Builder é um documento que define a imagem de base e os componentes que são aplicados à imagem de base para produzir a configuração desejada para a imagem AMI de saída. Você também cria um pipeline que distribui sua AMI personalizada. Para obter mais informações, consulte How EC2 Image Builder works no Guia do usuário do EC2 Image Builder.

Recomendamos usar uma das seguintes AMIs otimizadas para o Amazon ECS como “imagem pai” no EC2 Image Builder:

- Linux
  - AL2023 x86 otimizado para o Amazon ECS
  - AMI do Amazon Linux 2023 (arm64) otimizada para Amazon ECS
  - AMI do kernel 5 do Amazon Linux 2 otimizada para o Amazon ECS
  - AMI do Amazon Linux 2 x86 otimizada para o Amazon ECS
- Windows
  - Windows 2022 Full x86 otimizado para o Amazon ECS
  - Windows 2022 Core x86 otimizado para o Amazon ECS
  - Windows 2019 Full x86 otimizado para o Amazon ECS

- Windows 2019 Core x86 otimizado para o Amazon ECS
- Windows 2016 Full x86 otimizado para o Amazon ECS

Também recomendamos selecionar “Usar a versão mais recente do sistema operacional disponível”. O pipeline usará o versionamento semântico para a imagem pai, o que ajuda a detectar as atualizações de dependência em trabalhos programados automaticamente. Para obter mais informações, consulte Semantic versioning no Guia do usuário do EC2 Image Builder.

A AWS atualiza regularmente as imagens de AMI otimizadas para Amazon ECS com patches de segurança e a nova versão do agente de contêiner. Ao usar um ID de AMI como imagem pai na fórmula de imagem, você precisa verificar regularmente se há atualizações na imagem pai. Se houver atualizações, é preciso criar uma versão da fórmula com a AMI atualizada. Isso garante que suas imagens personalizadas incorporem a versão mais recente da imagem pai. Para obter informações sobre como criar um fluxo de trabalho para atualizar automaticamente as instâncias do EC2 no cluster do Amazon ECS com as AMIs recém-criadas, consulte How to create an AMI hardening pipeline and automate updates to your ECS instance fleet.

Você também pode especificar o nome do recurso da Amazon (ARN) de uma imagem pai publicada por meio de um pipeline gerenciado do EC2 Image Builder. A Amazon publica rotineiramente imagens de AMI otimizadas para Amazon ECS por meio de pipelines gerenciados. Essas imagens são acessíveis ao público. Você deve ter as permissões corretas para acessar a imagem. Ao usar um ARN de imagem em vez de uma AMI na fórmula do Image Builder, seu pipeline usa automaticamente a versão mais recente da imagem pai sempre que é executada. Essa abordagem elimina a necessidade de criar manualmente versões da fórmula para cada atualização.

Uso do ARN de imagem com a infraestrutura como código (IaC)

Você pode configurar a fórmula usando o console do EC2 Image Builder, a infraestrutura como código (por exemplo, o AWS CloudFormation) ou o AWS SDK. Ao especificar uma imagem pai na fórmula, você pode especificar um ID de AMI do EC2, um ARN de imagem do Image Builder, um ID de produto do AWS Marketplace ou uma imagem de contêiner. A AWS disponibiliza publicamente IDs de AMI e ARNs de imagem do Image Builder de AMIs otimizadas para Amazon ECS. Este é o formato de ARN da imagem:

```
arn:${Partition}:imagebuilder:${Region}:${Account}:image/${ImageName}/${ImageVersion}
```

A `ImageVersion` tem o formato a seguir. Substitua *`principal`*, *`secundário`* e *`patch`* pelos valores mais recentes.

```
<major>.<minor>.<patch>
```

Você pode substituir `major`, `minor` e `patch` por valores específicos ou usar o ARN sem versão de uma imagem, para que o pipeline permaneça atualizado com a versão mais recente da imagem pai. Um ARN sem versão usa o formato curinga 'x.x.x' para representar a versão da imagem. Essa abordagem permite que o serviço Image Builder seja resolvido automaticamente para a versão mais recente da imagem. Usar o ARN sem versão garante que sua referência sempre direcione para a imagem mais recente disponível, simplificando o processo de manutenção de imagens atualizadas na implantação. Ao criar uma fórmula com o console, o EC2 Image Builder identifica automaticamente o ARN da imagem pai. Ao usar o IaC para criar a fórmula, você deve identificar o ARN e selecionar a versão da imagem desejada ou usar o ARN sem versão para indicar a última imagem disponível. Recomendamos criar um script automatizado para filtrar e exibir somente imagens que estejam alinhadas com seus critérios. O script em Python a seguir mostra como recuperar uma lista de AMIs otimizadas para Amazon ECS.

O script aceita dois argumentos opcionais: `owner` e `platform`, com valores padrão de "Amazon" e "Windows", respectivamente. Os valores válidos do argumento do proprietário são: `Self`, `Shared`, `Amazon` e `ThirdParty`. Os valores válidos do argumento da plataforma são `Windows` e `Linux`. Por exemplo, se você executar o script com o argumento `owner` definido como `Amazon` e o `platform` definido como `Linux`, o script gera uma lista de imagens publicadas pela Amazon, incluindo imagens otimizadas para Amazon ECS.

```
import boto3
import argparse

def list_images(owner, platform):
    # Create a Boto3 session
    session = boto3.Session()

    # Create an EC2 Image Builder client
    client = session.client('imagebuilder')

    # Define the initial request parameters
    request_params = {
        'owner': owner,
        'filters': [
            {
                'name': 'platform',
                'values': [platform]
            }
```

```
        ]
    }

    # Initialize the results list
    all_images = []

    # Get the initial response with the first page of results
    response = client.list_images(**request_params)

    # Extract images from the response
    all_images.extend(response['imageVersionList'])

    # While 'nextToken' is present, continue paginating
    while 'nextToken' in response and response['nextToken']:
        # Update the token for the next request
        request_params['nextToken'] = response['nextToken']

        # Get the next page of results
        response = client.list_images(**request_params)

        # Extract images from the response
        all_images.extend(response['imageVersionList'])

    return all_images

def main():
    # Initialize the parser
    parser = argparse.ArgumentParser(description="List AWS images based on owner and
 platform")

    # Add the parameters/arguments
    parser.add_argument("--owner", default="Amazon", help="The owner of the images.
 Default is 'Amazon'.")
    parser.add_argument("--platform", default="Windows", help="The platform type of the
 images. Default is 'Windows'.")

    # Parse the arguments
    args = parser.parse_args()

    # Retrieve all images based on the provided owner and platform
    images = list_images(args.owner, args.platform)

    # Print the details of the images
    for image in images:
```

```
        print(f"Name: {image['name']}, Version: {image['version']}, ARN:
 {image['arn']}")

if __name__ == "__main__":
    main()
```

Uso do ARN de imagem com o AWS CloudFormation

Uma fórmula de imagem do Image Builder é um esquema que especifica a imagem pai e os componentes necessários para alcançar a configuração pretendida da imagem de saída. Use o recurso `AWS::ImageBuilder::ImageRecipe`. Defina o valor de `ParentImage` para o ARN de imagem. Use o ARN sem versão da imagem desejada para garantir que o pipeline sempre use a versão mais recente da imagem. Por exemplo, `.arn:aws:imagebuilder:us-east-1:aws:image/amazon-linux-2023-ecs-optimized-x86/x.x.x` A definição do recurso `AWS::ImageBuilder::ImageRecipe` a seguir usa um ARN de imagem gerenciada pela Amazon:

```
ECSRecipe:
    Type: AWS::ImageBuilder::ImageRecipe
    Properties:
      Name: MyRecipe
      Version: '1.0.0'
      Components:
        - ComponentArn: [<The component arns of the image recipe>]
      ParentImage: "arn:aws:imagebuilder:us-east-1:aws:image/amazon-linux-2023-ecs-
optimized-x86/x.x.x"
```

Para obter mais informações sobre o recurso `AWS::ImageBuilder::ImageRecipe`, consulte o Guia do usuário do AWS CloudFormation.

Você pode automatizar a criação de imagens no pipeline definindo a propriedade de `Schedule` do recurso `AWS::ImageBuilder::ImagePipeline`. O cronograma inclui uma condição inicial e uma expressão cron. Consulte mais informações em `AWS::ImageBuilder::ImagePipeline` no Guia de Usuário AWS CloudFormation.

O exemplo de `AWS::ImageBuilder::ImagePipeline` a seguir faz com que o pipeline execute uma compilação às 10h do Tempo Universal Coordenado (UTC) todos os dias. Defina os seguintes valores de `Schedule`:

- Defina `PipelineExecutionStartCondition` como `EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE`. A compilação é iniciada somente se os recursos dependentes, como a imagem pai ou os componentes, que usam o caractere curinga ‘x’ em suas versões semânticas, forem atualizados. Isso garante que a compilação incorpore as atualizações mais recentes desses recursos.
- Defina ScheduleExpression como a expressão cron (`0 10 * * ? *`).

```
ECSPipeline:
    Type: AWS::ImageBuilder::ImagePipeline
    Properties:
      Name: my-pipeline
      ImageRecipeArn: <arn of the recipe you created in previous step>
      InfrastructureConfigurationArn: <ARN of the infrastructure configuration
 associated with this image pipeline>
      Schedule:
        PipelineExecutionStartCondition:
 EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE
        ScheduleExpression: 'cron(0 10 * * ? *)'
```

Uso do ARN de imagem com o Terraform

A abordagem para especificar a imagem pai e o cronograma do pipeline no Terraform está alinhada com a do AWS CloudFormation. Use o recurso `aws_imagebuilder_image_recipe`. Defina o valor de `parent_image` para o ARN de imagem. Use o ARN sem versão da imagem desejada para garantir que o pipeline sempre use a versão mais recente da imagem. Para obter mais informações, consulte `aws_imagebuilder_image_recipe` na documentação do Terraform.

No bloco de configuração de agendamento do `aws_imagebuilder_image_pipeline resource`, defina o valor do argumento `schedule_expression` como uma expressão cron de sua escolha para especificar a frequência de execução do pipeline e defina a `pipeline_execution_start_condition` como `EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE`. Para obter mais informações, consulte `aws_imagebuilder_image_pipeline` na documentação do Terraform.

Usar containers de aprendizado profundo da AWS no Amazon ECS

O AWS Deep Learning Containers fornece um conjunto de imagens do Docker para treinar e fornecer modelos no TensorFlow e no Apache MXNet (Incubating) no Amazon ECS. O Deep

Learning Containers fornece ambientes otimizados com bibliotecas TensorFlow, NVIDIA CUDA (para instâncias de GPU) e Intel MKL (para instâncias de CPU). As imagens de contêiner para o Deep Learning Containers estão disponíveis no Amazon ECR para referência nas definições de tarefa do Amazon ECS. É possível usar o Deep Learning Containers junto com o Amazon Elastic Inference para reduzir os custos de inferência.

Para começar a usar o Deep Learning Containers sem o Elastic Inference no Amazon ECS, consulte Configuração do Amazon ECS no Guia do desenvolvedor do AMIs de deep learning da AWS.

Cotas de serviço do Amazon ECS

Quando você cria sua Conta da AWS, definimos cotas padrão (também denominadas limites) nesses recursos da AWS de acordo com a região. Se você tentar exceder a cota de um recurso, a solicitação falhará.

O console de Service Quotas é um local central onde é possível visualizar e gerenciar suas cotas de serviços da AWS e solicitar um aumento de cota para muitos dos recursos que você usa. Use as informações de cota que fornecemos para gerenciar seus recursos. Planeje a solicitação de aumentos das cotas com antecedência antes que sejam necessários.

Para obter mais informações sobre cotas do Amazon ECS e Fargate, consulte Endpoints e cotas do Amazon ECS no Referência geral da Amazon Web Services. Para obter informações sobre o controle de utilização de API do Amazon ECS, consulte Solicitar controle de utilização para a API do Amazon ECS.

Gerenciar cotas de serviço do Amazon ECS e do AWS Fargate no AWS Management Console

O Amazon ECS foi integrado ao Service Quotas, um serviço da AWS que permite visualizar e gerenciar as cotas em um local central. Para obter mais informações, consulte O que são cotas de serviço? no Guia do usuário do Service Quotas.

O Service Quotas facilita a pesquisa do valor das cotas de serviço do Amazon ECS.

AWS Management Console

Para visualizar as cotas de serviço do Amazon ECS e do Fargate usando o AWS Management Console

1. Abra o console Service Quotas em https://console.aws.amazon.com/servicequotas/.
2. No painel de navegação, selecione serviços da AWS.
3. Na lista Serviços da AWS, procure e selecione Amazon Elastic Container Service (Amazon ECS) ou AWS Fargate.

   Na lista Service quotas é possível ver o nome da service quota, o valor aplicado (se estiver disponível), a cota padrão da AWS e se o valor da cota é ajustável.

4. Para visualizar informações adicionais sobre uma Service Quota, como a descrição, escolha o nome da cota.
5. (Opcional) Para solicitar um aumento de cota, selecione a cota que deseja aumentar, selecione Solicitar Aumento de Cota, insira ou selecione as informações necessárias e, por fim, selecione Solicitar.

Para trabalhar mais com cotas de serviço usando o AWS Management Console, consulte o Guia do usuário do Service Quotas. Para solicitar o aumento da cota, consulte Requesting a Quota Increase (Solicitar um aumento de cota) no Guia do usuário do Service Quotas.

AWS CLI

Para visualizar as cotas de serviço do Amazon ECS e do Fargate usando o AWS CLI

Execute o comando a seguir para visualizar as cotas padrão do Amazon ECS.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code ecs \
    --output table
```

Execute o comando a seguir para visualizar as cotas padrão do Fargate.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code fargate \
    --output table
```

Execute o comando a seguir para visualizar as cotas aplicadas do Fargate.

```
aws service-quotas list-service-quotas \
    --service-code fargate
```

> **Note**
>
> O Amazon ECS não é compatível com cotas aplicadas.

Para obter mais informações sobre como trabalhar com cotas de serviço usando a AWS CLI, consulte a Referência de comandos da AWS CLI do Service Quotas. Para solicitar um aumento de quotas, consulte o comando `request-service-quota-increase` na Referência de comandos do AWS CLI.

Como lidar com as cotas de serviço e os limites de controle de utilização de API do Amazon ECS

O Amazon ECS está integrado a vários Serviços da AWS, incluindo o Elastic Load Balancing, o AWS Cloud Map e o Amazon EC2. Com essa forte integração, o Amazon ECS inclui vários recursos, como balanceamento de carga do serviço, Service Connect, rede de tarefas e ajuste de escala automático de clusters. O Amazon ECS e os outros Serviços da AWS que ele integra mantêm cotas de serviço e limites de taxa de API para garantir desempenho e utilização consistentes. As cotas de serviço também evitam o provisionamento acidental de mais recursos do que o necessário e protegem contra ações maliciosas que possam aumentar sua fatura.

Ao se familiarizar com as cotas de serviço e os limites de taxa de API da AWS, você pode planejar a escalabilidade das workloads sem se preocupar com a degradação inesperada do desempenho. Para obter mais informações, consulte Request throttling for the Amazon ECS API.

Ao escalar as workloads no Amazon ECS, recomendamos que você considere a cota de serviço a seguir.

- O AWS Fargate tem cotas que limitam o número de tarefas em execução simultânea em cada Região da AWS. Há cotas para tarefas sob demanda e do Fargate Spot no Amazon ECS. Cada cota de serviço também inclui todos os pods do Amazon EKS executados no Fargate.
- Em tarefas executadas nas instâncias do Amazon EC2, o número máximo de instâncias do Amazon EC2 que você pode registrar em cada cluster é 5 mil. Caso use o ajuste de escala automático de cluster do Amazon ECS com um provedor de capacidade de grupo do Auto Scaling, ou caso gerencie instâncias do Amazon EC2 para o cluster sem ajuda, essa cota pode virar um gargalo de implantação. Se precisar de mais capacidade, você pode criar mais clusters ou solicitar um aumento na cota de serviço.
- Se você usa o ajuste de escala automático de cluster do Amazon ECS com um provedor de capacidade de grupo do Auto Scaling, considere a cota `Tasks in the PROVISIONING state per cluster` ao escalar os serviços. Essa cota é o número máximo de tarefas no estado `PROVISIONING` de cada cluster para as quais os provedores de capacidade podem aumentar a

capacidade. Ao executar um grande número de tarefas ao mesmo tempo, você pode facilmente atingir essa cota. Um exemplo é se você implantar simultaneamente dezenas de serviços, cada um com centenas de tarefas. Quando isso acontece, o provedor de capacidade precisa executar novas instâncias de contêiner para posicionar as tarefas quando o cluster não tem capacidade suficiente. Enquanto o provedor de capacidade executa instâncias adicionais do Amazon EC2, é provável que o programador de serviços do Amazon ECS continue executando tarefas em paralelo. No entanto, essa atividade pode ter controle de utilização devido à capacidade insuficiente do cluster. O programador de serviços do Amazon ECS implementa uma estratégia de recuo e controle de utilização exponencial para repetir o posicionamento de tarefas à medida que novas instâncias de contêineres são executadas. Como resultado, você pode enfrentar tempos de implantação ou de aumento horizontal de escala mais lentos. Para evitar essa situação, você pode planejar as implantações de serviços em uma das opções a seguir. Implemente um grande número de tarefas que não exijam aumento da capacidade do cluster ou mantenha uma capacidade disponível do cluster para a execução de novas tarefas.

Além de considerar a cota de serviços do Amazon ECS ao escalar as workloads, considere também a cota de serviço dos outros Serviços da AWS integrados ao Amazon ECS.

Elastic Load Balancing

Você pode configurar os serviços do Amazon ECS para usar o Elastic Load Balancing para distribuir o tráfego uniformemente entre as tarefas. Para obter mais informações e as práticas recomendadas sobre como escolher um balanceador de carga, consulte Uso do balanceamento de carga para distribuir o tráfego de serviço do Amazon ECS.

Cotas de serviço do Elastic Load Balancing

Ao escalar as workloads, considere as cotas de serviço do Elastic Load Balancing a seguir. A maioria das cotas de serviço do Elastic Load Balancing é ajustável, e você pode solicitar um aumento no console do Service Quotas.

Application Load Balancer

Ao usar um Application Load Balancer, dependendo do seu caso de uso, pode ser necessário solicitar um aumento de cota para:

- A cota de `Targets per Application Load Balancer`, que é o número de destinos por trás do Application Load Balancer.

- A cota de `Targets per Target Group per Region`, que é o número de destinos por trás dos grupos de destino.

Para obter mais informações, consulte Quotas for your Application Load Balancers no Guia do usuário para Application Load Balancers.

Network Load Balancer

Há limitações mais rígidas no número de destinos que você pode registrar com um Network Load Balancer. Ao usar um Network Load Balancer, talvez você queira habilitar o suporte entre zonas, que vem com limitações adicionais de escalabilidade no `Targets per Availability Zone Per Network Load Balancer`, o número máximo de destinos por zona de disponibilidade para cada Network Load Balancer. Para obter mais informações, consulte Quotas for your Network Load Balancers no Guia do usuário para Network Load Balancers.

Controle de utilização da API do Elastic Load Balancing

Ao configurar um serviço do Amazon ECS para usar um balanceador de carga, as verificações de integridade do grupo de destino devem ser aprovadas para que o serviço seja considerado íntegro. Para realizar essas verificações de integridade, o Amazon ECS invoca as operações de API do Elastic Load Balancing em seu nome. Caso tenha um grande número de serviços configurados com balanceadores de carga na conta, você pode tornar as implantações de serviços mais lentas devido ao possível controle de utilização específico das operações de API `RegisterTarget`, `DeregisterTarget` e `DescribeTargetHealth` do Elastic Load Balancing. Quando ocorre o controle de utilização, ocorrem erros de controle de utilização nas mensagens de eventos do serviço do Amazon ECS.

Caso enfrente problemas de controle de utilização na API do AWS Cloud Map, entre em contato com o Suporte para obter orientação sobre como aumentar os limites de controle de utilização na API do AWS Cloud Map. Para obter mais informações sobre como monitorar e solucionar problemas desses erros de controle de utilização, consulte Lidar com problemas de controle de utilização do Amazon ECS.

Interfaces de rede elástica

Com as tarefas usando o modo de rede `awsvpc`, o Amazon ECS provisiona uma interface de rede elástica (ENI) exclusiva para cada tarefa. Quando os serviços do Amazon ECS usam um balanceador de carga do Elastic Load Balancing, essas interfaces de rede também são registradas como destinos no grupo de destino apropriado definido no serviço.

Cotas de serviço da interface de rede elástica

Ao executar tarefas que usam o modo de rede `awsvpc`, uma interface de rede elástica exclusiva é anexada a cada tarefa. Caso essas tarefas precisem ser acessadas pela Internet, atribua um endereço IP público à interface de rede elástica dessas tarefas. Ao escalar as workloads do Amazon ECS, considere estas duas cotas importantes:

- A cota de `Network interfaces per Region`, que é o número máximo de interfaces de rede em uma Região da AWS para sua conta.
- A cota de `Elastic IP addresses per Region`, que é o número máximo de endereços IP elásticos em uma Região da AWS.

Ambas as cotas de serviço são ajustáveis e você pode solicitar um aumento delas no console do Service Quotas. Para obter mais informações, consulte Amazon VPC service quotas no Guia do usuário da Amazon Virtual Private Cloud.

Em workloads do Amazon ECS hospedadas em instâncias do Amazon EC2, ao executar tarefas que usam o modo de rede `awsvpc`, considere a cota de serviço `Maximum network interfaces`, o número máximo de instâncias de rede para cada instância do Amazon EC2. Essa cota limita o número de tarefas que você pode posicionar em uma instância. Não é possível ajustar a cota e ela não está disponível no console do Service Quotas. Para obter mais informações, consulte Endereços IP por interface de rede por tipo de instância no Guia do usuário do Amazon EC2.

Embora você não possa alterar o número de interfaces de rede que podem ser anexadas a uma instância do Amazon EC2, é possível usar o recurso de truncamento da interface de rede elástica para aumentar o número de interfaces de rede disponíveis. Por exemplo, por padrão, uma instância `c5.large` pode ter até três interfaces de rede. A interface de rede primária da instância conta como uma. Portanto, é possível anexar mais duas interfaces de rede à instância. Como cada tarefa que usa o modo de rede `awsvpc` requer uma interface de rede, normalmente só é possível executar duas dessas tarefas nesse tipo de instância. Isso pode levar à subutilização da capacidade do cluster. Se você habilitar o truncamento da interface de rede elástica, poderá aumentar a densidade da interface de rede para posicionar um número maior de tarefas em cada instância. Com o truncamento ativado, uma instância `c5.large` pode ter até 12 interfaces de rede. A instância tem a interface de rede primária e o Amazon ECS cria e anexa uma interface de rede de “truncamento” à instância. Como resultado, com essa configuração, você pode executar dez tarefas na instância em vez das duas tarefas padrão. Para obter mais informações, consulte Aumento das interfaces de rede de instâncias de contêiner do Linux no Amazon ECS.

Controle de utilização da API de interface de rede elástica

Ao executar tarefas que usam o modo de rede `awsvpc`, o Amazon ECS depende das APIs do Amazon EC2 mencionadas a seguir. Cada uma dessas APIs tem controles de utilização de API diferentes. Para obter mais informações, consulte Request throttling for the Amazon EC2 API na Referência de APIs do Amazon EC2.

- CreateNetworkInterface
- AttachNetworkInterface
- DetachNetworkInterface
- DeleteNetworkInterface
- DescribeNetworkInterfaces
- DescribeVpcs
- DescribeSubnets
- DescribeSecurityGroups
- DescribeInstances

Se as chamadas de API do Amazon EC2 receberem controle de utilização durante os fluxos de trabalho de provisionamento da interface de rede elástica, o programador de serviços do Amazon ECS repete automaticamente com recuos exponenciais. Às vezes, essas repetições automáticas podem levar a um atraso na execução de tarefas, resultando em velocidades de implantação mais lentas. Quando ocorrer o controle de utilização da API, você verá a mensagem `Operations are being throttled. Will try again later.` nas mensagens de eventos do serviço. Caso atenda consistentemente aos controle de utilização de API do Amazon EC2, entre em contato com o Suporte para obter orientações sobre como aumentar os limites de controle de utilização da API. Para obter mais informações sobre como monitorar e solucionar problemas de controle de utilização, consulte Handling throttling issues.

AWS Cloud Map

A descoberta de serviços do Amazon ECS e o Service Connect usam APIs do AWS Cloud Map para gerenciar namespaces para os serviços do Amazon ECS. Se os serviços tiverem um grande número de tarefas, considere as recomendações a seguir.

Cotas de serviço do AWS Cloud Map

Quando os serviços do Amazon ECS são configurados para usar a descoberta de serviços ou o Service Connect, a cota de `Tasks per service`, que é o número máximo de tarefas do serviço, é afetada pela cota de serviço de `Instances per service` do AWS Cloud Map, que é o número máximo de instâncias desse serviço. Especificamente, a cota de serviço do AWS Cloud Map diminui a quantidade de instâncias que você pode executar para, no máximo, 1.000 instâncias por serviço. Não é possível alterar a cota do AWS Cloud Map. Para obter mais informações, consulte as Service Quotas do AWS Cloud Map.

Controle de utilização da API do AWS Cloud Map

O Amazon ECS chama as APIs `ListInstances`, `GetInstancesHealthStatus`, `RegisterInstance` e `DeregisterInstance` do AWS Cloud Map em seu nome. Elas ajudam na descoberta de serviços e realizam verificações de integridade ao executar uma tarefa. Quando vários serviços que usam a descoberta de serviços com um grande número de tarefas são implantados ao mesmo tempo, isso pode resultar na ultrapassagem dos limites de controle de utilização da API do AWS Cloud Map. Quando isso acontecer, é provável que surja a mensagem `Operations are being throttled. Will try again later` nas mensagens de eventos do serviço do Amazon ECS e que a velocidade de implantação e execução de tarefas fique mais lenta. O AWS Cloud Map não documenta os limites de controle de utilização dessas APIs. Caso enfrente problemas de controle de utilização, entre em contato com o Suporte para obter orientações sobre como aumentar os limites de controle de utilização da API. Para obter mais recomendações sobre como monitorar e solucionar esses erros de controle de utilização, consulte Lidar com problemas de controle de utilização do Amazon ECS.

Referência da API do Amazon ECS

Além do AWS Management Console e da AWS Command Line Interface (AWS CLI), o Amazon ECS também fornece uma API. É possível usar a API para automatizar tarefas de gerenciamento de recursos do Amazon ECS.

- Para obter uma lista de operações de API pelo recurso Amazon ECS, consulte Ações pelo recurso Amazon ECS.
- Para obter uma lista alfabética de operações da API, consulte Ações.
- Para obter uma lista alfabética de tipos de dados, consulte Tipos de dados.
- Para obter uma lista de parâmetros de consulta comuns, consulte Parâmetros comuns.
- Para obter descrições dos códigos de erro, consulte Erros comuns.

Para obter mais informações sobre a AWS CLI, consulte a Referência da AWS Command Line Interface para o Amazon ECS.

Histórico do documento

A tabela a seguir descreve as principais atualizações e os novos recursos do Guia do desenvolvedor do Amazon Elastic Container Service. Também atualizamos a documentação com frequência para abordar os comentários enviados por você.

Alteração	Descrição	Data
Adicione permissões à política gerenciada AmazonEC2ContainerServiceforEC2Role.	A política `AmazonEC2ContainerServiceforEC2Role` gerenciada do IAM foi atualizada para incluir a permissão `ecs:ListTagsForResource`. Essa permissão permite que o agente do Amazon ECS recupere tags de instâncias de tarefas e contêineres por meio do endpoint de metadados da tarefa (`${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`). Para obter mais informações, consulte AmazonEC2ContainerServiceforEC2Role.	4 de agosto de 2025
Adicione permissões à política AmazonECSInfrastructureRolePolicyForLoadBalancers.	A política `AmazonECSInfrastructureRolePolicyForLoadBalancers` gerenciada do IAM foi atualizada com novas permissões para descrever, cancelar o registro e registrar grupos de destino.	25 de julho de 2025
Suporte à atualização de um controlador de implantação de serviços	O Amazon ECS é compatível com a atualização de um controlador de implantação de serviços, permitindo que você alterne entre diferentes tipos de implantação de um serviço existente. Para obter mais informações, consulte Atualizar parâmetros de serviço do Amazon ECS.	15 de julho de 2025
Implantações azul/verde nativas do Amazon ECS	O Amazon ECS é compatível com implantações azul/verde nativas, permitindo que você valide novas revisões de serviços antes de direcionar o tráfego de produção para elas. Para obter mais informações, consulte Amazon ECS blue/green deployments.	15 de julho de 2025

Alteração	Descrição	Data
Adicione a nova política gerenciada AmazonECS InfrastructureRole PolicyForLoadBalancers.	Essa política fornece acesso a outros recursos de serviços da AWS necessários para gerenciar balanceadores de carga associados a workloads do Amazon ECS em seu nome. Para obter mais informações, consulte AmazonECSInfrastructureRole PolicyForLoadBalancers.	15 de julho de 2025
Suporte para SOCI Index Manifest v2	Os clientes do Fargate que quiserem usar o SOCI só poderão usar o SOCI Index Manifest v2. Os clientes existentes que já usaram o SOCI no Fargate poderão continuar usando o SOCI Index Manifest v1, no entanto, é altamente recomendável que esses clientes migrem para o SOCI Index Manifest v2. Para obter mais informações, consulte Carregamento lento de imagens de contêiner usando Seekable OCI (SOCI).	02 de julho de 2025
Inclusão da ID da tarefa em eventos de ação de serviço gerados em decorrência de falhas de integridade.	Para obter mais informações, consulte Mensagens de evento não íntegro de serviço do Amazon ECS.	30 de junho de 2025
Atualize para o modo de driver de log padrão.	O modo padrão de registro em log do Amazon ECS foi atualizado de `blocking` para `non-blocking` . Você pode substituir esse modo padrão especificando um `mode` na definição de contêiner `logConfiguration` ou configurar seu próprio padrão definindo a configuração de conta `defaultLogDriverMode` usando o console do Amazon ECS ou a AWS CLI. Para obter mais informações, consulte Modo de driver de logs padrão.	25 de junho de 2025

Alteração	Descrição	Data
Adicione permissões à política AmazonECS ServiceRolePolicy.	A política gerenciada do IAM `AmazonECS ServiceRolePolicy` foi atualizada com novas permissões AWS Cloud Map, permitindo que o Amazon ECS atualize atributos de serviço AWS Cloud Map para serviços gerenciados pelo Amazon ECS.	24 de junho de 2025
Suporte para a AMI do Windows Server 2025 Full otimizada para Amazon ECS e AMI do Windows Server 2025 Core otimizada para Amazon ECS.	Para obter mais informações consulte AMIs do Windows otimizadas para Amazon ECS.	20 de junho de 2025
Suporte à atualização de uma estratégia de provedor de capacidade de serviço.	Agora, você pode atualizar uma estratégia de provedor de capacidade de serviço. Para obter mais informaçõ es, consulte Atualizar parâmetros de serviço.	12 de junho de 2025
Adicionar permissões a AmazonECSInfrastru ctureRolePolicyFor Volumes	A política `AmazonECSInfrastructureRole PolicyForVolumes` foi atualizada para adicionar a permissão `ec2:DescribeInstances` . Essa permissão ajuda a impedir a colisão de nome de dispositivo para volumes do Amazon EBS anexados a tarefas do Amazon ECS em execução na mesma instância de contêiner.	2 de junho de 2025
Suporte à taxa de inicializ ação de volume para volumes do Amazon EBS associados a tarefas do Amazon ECS.	Agora você pode definir uma `volumeInt iializationRate` para determinar a taxa, em MiB/s, na qual os dados são obtidos de um snapshot de um volume existente do Amazon EBS para criar novos volumes para anexação às tarefas do Amazon ECS. Para obter mais informações, consulte volumeIni tializationRate.	13 de maio de 2025

Alteração	Descrição	Data
Suporte à interrupção da implantação de um serviço.	Agora é possível interromper a implantação de um serviço. Para obter mais informações, consulte Parar implantações de serviços do Amazon ECS.	2 de maio de 2025
Suporte a endpoints de pilha dupla do Amazon ECS.	Agora é possível usar endpoints de pilha dupla para interagir com o Amazon ECS. Os endpoints de pilha dupla oferecem suporte a solicitações ao Amazon ECS tanto via Internet Protocol versão 4 (IPv4) quanto via Internet Protocol versão 6 (IPv6). Para obter mais informações, consulte Usar endpoints de pilha dupla do Amazon ECS.	30 de abril de 2025
Suporte para configuração de um modo de driver de logs padrão no nível da conta em uma Região da AWS.	Agora você pode configurar um modo de entrega de mensagens de log padrão de um contêiner para o driver de logs que você especificar na `logConfiguration` usando a configuração de conta `defaultLogDriverMode`. Para obter mais informações, consulte Modo de driver de logs padrão.	17 de abril de 2025
Suporte para migração de um serviço com um formato curto de ARN para o formato longo de ARN.	Agora você pode migrar um serviço com um formato curto de ARN para o formato longo de ARN. Para obter mais informações, consulte Migrate an Amazon ECS short service ARN to a long ARN.	13 de fevereiro de 2025
Suporte para três tipos de injeção de falhas no Amazon ECS e Fargate.	Agora, você pode utilizar a porta de buraco negro de rede, latência de rede e injeção de falhas por perda de pacotes de rede com o Amazon ECS e o Fargate. Para obter mais informações, consulte Utilizar o AWS Fault Injection Service para testar suas workloads do Amazon ECS.	19 de dezembro de 2024

Alteração	Descrição	Data
Suporte para o CloudWatch Container Insights com observabilidade aprimorada.	Você pode utilizar o CloudWatch Container Insights com observabilidade aprimorada. Para obter mais informações, consulte Monitorar contêineres do Amazon ECS utilizando o Container Insights com observabilidade aprimorada.	2 de dezembro de 2024
Suporte a escalabilidade preditiva.	É possível usar a escalabilidade preditiva para antecipar necessidades futuras e aumentar proativamente as tarefas em seu serviço, conforme necessário. Para obter mais informações, consulte Usar padrões históricos para escalar serviços do CloudWatch com escalabilidade preditiva.	21 de novembro de 2024
Suporte ao rebalanceamento de serviços.	Agora você pode fazer com que o Amazon ECS equilibre automaticamente suas tarefas de serviço em todas as zonas de disponibilidade. Para obter mais informações, consulte Balancear um serviço do Amazon ECS entre zonas de disponibilidade.	19 de novembro de 2024
Suporte à configuração de resolução de tags de imagem de contêiner para resumos de imagem.	Agora é possível configurar a resolução das tags de imagem do contêiner para resumos de cada contêiner em tarefas que fazem parte de um serviço. Para obter mais informações, consulte Resolução da imagem do contêiner e `versionConsistency`.	19 de novembro de 2024
Adicionar uma nova política do IAM `AmazonECSInfrastructureRolePolicyForVpcLattice`	A política `AmazonECSInfrastructureRolePolicyForVpcLattice` fornece acesso a outros recursos do serviço da AWS necessários para gerenciar volumes associados a workloads do Amazon ECS em seu nome.	18 de novembro de 2024

Alteração	Descrição	Data
O VPC Lattice agora está disponível para serviços do Amazon ECS.	Agora você pode usar o VPC Lattice para padronizar a conectividade entre serviços, a segurança e a observabilidade. Para obter mais informações, consulte Usar o Amazon VPC Lattice para conectar, observar e proteger seus serviços do Amazon ECS.	18 de novembro de 2024
Suporte a histórico de serviços usando implantações de serviços e revisões de serviços.	É possível usar implantações e revisões de serviços para visualizar o histórico de implantações. Para obter mais informações, consulte Visualizar o histórico de serviços usando implantações de serviços do Amazon ECS e Revisões de serviços do Amazon ECS.	30 de outubro de 2024
Suporte a volumes do Amazon EBS para o tipo de lançamento do EC2 usando o sistema operacional Windows.	O Amazon ECS oferece suporte a volumes do Amazon EBS anexados a tarefas executadas em instâncias do EC2 com o sistema operacional Windows. Para obter mais informações, consulte Usar volumes do Amazon EBS com o Amazon ECS.	24 de outubro de 2024
Adição de permissões a AmazonECSInfrastructureRolePolicyForVolumes.	A política `AmazonECSInfrastructureRolePolicyForVolumes` foi atualizada para permitir que os clientes criem um volume do Amazon EBS com base em um snapshot. Para obter mais informações, consulte AmazonECSInfrastructureRolePolicyForVolumes.	10 de outubro de 2024
O Fargate Spot está disponível para Linux ARM.	O suporte ao Fargate Spot no Linux ARM foi adicionado ao Amazon ECS.	6 de setembro de 2024

Alteração	Descrição	Data
Política de reinicialização de contêineres para contêineres individuais em tarefas do Amazon ECS.	Você pode habilitar políticas de reinicialização para contêineres essenciais e não essenciais determinados em sua definição de tarefa para superar falhas transitórias com mais rapidez e manter a disponibilidade das tarefas. Para obter mais informações, consulte Reiniciar contêineres individuais em tarefas do Amazon ECS com políticas de reinicialização de contêineres.	15 de agosto de 2024
Adicionou permissões a the section called “AmazonECS_FullAccess”	A política `AmazonECS_FullAccess` foi atualizada para adicionar permissões `iam:PassRole` para perfis do IAM para um perfil chamado `ecsInfrastructureRole`. Este é o perfil padrão do IAM criado pelo AWS Management Console, destinado a ser usado como um perfil de infraestrutura do ECS que permite ao Amazon ECS gerenciar volumes do Amazon EBS anexados às tarefas do ECS.	13 de agosto de 2024
Suporte para Amazon Linux 2023, CentOS Stream 9, Fedora 40 e Ubuntu 24 adicionado ao ECS Anywhere	O suporte para os sistemas operacionais Amazon Linux 2023, CentOS Stream 9, Fedora 40 e Ubuntu 24 foi adicionado ao ECS Anywhere. Para obter mais informações, consulte Sistemas operacionais e arquiteturas de sistema compatíveis.	23 de julho de 2024
Resolução de imagem de contêiner para serviços que usam o tipo de implantação atualização contínua (ECS).	Para garantir que todas as tarefas em um serviço que usa o controlador de implantação de atualização contínua usem a mesma imagem de contêiner, o Amazon ECS resolve nomes de imagens de contêiner e quaisquer tags de imagem especificadas na definição da tarefa para resumos de imagens de contêiner. Para obter mais informações, consulte Resolução da imagem de contêiner.	10 de julho de 2024

Alteração	Descrição	Data
Suporte para Debian 11 e Debian 12 adicionado ao ECS Anywhere	O suporte para os sistemas operacionais Debian 11 e Debian 12 foi adicionado ao ECS Anywhere. Para obter mais informações, consulte Sistemas operacionais e arquiteturas de sistema compatíveis.	28 de março de 2024
Suporte a gMSA para contêineres do Linux no Fargate	O Amazon ECS oferece suporte à autenticação do Active Directory para contêineres do Linux no Fargate por meio de um tipo especial de conta de serviço denominada Conta de serviço gerenciada por grupo (gMSA). Para obter mais informações, consulte Using gMSA for Linux containers on Fargate.	5 de março de 2024
Métricas do CloudWatch adicionadas aos volumes do Amazon EBS anexados às tarefas	O Amazon ECS agora publica as métricas do CloudWatch para a utilização do armazenamento do Amazon EBS em tarefas que têm um volume do Amazon EBS anexado. Para obter mais informações, consulte Amazon ECS CloudWatch metrics.	8 de fevereiro de 2024
TLS do Service Connect	Agora você pode usar o TLS com o Service Connect.	22 de janeiro de 2024
Política gerenciada de TLS do Service Connect	Foi adicionada a nova política AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity.	22 de janeiro de 2024
Atualização da configuração de tempo limite do Service Connect	A configuração de tempo limite do Service Connect agora pode ser atualizada e inclui dois parâmetros opcionais: `idleTimeout` e `perRequestTimeout`.	22 de janeiro de 2024
Drenagem de instâncias gerenciadas do Amazon ECS	Você pode usar a drenagem de instâncias gerenciadas do Amazon ECS para facilitar o encerramento normal das instâncias do Amazon ECS.	19 de janeiro de 2024

Alteração	Descrição	Data
Suporte ao Ubuntu 22 adicionado para o ECS Anywhere	O suporte ao sistema operacional Ubuntu 22 foi adicionado ao ECS Anywhere. Para obter mais informações, consulte Sistemas operacionais e arquiteturas de sistema compatíveis.	16 de janeiro de 2024
Adicionar política do IAM `AmazonECSInfrastructureRolePolicyForVolumes`	O `AmazonECSInfrastructureolePolicyForVolumes` foi adicionado. A política concede as permissões necessárias ao Amazon ECS para fazer chamadas de API da AWS a fim de gerenciar volumes do Amazon EBS associados às workloads do Amazon ECS.	11 de janeiro de 2024
Volume de dados do Amazon EBS para uma tarefa do Amazon ECS	Você pode configurar um volume de dados do Amazon EBS por tarefa durante a implantação para anexar a tarefas autônomas do Amazon ECS ou tarefas gerenciadas por um serviço do ECS. Configurar um volume na implantação permite criar definições de tarefas reutilizáveis não restritas a tipos ou configurações de volume específicos. Os volumes do Amazon EBS fornecem um armazenamento em blocos altamente disponível, econômico, durável e de alto desempenho para workloads em contêineres com uso intenso de dados.	11 de janeiro de 2024
O console clássico do Amazon ECS chegou ao fim da vida útil	O console do Amazon ECS chegou ao fim da vida útil.	4 de dezembro de 2023
Política atualizada	A política do IAM AmazonECSServiceRolePolicy gerenciada foi atualizada com novas permissões de `events` e permissões adicionais de `autoscaling` e `autoscaling-plans`.	4 de dezembro de 2023

Alteração	Descrição	Data
Suporte ao monitoramento de runtime	Você pode usar o monitoramento de runtime para monitorar as workloads do Amazon ECS e identificar comportamentos maliciosos ou não autorizados. Para obter mais informações, consulte Runtime Monitoring.	26 de novembro de 2023
Política atualizada	A política do IAM gerenciada `AmazonECSServiceRolePolicy` foi atualizada para permitir o acesso à API AWS Cloud Map `DiscoverInstancesRevision` .	4 de outubro de 2023
Configuração de retirada de tarefas do AWS Fargate	É possível configurar o período de espera até que as tarefas do Fargate sejam retiradas. Para obter mais informações, consulte Manutenção de tarefas do AWS Fargate.	5 de setembro de 2023
Parâmetros adicionais de definição de tarefa no AWS Fargate	Adicionado suporte no AWS Fargate para `pidMode` e `systemControls` na versão da plataforma Linux `1.4.0`. Para obter mais informações, consulte Definições de tarefas.	9 de agosto de 2023
Redesenho da página de definição de tarefas do console do Amazon ECS	A página de definição de tarefas do console do Amazon ECS foi redesenhada e contém opções adicionais. Para obter mais informações, consulte Criação de uma definição de tarefa com a utilização do console.	26 de julho de 2023
O Fargate oferece suporte a carregamento lento com índices Seekable OCI	O AWS Fargate está introduzindo índices Seekable OCI (SOCI). Com o SOCI, os contêineres gastam apenas alguns segundos na extração da imagem antes de serem iniciados, fornecendo tempo para a configuração do ambiente e a instanciação da aplicação enquanto a imagem é baixada em segundo plano. Para obter mais informações, consulte Carregamento lento de imagens de contêiner usando Seekable OCI (SOCI).	17 de julho de 2023

Alteração	Descrição	Data
Suporte aprimorado para gMSA em Linux e Windows	A definição da tarefa tem um novo campo `credentialSpecs` para gMSA para Linux e Windows. Um novo tutorial completo para gMSA sem domínio no Windows foi adicionado. Consulte Tutorial: uso de contêineres do Windows com gMSA sem domínio usando a AWS CLI Para obter mais informações, consulte Uso de gMSAs para contêineres de Linux e Uso de gMSAs para contêineres de Windows.	14 de julho de 2023
Documentação aprimorada das versões do ECS Agent	A documentação das versões do agente do Amazon ECS foi atualizada. Recomendamos que você use a versão `v20.10.13` ou posterior do Docker com a versão mais recente do agente de contêiner do Amazon ECS. As versões lançadas e as alterações no agente estão disponíveis no GitHub. Para obter mais informações, consulte Versões do agente de contêiner de Linux do Amazon ECS.	20 de junho de 2023
Disponibilidade de regiões atualizada para suporte ao ARM64 do Fargate	A disponibilidade de regiões para o ARM64 do Fargate foi atualizada. Para obter mais informações, consulte Considerações.	19 de junho de 2023
Melhorias na documentação do ajuste de escala automático de clusters	A documentação para a escalabilidade do Amazon ECS do Amazon EC2 Auto Scaling tem melhorias significativas de precisão e legibilidade. Para obter mais informações, consulte Ajuste de escala automático de cluster no Amazon ECS.	4 de maio de 2023

Alteração	Descrição	Data
Autorização de atribuição de tags para a criação de recursos.	Os usuários devem ter permissões para ações que criam o recurso, como `ecsCreateCluster` . Quando você cria um recurso e especifica tags para esse recurso, a AWS executa uma autorização adicional para verificar se há permissões para criar tags. Para obter mais informações, consulte Autorização para a atribuição de tags e Conceder permissão para atribuir tags a recursos na criação.	18 de abril de 2023
Supporte para gMSA para contêineres do Linux no EC2	É possível usar o gMSA para se autenticar no Active Directory para contêineres do Linux no EC2. Para obter mais informações, consulte Uso de gMSAs para contêineres do Linux.	14 de abril de 2023
Suporte ao armazenamento temporário para contêineres do Windows no AWS Fargate	É possível usar armazenamento temporário para contêineres do Windows no AWS Fargate. Para obter mais informações, consulte Armazenamento de tarefas do Fargate.	14 de abril de 2023
Suporte do AWS Cost Management para dados CUR em nível de tarefa	É possível ativar o custo em nível de tarefa e o uso de recursos nos Relatórios de custo e uso. Isso adiciona dados de alocação de custos divididos para tarefas executadas no AWS Fargate e no EC2. Para obter mais informações, consulte Relatórios de custo e uso em nível de tarefa.	12 de abril de 2023
AMI do Amazon Linux 2023 otimizada para Amazon ECS	É possível implantar workloads na AMI otimizada para Amazon ECS do Amazon Linux 2023. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.	10 de abril de 2023

Alteração	Descrição	Data
Padrão Federal de Processamento de Informações (FIPS) 140 do AWS Fargate	É possível implantar workloads no Amazon ECS no AWS Fargate de forma compatível com o Federal Information Processing Standard (FIPS) 140. Para obter mais informações, consulte Padrão Federal de Processamento de Informações (FIPS-140) do AWS Fargate.	10 de abril de 2023
Exclusão de definição de tarefa	É possível excluir uma definição de tarefa usando o console do Amazon ECS, o SDK e a AWS CLI. Para obter mais informações, consulte Exclusão de uma revisão de definição de tarefa usando o console e Definições de tarefas.	24 de fevereiro de 2023
Recomendações de serviços do AWS Fargate no Compute Optimizer	O AWS Compute Optimizer gera recomendações de tarefas e tamanho de contêineres com base na utilização de tarefas em execução em serviços do Amazon ECS no AWS Fargate. Para obter mais informações, consulte Visualização de recomendações para serviços do Amazon ECS no Fargate.	27 de janeiro de 2023
Console do Amazon ECS	Agora, o novo console do Amazon ECS é o console padrão. Para obter mais informações, consulte Novo console do Amazon ECS.	19 de janeiro de 2023
Atualizada a política `AmazonECS_FullAccess` do IAM	A política `AmazonECS_FullAccess` do IAM foi atualizada para incluir permissões para adicionar tags aos balanceadores de carga durante a criação. Para obter mais informações, consulte AmazonECS_FullAccess.	4 de janeiro de 2023
Usar alarmes do CloudWatch para detectar falhas na implantação de serviço do Amazon ECS	É possível configurar o Amazon ECS para definir que a implantação falhou quando ele detectar que um alarme especificado do CloudWatch entrou no estado de ALARM (ALARME). Para obter mais informações, consulte the section called “Implantações de atualização cumulativa”.	19 de dezembro de 2022

Alteração	Descrição	Data
Suporte para mapeamento de portas de contêineres	É possível definir um intervalo de números de portas no contêiner vinculado ao intervalo de portas host mapeado dinamicamente. Para obter mais informações, consulte the section called “Mapeamentos de porta”.	15 de dezembro de 2022
Disponibilidade geral do Amazon ECS Service Connect	Esse recurso adiciona descoberta de serviços e malha de serviços que são controladas pelas implantações de serviço do Amazon ECS. Para obter mais informações, consulte the section called “ Service Connect”.	27 de novembro de 2022
A experiência do novo console do Amazon ECS para definições de tarefa foi atualizada	A experiência do novo console do Amazon ECS agora contém um editor de JSON para definições de tarefas. Para obter mais informações, consulte the section called “Criação de uma definição de tarefa usando o console”.	27 de outubro de 2022
A experiência do novo console do Amazon ECS para definições de tarefa foi atualizada	A experiência do novo console do Amazon ECS agora contém um editor de JSON para definições de tarefas. Para obter mais informações, consulte the section called “Criação de uma definição de tarefa usando o console”.	27 de outubro de 2022
A experiência do novo console do Amazon ECS foi atualizada	A experiência do novo console do Amazon ECS foi atualizada com parâmetros adicionais de serviço e tarefa. Para obter mais informações, consulte the section called “Criação de uma implantação de atualização contínua” e the section called “Execução de uma aplicação como uma tarefa”.	7 de outubro de 2022
Novas informações no endpoint de metadados de tarefas versão 4	O endpoint de metadados de tarefas versão 4 agora contém o ID da VPC e o nome do serviço. Para obter mais informações, consulte the section called “Endpoint de metadados de tarefas versão 4”.	7 de outubro de 2022

Alteração	Descrição	Data
Tamanhos da nova definição de tarefa	O Amazon ECS no Fargate agora oferece suporte aos tamanhos de tarefas de 8 vCPU e 16 vCPU. Para obter mais informações, consulte the section called “Tamanho da tarefa”.	16 de setembro de 2022
Páginas da CLI do ECS arquivadas	A documentação da CLI do ECS foi arquivada. Recomendamos usar o AWS Copilot para suas necessidades de ferramentas de linha de comando. Para obter mais informações, consulte Criar de recursos do Amazon ECS usando a interface de linha de comando do AWS Copilot.	15 de setembro de 2022
Novas cotas do Fargate	O Fargate está fazendo a transição de cotas baseadas em contagem de tarefas para cotas baseadas em vCPU. Para obter mais informações, consulte Endpoints e cotas do Amazon ECS.	8 de setembro de 2022
Suporte para grupos de alta atividade do Amazon EC2 Auto Scaling.	Agora, você pode usar grupos de alta atividade do Amazon EC2 Auto Scaling para aumentar a escala na horizontal das suas aplicações mais rapidamente e economizar custos. Para obter mais informações, consulte Configuração de instâncias inicializadas previamente para o grupo do Amazon ECS Auto Scaling.	23 de março de 2022
Suporte para instâncias do Windows no ECS Anywhere.	O ECS Anywhere agora oferece suporte a instâncias do Windows. Para obter mais informações, consulte Clusters do Amazon ECS para o tipo de inicialização externa.	3 de março de 2022
Adicionado suporte ao ECS Exec para instâncias externas	O ECS Exec agora oferece suporte a instâncias externas. Para obter mais informações, consulte Monitoramento de contêineres do Amazon ECS com o ECS Exec.	24 de janeiro de 2022

Alteração	Descrição	Data
A experiência do novo console do Amazon ECS atualizada	A experiência do novo console do Amazon ECS oferece suporte à criação e exclusão de um cluster, atualização de uma definição de tarefa e cancelamento do registro de uma definição de tarefa. Para ter mais informações, consulte Criação de um cluster do Amazon ECS para o tipo de inicialização do Fargate, Exclusão de um cluster do Amazon ECS, Atualizar uma definição de tarefa do Amazon ECS usando o console e Cancelar registro de uma revisão de definição de tarefa do Amazon ECS usando o console.	8 de dezembro de 2021
A experiência do novo console do Amazon ECS atualizada	A experiência do novo console do Amazon ECS oferece suporte à criação de uma definição de tarefa. Para obter mais informações, consulte Criar uma definição de tarefa do Amazon ECS usando o console.	23 de novembro de 2021
O Amazon ECS oferece suporte à arquitetura ARM de 64 bits para Linux.	O Amazon ECS oferece suporte à arquitetura de CPU ARM de 64 bits para o sistema operacional Linux. Para obter mais informações, consulte the section called “Definições de tarefa para workloads do ARM de 64 bits”.	23 de novembro de 2021
Suporte do Amazon ECS para a opção log-driver-buffer-limit do fluentd	O Amazon ECS oferece suporte à opção `log-driver-buffer-limit` do fluentd. Para obter mais informações, consulte Envio de logs do Amazon ECS para um serviço da AWS ou para uma AWS Partner.	22 de novembro de 2021
Script de compilação da AMI do Linux otimizada para o Amazon ECS	O Amazon ECS abriu o código dos scripts de compilação usados para criar as variantes do Linux da AMI otimizada para o Amazon ECS. Para obter mais informações, consulte Script de compilação da AMI do Linux otimizada para o Amazon ECS.	19 de novembro de 2021

Alteração	Descrição	Data
Integridade de instâncias do contêiner	O Amazon ECS adiciona suporte para o monitoramento de integridade de instâncias de contêiner. Para obter mais informações, consulte Monitoramento da integridade da instância de contêiner do Amazon ECS.	10 de novembro de 2021
Suporte para Exec do Amazon ECS para Windows	O Exec do Amazon ECS oferece suporte ao Windows. Para obter mais informações, consulte Monitoramento de contêineres do Amazon ECS com o ECS Exec.	1º de novembro de 2021
Supporte para contêineres do Windows no Fargate.	O Amazon ECS oferece suporte para contêineres do Windows no Fargate. Para obter mais informações, consulte Log de alterações da versão da plataforma Windows para o Fargate.	28 de outubro de 2021
Suporte a GPU para instâncias externas no Amazon ECS Anywhere	O Amazon ECS oferece suporte à especificação de requisitos de GPU na definição de tarefa para tarefas executadas em instâncias externas. Para obter mais informações, consulte Definições de tarefa do Amazon ECS para workloads de GPU e Registro de uma instância externa para um cluster do Amazon ECS.	8 de outubro de 2021
Suporte do modo de rede `awsvpc` no Windows	O Amazon ECS oferece suporte ao modo de rede `awsvpc` no Windows. Para obter mais informações, consulte Alocar uma interface de rede para uma tarefa do Amazon ECS.	15 de julho de 2021
Disponibilidade geral do Bottlerocket	O Amazon ECS oferece suporte a uma variante da AMI otimizada para o Amazon ECS do sistema operacional Bottlerocket que é fornecida como uma AMI. Para obter mais informações, consulte AMIs Bottlerocket otimizadas para Amazon ECS.	30 de junho de 2021
Atualização de tarefas programadas do Amazon ECS	O Amazon EventBridge adicionou suporte para parâmetros adicionais ao criar regras que acionam tarefas programadas do Amazon ECS.	25 de junho de 2021

Alteração	Descrição	Data
Políticas gerenciadas pela AWS para o Amazon ECS	O Amazon ECS adicionou documentação de políticas gerenciadas pela AWS para funções vinculadas a serviços. Para obter mais informações, consulte Políticas gerenciadas pela AWS para o Amazon Elastic Container Service.	8 de junho de 2021
Conceitos básicos da AWS CDK	Adicionado um guia de conceitos básicos para usar o AWS CDK com o Amazon ECS. Para obter mais informações, consulte Criar recursos do Amazon ECS usando o AWS CDK.	27 de maio de 2021
Amazon ECS Anywhere	O Amazon ECS adicionou suporte para registrar um servidor on-premises ou uma máquina virtual (VM) no cluster. Para obter mais informações, consulte Clusters do Amazon ECS para o tipo de inicialização externa.	25 de maio de 2021
AMI do Windows Server 20H2 Core otimizada para Amazon ECS	O Amazon ECS adicionou suporte para uma nova variante da AMI otimizada para Amazon ECS para Windows Server 20H2 Core. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.	19 de abril de 2021
Amazon ECS Exec	O Amazon ECS lançou uma nova ferramenta de depuração chamada ECS Exec. Para obter mais informações, consulte Monitoramento de contêineres do Amazon ECS com o ECS Exec.	15 de março de 2021
Suporte à politica do endpoint da VPC	O Amazon ECS agora oferece suporte às políticas de endpoint da VPC. Para obter mais informações, consulte Criar uma política de endpoint da VPC para o Amazon ECS.	11 de janeiro de 2021

Alteração	Descrição	Data
Nova experiência do console	O Amazon ECS lançou uma nova experiência de console que oferece suporte à criação ou atualizaç ão de um serviço ou à execução de uma tarefa autônoma. Para obter mais informações, consulte Criação de uma implantação de atualização contínua do Amazon ECS e Execução de uma aplicação como uma tarefa do Amazon ECS.	28 de dezembro de 2020
Atualizar provedor de capacidade	O Amazon ECS adicionou suporte para a atualizaç ão de um provedor de capacidade de grupo do Auto Scaling existente.	23 de novembro de 2020
Agora, o ECS oferece suporte ao Amazon FSx for Windows File Server para tarefas do Windows	O Amazon ECS adicionou suporte para especificar volumes do Amazon FSx for Windows File Server nas definições de tarefa do Windows. Para obter mais informações, consulte Uso de volumes do FSx para Windows File Server com Amazon ECS.	11 de novembro de 2020
Adicionado suporte ao modo de pilha dupla da VPC	O Amazon ECS adicionou suporte para o uso de uma VPC no modo de pilha dupla com tarefas usando o modo de rede `awsvpc`, que fornece suporte para endereços IPv6. Para obter mais informações, consulte Usar uma VPC no modo de pilha dupla.	5 de novembro de 2020
Atualização do endpoint de metadados de tarefas v4	O Amazon ECS adicionou metadados à saída do endpoint de metadados de tarefas v4. Para obter mais informações, consulte Endpoint de metadados de tarefas do Amazon ECS versão 4.	5 de novembro de 2020
Suporte para Local Zones e zonas do Wavelength	O Amazon ECS adicionou suporte para workloads em Local Zones e zonas do Wavelength. Para obter mais informações, consulte Aplicações do Amazon ECS em sub-redes compartilhadas, zonas locais e zonas do Wavelength.	4 de setembro de 2020

Alteração	Descrição	Data
Variante do Amazon ECS da AMI do Bottlerocket	O Bottlerocket é um sistema operacional de código aberto baseado em Linux que foi criado especific amente pela AWS para executar contêineres. É fornecida uma variante da AMI otimizada para Amazon ECS do sistema operacional Bottlerocket que é fornecida como uma AMI que pode ser usada quando forem iniciadas instâncias de contêiner do Amazon ECS. Para obter mais informações, consulte AMIs Bottlerocket otimizadas para Amazon ECS.	31 de agosto de 2020
Endpoint de metadados da tarefa versão 4 atualizado para dados estatísticos de taxa de rede	O endpoint de metadados da tarefa versão 4 foi atualizado para fornecer dados estatísticos de taxa de rede para tarefas do Amazon ECS que usam os modos de rede `awsvpc` ou `bridge` hospedados em instâncias do Amazon EC2 que executam pelo menos a versão `1.43.0` do agente de contêiner. Para obter mais informações, consulte Endpoint de metadados de tarefas do Amazon ECS versão 4.	10 de agosto de 2020
Métricas de uso do Fargate	O AWS Fargate fornece métricas de uso do CloudWatch que fornecem visibilidade do uso, pela sua conta, de recursos sob demanda do Fargate e do Fargate Spot. Para obter mais informações, consulte Métricas de uso.	3 de agosto de 2020
AWS Copilot versão 0.1.0	O novo AWS Copilot CLI foi lançado, fornecendo comandos de alto nível para simplificar a modelagem , a criação, o lançamento e o gerenciamento de aplicações em contêineres no Amazon ECS em um ambiente de desenvolvimento local. Para obter mais informações, consulte Criar de recursos do Amazon ECS usando a interface de linha de comando do AWS Copilot.	9 de julho de 2020

Alteração	Descrição	Data
Programação de substituição de versões da plataforma do AWS Fargate	O programa de substituição da versão da plataforma do Fargate foi adicionado. Para obter mais informações, consulte Descontinuação da versão da plataforma Linux do AWS Fargate.	8 de julho de 2020
Expansão da região do AWS Fargate	O Amazon ECS no AWS Fargate foi expandido para a região da Europa (Milão).	25 de junho de 2020
AMI do Amazon Linux 2 (Neuron) otimizada para Amazon ECS lançada	O Amazon ECS lançou uma AMI do Amazon Linux 2 (Neuron) otimizada para o Amazon ECS para workloads inferenciais. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.	24 de junho de 2020
Adicionado suporte para exclusão de provedores de capacidade	O Amazon ECS adicionou suporte para excluir provedores de capacidade de grupo do Auto Scaling.	11 de junho de 2020
Atualização da versão 1.4.0 da plataforma do AWS Fargate	A partir de 28 de maio de 2020, qualquer nova tarefa do Fargate que seja iniciada usando a versão 1.4.0 da plataforma terá seu armazenamento temporário de 20 GB criptografado com um algoritmo de criptografia AES-256 usando uma chave de criptografia gerenciada pelo AWS Fargate. Para obter mais informações, consulte Armazenamento efêmero de tarefas do Fargate para o Amazon ECS.	28 de maio de 2020
Suporte a arquivo de variáveis de ambiente	Foi adicionado suporte para especificar arquivos de variáveis de ambiente em uma definição de tarefa, o que permite adicionar variáveis de ambiente em massa a seus contêineres. Para obter mais informações, consulte Transferência de uma variável de ambiente individual para um contêiner do Amazon ECS.	18 de maio de 2020

Alteração	Descrição	Data
Expansão da região do AWS Fargate	A AWS Fargate com Amazon ECS foi expandido para a região da África (Cidade do Cabo).	11 de maio de 2020
Cota de serviço atualizada	A seguinte cota de serviço foi atualizada: • Os clusters por conta foram elevados de `2,000` para `10,000`. Para obter mais informações, consulte Endpoints e cotas do Amazon ECS.	17 de abril de 2020

Alteração	Descrição	Data
Versão 1.4.0 da plataforma do AWS Fargate	Lançada a versão 1.4.0 da plataforma do AWS Fargate, contendo os seguintes recursos: • Adicionado suporte para usar volumes do sistema de arquivos do Amazon EFS para o armazenamento de tarefas persistentes. Para obter mais informações, consulte Uso de volumes do Amazon EFS com o Amazon ECS. • O armazenamento efêmero de tarefas foi aumentado para 20 GB. Para obter mais informações, consulte Armazenamento efêmero de tarefas do Fargate para o Amazon ECS. • O comportamento de tráfego de rede de e para tarefas foi atualizado. A partir da versão 1.4 da plataforma, todas as tarefas do Fargate recebem uma única interface de rede elástica (conhecida como ENI de tarefa), e todo o tráfego de rede flui por essa ENI dentro da VPC e será visível por meio dos logs de fluxo da VPC. Para obter mais informações, consulte Opções de rede de tarefas do Amazon ECS para o tipo de execução do Fargate. • As ENIs de tarefas adicionam suporte para quadros jumbo. As interfaces de rede são configuradas com uma unidade de transmissão máxima (MTU), que se refere ao tamanho da maior carga útil que cabe em um único quadro. Quanto maior a MTU, maior será a carga útil do aplicativo que pode caber em um único quadro, o que reduz a sobrecarga por quadro e aumenta a eficiência. O suporte a quadros jumbo reduzirá a sobrecarga quando o caminho de rede entre a tarefa e o destino oferecer suporte a quadros	8 de abril de 2020

Alteração	Descrição	Data
	jumbo, assim como todo o tráfego que permanece dentro da VPC. • O CloudWatch Container Insights inclui métricas de performance de rede para tarefas do Fargate. Para obter mais informações, consulte Monitorar contêineres do Amazon ECS utilizando o Container Insights com observabilidade aprimorada. • Adicionado suporte para o endpoint de metadados de tarefas v4, que fornece informações adicionais para as tarefas do Fargate, incluindo dados estatísticos de rede da tarefa e em qual zona de disponibilidade a tarefa está sendo executada. Para obter mais informações, consulte Endpoint de metadados de tarefas do Amazon ECS versão 4. • Suporte adicionado para o parâmetro Linux `SYS_PTRACE` nas definições de contêiner. Para obter mais informações, consulte Parâmetros do Linux. • O agente de contêiner do Fargate substitui o uso do agente de contêiner do Amazon ECS do para todas as tarefas do Fargate. Essa alteração não deve afetar a forma como suas tarefas são executadas. • O runtime do contêiner agora está usando containerd em vez do docker. Essa alteração não deve afetar a forma como suas tarefas são executadas. Você perceberá que algumas mensagens de erro provenientes do runtime do contêiner deixarão de mencionar o Docker e mostrarão erros mais gerais.	

Alteração	Descrição	Data
	Para obter mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.	
Suporte do sistema de arquivos do Amazon EFS para volumes de tarefas	O sistema de arquivos do Amazon EFS pode ser usado como volumes de dados para as tarefas do Amazon ECS e do Fargate. Para obter mais informaçõ es, consulte Uso de volumes do Amazon EFS com o Amazon ECS.	8 de abril de 2020
Versão 4 do endpoint de metadados de tarefas do Amazon ECS	A partir da versão 1.39.0 do agente de contêiner do Amazon ECS e da versão 1.4.0 da plataforma do Fargate, uma variável de ambiente denominad a `ECS_CONTAINER_METADATA_URI_V4` é injetada em cada contêiner de uma tarefa. Quando você consultar o endpoint de metadados de tarefas versão 4, vários metadados de tarefas e estatísticas do Docker estarão disponíveis para tarefas. Para obter mais informações, consulte Endpoint de metadados de tarefas do Amazon ECS versão 4.	8 de abril de 2020
Suporte para versões específicas de segredos do Secrets Manager a serem injetadas como variáveis de ambiente	Adicionado suporte para especificar dados sigilosos usando versões específicas de segredos do Secrets Manager. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.	24 de fevereiro de 2020
Adicionadas opções de configuração de implantação do CodeDeploy para implantações azuis/ver des	O serviço do CodeDeploy adicionou novas configura ções de implantação linear e canary para o tipo de implantação do Amazon ECS. A capacidade de definir configurações de implantação personalizadas também está disponível. Para obter mais informações, consulte Implantações azul/verde do CodeDeploy para o Amazon ECS.	6 de fevereiro de 2020

Alteração	Descrição	Data
Adicionamos o parâmetro de definição de tarefa efsVolumeConfiguration	O parâmetro de definição de tarefa `efsVolume Configuration` está em pré-visualização pública, o que facilita o uso de sistemas de arquivos do Amazon EFS com as tarefas do Amazon ECS. Para obter mais informações, consulte Uso de volumes do Amazon EFS com o Amazon ECS.	17 de janeiro de 2020
Atualizado o comportamento do registro do agente de contêiner do Amazon ECS	Foi atualizado o comportamento dos locais e da alternância do registro do agente de contêiner do Amazon ECS. Para obter mais informações, consulte Parâmetros de configuração do log do agente de contêiner do Amazon ECS.	13 de janeiro de 2020
Fargate Spot	Adicionado o suporte do Amazon ECS para executar tarefas usando o Fargate Spot. Para obter mais informações, consulte Clusters do Amazon ECS para Fargate.	3 de dezembro de 2019
Auto Scaling de cluster	A autoescalabilidade de clusters do Amazon ECS permite um maior controle sobre como escalar tarefas dentro de um cluster. Para obter mais informações, consulte Gerenciamento automático da capacidade do Amazon ECS com ajuste de escala automático de cluster.	3 de dezembro de 2019
Provedores de capacidade de cluster	Os provedores de capacidade de cluster do Amazon ECS determinam a infraestrutura a ser usada para as tarefas. Para obter mais informações, consulte Clusters do Amazon ECS.	3 de dezembro de 2019
Criar um cluster em um AWS Outposts	O Amazon ECS agora oferece suporte para a criação de clusters no AWS Outposts. Para obter mais informações, consulte the section called “Amazon Elastic Container Service no AWS Outposts”.	3 de dezembro de 2019

Alteração	Descrição	Data
Eventos de ação do serviço	O Amazon ECS agora envia eventos para o Amazon EventBridge quando determinadas ações de serviço ocorrem. Para obter mais informações, consulte Eventos de ação do serviço do Amazon ECS.	25 de novembro de 2019
A AMI otimizada para GPU do Amazon ECS oferece suporte a instâncias G4	O Amazon ECS adicionou suporte para a família de tipos de instânciagG4 ao usar a AMI otimizada para GPU do Amazon ECS. Para obter mais informações, consulte Definições de tarefa do Amazon ECS para workloads de GPU.	8 de outubro de 2019
FireLens para Amazon ECS	O FireLens para Amazon ECS está disponível para o público. O FireLens para Amazon ECS permite usar parâmetros de definição de tarefa para encaminhar logs para um serviço da AWS ou para o destino de um parceiro para armazenamento e análise de logs. Para obter mais informações, consulte Envio de logs do Amazon ECS para um serviço da AWS ou para uma AWS Partner.	30 de setembro de 2019
Expansão da região do AWS Fargate	O AWS Fargate com Amazon ECS foi expandido para as regiões da Europa (Paris), Europa (Estocolmo) e Oriente Médio (Bahrein).	30 de setembro de 2019
Deep Learning Containers com Elastic Inference no Amazon ECS	O Amazon ECS oferece suporte à anexação de aceleradores do Amazon Elastic Inference a contêineres para tornar a execução de workloads de inferência de aprendizado profundo mais eficiente.	3 de setembro de 2019

Alteração	Descrição	Data
FireLens para Amazon ECS	O FireLens para Amazon ECS está em pré-visualização pública. O FireLens para Amazon ECS permite usar parâmetros de definição de tarefa para encaminhar logs para um serviço da AWS ou para o destino de um parceiro para armazenamento e análise de logs. Para obter mais informações, consulte Envio de logs do Amazon ECS para um serviço da AWS ou para uma AWS Partner.	30 de agosto de 2019
CloudWatch Container Insights	O CloudWatch Container Insights já está disponível para o público em geral. Ele permite coletar, agregar e resumir métricas e logs de seus aplicativos e microsserviços em contêineres. Para obter mais informações, consulte Monitorar contêineres do Amazon ECS utilizando o Container Insights com observabilidade aprimorada.	30 de agosto de 2019
Configuração de troca no nível do contêiner	O Amazon ECS adicionou suporte para controlar o uso de espaço de memória de troca em suas instâncias de contêiner do Linux no nível do contêiner. Usando uma configuração de troca por contêiner, cada contêiner em uma definição de tarefa pode ter a troca habilitada ou desabilitada, e, para aqueles que a têm habilitada, a quantidade máxima de espaço de troca usado pode ser limitada. Para obter mais informações, consulte Gerenciar o espaço de memória de troca de contêiner no Amazon ECS.	16 de agosto de 2019
Expansão da região do AWS Fargate	O AWS Fargate com Amazon ECS foi expandido para a região Ásia-Pacífico (Hong Kong).	6 de agosto de 2019

Alteração	Descrição	Data
Entroncamento da interface de rede elástica	Adicionados tipos de instância do Amazon EC2 compatíveis com o recurso de entroncamento de ENI. Para obter mais informações, consulte Instâncias com suporte para o aumento de interfaces de rede de contêineres do Amazon ECS.	1 de agosto de 2019
Registro de vários grupos de destino com um serviço	Adicionado suporte para especificar vários grupos de destino em uma definição de serviço. Para obter mais informações, consulte Registrar vários grupos de destino em um serviço Amazon ECS.	30 de julho de 2019
Especificar dados sigilosos usando segredos do Secrets Manager	Adicionado tutorial para especificar dados sigilosos usando segredos do Secrets Manager. Para obter mais informações, consulte Especificar dados confidenciais usando segredos do Secrets Manager no Amazon ECS.	20 de julho de 2019
CloudWatch Container Insights	O Amazon ECS adicionou suporte para o CloudWatch Container Insights. Para obter mais informações, consulte Monitorar contêineres do Amazon ECS utilizando o Container Insights com observabilidade aprimorada.	9 de julho de 2019
Permissões no nível do recurso para serviços e conjuntos de tarefas do Amazon ECS	O Amazon ECS expandiu o suporte às permissões no nível do recurso para serviços e conjuntos de tarefas do Amazon ECS. Para obter mais informações, consulte Como o Amazon Elastic Container Service funciona com o IAM.	27 de junho de 2019
Nova AMI com patch otimizada para Amazon ECS para AWS-2019-005	O Amazon ECS atualizou a AMI otimizada para Amazon ECS para lidar com as vulnerabilidades descritas no AWS-2019-005.	17 de junho de 2019

Alteração	Descrição	Data
Entroncamento da interface de rede elástica	O Amazon ECS apresenta suporte para iniciar instâncias de contêiner usando tipos de instância do Amazon EC2 compatíveis que tenham aumentado a densidade da interface de rede elástica (ENI). Usar esses tipos de instância e escolher a configuração da conta `awsvpcTrunking` fornecem maior densidade de ENI em instâncias de contêiner recém-iniciadas, o que permite colocar mais tarefas em cada instância de contêiner. Para obter mais informações, consulte Aumento das interfaces de rede de instâncias de contêiner do Linux no Amazon ECS.	6 de junho de 2019
Atualização da versão 1.3.0 da plataforma do AWS Fargate	A partir de 1.º de maio de 2019, todas as novas tarefas do Fargate que iniciadas oferecem suporte ao driver de log `splunk`, além do driver de log `awslogs`. Para obter mais informações, consulte Armazenamento e registro.	1º de maio de 2019
Atualização da versão 1.3.0 da plataforma do AWS Fargate	A partir de 1.º de maio de 2019, todas as novas tarefas do Fargate iniciadas oferecem suporte à referênci a a dados sigilosos na configuração de log de um contêiner usando o parâmetro `secretOptions` de definição do contêiner. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.	1º de maio de 2019
Atualização da versão 1.3.0 da plataforma do AWS Fargate	A partir de 2 de abril de 2019, todas as novas tarefas do Fargate iniciadas oferecem suporte à injeção de dados sigilosos nos seus contêineres. Isso é feito armazenando seus dados sigilosos em segredos do AWS Secrets Manager ou em parâmetros do AWS Systems Manager Parameter Store e fazendo referência a eles na definição do contêiner. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.	2 de abril de 2019

Alteração	Descrição	Data
Atualização da versão 1.3.0 da plataforma do AWS Fargate	A partir de 27 de março de 2019, qualquer nova tarefa do Fargate lançada pode usar parâmetros adicionais de definição de tarefa que permitem que você defina uma configuração de proxy, dependências para inicialização e desligamento de contêiner, além de um valor de tempo limite de início e interrupção por contêiner. Para ter mais informações, consulte Configuração do proxy, Dependência de contêiner e Tempos limite de contêiner.	27 de março de 2019
O Amazon ECS apresenta o tipo de implantação externa	O tipo de implantação externa permite que você use qualquer controlador de implantação de terceiros para o controle total do processo de implantação para um serviço do Amazon ECS. Para obter mais informações, consulte Implantação de serviços do Amazon ECS usando um controlador de terceiros.	27 de março de 2019
AWS Deep Learning Containers no Amazon ECS	AWS Deep Learning Containers são um conjunto de imagens do Docker para treinar e fornecer modelos em TensorFlow no Amazon Elastic Container Service (Amazon ECS). Deep Learning Containers fornecem ambientes otimizados com TensorFlow e bibliotecas Intel MKL (para instâncias de CPI), Nvidia CUDA (para instâncias de GPU) e estão disponíveis no Amazon ECR. Para obter mais informações, consulte Usar containers de aprendizado profundo da AWS no Amazon ECS.	27 de março de 2019
O Amazon ECS apresenta um melhor gerenciamento de dependências de contêiner	O Amazon ECS apresenta parâmetros adicionais de definição de tarefa que permitem definir dependências para inicialização e desligamento de contêineres, além de um valor de tempo limite de início e de interrupção por contêiner. Para obter mais informações, consulte Dependência de contêiner.	7 de março de 2019

Alteração	Descrição	Data
O Amazon ECS apresenta a API `PutAccountSettingDefault`	O Amazon ECS apresenta a API `PutAccountSettingDefault`, que permite que o usuário defina o status de aceitação do formato padrão de ARN/ID para todos os usuários e perfis na conta. Anteriormente, definir o status de aceitação padrão da conta exigia o uso do proprietário da conta. Para obter mais informações, consulte Nomes de recursos da Amazon (ARNs) e IDs.	8 de fevereiro de 2019
O Amazon ECS é compatível com workloads de GPU	O Amazon ECS é compatível com workloads de GPU, permitindo que você crie clusters com instâncias de contêiner habilitadas para GPU. Em uma definição de tarefa, você pode especificar o número de GPUs necessárias e o agente do ECS atribuirá GPUs físicas ao contêiner. Para obter mais informações, consulte Definições de tarefa do Amazon ECS para workloads de GPU.	4 de fevereiro de 2019
O Amazon ECS expandiu o suporte a segredos	O Amazon ECS expandiu o suporte para o uso de segredos do AWS Secrets Manager diretamente nas definições de tarefa para injetar dados sigilosos nos contêineres. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.	21 de janeiro de 2019

Alteração	Descrição	Data
Endpoints da VPC de interface (AWS PrivateLink)	Adicionado suporte para a configuração de endpoints da VPC de interface com tecnologia AWS PrivateLink. Isso permite a criação de uma conexão privada entre a VPC e o Amazon ECS, sem necessidade de acesso pela Internet, por meio de uma instância NAT, de uma conexão VPN ou do AWS Direct Connect. Para obter mais informações, consulte Endpoints da VPC de interface (AWS PrivateLink).	26 de dezembro de 2018
Versão 1.3.0 da plataforma do AWS Fargate	Lançada nova versão da plataforma do AWS Fargate, que contém: • Suporte adicionado para o uso dos parâmetros do Parameter Store do AWS Systems Manager para injetar dados confidenciais em seus contêineres. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS. • Adicionada a reciclagem de tarefas para as tarefas do Fargate, que é o processo de atualização das tarefas que fazem parte de um serviço do Amazon ECS. Para obter mais informações, consulte Retirada e manutenção de tarefas para AWS Fargate no Amazon ECS. Para obter mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.	17 de dezembro de 2018

Alteração	Descrição	Data
Service limits atualizados	Os seguintes service limits foram atualizados: • O número de clusters por região, por conta, foi elevado de `1000` para `2000`. • O número de instâncias de contêiner por cluster foi elevado de `1000` para `2000`. • O número de serviços por cluster foi elevado de `500` para `1000`. Para obter mais informações, consulte Cotas de serviço do Amazon ECS.	14 de dezembro de 2018
Expansão da região do AWS Fargate	O AWS Fargate com Amazon ECS foi expandido para as regiões Ásia-Pacífico (Mumbai) e Canadá (Central). Para obter mais informações, consulte Regiões com suporte para Amazon ECS no AWS Fargate.	7 de dezembro de 2018
Implantações azuis/verdes do Amazon ECS	O Amazon ECS adicionou suporte para implantações azuis/verdes usando o CodeDeploy. Esse tipo de implantação permite que você verifique uma nova implantação de um serviço antes de enviar tráfego de produção para ele. Para obter mais informações, consulte Implantações azul/verde do CodeDeploy para o Amazon ECS.	27 de novembro de 2018
Lançada a AMI do Amazon Linux 2 (arm64) otimizada para Amazon ECS	O Amazon ECS lançou uma AMI do Amazon Linux 2 otimizada para Amazon ECS para arquitetura arm64. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.	26 de novembro de 2018

Alteração	Descrição	Data
Adicionado suporte para sinalizadores do Docker adicionais em definições de tarefa	O Amazon ECS introduziu suporte para os seguintes sinalizadores do Docker em definições de tarefa: • Modo IPC • Modo PID	16 de novembro de 2018
Suporte a segredos do Amazon ECS	Adicionado suporte ao Amazon ECS para o uso dos parâmetros do AWS Systems Manager Parameter Store para injetar dados sigilosos nos contêineres. Para obter mais informações, consulte Transferência de dados confidenciais para um contêiner do Amazon ECS.	15 de novembro de 2018
Marcação de recursos	O Amazon ECS adicionou suporte para adicionar etiquetas de metadados aos seus serviços, definições de tarefa, tarefas, clusters e instâncias de contêiner. Para obter mais informações, consulte Marcação de recursos do Amazon ECS.	15 de novembro de 2018
Expansão da região do AWS Fargate	O AWS Fargate com Amazon ECS foi expandido para as regiões Oeste dos EUA (Norte da Califórnia) e Ásia-Pacífico (Seul). Para obter mais informações, consulte AWS Fargate para o Amazon ECS.	7 de novembro de 2018

Alteração	Descrição	Data
Service limits atualizados	Os seguintes service limits foram atualizados: • O número de tarefas que usam o tipo de inicializ ação do Fargate, por região e por conta foi elevado de 20 para 50. • O número de endereços IP públicos de tarefas que usam o tipo de inicialização do Fargate foi elevado de 20 para 50. Para obter mais informações, consulte Cotas de serviço do Amazon ECS.	31 de outubro de 2018
Expansão da região do AWS Fargate	O AWS Fargate com Amazon ECS foi expandido para a região da Europa (Londres). Para obter mais informações, consulte AWS Fargate para o Amazon ECS.	26 de outubro de 2018
Lançada a AMI do Amazon Linux 2 otimizada para Amazon ECS	O Amazon ECS oferece AMIs do Linux que são otimizadas para o serviço em duas variantes. A versão mais recente e recomendada é baseada em x;. O Amazon ECS também oferece AMIs que são baseadas na AMI do Amazon Linux, mas recomenda mos migrar as workloads para a variante do Amazon Linux 2, já que o suporte para a AMI do Amazon Linux terminará no máximo em 30 de junho de 2020. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.	18 de outubro de 2018

Alteração	Descrição	Data
Versão 3 do endpoint de metadados de tarefas do Amazon ECS	A partir da versão 1.21.0 do agente de contêineres do Amazon ECS, o agente injeta uma variável de ambiente denominada `ECS_CONTAINER_METADATA_URI` em cada contêiner de uma tarefa. Quando você consulta a versão 3 do endpoint de metadados de tarefas, vários metadados de tarefas e Dados estatísticos do Docker estão disponíveis para as tarefas que usam o modo de rede `awsvpc` em um endpoint HTTP fornecido pelo agente de contêiner do Amazon ECS. Para obter mais informações, consulte Monitoramento de workloads usando metadados do Amazon ECS.	18 de outubro de 2018
Expansão da região de descoberta do serviço do Amazon ECS	A descoberta do serviço do Amazon ECS expandiu o suporte para as regiões Canadá (Central), América do Sul (São Paulo), Ásia-Pacífico (Seul), Ásia-Pacífico (Mumbai) e Europa (Paris). Para obter mais informações, consulte Uso da descoberta de serviços para conectar serviços do Amazon ECS com nomes DNS.	27 de setembro de 2018
Adicionado suporte para sinalizadores do Docker adicionais em definições de contêiner	O Amazon ECS introduziu suporte para os seguintes sinalizadores do Docker em definições de contêiner: • Controles do sistema • Interativo • Pseudoterminal	17 de setembro de 2018

Alteração	Descrição	Data
Suporte de autenticação de registro privado para Amazon ECS usando tarefas do AWS Fargate	O Amazon ECS introduziu suporte para autenticação de registro privado de tarefas do Fargate usando o AWS Secrets Manager. Esse recurso permite que você armazene suas credenciais de forma segura e, então, referencie-as em sua definição de contêiner, o que permite que suas tarefas usem imagens privadas. Para obter mais informações, consulte Uso de imagens de contêiner que não são da AWS no Amazon ECS.	10 de setembro de 2018
Expansão da região de descoberta do serviço do Amazon ECS	A descoberta de serviços do Amazon ECS expandiu suporte para as regiões Ásia-Pacífico (Singapura), Ásia-Pacífico (Sydney), Ásia-Pacífico (Tóquio), Europa (Frankfurt) e Europa (Londres). Para obter mais informações, consulte Uso da descoberta de serviços para conectar serviços do Amazon ECS com nomes DNS.	30 de agosto de 2018
Tarefas programadas com suporte a tarefas do Fargate	O Amazon ECS introduziu suporte para tarefas programadas para o tipo de inicialização do Fargate.	28 de agosto de 2018
Autenticação de registro privado usando o suporte do AWS Secrets Manager	O Amazon ECS introduziu suporte para autenticação de registro privado usando o AWS Secrets Manager. Esse recurso permite que você armazene suas credenciais de forma segura e, então, referencie-as em sua definição de contêiner, o que permite que suas tarefas usem imagens privadas. Para obter mais informações, consulte Uso de imagens de contêiner que não são da AWS no Amazon ECS.	16 de agosto de 2018

Alteração	Descrição	Data
Adição de suporte a volume do Docker	O Amazon ECS apresentou o suporte para volumes do Docker. Para obter mais informações, consulte Opções de armazenamento para tarefas do Amazon ECS.	9 de agosto de 2018
Expansão da região do AWS Fargate	O AWS Fargate com Amazon ECS foi expandido para as regiões da Europa (Frankfurt), Ásia-Pacífico (Singapura) e Ásia-Pacífico (Sydney). Para obter mais informações, consulte AWS Fargate para o Amazon ECS.	19 de julho de 2018

Alteração	Descrição	Data
Adicionadas estratégias do programador de serviço do Amazon ECS	O Amazon ECS introduziu o conceito de estratégias de programador de serviço. Há duas estratégias de programador de serviços disponíveis: • `REPLICA`: a estratégia de programação de réplica coloca e mantém o número desejado de tarefas no seu cluster. Por padrão, o programador de serviço distribui tarefas por zonas de disponibilidade. É possível usar estratégias e limitações de posicionamento de tarefas para personalizar decisões de posicionamento de tarefa. Para obter mais informações, consulte Estratégia de agendamento de réplicas. • `DAEMON`: a estratégia de programação do daemon implanta exatamente uma tarefa em cada instância de contêiner ativa que atenda a todas as restrições de posicionamento de tarefas que você especifica no seu cluster. Ao usar essa estratégia, não há necessidade de especificar um número desejado de tarefas, uma estratégia de posicionamento de tarefas ou usar políticas de Auto Scaling do serviço. Para obter mais informações, consulte Estratégia de agendamento de daemon. **Note** As tarefas do Fargate não são compatíveis com a estratégia de programação do `DAEMON`. .	12 de junho de 2018

Alteração	Descrição	Data
Agente de contêiner do Amazon ECS v1.18.0	Lançada nova versão do agente do contêiner do Amazon ECS, que adicionou a seguinte funcionalidade: • Adicionado suporte para personalizar o comportamento de pull de imagem do agente de contêiner usando o parâmetro `ECS_IMAGE_PULL_BEHAVIOR` . Para obter mais informações, consulte Configuração do agente de contêiner do Amazon ECS. Para obter mais informações, consulte amazon-ecs-agent github.	24 de maio de 2018
Adicionado suporte para modos de rede `bridge` e `host` ao configurar a descoberta de serviço	Adicionado suporte para configurar a descoberta de serviço para serviços do Amazon ECS usando definições de tarefa que especificam o modo de rede `bridge` ou `host`. Para obter mais informações, consulte Uso da descoberta de serviços para conectar serviços do Amazon ECS com nomes DNS.	22 de maio de 2018
Adicionado suporte para parâmetros de metadados adicionais da AMI otimizada para Amazon ECS	Adicionados subparâmetros que permitem recuperar de forma programática o ID da AMI otimizada para Amazon ECS, o nome da imagem, o sistema operacional, a versão do agente de contêiner e a versão do runtime. Consulte os metadados usando a API do Systems Manager Parameter Store. Para obter mais informações, consulte Recuperação de metadados da AMI do Linux otimizada para o Amazon ECS.	9 de maio de 2018

Alteração	Descrição	Data
Expansão da região do AWS Fargate	O AWS Fargate com o Amazon ECS foi expandido para as regiões Leste dos EUA (Ohio), Oeste dos EUA (Oregon) e Oeste da Europa (Irlanda). Para obter mais informações, consulte AWS Fargate para o Amazon ECS.	26 de abril de 2018
Recuperação dos metadados da AMI otimizada para Amazon ECS	Adicionada capacidade de recuperação programática dos metadados da AMI otimizada para Amazon ECS usando a API do Systems Manager Parameter Store. Para obter mais informações, consulte Recuperação de metadados da AMI do Linux otimizada para o Amazon ECS.	10 de abril de 2018
Versão da plataforma do AWS Fargate	Lançada nova versão da plataforma do AWS Fargate, que contém: • O suporte adicionado para Monitoramento de workloads usando metadados do Amazon ECS. • O suporte adicionado para Verificação de integridade. • Adicionado o suporte para Uso da descoberta de serviços para conectar serviços do Amazon ECS com nomes DNS Para obter mais informações, consulte Versões da plataforma do Fargate para o Amazon ECS.	26 de março de 2018
Descoberta de serviço do Amazon ECS	Adicionada integração com o Route 53 para oferecer suporte à descoberta de serviço do Amazon ECS. Para obter mais informações, consulte Uso da descoberta de serviços para conectar serviços do Amazon ECS com nomes DNS.	22 de março de 2018

Alteração	Descrição	Data
Suporte a shm-size e tmpfs do Docker	Adicionado suporte aos parâmetros shm-size e tmpfs do Docker em definições de tarefa do Amazon ECS. Para obter mais informações sobre a sintaxe de CLI do ECS atualizada, consulte Parâmetros do Linux.	20 de março de 2018
Verificações de integrida de do contêiner	Inclusão de suporte às verificações de integridade do Docker nas definições de contêiner. Para obter mais informações, consulte Verificação de integridade.	8 de março de 2018
AWS Fargate	Adicionada visão geral do Amazon ECS com AWS Fargate. Para obter mais informações, consulte AWS Fargate para o Amazon ECS.	22 de fevereiro de 2018
Endpoint de metadados de tarefas do Amazon ECS	A partir da versão 1.17.0 do agente de contêiner do Amazon ECS, vários metadados de tarefas e dados estatísticos do Docker estão disponíveis para tarefas que usam o modo de rede `awsvpc` em um endpoint HTTP fornecido pelo agente de contêiner do Amazon ECS. Para obter mais informações, consulte Monitoramento de workloads usando metadados do Amazon ECS.	8 de fevereiro de 2018
Auto Scaling de serviços do Amazon ECS usando políticas de monitoram ento do objetivo	Adicionado suporte ao Auto Scaling de serviços do ECS usando políticas de monitoramento do objetivo no console do Amazon ECS. Para obter mais informaçõ es, consulte Use uma métrica de destino para escalar os serviços do Amazon ECS. Remoção do tutorial anterior de escalabilidade em etapas do assistente de primeira execução do ECS. Ele foi substituído pelo novo tutorial de rastreamento de destino.	8 de fevereiro de 2018

Alteração	Descrição	Data
Suporte para o Docker 17.09	Adicionado suporte para Docker 17.09. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.	18 de janeiro de 2018
Novo comportamento do programador de serviços	Informações atualizadas sobre o comportamento de tarefas de serviços que não puderam ser iniciadas. Foi documentada uma nova mensagem de evento de serviço que é acionada quando uma tarefa de serviço apresenta falhas consecutivas.	11 de janeiro de 2018
Período de espera da inicialização da verificação de integridade do Elastic Load Balancing	Adicionada capacidade de especificar um período de espera para verificações de integridade.	27 de dezembro de 2017
CPU e memória em nível de tarefa	Adicionado suporte para especificação de CPU e memória em nível de tarefa nas definições de tarefa. Para obter mais informações, consulte TaskDefinition.	12 de dezembro de 2017
Função de execução de tarefas	O agente de contêiner do Amazon ECS faz chamadas para as ações de API do Amazon ECS em seu nome. Dessa forma, ele exige uma política e uma função do IAM para o serviço saber se o agente pertence a você. As ações a seguir estão cobertas pela tarefa função de execução: • Chamadas para o Amazon ECR extrair a imagem de contêiner • Chamadas para o CloudWatch para armazenar os logs da aplicação do contêiner Para obter mais informações, consulte Função do IAM de execução de tarefas do Amazon ECS.	7 de dezembro de 2017

Alteração	Descrição	Data
Contêineres do Windows têm suporte para GA	Adicionado suporte para contêineres do Windows Server 2016. Para obter mais informações, consulte Variantes da AMI otimizada para Amazon ECS.	5 de dezembro de 2017
AWS Fargate disponível para o público	Adicionado suporte para iniciar serviços do Amazon ECS usando o tipo de inicialização do Fargate. Para obter mais informações, consulte Tipos de inicialização do Amazon ECS.	29 de novembro de 2017
Mudança de nome do Amazon ECS	O Amazon Elastic Container Service foi renomeado (anteriormente, Amazon EC2 Container Service).	21 de novembro de 2017
Redes de tarefas	Os recursos de redes de tarefas fornecidos pelo modo de rede awsvpc fornecem às tarefas do Amazon ECS as mesmas propriedades de redes que as instâncias do Amazon EC2. Quando você usa o modo de rede awsvpc em suas definições de tarefa, todas as tarefas iniciadas a partir dessa definição obterão sua própria interface de rede elástica, um endereço IP privado primário e um nome de host DNS interno. O recurso de rede de tarefas simplifica a rede de contêineres e oferece a você mais controle sobre como aplicativos em contêineres se comunicam entre si e com outros serviços dentro das suas VPCs. Para obter mais informações, consulte Opções de redes de tarefas do Amazon ECS para o tipo de inicialização do EC2.	14 de novembro de 2017
Metadados do contêiner do Amazon ECS	Agora, os contêineres do Amazon ECS podem acessar metadados, como o contêiner do Docker ou ID de imagem, configuração de redes ou ARNs da Amazon. Para obter mais informações, consulte Arquivo de metadados de contêiner do Amazon ECS.	2 de novembro de 2017

Alteração	Descrição	Data
Suporte para o Docker 17.06	Adicionado suporte para Docker 17.06. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.	2 de novembro de 2017
Suporte para sinalizadores do Docker: device e init	Adicionado suporte para recursos de device e init do Docker nas definições de tarefa usando o parâmetro `LinuxParameters` (`devices` e `initProcessEnabled`). Para obter mais informações, consulte LinuxParameters.	2 de novembro de 2017
Suporte para sinalizadores do Docker: cap-add e cap-drop	Adicionado suporte para recursos de cap-add e cap-drop do Docker nas definições de tarefa usando o parâmetro `LinuxParameters` (`capabilities`). Para obter mais informações, consulte LinuxParameters.	22 de setembro de 2017
Suporte para Network Load Balancer	O Amazon ECS adicionou suporte a network load balancers no console do Amazon ECS.	7 de setembro de 2017
Substituições do RunTask	Adicionado suporte para substituições de definições de tarefa ao executar uma tarefa. Isso permite executar uma tarefa ao alterar uma definição de tarefa sem precisar criar uma nova revisão de definição de tarefa. Para obter mais informações, consulte Execução de uma aplicação como uma tarefa do Amazon ECS.	27 de junho de 2017
Tarefas programadas do Amazon ECS	Adicionado suporte para programar tarefas usando cron.	7 de junho de 2017
Instâncias spot no console do Amazon ECS	Adicionado suporte para a criação de instâncias de contêineres de frota spot no console do Amazon ECS. Para obter mais informações, consulte Iniciar uma instância de contêiner do Linux do Amazon ECS.	6 de junho de 2017

Alteração	Descrição	Data
Notificação do Amazon SNS para novas versões da AMI otimizada para Amazon ECS	Adicionada capacidade para assinatura de notificaç ões do SNS sobre novas versões da AMI otimizada para Amazon ECS.	23 de março de 2017
Microsserviços e trabalhos em lote	Adicionada documentação para dois casos de uso comum para microsserviços e trabalhos em lote do Amazon ECS.	Fevereiro de 2017
Drenagem de instâncias de contêineres	Adicionado suporte para drenagem de instâncias de contêineres que fornece um método para remover instâncias de contêineres de um cluster. Para obter mais informações, consulte Drenagem de instâncias de contêiner do Amazon ECS.	24 de janeiro de 2017
Suporte para o Docker 1.12	Adicionado suporte para Docker 1.12. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.	24 de janeiro de 2017
Novas estratégias de posicionamento de tarefas	Adicionado suporte para estratégias de posiciona mento de tarefas: posicionamento com base em atributos, agrupamento, distribuição de zona de disponibilidade e um por host. Para obter mais informações, consulte Uso de estratégias para definir o posicionamento de tarefas do Amazon ECS.	29 de dezembro de 2016
Suporte para contêiner es do Windows na versão beta	Adicionado suporte para contêineres do Windows Server 2016 (beta). Para obter mais informações, consulte Variantes da AMI otimizada para Amazon ECS.	20 de dezembro de 2016
Suporte para Blox OSS	Adicionado suporte para Blox OSS, que habilita programadores de tarefas personalizados. Para obter mais informações, consulte Programação de contêiner es no Amazon ECS.	1° de dezembro de 2016

Alteração	Descrição	Data
Sequência de eventos do Amazon ECS para CloudWatch Events	Agora o Amazon ECS envia alterações de instâncias de contêineres e de status de tarefas ao CloudWatch Events. Para obter mais informações, consulte Automatização de respostas a erros do Amazon ECS usando o EventBridge.	21 de novembro de 2016
Registro de contêiner do Amazon ECS no CloudWatch Logs	Adicionado suporte para que o driver awslogs envie fluxos de log de contêineres para o CloudWatch Logs. Para obter mais informações, consulte Envio de logs do Amazon ECS para o CloudWatch .	12 de setembro de 2016
Serviços do Amazon ECS com suporte ao Elastic Load Balancing para portas dinâmicas	Adicionado suporte para um load balancer para oferecer suporte a várias instâncias:combinações de portas por listener, o que aumenta a flexibilidade para contêineres. Agora, você pode permitir que o Docker defina dinamicamente a porta de host do contêiner e o programador do ECS registra a porta da instância com o load balancer. Para obter mais informaçõ es, consulte Uso do balanceamento de carga para distribuir o tráfego de serviço do Amazon ECS.	11 de agosto de 2016
Funções do IAM para tarefas do Amazon ECS	Adicionado suporte para associar funções do IAM a uma tarefa. Isto fornece permissões mais refinadas para os contêineres em vez de uma única função para uma instância de contêineres completa. Para obter mais informações, consulte Perfil do IAM para tarefas do Amazon ECS.	13 de julho de 2016
Suporte para o Docker 1.11	Adicionado suporte para Docker 1.11. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.	31 de maio de 2016
Escalabilidade automátic a de tarefas	O Amazon ECS adicionou suporte para escalabilidade automática das tarefas executadas por um serviço. Para obter mais informações, consulte Como escalar automaticamente o serviço do Amazon ECS.	18 de maio de 2016

Alteração	Descrição	Data
Filtragem de definição de tarefa com base em família de tarefas	Adicionado suporte para filtragem de uma lista de definição de tarefa com base na família de definição de tarefa. Para obter mais informações, consulte ListTaskDefinitions.	17 de maio de 2016
Registro de contêineres do Docker e agentes do Amazon ECS	O Amazon ECS adicionou a capacidade de enviar logs de agentes do ECS e de contêineres do Docker de instâncias de contêineres para o CloudWatch Logs para simplificar a solução de problemas.	5 de maio de 2016
A AMI otimizada para ECS agora é compatível com Amazon Linux 2016.03.	Adicionado suporte à AMI otimizada para ECS para Amazon Linux 2016.03. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.	5 de abril de 2016
Suporte para o Docker 1.9	Adicionado suporte para Docker 1.9. Para obter mais informações, consulte AMIs do Linux otimizadas para o Amazon ECS.	22 de dezembro de 2015
Métricas do CloudWatch para reserva de CPU e memória do cluster	O Amazon ECS adicionou métricas personalizadas do CloudWatch para reserva de CPU e memória do cluster.	22 de dezembro de 2015
Nova experiência de primeira execução do Amazon ECS	A experiência de primeira execução do console do Amazon ECS adicionou a criação da função de zero clique.	23 de novembro de 2015
Colocação de tarefas nas zonas de disponibilidade	O programador de serviços do Amazon ECS adicionou suporte para posicionamento de tarefas em zonas de disponibilidade.	8 de outubro de 2015

Alteração	Descrição	Data
Métricas do CloudWatch para clusters e serviços do Amazon ECS	O Amazon ECS adicionou métricas personalizadas do CloudWatch para utilização de CPU e memória para cada instância de contêiner, serviço e família de definição de tarefa em um cluster. Essas novas métricas podem ser usadas para dimensionar instâncias de contêineres em um cluster usando grupos do Auto Scaling ou para criar alarmes personalizados do CloudWatch.	17 de agosto de 2015
Suporte para porta de UDP	Adicionado suporte para portas de UDP em definições de tarefa.	7 de julho de 2015
Substituições de variáveis de ambiente	Adicionado suporte para deregisterTaskDefinition e substituições de variáveis de ambientes para runTask.	18 de junho de 2015
Atualizações automatizadas do agente do Amazon ECS	Adicionada capacidade para ver a versão do agente do ECS que está em execução em uma instância de contêiner. Também é capaz de atualizar o agente do ECS a partir do AWS Management Console, da AWS CLI e do SDK.	11 de junho de 2015
Integração do programador de serviço do Amazon ECS e do Elastic Load Balancing	Adicionada capacidade para definir um serviço e associar este serviço a um balanceador de carga do Elastic Load Balancing.	9 de abril de 2015
Amazon ECS disponível para o público	Amazon ECS disponível para o público nas regiões Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Ásia-Pacífico (Tóquio) e Europa (Irlanda).	9 de abril de 2015